Placement of Social Digital Twins at the Edge for Beyond 5G IoT Networks

Olga Chukhno, *Graduate Student Member, IEEE*, Nadezhda Chukhno, *Graduate Student Member, IEEE*, Giuseppe Araniti, *Senior Member, IEEE*, Claudia Campolo, *Senior Member, IEEE*, Antonio Iera, *Senior Member, IEEE*, and Antonella Molinaro, *Senior Member, IEEE*

***Abstract*—As the fifth-generation (5G) and beyond (5G+/6G) networks move forward, and a wide variety of new advanced Internet of Things (IoT) applications are offered, effective methodologies for discovering time-relevant information, services, and resources are being demanded. To this end, computing-, storage-, and battery-constrained IoT devices are progressively augmented via digital twins (DTs) hosted on edge servers. According to recent research results, a further feature these devices may acquire is social behavior; this latter offers enormous possibilities for fast and trustworthy service discovery, although it requires new orchestration policies of DTs at the network edge. This work addresses the dynamic placement of DTs with social capabilities [social digital twins (SDTs)] at the edge, by providing an optimal solution under IoT device mobility and by accounting for edge network deployment specifics, types of devices, and their social peculiarities. The optimization problem is formulated as a particular case of the quadratic assignment problem (QAP); also, an approximation algorithm is proposed and two relaxation techniques are applied to reduce computation complexity. Results show that the proposed placement policy ensures a latency among SDTs up to 1.4 times lower than the one obtainable with a traditional proximity-based only placement while still guaranteeing appropriate proximity between physical devices and their virtual counterparts. Moreover, the proposed heuristic closely approximates the optimal solution while guaranteeing the lowest computational time.**

***Index Terms*—Digital twin (DT), edge computing, orchestration, Social Internet of Things (SIoT).**

Manuscript received 11 June 2022; accepted 30 June 2022. Date of publication 13 July 2022; date of current version 21 November 2022. This work was supported by the European Union's Horizon 2020 Research and Innovation Programme under the Marie Skłodowska Curie Grant under Agreement 813278 (A-WEAR: A Network for Dynamic Wearable Applications With Privacy Constraints). *(Corresponding author: Olga Chukhno.)*

Olga Chukhno is with DIIES, University Mediterranea of Reggio Calabria and CNIT, 89060 Reggio Calabria, Italy, and also with the ITC Faculty, Electrical Engineering, Tampere University, 33014 Tampere, Finland (e-mail: olga.chukhno@unirc.it).

Nadezhda Chukhno is with DIIES, University Mediterranea of Reggio Calabria and CNIT, 89060 Reggio Calabria, Italy, and also with the Institute of New Imaging Technologies, Universitat Jaume I, 12071 Castellón de la Plana, Spain (e-mail: nadezda.chukhno@unirc.it).

Giuseppe Araniti and Claudia Campolo are with DIIES, University Mediterranea of Reggio Calabria and CNIT, 89060 Reggio Calabria, Italy (e-mail: araniti@unirc.it; claudia.campolo@unirc.it).

Antonio Iera is with DIMES, University of Calabria, 87030 Arcavacata, Italy, and also with CNIT, 43124 Parma, Italy (e-mail: antonio.iera@dimes.unical.it).

Antonella Molinaro is with DIIES, University Mediterranea of Reggio Calabria and CNIT, 89060 Reggio Calabria, Italy, and also with the CentraleSupélec, Université Paris-Saclay, 91190 Gif-sur-Yvette, France (e-mail: antonella.molinaro@unirc.it).

Digital Object Identifier 10.1109/JIOT.2022.3190737

I. INTRODUCTION

BY 2050, there will be 24 billion interconnected devices [1], meaning almost every object around us will be connected to the Internet, such as wearable devices, mobile phones, robots, electric meters, cars, and streetlights. Such devices will allow end users to enjoy a wide variety of innovative applications, ranging, for instance, from augmented reality (AR)/extended reality (XR) to autonomous assistance navigation [2], demanding high throughput, low latency, high reliability, and pervasive availability.

Although several countries are rolling out the fifth-generation (5G) network, the requirements of most of the applications mentioned above are still hardly supported, hence motivating the research community to look ahead to beyond-5G solutions toward 2030. Among them, digital twins (DTs), acting as high-fidelity digital mirrors of physical entities, appear as the game changer to fully enable the digital transformation and cope with the increasing connectivity, computing, and storage demands of massively deployed heterogeneous Internet of Things (IoT) devices in 5G and beyond network scenarios [3].

As examined in [4], connectivity between physical and virtual counterparts, i.e., DTs, is an open issue attracting considerable interest, especially with a view to guaranteeing real-time data transfer [5]. To this aim, there is a wide consensus on placing DTs at the network edge so to ensure low-latency interactions with their physical counterparts located in proximity [6], [7]. However, the decision about the proper placement of DTs has to account for the limited and heterogeneous resources at edge servers. Such decision becomes even more complicated when considering mobile devices in the physical realm that constantly trigger DT migrations among edge servers to ensure proximity to the physical devices. Recent works have addressed these issues; for instance, deep reinforcement learning (DRL) and an algorithm based on iteratively solving a series of minimum graph cuts have been leveraged, respectively, in [8] and [9], to approximate the optimal placement solution.

A further feature, which adds constraints to the placement policy, is the possibility for physical devices to establish mutual social relationships, e.g., according to the Social IoT (SIoT) paradigm, which has gained great popularity in recent years in the IoT research arena [10], [11]. A social network of devices is created by establishing and maintaining different types of relationships, such as co-location, ownership, and parental, among others [10]. Indeed, IoT applications can be conceived so to leverage data and services provided by "friend" devices. Their discovery, by navigating the social network graph, may be facilitated if we introduce so-called social DTs (SDTs) at the edge, used to expose the mentioned resource/services on behalf of the physical devices and also to keep trace of the social relationships dynamically established among their physical counterparts. As a consequence, a wise placement of SDTs must also be implemented to make the social network browsing quicker and more effective.

In light of the above, it is clear that the dynamic placement of DTs with social capabilities at the edge is a challenging decision to be taken when both user- and operator-centric requirements need to be simultaneously satisfied. In our previous work in [12], the issue has been addressed through the formulation of an initial optimization problem under basic conditions and without considering SDT mobility, as the main objective was to provide a proof of concept of the introduced paradigm. Differently, in this work, we propose a significantly extended study, which accounts for more realistic operational conditions and provides the contributions summarized as follows.

1) The design of an SIoT-Edge framework, closely aligned with the multiaccess edge computing (MEC) architecture, standardized by the European Telecommunications Standards Institute (ETSI) [13], wherein the proposed SDT placement, named enhanced social-aware closest edge placement (eSoCEP), is conceived as a functionality of the ETSI mobile-edge (ME) orchestrator.
2) The formulation of the optimal placement of SDTs as a quadratic assignment problem (QAP), which extends the preliminary formulation in [12] by accounting for different types of IoT devices, their social features, mobility patterns, and the limited computing resources of edge servers.
3) The design of an approximation scheme to find near-optimal solutions and the application of approximation techniques addressing the challenge of the NP-hardness of the formulated optimization problem.
4) The evaluation of the performance of the proposed algorithm against the formulated optimal solution through joined CPLEX and MATLAB simulations.
5) The analysis of the time-dependent behavior of the SIoT-Edge system under conditions of device mobility for the dynamic SDT placement problem, and the consequent selection of the proper time interval duration between consecutive runs of the SDT placement strategy for the scenario of interest.
6) The implementation of an extensive performance evaluation campaign aiming at studying the impact of the input parameters and of the social relationship types and device categories on meaningful performance metrics.

The remainder of this work is organized as follows. An overview of related work is provided in Section II. In Section III, we illustrate the SIoT-Edge framework. In Section IV, we present the reference system model used for the solution of the dynamic SDT placement problem. Section V formulates the optimization problem as a particular case of the QAP. Section VI proposes an approximation algorithm and describes relaxation techniques for the SDT placement. Simulation results are reported in Section VII. The main findings of the study are summarized in Section VIII. Finally, in Section IX, conclusions are drawn and hints on future works are provided.

II. Background and Motivations

Most existing works on the SIoT have focused on smart objects and their interconnection via social relationships, as well as on specific requirements and constraints raised in this domain. Examples of addressed research on this topic are IoT objects' social behavior understanding [14], service discovery [15], trustworthiness management [16], and critical security and privacy provisioning [17], among others. Thanks to more recent frameworks, platforms, and architectures [11], this research orientation is solid in its current form, as knowledge has already been accumulated through both theoretical and experimental studies.

As for DTs, in [3], its general concept has been examined by considering different functions, configurations, and patterns. In [18], a DT wireless network model has been designed to transfer data processing and computation to the network edge in real time. A vision of DT edge networks has been proposed in [19], where the offloading scheme is based on DRL, and DTs aid the offloading decision by estimating the states of edge servers. In [20], a framework has been introduced for improving the energy efficiency of services in a MEC system, where DTs are used to train the deep learning algorithm. A framework to track mobility has been presented in [21], where DTs retrieve mobility data of physical entities. In [22], a content caching mechanism with DT support has been designed for socially oriented vehicular edge networks.

A. Placement at the Edge

The problem of service placement, both in general and in the specific case of DTs, has been widely discussed in the scientific literature, and numerous solutions have appeared. Some existing works have investigated the possibilities of edge infrastructures to satisfy the latency constraints between a physical device and its DT [12]. In [7], a cloudlet placement strategy has been proposed, taking into account the cost of deploying edge servers and the end-to-end latency between physical objects and their virtual entities. In [23], a service placement framework has been designed to achieve a tradeoff between latency and migration costs. In [24], a joint optimization on the service placement and access network selection has been studied to improve the Quality of Service (QoS). In [9], a service entity placement problem considering activation, placement, proximity, and co-location costs has been formulated and solved with an iterative algorithm.

Moreover, machine learning (ML) techniques for entity/service placement at the network edge have been recently investigated in [25] as a means to efficiently solve complex edge placement optimization problems, instead of formulating mathematically tractable heuristics. Decision

TABLE I
DT PLACEMENT SOLUTIONS IN THE LITERATURE VERSUS OUR WORK: COMPARISON OF SUPPORTED FEATURES

Ref., year	**Device mobility**	**Social features**	**Cost function to be optimized**	**Device-DT delay constraint**	**Edge capacity constraints**	**Device heterogeneity**	**Edge heterogeneity**	**Solution to the optimization problem**
[7], 2019	Yes	No	Sum of cloudlet cost and device-DT latency	No	Yes	No	Yes	Lagrangian heuristic algorithm
[8], 2021	Yes	No	Sum of DT initialization (device-DT delay) and synchronization delay	Yes	Yes	Yes	Yes	DRL-based algorithm
[9], 2018	Yes	Twitter social graph	Sum of activation, placement, proximity (device-DT delay), and colocation cost	No	No	Yes	Yes	Iterative solution of a series of minimum graph cuts
[23], 2018	Yes	No	Sum of computing and communication delay (device-DT latency)	No	No	Yes	Yes	Distributed approximation scheme
[24], 2019	Yes	No	Sum of access, switching and communication (device-DT) delay	No	Yes	Yes	Yes	Iteration-based algorithm
[12], 2020	No	SIoT relationships from Santander dataset	Sum of device-SDT delay and inter-friend SDT delay	Yes	Yes	Yes	No	Not provided
Our work	**Yes**	**SIoT relationships from Santander dataset**	**Sum of device-SDT delay and inter-friend SDTs delay, with the latter one weighted by frequency of data exchange among friend SDTs**	**Yes**	**Yes**	**Yes**	**Yes**	**Graph-based heuristic**

tree-based algorithms have shown to be the best performing among supervised learning schemes. DRL and transfer learning are used in [8] to find the solutions of the optimal DT placement and migration problems, respectively.

Table I summarizes the main features of the closest related works, by highlighting the differences with our work. It emerges that the past literature does not cover the full convergence of virtualization and socialization capabilities of future IoT devices and applications, by also accounting for heterogeneous and resource-limited edge computing environments.

B. Contributions of the Work

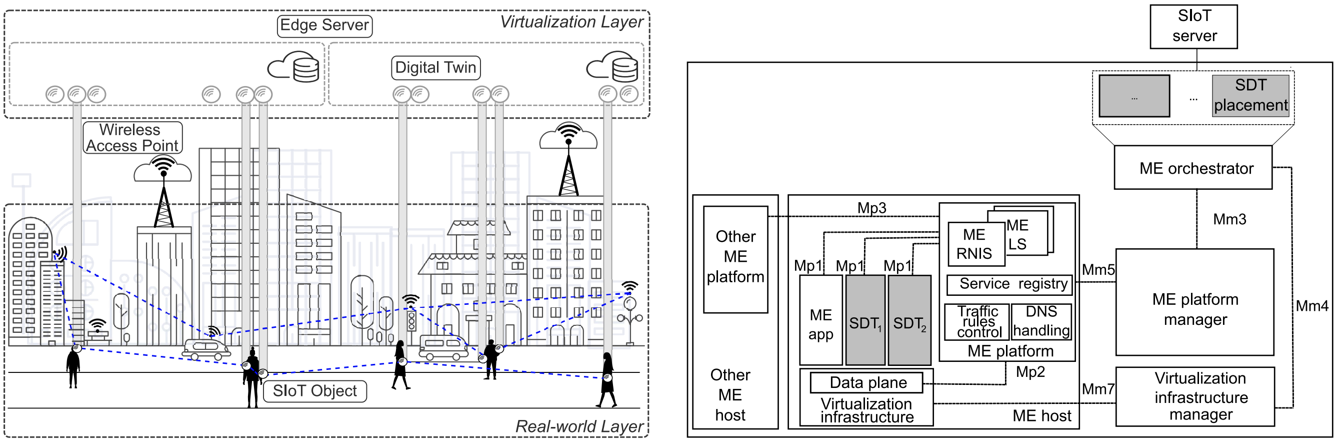
This work aims to fill the aforementioned critical gap by contributing a framework and methodology for the dynamic placement of SDTs that build upon the SIoT, edge computing, and DT concepts. More specifically, SDTs should be placed as closer as possible to their physical counterparts to ensure low-latency interactions between IoT objects and corresponding SDTs. This can also help to reduce the amount of traffic traversing the edge network segment. In addition, social relationships have to be accounted for, since SDTs may likely need to interact with each other to offer SIoT-based services. SIoT objects may need to quickly interoperate and discover services with a low data footprint on the edge infrastructure by querying the social network. Hence, SDTs of friend SIoT devices should preferably be placed in close edge servers.

Both aspects have been accounted for in our early study in [12], where an optimal "static" SDT placement problem was formulated. However, the optimization was subject to strong assumptions which, although tolerable for carrying out an initial proof of concept, did not allow to exploit the proposed model in real scenarios, mainly due to the following limitations, which this article overcomes. *First*, a simple proximity-driven only SDT placement may not be feasible due to the limited resources of edge servers; it gets increasingly more difficult as the demands of the SDTs in terms of computing/storage resources get higher and heterogeneous, as expected under realistic circumstances. *Second*, the inter-SDTs latency should be minimized not in a myopic manner, but rather accounting for the specific relationship existing between the corresponding physical devices, which entails the exchange of data with different intensiveness. *Third*, efficient use of available resources plays a key role in satisfying the requirements of both users and operators. *Fourth*, a placement decision cannot be static. Instead, it shall account for the mobility patterns of physical IoT devices that may trigger repeated migrations of SDTs among different edge servers. *Finally,* the design and validation of an efficient heuristic solution were missing there, which are, instead, among the additional contributions of the present work toward the practical deployments of the proposed framework.

III. SIoT-EDGE FRAMEWORK: OVERVIEW

This section presents a general overview of the proposed SIoT-Edge framework for the dynamic placement of SDTs. Similarly to our early study [12], the reference architecture consists of a *real-world layer* and a *virtualization layer* [see Fig. 1(a)].[1]

[1]The description of the architecture is here briefly reported to make this article self-contained.



(a) (b)

Fig. 1. SIoT-Edge framework. (a) Reference architecture. (b) Envisioned components within the ETSI MEC architecture.

The *real-world layer* represents the physical world that accommodates IoT objects interconnected with each other and to other entities through connectivity facilities. Social relationships among objects are assumed, which are set up according to the SIoT paradigm [10]. For instance, the co-location object relationship (C-LOR) is established among objects located in the same place. The ownership object relationship (OOR) specifies connections among objects that belong to the same owner. The parental object relationship (POR) is defined among objects belonging to the same production batch. Social object relationships (SOR) is established due to sporadic or continuous contact of users/devices.

The *virtualization layer* is responsible for hosting the digital counterparts of physical devices [26], i.e., the SDTs. They offer the typical functionalities a digital counterpart provides, including caching and aggregation of the raw data transmitted by the IoT device, before IoT applications can process them. In addition, the proposed SDT stores metadata describing the device type and the established SIoT relationships. An IoT device, willing to query friend devices, discover services offered by them, and/or push data to them, needs to read the friendship information stored in its SDT. Once such a piece of information is retrieved, the SDT itself can interact with its peers on behalf of the physical device.

We assume that SDTs are deployed as a virtualized ME app (e.g., through containers) and instantiated in edge servers. The latter ones, referred to as ME hosts, in agreement with the ETSI MEC architecture [13], may be associated with base stations (BSs)/access points (APs).

In order to align our proposal with the ETSI MEC architecture [13], the SIoT-Edge framework components, as shown in Fig. 1(b), are considered.

In the ETSI MEC architecture, the ME orchestrator has visibility of the resources and capabilities of the entire edge network, made up of several ME hosts, and determines the most suitable ME hosts for instantiating the applications (i.e., ME apps) according to the application requirements (e.g., latency, processing, memory, etc.), available resources, and mobility conditions. In case a virtualized application needs to be relocated, the orchestrator triggers the migration procedure.

In the envisioned framework, the ME orchestrator is in charge of selecting the ME hosts wherein each SDT should be placed [see the corresponding functional module in Fig. 1(b)]. Besides, in our design, the ME orchestrator may interact with an external SIoT server to get information about the current social relationships established by any given physical device to decide the most suitable placement of its SDT. In particular, the SIoT server may record the profiles of SIoT devices, their relationships, as well as activities. The location information of the objects can also be managed and then updated in the profile on the SIoT server.

IV. System Model

This section outlines the reference system model and summarizes our modeling assumptions. Some of them are inherited from our previous work in [12] and shortly recalled here to make this article self-contained. Others have been specifically added to match the additional contributions of the work. Unlike [12], the system is assumed to operate according to a discrete timing based on a sequence of time slots $t \in \mathcal{T} = \{0, \ldots, T\}$ with the duration of τ (in minutes), introduced to capture the mobility features and offer dynamic decisions. The assumption is quite common in [7], [23], and [27]. Table II lists the basic notations employed in this work.

A. SIoT Devices and SDTs

We consider N IoT devices within the coverage area of M wireless APs (e.g., BSs and APs). At a given time instant, an IoT device i is assumed to be connected to a single AP/BS, particularly the closest one [28]. A device i can move and change the connectivity point [see Fig. 2(b)].

Considered devices establish relationships according to the SIoT paradigm. The resulting social network is represented by a social-based graph $G_P(t) = (V_P(t), E_P(t))$. The set of vertices in the graph $G_P(t)$, i.e., $V_P(t)$, corresponds to the IoT devices connected by links in set $E_P(t)$. The probability $p_{ij}(t)$, $0 \le p_{ij}(t) \le 1$, reflects the intensiveness of the data exchange between IoT devices i and j, and is associated with SIoT links. It is straightforward to assume that the $p_{ij}(t)$ value is strongly

TABLE II
SUMMARY OF THE MAIN NOTATIONS

Notation	Description
N	Number of IoT devices
M	Number of edge servers
$G_P(t) = (V_P(t), E_P(t))$	Weighted undirected graph of physical IoT devices
$p_{ij}(t)$	Probability of data exchange between IoT devices $i, j \in V_P(t)$
$aCPU_k$	CPU capability of edge server $k \in V_S(t)$
aD_k	Disk capability of edge server $k \in V_S(t)$
$aRAM_k$	Memory capability of edge server $k \in V_S(t)$
$CPU_i(t)$	CPU requirement to execute the SDT of physical IoT device $i \in V_P(t)$
$D_i(t)$	Disk requirement to execute the SDT of physical IoT device $i \in V_P(t)$
$RAM_i(t)$	Memory requirement to execute the SDT of physical IoT device $i \in V_P(t)$
$G_S(V_S(t), E_S(t))$	Weighted undirected graph of edge servers
$L_{ik}(t)$	Latency between physical device $i \in V_P(t)$ and its SDTs placed at edge server $k \in V_S(t)$
$L_{kl}(t)$	Latency between edge servers $k, l \in V_S(t)$
$d_{ik}(t)$	Physical distance between IoT device $i \in V_P(t)$ and edge server $k \in V_S(t)$ (that hosts its SDT)
$d_{kl}(t)$	Physical distance between SDTs deployed at edge servers $k, l \in V_S(t)$
$x_{ik}(t)$	Binary variable taking the value 1 if SDT of device $i \in V_P(t)$ is mapped to edge server $k \in V_S(t)$(t)
$L_{\max_i}$	Maximum latency between physical device $i \in V_P(t)$ and its SDT deployed at at edge server $k \in V_S(t)$
THR_{CPU}	Threshold value of CPU utilization
THR_D	Threshold value of disk storage utilization
THR_{RAM}	Threshold value of RAM utilization
$C_{ikjl}(t)$	Cost of connections between devices $i, j \in V_P(t)$ and their SDTs placed at edge servers $k, l \in V_S(t)$

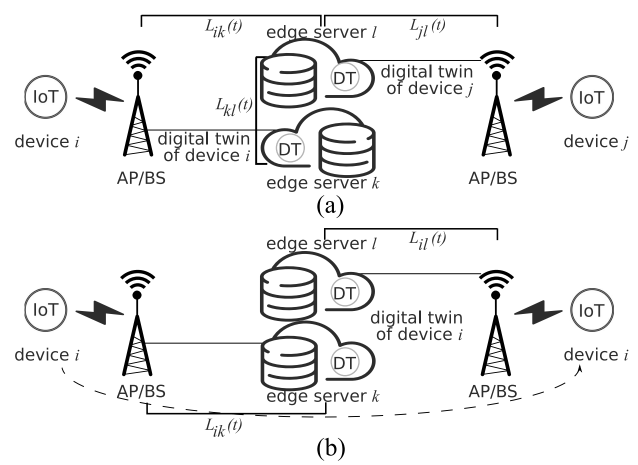


Fig. 2. Example of dynamic SDT placement problem: (a) illustration of SDT placement and (b) illustration of SDT's migration.

correlated with the specific kind of social relationship established between the two physical devices [29]. IoT devices tied by an OOR may need to frequently share data about the smart home/the smart car of the owner as well as her habits, preferences, and health status. This would not be the case for POR. In case more than one relationship is established between two devices, the maximum $p_{ij}(t)$ value is utilized.

B. Network Edge

We consider an edge infrastructure consisting of M edge servers associated with each wireless AP/BS [28] and, unlike [7] and [12], equipped with heterogeneous capabilities. An edge server k has a finite amount of CPU, disk, and RAM resources, denoted as $a\text{CPU}_k$, aD_k, and $a\text{RAM}_k$, respectively [30]. Such servers are in charge of hosting SDTs, which are in turn associated with IoT devices. SDTs can store data and perform some processing, having specific CPU, disk, and RAM demands. The latter ones are indicated for an SDT i as $CPU_i(t)$, $D_i(t)$, and $RAM_i(t)$.

The edge network is represented by graph $G_S = (V_S(t), E_S(t))$, where $V_S(t)$ is a finite set of edge servers, whereas $E_S(t)$ is a set of links between the edge servers. We reasonably assume that the number of IoT devices is larger than the number of edge servers, $|V_P(t)| = N > |V_S(t)| = M$, which does not limit the generality of the presentation.

The latency between each pair of edge servers k, l is given by $L_{kl}(t)$. Similar to [7] and [9], $L_{kl}(t)$ is estimated to be proportional to the distance between them. We denote as $L_{ik}(t)$ the latency between device i and its SDT located at edge server k. Similar to [31], for $L_{ik}(t)$, we neglect the delay over the radio interface and consider the latency between a BS that covers the device and an edge server that hosts the corresponding SDT. Hence, $L_{ik}(t)$ and $L_{kl}(t)$ are estimated as in [32]

$$L_{ik}(t) = \epsilon d_{ik}(t) \tag{1}$$
$$L_{kl}(t) = \epsilon d_{kl}(t) \tag{2}$$

where ϵ is the distance to latency mapping coefficient, $d_{ik}(t)$ is the physical distance between the BS that serves device i and edge server k, whereas $d_{kl}(t)$ is the distance between edge servers k and l.

V. Optimal Social Digital Twin Placement

We formulate the optimal SDT placement problem by targeting the following main objectives.

1) To jointly minimize the latency between each IoT device and its relevant SDT placed at an edge server and the latency between friends SDTs, while accounting for the relationship existing between the corresponding physical devices, and, hence, for the intensiveness of the expected data exchange.
2) To ensure that delay bounds on the interactions between IoT devices and their SDTs are met, whenever requested.
3) To guarantee the effective utilization of the available resources for heterogeneous SDT demands.

Hence, we define the objective function to be optimized and the relevant constraints in the following.

A. Objective Function

We define the objective function as a cost to be minimized given the sum of two latency contributions. The first component includes the latency experienced between a physical device and the corresponding SDT and is given by

$$C_1(t) = \sum_{i \in V_P} \sum_{k \in V_S} x_{ik}(t) L_{ik}(t) \tag{3}$$

where $x_{ik}(t)$ is the SDT placement decision variable and is equal to 1 if the SDT of device i is placed at edge server k at time slot t; otherwise, $x_{ik}(t) = 0$, i.e.,

$$x_{ik}(t) \in \{0, 1\} \quad \forall\, i \in V_P \quad \forall\, k \in V_S \quad \forall\, t \in \mathcal{T}. \tag{4}$$

The second component of the objective function includes the latency among SDTs of friend devices at time slot t, i.e., the time required to communicate and discover services querying the friends on the virtualization layer, and is as follows:

$$C_2(t) = \sum_{i \in V_P} \sum_{k \in V_S} \sum_{j \in V_P} \sum_{l \in V_S} x_{ik}(t) x_{jl}(t) p_{ij}(t) L_{kl}(t) \tag{5}$$

where $x_{jl}(t)$ is the SDT placement decision variable, $x_{jl}(t) \in \{0, 1\}$.

Hence, we define the total cost at time slot t as

$$C(t) = C_1(t) + C_2(t). \tag{6}$$

B. Constraints

1) Allocation Constraint: Allocation constraint is responsible for the placement of SDTs without replication. Since each SDT is allocated only to one edge server, we have the following constraint for SDT placement decision $x_{ik}(t)$:

$$\sum_{k \in V_S} x_{ik}(t) = 1 \quad \forall\, i \in V_P \quad \forall\, t \in \mathcal{T}. \tag{7}$$

2) Latency Constraint: Latency constraint is compliant with the idea of meeting the proximity constraint for the SDT of a given physical device and preserves a limitation on the latency between IoT device i and its SDT deployed at edge server k, which is upper bounded by $L_{\max_i}$, i.e.,

$$L_{ik}(t) \le L_{\max_i} \quad \forall\, i \in V_P \quad \forall\, k \in V_S \quad \forall\, t \in \mathcal{T}. \tag{8}$$

3) Resource Utilization Constraints: Resource utilization constraints guarantee efficient resource utilization while preventing the overload of a given edge server k. The constraints ensure SDT placement according to edge server resource availability and guarantee that the capacity constraint (i.e., CPU, THR_{CPU}, disk storage, THR_D, and RAM, THR_{RAM}, utilization) for each edge server at time slot t is not violated when multiple IoT devices simultaneously share the computing resources to host the corresponding SDTs at edge servers $\forall\, k \in V_S\ \forall\, t \in \mathcal{T}$

$$\sum_{i \in V_P} \frac{x_{ik}(t) CPU_i(t)}{a\text{CPU}_k} \le THR_{\text{CPU}} \tag{9}$$

$$\sum_{i \in V_P} \frac{x_{ik}(t) D_i(t)}{aD_k} \le THR_D \tag{10}$$

$$\sum_{i \in V_P} \frac{x_{ik}(t) RAM_i(t)}{a\text{RAM}_k} \le THR_{\text{RAM}}. \tag{11}$$

C. Problem Formulation

The SDT placement problem can further be formulated as follows:

$$\begin{aligned} &\min \quad C(t) \\ &\text{s.t.} \quad (4),\ (7), (8), (9), (10), (11). \end{aligned} \tag{12}$$

In each time slot t, an optimal placement can be obtained when solving (12) with the exhaustive search.

D. Complexity Analysis

Lemma 1: Optimal SDT placement in the dynamic large-scale SIoT-Edge environment problem is NP-hard.

Proof: We conducted the proof in our previous work in [12] via a polynomial-time reduction from the QAP, which is known to be NP-hard [33]. ■

E. Linearization

We aim to remove the nonlinearity of function (12), specifically (5), and perform the linearization of the objective function. When elaborating $C_2(t)$, we first denote the cost contributions related to the latency between the SDTs of IoT devices i and j placed at edge servers k and l at time slot t, respectively, as $C_{ikjl}(t)$, by replacing $p_{ij}(t) L_{kl}(t)$. We reformulate (5) as follows [34]:

$$\begin{aligned} &\sum_{i \in V_P} \sum_{k \in V_S} \sum_{j \in V_P} \sum_{l \in V_S} x_{ik}(t) x_{jl}(t) C_{ikjl}(t) \\ &\quad = \sum_{i \in V_P} \sum_{k \in V_S} x_{ik}(t) \sum_{j \in V_P} \sum_{l \in V_S} x_{jl}(t) C_{ikjl}(t). \end{aligned} \tag{13}$$

We then define $x_{ik}(t) \sum_{j \in V_P} \sum_{l \in V_S} x_{jl}(t) C_{ikjl}(t)$ by introducing $F_{ik}(t)$ and express the minimization of $C^L(t)$

$$\min \quad C^L(t) = C_1(t) + \sum_{i \in V_P} \sum_{k \in V_S} F_{ik}(t) \tag{14}$$

$$\text{s.t.} \quad f_{ik}(t) x_{ik}(t) + \sum_{j \in V_P} \sum_{l \in V_S} x_{jl}(t) C_{ikjl}(t) - F_{ik}(t) \le f_{ik}(t) \tag{15}$$

$$F_{ik}(t) \ge 0 \quad \forall\, i \in V_P \quad \forall\, k \in V_S \quad \forall\, t \in \mathcal{T} \tag{16}$$

where $f_{ik}(t)$ is given by

$$f_{jl}(t) = \sum_{j \in V_P} \sum_{l \in V_S} C_{ikjl}(t). \tag{17}$$

VI. Approximation Solutions

Computing the optimal policy solution for the SDT placement is very difficult in practical cases. Indeed, as mentioned in Section V-D, it is an NP-hard problem. The solution via exhaustive search suggested in Section V can provide an optimal configuration for a small size network, but it turns to be of no practical use for large networks. This is why in this section we aim to design an approximation solution for the SDT placement problem that is simpler to compute while still achieving close-to-optimal performance. The algorithm carries out a relaxing transformation to simplify the original problem, and a graph-based heuristic is derived from it. We then introduce alternative known approximation techniques, i.e., the associated local branching (LB) and relaxation-induced neighborhood search (RINS) heuristics, which are also applied to our problem.

A. Proposed Graph-Based Heuristic

The QAP has been drawing researchers' attention worldwide because of its practical and theoretical importance as

Algorithm 1: Graph-Based Heuristic

1 **Input:** $G_P(t) = G_P(V_P(t), E_P(t))$;
  $G_S(t) = G_S(V_S(t), E_S(t))$;
2 **Output:** $V_P(t) \rightarrow V_S(t)$;
3 find sets of connected components $G_{P'}(t) = G_P(V_{P'}(t))$, $|G_{P'}(t)| = n$ such that $V_{P'}(t) \subseteq V_P(t), E_{P'}(t) \subseteq E_P(t)$,
  $\forall u, v \in V_{P'}(t) \; \exists \; (u, v)$,
  $\forall u \in V_{P'}(t), w \notin V_{P'}(t) \; \nexists \; (u, w)$;
4 $MAX_W \leftarrow 0$;
5 **while** $G_{P'}(t) \neq \emptyset$ **do**
6   **for** $m = 1{:}|G_{P'}(t)|$ **do**
7   find $W(t, m) = \sum_{i,j \in G_{P'}(t,m)} p_{ij}(t)$;
8   **if** $MAX_W < W(t, m)$ **then**
9       $MAX_W \leftarrow W(t, m)$;
10       $G_{P'_{\max}}(t) \leftarrow G_{P'}(t, m)$;
11   **end**
12   **end**
13   find spanning subgraph $T(G_{P'_{\max}}(t))$ such that $V_P(T) = V_P(G_{P'_{\max}}(t)) \wedge E_P(T) \subseteq E_P(G_{P'_{\max}}(t))$, $|E_P(T)| = [V_P(G_{P'_{\max}}(t))| - 1$;
14   $G_{P'}(t) \leftarrow G_{P'}(t) \setminus G_{P'_{\max}}(t)$;
15   find optimal mapping $\Pi(t) = \{\pi(t){:}V_P(T) \rightarrow V_S\}$;
16 **end**
17 **return** $V_P(t) \rightarrow V_S(t)$.

well as its complexity. The QAP is one of the most challenging combinatorial optimization problems. However, to the best of our knowledge, there is no theoretical proof concerning quality and computational time convergence, especially for large-scale dimension problems. We focus on achieving low enough execution time in this work while ensuring reasonable approximation to the optimal solution.

For the SDTs placement problem, we present an approximation algorithm, which is based on a graph-theoretic solution. The pseudocode of the graph-based heuristic is listed in Algorithm 1 and it is executed for each time slot $t \in \mathcal{T}$.

The formulation of the approximation solution in terms of graph theory is as follows. Let $G_P(t)$ be a weighted connected graph, $V_P(t)$ be the set of vertices of graph $G_P(t)$ corresponding to the SDTs, and $E_P(t)$ be the set of links of the graph $G_P(t)$ defining the connections between the SDTs allocated at edge servers. Let $V_S(t)$ be a finite set of positions intended for assigning vertices of the graph $G_S(t)$ corresponding to the set of edge servers.

Algorithm 1 starts with the definition of a connected component $G_{P'}(t)$ of graph $G_P(t)$, i.e., identification of individual connectivity components (line 3). It allows defining the number of strongly connected components in which a path from each vertex to another vertex exists. The algorithm considers each component separately (lines 5–16), starting from the strongest one between vertices of a component (lines 6–12).

Next, Algorithm 1 finds an approximating spanning subgraph or, in other words, a spanning tree (line 13). In the mathematical field of graph theory, a spanning tree T of an undirected graph $G_{P'}(t)$ is a subgraph that is a tree, which includes all of the vertices of $G_{P'}(t)$ with a minimum possible number of links. If all of the links of $G_{P'}(t)$ are links of a spanning tree T of $G_{P'}(t)$, then $G_{P'}(t)$ is a tree and is identical to T. The advantages of spanning tree usage and, therefore, problem simplification are as follows. First, constructing a spanning tree takes a polynomial time when using well-known algorithms (e.g., Boruvka's, Prim's, Kruskal's, reverse-delete, greedy algorithms, etc.). Second, the problem of tree placement can be solved relatively quickly.

We then perform mapping $\Pi(t)$ (line 15) by placing the vertices of graph $G_P(t)$, assuming that vertex $i \in V_P(t)$ is allocated in the position $\pi(i) \in V_S(t)$, such that any vertex of $V_S(t)$ can either accommodate vertices of $V_P(t)$ or accommodate no vertices. The set of all mappings of set $V_P(t)$ into set $V_S(t)$ is given by

$$\Pi(t) = \{\pi(t) : V_P(t) \rightarrow V_S(t)\}. \tag{18}$$

We specify the following parameters. First, the distance $L_{ik}(t)$ between vertex $i \in V_P(t)$ and position $k \in V_S(t)$, defined in terms of the latency of the connectivity between physical device i and its SDTs placed at edge server k. We refer to as distance $L_{ik}(t)$ the cost of placing vertex $i \in V_P(t)$ in position $k \in V_S(t)$. We then define: the weight of the edge $p_{ij}(t)$ associated with the probability of data exchanging between IoT devices; the distance L_{kl} between positions $k, l \in V_S(t)$, defined in terms of the latency of the connectivity between edge server k and edge server l. The cost of communication between vertices $i, j \in V_P(t)$ placed in positions $k, l \in V_S(t)$ corresponds to $C_{ikjl}(t) = p_{ij}(t)L_{kl}(t)$.

As we aim to allocate the vertices of graph $G_P(t)$ in positions $V_S(t)$ by minimizing the total cost of placing the vertices $V_P(t)$ to positions $V_S(t)$, the problem is formulated in terms of mappings as follows:

$$\min_{\pi(t) \in \Pi(t)} \left\{ \sum_{i \in V_P(t)} L(i, \pi(i)) + \sum_{i \in V_P(t)} \sum_{j \in V_P(t)} C(i, j, \pi(i), \pi(j)) \right\}. \tag{19}$$

Algorithm 1 determines the most suitable positions to host vertices of the spanning tree (T) to minimize the cost of spanning tree's vertices allocated to the set of positions $G_S(t)$ (19). The Algorithm terminates when all vertices of $G_P(t)$ are mapped onto positions belonging to $G_S(t)$ by satisfying constraints (4) and (7)–(11).

Complexity Analysis: The computational complexity of the proposed algorithm (Algorithm 1) is given by

$$O(n) \cdot O(n) = O\left(n^2\right)$$

where n is the complexity due to the *while* cycle over all $|G_p(t)| = n$ vertices of graph $G_P(t)$ in the worst case when the number of connected components of graph $G_P(t)$ is equal to the number of graph vertices (lines 5–16). For the second component, which is inside the *while* cycle, n is the complexity due to the searching for the largest connected component in terms of the number of communications between vertices (lines 6–12). Since initially $|G_{p'}(t)| = n$ and at each iteration one of the elements is removed, this inner loop runs n times first, then $n-1, n-2$, and so on until at the last iteration, the inner loop runs only once. The complexity of the sum $1+2+\cdots+(n-1)+n$ is problematic to be precisely determined (lines 6–12). Instead, we find an upper limit for it, which is $O(n)$. This means that every time the inner loop goes exactly n times. The complexity of lines 13–16 is $O(1)$, but it runs within *while* cycle, therefore, $O(n \cdot 1) = O(n)$, which does not change the complexity of the first component. As a result, the operations' numbers' upper limit is in $O(n^2)$.

In contrast, the problem of optimal SDT placement in the dynamic large-scale SIoT-Edge environment has been formulated as a QAP, which is NP-hard. The NP-hard problems can be solved but not in polynomial time, i.e., there are no solutions that can offer the result within $O(n^k)$ for any constant $k \geq 2$. Moreover, QAP is one of the most challenging combinatorial optimization problems [35]. While theoretical, algorithmic, and technological breakthroughs have resulted in large increases in the sizes of solvable issues for many well-known NP-hard problems, QAP has remained a stand-alone class that appears to defy all attempts to solve it except for very small sizes [35]. General form of QAP requires the specification of $O(n^4)$ cost terms for $C_2(t)$ [36]. Therefore, the lower limit of the number of operations for the dynamic large-scale SIoT-Edge environment problem is in $O(n^4)$.

Hence, the conceived heuristic ensures a significantly lower theoretical complexity than the optimal problem formulation.

B. Classic Relaxation Techniques

The solution of the problem (12) can also be adopted by utilizing branch-and-cut or branch-and-bound techniques. In this work, we consider LB heuristic and RINS heuristic and compare them with the optimal solution as well as the proposed approximation algorithm.

1) LB Heuristic: LB is based on the idea of changing neighborhoods during the search for a better solution [37]. LB is a technique designed, in principle, as an exact method. However, if the total time allocated to solve a given instance is reached before an optimal solution is found and its optimality is proved, then LB stops at the time with the best solution known as output (or, possibly, with no feasible solution).

2) RINS Heuristic: RINS is a heuristic that explores a neighborhood of the incumbent solution to find a new, improved incumbent [38]. RINS constructs a promising neighborhood using information contained in the continuous relaxation of the mixed-integer programming (MIP) model. The neighborhood depends on the current incumbent solution and the current fractional solution of a branch-and-cut node. The neighborhood is formulated and explored as another MIP. This optimization is truncated by limiting the number of nodes explored in the search tree.

VII. Experimental Evaluation

This section aims to assess the performance of the proposed SDT placement optimization strategy. First, we describe the conducted simulation campaign, including the considered scenario, settings, benchmark schemes, and metrics of interest. Then, we compare the results achieved through the optimal solution, the approximation solutions, i.e., the graph-based and the LB and RINS heuristics, and some benchmark placement schemes, by means of a simulator tool based on the IBM ILOG CPLEX Optimization Studio 12.10.0 and MATLAB R2021b software. Simulation settings are reported in the remainder of this section and are summarized in Table III.

TABLE III
System Model Parameters

Deployment	
Area of interest	**Area:** Santander, Spain [39] **Size:** 4000 m x 4000 m [39]
Users	**Number:** 50 (100) **Mobility pattern:** SWIM [39], [40]
IoT Devices	**Smartphones:** 6% (12%) [39] **Cars:** 15% (14%) [39] **Tablets:** 12% (11%) [39] **Smart Fitness:** 20% (24%) [39] **Smartwatches:** 29% (25%) [39] **PCs (static):** 1% (6%) [39] **Printers (static):** 10% (2%) [39] **Home Sensors (static):** 7% (6%) [39] **Total number:** 113 (328) [39]
Social network	**Probability of data exchange:** 1 (OOR), 0.1 (C-LOR), 0.1 (SOR), 0.1 (POR) *: 1 (OOR), 1 (C-LOR), 0.1 (SOR), 0.1 (POR)
Radio	
Base stations	**Cell layout:** 3GPP hexagonal grid [41] **Number of BSs:** 8 [41] **Cell area radius:** 450 m [41] **Intersite distance:** 1350 m [41]
Edge	
Edge servers	**Deployment:** Co-location with BSs [28] **Number:** Equal to number of BSs [28] **Distance:** Geographical distance [7], [9] **Latency:** Proportional to the distance (Eq. (2) with ϵ 3.33 ms/km [32]) **CPU capability:** 24000 MIPS [42] **Disk capability:** 2 TB [42] **RAM capability:** 24 GB [42]
Resource utilisation constraints	**CPU utilization threshold:** 0.6 [43] **Disk utilization threshold:** 0.9 [43] **RAM utilization threshold:** 0.9 [43]
SDTs	
Disk	**Disk demands:** Uniformly distributed in $[10, 50]$ GB [30]
CPU demands / RAM demands	**High-CPU medium instance:** 2000 MIPS/0.85 GB [44] **Extra large instance:** 2500 MIPS/3.75 GB [44] **Small instance:** 1000 MIPS/1.7 GB [44] **Micro instance:** 500 MIPS/613 MB [44]
IoT-Edge	
IoT devices-SDTs	**Distance:** Geographical distance [7], [9] **Latency:** Proportional to the distance with ϵ 3.33 ms/km [32]
Proximity constraint	**Physical device-SDT maximum latency:** Uniformly distributed in $[1, 10]$ ms [45]

A. Simulation Settings

1) Network and Edge Deployments: Similar to [39], we consider the city center of Santander (Spain), which roughly has an area of 4 km $\times$ 4 km. We assume the hexagonal grid cellular layout, in agreement with the Third Generation Partnership Project (3GPP) specifications [41], with $M = 8$ BSs. An edge server is associated with each BS, in agreement with ETSI documents [28].

2) SIoT Devices and Mobility: We evaluate the proposal on the basis of a realistic object behavior taken from the large data set[2] generated in [39] that tracks device interactions based on real IoT objects[3] and the small world in motion (SWIM) mobility model [40]. SWIM is based on a simple intuition on human mobility, i.e., people go more often to places close to their home and to the most popular places. We consider two device density settings (i.e., portions of the data set), namely, 50 and 100 users with $N = 113$ and $N = 328$ heterogeneous IoT devices, respectively, spanning from static devices to consumer devices carried by users moving in the selected area of interest (see Table III for the percentages of each category of IoT devices).

3) SDT Demands: An SDT is paired to each IoT device and implemented as a container in edge facilities [46]. Without loss of generality, to account for the heterogeneity of IoT devices, we associate SDTs with four types of containers (based on the corresponding device types) according to the CPU demands [44] as presented in Table III. For example, cars with autonomous assistance navigation and smartphones might require high CPU for their SDTs; whereas SDTs of smartwatches and sensors and tablets, smart fitness devices, and printers might be associated with small and micro instances, respectively. As a further requirement, the maximum latency between a physical device and its SDT is uniformly distributed in the interval [1, 10] ms [45]. The latency is derived according to the geographical distance between any two entities deployed in the reference area [7], [9], [32] as described in Section IV-B.

4) SIoT Features: Social relationships are associated with each device in the data set. For the extracted 113 devices, the percentage of established relationships corresponds to 50%, 21%, 15%, and 14%, for OOR, C-LOR, SOR, and POR, respectively. Whereas for the 328 devices setup, the percentage of established relationships is as follows: 60% OOR, 8% C-LOR, 1% SOR, and 31% POR.

Unlike all the aforementioned settings, there are no specific clues about how to set the parameter $p_{ij}(t)$. Hence, without loss of generality, we consider a set of representative values throughout the simulation campaign to understand their impact on the conceived SDT placement strategy compared to solutions oblivious of the social relationship information (see the following). In particular, we fix $p_{ij}(t) = 0.1$ for SOR and POR. The rationale behind these values is that IoT devices, either sporadically coming into contact (i.e., tied by an SOR relationship) or belonging to the same brand or product batch (i.e., tied by a POR relationship), are reasonably expected not to exchange big amount of data with a high frequency, but only upon some specific events, when compared to the other friendship types. For instance, devices tied by a POR may seldom exchange software updates. Data exchanges among objects of the same owner, i.e., those tied by an OOR relationship, may be frequent and huge, e.g., to synchronize personal/health data, to monitor the smart home, etc. Hence, we set for OOR $p_{ij}(t) = 1$, and we vary it to be 0.1 and 1 for those devices tied by a C-LOR. However, it is worth remarking that the strategy is flexible enough to accommodate other settings.

[2]http://www.social-iot.org

[3]https://www.fiware.org

5) Simulation Time: The entire simulation covers a timelapse of 5 h. Within this period, we first simulate time slots t of duration equal to $\tau = 5$ min [23]. Then, we vary this setting up to 30 min in steps of 5 min. We determine the best time slot duration based on the analysis of simulation results. During each time slot t, we assume that the SDT placement does not change [23], [27].

6) Benchmark Schemes: We compare the optimal solution (labeled in the curves as *eSoCEP*) with the following approximation solutions and placement strategies.

1) *Approximation Techniques:*
   a) Proposed graph-based heuristic, as per Algorithm 1, labeled in the curves as *eSoCEP Heuristic* (see Section VI-A).
   b) LB [37], labeled in the curves as *LB* (see Section VI-B1).
   c) RINS [38], labeled in the curves as *RINS* (see Section VI-B2).
2) *Placement Strategies:*
   a) Social-aware closest edge placement (SoCEP) of SDTs [12], labeled in the curves as *SoCEP*, which only takes into account proximity and social requirements without accounting for the specific nature of SIoT links, types of devices, network deployment specifics, and mobility. We also simulate the graph-based heuristic for SoCEP, labeled in the curves as *SoCEP Heuristic*.
   b) Closest edge placement (CEP), labeled in the curves as *CEP*, widely leveraged in the literature and also known as the *FollowMe* strategy, is representative of the approaches in [7], [9] and considered in [8] and [47] as a benchmark solution. According to CEP, SDTs paired with physical devices are always placed at the nearest edge server, by neglecting social features.
   c) Static placement, labeled in the curves as *No Migration*, is a strategy according to which SDTs are initially placed at the nearest edge server and keep the same placement throughout the whole simulation duration without migration possibilities [23], [48].

It should be noticed that, for the sake of fair comparison, a dynamic placement is also triggered at every time slot for both SoCEP[4] and CEP.

7) Metrics of Interest: We evaluate the performance of proposed and benchmark solutions by leveraging the following metrics.

1) *Average Latency Between IoT Devices and Their SDTs:* We measure this metric as the average latency between a physical device and the edge server hosting the corresponding SDT.
2) *Average Latency Among Friend SDTs:* This metric refers to the average delay experienced between the SDTs of friend devices.

[4]A static placement was assumed in the original design in [12].

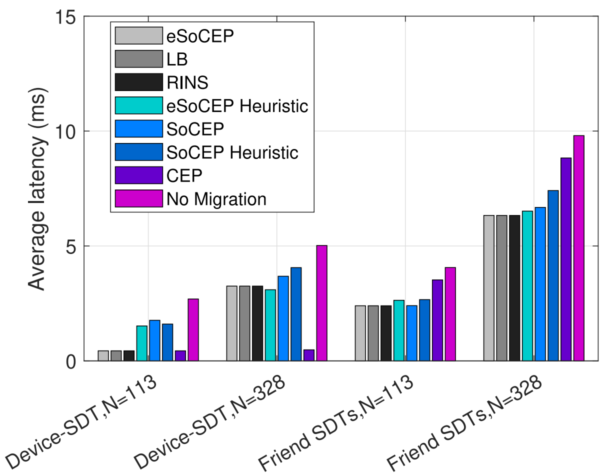


Fig. 3. Average latency: time interval duration $\tau = 5$ min [23], probability of data exchange $p_{ij}(t) = 1$ for OOR and $p_{ij}(t) = 0.1$ for C-LOR, SOR, and POR, and average number of friend IoT devices $f = 4$ for $N = 113$ and $f = 5$ for $N = 328$.

3) *Total Number of Migrated SDTs:* This metric refers to the total number of SDTs migrated from one edge server to another during the whole simulation.

B. Heuristic Validation and Complexity Analysis

The first set of results aims to validate the effectiveness and efficiency of the proposed eSoCEP heuristic (for both $N = 113$ and $N = 328$) when compared to the optimal solution, the considered relaxation techniques (LB and RINS), and the benchmarks solutions. As one may observe in Fig. 3, eSoCEP and eSoCEP Heuristic preserve close values for all considered metrics and under all device density settings. This is especially true for the latency among friend SDTs for which values are significantly lower compared to the CEP and No Migration benchmarks. Instead, the device-SDT latency, although higher for the eSoCEP Heuristic compared to the optimal solution, is in any case bound by the proximity constraint in (8).

Furthermore, we analyze the average latency among friend SDTs per relationship type, as illustrated in Fig. 4(a) and (b). The optimal eSoCEP solution guarantees zero latency among OOR friends for $N = 113$ and the lowest values for $N = 328$. This implies that SDTs of friend devices are co-located in the same edge server, in alignment with the targeted objectives, well captured by the parameter $p_{ij}(t)$ set equal to 1. Higher latency values are measured for the other kinds of relationships. In particular, the highest latency values are experienced among POR friends in the case of $N = 113$ IoT devices, because devices establishing such a kind of relationship are more likely spread throughout the topology.

As a next step, we assess the computational complexity. To this aim, in Fig. 5, we report the algorithms' running time as a function of the number of devices in the system. We evaluate placement strategies via simulations on an Intel Xeon CPU E5 − 2620 v4 at 2.10 GHz with 19.7-GB RAM.

We start with the comparison of eSoCEP and SoCEP heuristics with the optimal solver. In eSoCEP, the introduction of stricter constraints on latency and resource usage (see

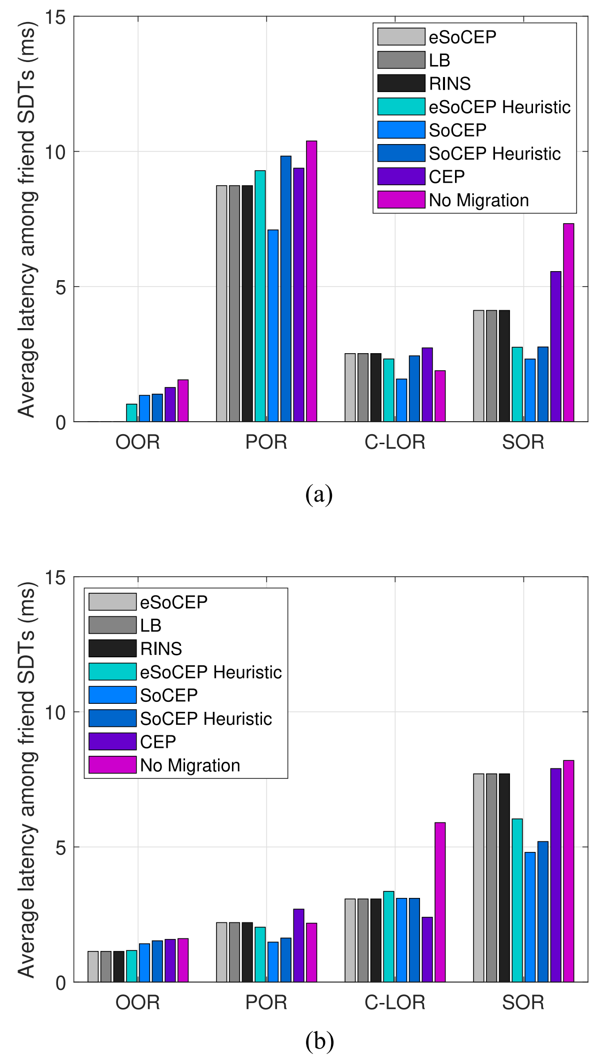


Fig. 4. Average latency among friend SDTs per relationship type: time interval duration $\tau = 5$ min [23] and probability of data exchange $p_{ij}(t) = 1$ for OOR and $p_{ij}(t) = 0.1$ for C-LOR, SOR, and POR. (a) $N = 113$, average number of friend IoT devices f=4. (b) $N = 328$, average number of friend IoT devices f=5.

Section V-B) as well as a linearization of the optimization function, which allows for faster optimal solution search (see Section V-E), lead to significant reduction in complexity. Contrarily to eSoCEP, the SoCEP strategy fails to scale as the number of devices and the average number of friends increase. Moreover, we note that LB and RINS (dashed lines) applied to the eSoCEP solution, respectively, decrease the running time on average by 7.5% and 11.5% compared to eSoCEP for the average number of friend IoT devices f equal to 4, and by 8.8% and 11.3% for $f = 7$.

From results in Fig. 5, it further emerges that the eSoCEP heuristic outperforms all the considered placement strategies and approximation solutions. It offers on average 43.3% and 46.9% reduction of the running time compared to the optimal solution for $f = 4$ and $f = 7$, respectively. For $N = 328$, such reduction is up to nearly 94%. Here, approximated spanning subgraph usage is beneficial for the following reasons. First, it is possible to construct a spanning subgraph in polynomial time by using well-known algorithms (see Section VI).

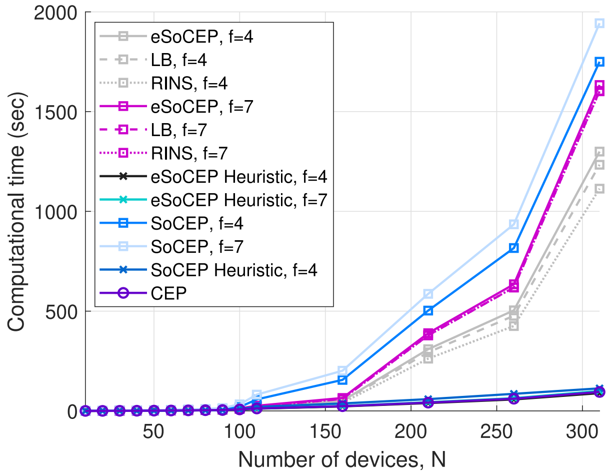


Fig. 5. Computational time when varying the number of IoT devices N and the average number of friend IoT devices $f = 4$ and $f = 7$.

Second, the problem of tree placement can be solved relatively quickly. The eSoCEP heuristic preserves low enough computational time regardless of f. Such a finding concerning the scalability of the proposed heuristic with the number of friends is particularly relevant, since IoT devices are expected to establish many relationships. What is more interesting is that the eSoCEP heuristic, representative of a more sophisticated SDT placement strategy, incurs the same computation time as the simplest CEP approach (violet line), whose detailed results are reported in the following.

C. Impact of the Time Slot Duration

So far, we have analyzed the results of the dynamic placement strategies when fixing the time slot to 5 min, similar to [23]. In this section, we assess the results when varying the duration of the time slot (i.e., the frequency of the optimization recomputation) for the reference scenario.

Fig. 6 reports the average latency contributions for different time interval duration, τ. In particular, in Fig. 6(a), it can be observed that the latency between the physical devices and their digital counterparts is sensitive to the setting of the time slot, whatever the considered dynamic placement strategy. The larger the time slot duration, the less frequent is the replacement of the SDTs. Hence, all the considered placement strategies hardly capture the device mobility, as witnessed by the increasing latency values when the time slot duration increases. This is especially true for the CEP strategy, whose latency values are almost doubled when passing from $\tau = 5$ min to $\tau = 30$ min. The CEP strategy exhibits the lowest latency, being specifically conceived to ensure that IoT devices have their SDT hosted in close proximity, neglecting the existence of social relationships. The proposed eSoCEP strategy outperforms SoCEP. If a static SDT placement (No Migration) strategy is enforced, the poorest performance is achieved with latency values that are nearly doubled compared to those measured for eSoCEP (averaged values through the entire simulation, not shown in the figure, are in the order of 3 ms).

When considering the latency among friend SDTs, Fig. 6(b), both SoCEP and eSoCEP outperform other schemes, thus confirming their capability to account for social relationships demands, as targeted by the formulated problem. For the No Migration policy, delay values close to 4 ms are measured, although not reported in the Figure to reduce cluttering.

From Fig. 6(c), interestingly, it emerges that the proposed eSoCEP solution is more efficient than SoCEP. It can be observed that it always triggers a lower number of migration events compared to SoCEP. As a consequence, the overhead incurred by migration procedures is lower, i.e., a lower amount of data is exchanged over backhaul links interconnecting the edge servers acting as the source and destination of migrating SDTs. It can be further noticed that, unlike latency metrics, the total number of migration events is highly sensitive to the settings of the time slot, t. Such a finding suggests to further investigate the results for a specific τ value, i.e., 20 min, which allows for achieving a good tradeoff between the number of migration events and the latency metrics.

D. Impact of the Device Type

Under such settings, it is worth understanding the SDT migration dynamics per category of IoT devices. Fig. 7 shows the percentage of migration events at the end of each time slot, when a replacement is triggered, for two representative device categories, i.e., smartwatches and home sensors. Not surprisingly, the SDTs of smartwatches migrate more than the SDTs of home sensors. You can notice that, although associated with static devices, the latter may also migrate for SoCEP and eSocEP to satisfy the targeted constraints of proximity among SDTs of friend devices. Also, for the considered device categories, eSoCEP is more efficient than SoCEP.

To further shed light on the latency performance, the latency among friend SDTs is reported in Fig. 8 per relationship type. Here, we consider two settings for the proposed heuristic, namely, for bars labeled as eSoCEP Heuristic*, probability of data exchange corresponds to $p_{ij}(t) = 1$ for OOR and C-LOR and $p_{ij}(t) = 0.1$ for SOR and POR, whereas for bars labeled as eSoCEP Heuristic, $p_{ij}(t) = 1$ for OOR and $p_{ij}(t) = 0.1$ for C-LOR, SOR, and POR.

Let us remember that SoCEP does not distinguish the relationship type when placing SDTs. In alignment with the targeted objectives, among the compared schemes, eSoCEP (both considered settings) provides the lowest latency values for SDTs associated with IoT devices establishing OOR relationships. The eSoCEP Heuristic* achieves a lower latency value for C-LOR as well, when compared to the other dynamic strategies, at the expense of slightly higher latency values for SOR and POR. Interestingly, latency values achieved by social-aware strategies for SOR mainly established by mobile devices, are more than halved, compared to the No Migration approach. Slightly higher values than SoCEP are experienced by eSoCEP only for the latency among SDTs associated with IoT devices establishing POR relationships, which, however, are expected to interact at a lower extent. Hence, the results in Fig. 8 confirm the flexibility of the proposed

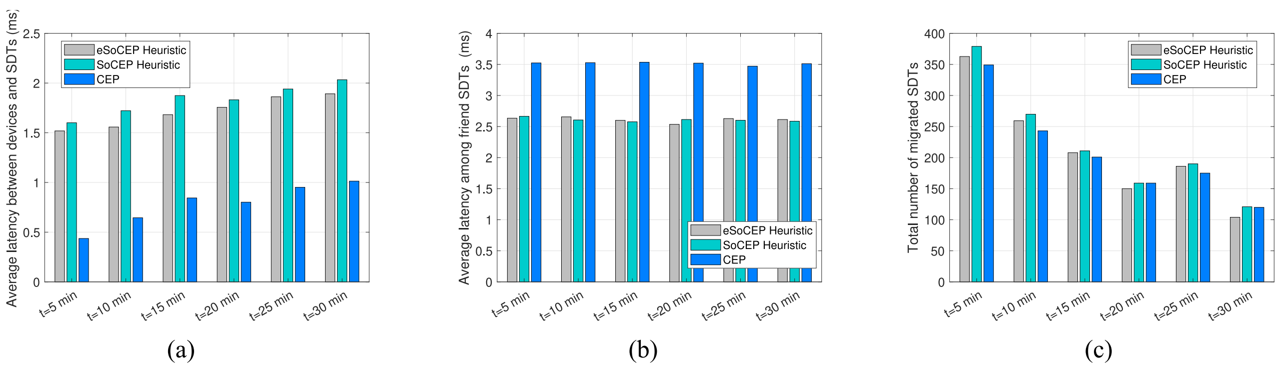


Fig. 6. Metrics of interest for different time interval duration τ: average number of friend IoT devices f=4 and probability of data exchange $p_{ij}(t) = 1$ for OOR and $p_{ij}(t) = 0.1$ for C-LOR, SOR, and POR, $N = 113$. (a) Latency between IoT devices and SDTs. (b) Latency among friend SDTs. (c) Total number of migrated SDTs during the simulation.

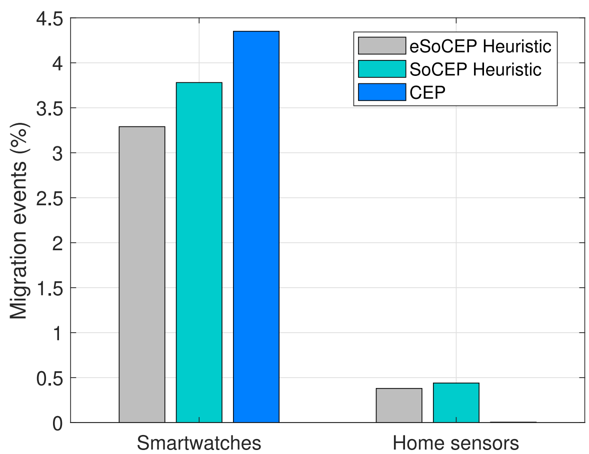


Fig. 7. Percentage of migration events for different device types: average number of friend IoT devices $f = 4$, $\tau = 20$ min, and probability of data exchange $p_{ij}(t) = 1$ for OOR and $p_{ij}(t) = 0.1$ for C-LOR, SOR, and POR, $N = 113$.

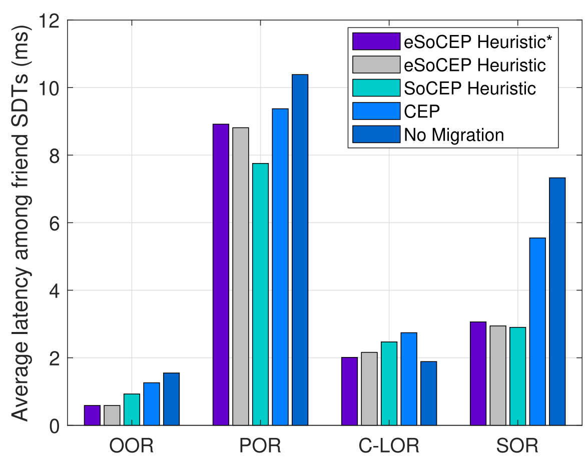


Fig. 8. Average latency among friend SDTs per relationship type: average number of friend IoT devices $f = 4$, $\tau = 20$ min, and probability of data exchange $p_{ij}(t) = 1$ for OOR and C-LOR and $p_{ij}(t) = 0.1$ for SOR and POR (eSoCEP Heuristic*) and $p_{ij}(t) = 1$ for OOR and $p_{ij}(t) = 0.1$ for C-LOR, SOR, and POR (eSoCEP Heuristic), $N = 113$.

Heuristic that allows for minimizing the latency in the data exchange based on its intensiveness for a given relationship type.

VIII. Discussion

The proposed placement, eSoCEP, satisfactorily targets the objective of ensuring the lowest latency among SDTs of friend devices that are more likely to exchange data, e.g., those tied by OOR while guaranteeing appropriate proximity between physical devices and their physical counterparts. A lower latency among SDTs has a twofold benefit: 1) the pressure on the network is low when SDTs exchange data because packets traverse a lower number of links and 2) quick interactions are ensured among them, which is crucial for service discovery procedures entailing the browsing of the social graph.

Compared to the benchmark solutions, the eSoCEP heuristic is shown to be efficient in terms of computation time, being its execution even faster than the most popular placement strategy, CEP, which is myopic w.r.t. the need to ensure proximity among SDTs of friend devices. Moreover, the eSoCEP heuristic is faster than the heuristic for our previous proposal, SoCEP, and also more efficient in terms of placement decisions.

eSoCEP incurs a lower number of migrations w.r.t. SoCEP. Hence, a lower communication footprint is incurred, since whenever a migration is triggered to match a new placement decision, data needs to be exchanged from the source to the target edge server.

The proposal is currently aimed to minimize the cost function, which jointly accounts for device-SDT latency and interfriend SDT latency. The proximity between the physical devices and their virtual counterparts is however ensured by a hard delay constraint.

Nonetheless its potentials and promising achievements, the proposal has room for improvement.

First, the formulated problem could be extended to overstep a possible limitation as, in this stage, only the communication latency is considered. The computation delay experienced at the selected edge server to perform the related IoT devices augmentation tasks is not currently accounted for in the problem. The formulation is quite flexible to accommodate such a further latency contribution. However, in order to do this, either assumptions are needed concerning the workload of each SDT associated with a given IoT device (as in [8]) or realistic patterns should be derived from actual IoT deployments. This issue will be among the subject matters of future work.

Moreover, the considered data set assumes that social relationships are static. However, they may evolve (be created and deleted) over time [49] and hence, placement decisions should vary according to the relationships' lifecycle. The chosen time slot values are short enough to capture well such dynamics and hence, SDTs can be moved to follow variations in the social relationships with no impact on the data exchange performance. Nonetheless, such replacements occur at the expenses of higher migration costs, which should also be considered.

IX. Conclusion and Future Work

In this work, we have developed a framework for the dynamic placement of DTs associated with physical IoT devices establishing social relationships. The conceived placement strategy accounts for the social features of IoT devices, their mobility patterns, and the limited computing resources of edge servers. The optimal placement of SDTs has been formulated as a QAP. We designed a heuristic and applied approximation techniques addressing the challenge of the optimization problem complexity. Numerical results demonstrated that relaxation techniques approximate very well the exact solution, whereas the proposed graph-based heuristic allows for preserving polynomial time complexity while also keeping results close to the optimal solution.

Future research directions include, among others, ML algorithms for mobility- and time-dependent SDTs (re)allocation, where the proposed placement might be utilized to generate the training data. Among ML algorithms, it could be worth to investigate DRL, in alignment with recent literature trends, to enhance SDTs placement efficiency.

Furthermore, the placement decision might be extended toward a green and sustainable dimension.

References

[1] "Why IoT Changes Everything." Ericsson. 2021. [Online]. Available: https://www.ericsson.com/

[2] L. U. Khan, W. Saad, D. Niyato, Z. Han, and C. S. Hong, "Digital-twin-enabled 6G: Vision, architectural trends, and future directions," *IEEE Commun. Mag.*, vol. 60, no. 1, pp. 74–80, Jan. 2022.

[3] H. X. Nguyen, R. Trestian, D. To, and M. Tatipamula, "Digital twin for 5G and beyond," *IEEE Commun. Mag.*, vol. 59, no. 2, pp. 10–15, Feb. 2021.

[4] B. R. Barricelli, E. Casiraghi, and D. Fogli, "A survey on digital twin: Definitions, characteristics, applications, and design implications," *IEEE Access*, vol. 7, pp. 167653–167671, 2019.

[5] M. Mashaly, "Connecting the twins: A review on digital twin technology & its networking requirements," *Procedia Comput. Sci.*, vol. 184, pp. 299–305, May 2021, doi: 10.1016/j.procs.2021.03.039. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877050921006694

[6] L. Atzori *et al.*, "SDN & NFV contribution to IoT objects Virtualization," *Comput. Netw.*, vol. 149, pp. 200–212, Feb. 2019.

[7] Q. Fan and N. Ansari, "On cost aware cloudlet placement for mobile edge computing," *IEEE/CAA J. Automatica Sinica*, vol. 6, no. 4, pp. 926–937, Jul. 2019.

[8] Y. Lu, S. Maharjan, and Y. Zhang, "Adaptive edge association for wireless digital twin networks in 6G," *IEEE Internet Things J.*, vol. 8, no. 22, pp. 16219–16230, Nov. 2021.

[9] L. Wang, L. Jiao, T. He, J. Li, and M. Mühlhäuser, "Service entity placement for social virtual reality applications in edge computing," in *Proc. IEEE INFOCOM Conf. Comput. Commun.*, 2018, pp. 468–476.

[10] L. Atzori, A. Iera, G. Morabito, and M. Nitti, "The social Internet of Things (SIOT)—When social networks meet the Internet of Things: Concept, architecture and network characterization," *Comput. Netw.*, vol. 56, no. 16, pp. 3594–3608, 2012.

[11] L. Atzori, A. Iera, and G. Morabito, "Sociocast: A new network primitive for IoT," *IEEE Commun. Mag.*, vol. 57, no. 6, pp. 62–67, Jun. 2019.

[12] O. Chukhno, N. Chukhno, G. Araniti, C. Campolo, A. Iera, and A. Molinaro, "Optimal placement of social digital twins in edge IoT networks," *Sensors*, vol. 20, no. 21, p. 6181, 2020.

[13] *Mobile Edge Computing (MEC); Framework and Reference Architecture*, ETSI Standard GS MEC 003 v1.1.1, Mar. 2016.

[14] S. Dhelim, H. Ning, F. Farha, L. Chen, L. Atzori, and M. Daneshmand, "IoT-enabled social relationships meet artificial social intelligence," *IEEE Internet Things J.*, vol. 8, no. 24, pp. 17817–17828, Dec. 2021.

[15] A. Khanfor, H. Ghazzai, Y. Yang, M. R. Haider, and Y. Massoud, "Automated service discovery for social Internet-of-Things systems," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, 2020, pp. 1–5.

[16] R. K. Chahal, N. Kumar, and S. Batra, "Trust management in social Internet of Things: A taxonomy, open issues, and challenges," *Comput. Commun.*, vol. 150, pp. 13–46, Jan. 2020.

[17] X. Gan, Y. Li, Y. Huang, L. Fu, and X. Wang, "When crowdsourcing meets social IoT: An efficient privacy-preserving incentive mechanism," *IEEE Internet Things J.*, vol. 6, no. 6, pp. 9707–9721, Dec. 2019.

[18] Y. Lu, X. Huang, K. Zhang, S. Maharjan, and Y. Zhang, "Low-latency federated learning and blockchain for edge association in digital twin empowered 6G networks," *IEEE Trans. Ind. Informat.*, vol. 17, no. 7, pp. 5098–5107, Jul. 2021.

[19] W. Sun, H. Zhang, R. Wang, and Y. Zhang, "Reducing offloading latency for digital twin edge networks in 6G," *IEEE Trans. Veh. Technol.*, vol. 69, no. 10, pp. 12240–12251, Oct. 2020.

[20] R. Dong, C. She, W. Hardjawana, Y. Li, and B. Vucetic, "Deep learning for hybrid 5G services in mobile edge computing systems: Learn from a digital twin," *IEEE Trans. Wireless Commun.*, vol. 18, no. 10, pp. 4692–4707, Oct. 2019.

[21] C. Campolo, G. Genovese, A. Molinaro, and B. Pizzimenti, "Digital twins at the edge to track mobility for MaaS applications," in *Proc. IEEE/ACM 24th Int. Symp. Distrib. Simul. Real Time Appl. (DS-RT)*, 2020, pp. 1–6.

[22] K. Zhang, J. Cao, S. Maharjan, and Y. Zhang, "Digital twin empowered content caching in social-aware vehicular edge networks," *IEEE Trans. Comput. Social Syst.*, vol. 9, no. 1, pp. 239–251, Feb. 2022.

[23] T. Ouyang, Z. Zhou, and X. Chen, "Follow me at the edge: Mobility-aware dynamic service placement for mobile edge computing," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 10, pp. 2333–2345, Oct. 2018.

[24] 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 INFOCOM Conf. Comput. Commun.*, 2019, pp. 1459–1467.

[25] Y. Zhao, Y. Li, X. Zhang, G. Geng, W. Zhang, and Y. Sun, "A survey of networking applications applying the software defined networking concept based on machine learning," *IEEE Access*, vol. 7, pp. 95397–95417, 2019.

[26] C. Campolo, G. Genovese, A. Iera, and A. Molinaro, "Virtualizing AI at the distributed edge towards intelligent IoT applications," *J. Sens. Actuator Netw.*, vol. 10, no. 1, p. 13, 2021.

[27] X. Chen *et al.*, "Dynamic service migration and request routing for microservice in multi-cell mobile edge computing," *IEEE Internet Things J.*, early access, Jan. 4, 2022, doi: 10.1109/JIOT.2022.3140183.

[28] S. Kekki *et al.*, "MEC in 5G networks," ETSI, Sophia Antipolis, France, White Paper 28, 2018.

[29] L. Atzori *et al.*, "Enhancing identifier/locator splitting through social Internet of Things," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 2974–2985, Apr. 2019.

[30] X. Liu, B. Cheng, and S. Wang, "Availability-aware and energy-efficient virtual cluster allocation based on multi-objective optimization in cloud datacenters," *IEEE Trans. Netw. Service Manag.*, vol. 17, no. 2, pp. 972–985, Jun. 2020.

[31] Q. Fan and N. Ansari, "Application aware workload allocation for edge computing-based IoT," *IEEE Internet Things J.*, vol. 5, no. 3, pp. 2146–2153, Jun. 2018.

[32] R. Landa, J. T. Araújo, R. G. Clegg, E. Mykoniati, D. Griffin, and M. Rio, "The large-scale geography of Internet round trip times," in *Proc. IFIP Netw. Conf.*, 2013, pp. 1–9.

[33] T. C. Koopmans and M. Beckmann, "Assignment problems and the location of economic activities," *Econometrica J. Econometr. Soc.*, vol. 25, no. 1, pp. 53–76, 1957.

[34] L. Kaufman and F. Broeckx, "An algorithm for the quadratic assignment problem using bender's decomposition," *Eur. J. Oper. Res.*, vol. 2, no. 3, pp. 207–211, 1978.
[35] S. Sahni and T. Gonzalez, "P-complete approximation problems," *J. ACM*, vol. 23, no. 3, pp. 555–565, 1976.
[36] E. L. Lawler, "The quadratic assignment problem," *Manage. Sci.*, vol. 9, no. 4, pp. 586–599, 1963.
[37] M. Fischetti and A. Lodi, "Local branching," *Math. Program.*, vol. 98, no. 1, pp. 23–47, 2003.
[38] E. Danna, E. Rothberg, and C. Le Pape, "Exploring relaxation induced neighborhoods to improve MIP solutions," *Math. Program.*, vol. 102, no. 1, pp. 71–90, 2005.
[39] C. Marche, L. Atzori, V. Pilloni, and M. Nitti, "How to exploit the social Internet of Things: Query generation model and device profiles' dataset," *Comput. Netw.*, vol. 174, Jun. 2020, Art. no. 107248.
[40] A. Mei and J. Stefa, "SWIM: A simple model to generate small mobile worlds," in *Proc. IEEE INFOCOM*, 2009, pp. 2106–2113.
[41] "Universal mobile telecommunications system (UMTS); radio frequency (RF) system scenarios (release 13)," 3GPP, Sophia Antipolis, France, Rep. TR 25.942, Jan. 2016.
[42] H. Zhao, J. Wang, F. Liu, Q. Wang, W. Zhang, and Q. Zheng, "Power-aware and performance-guaranteed virtual machine placement in the cloud," *IEEE Trans. Parallel Distrib. Syst.*, vol. 29, no. 6, pp. 1385–1400, Jun. 2018.
[43] Z. Ma, S. Shao, S. Guo, Z. Wang, F. Qi, and A. Xiong, "Container migration mechanism for load balancing in edge network under power Internet of Things," *IEEE Access*, vol. 8, pp. 118405–118416, 2020.
[44] A. Beloglazov and R. Buyya, "Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in cloud data centers," *Concurrency Comput. Pract. Exp.*, vol. 24, no. 13, pp. 1397–1420, 2012.
[45] R. Ford, M. Zhang, M. Mezzavilla, S. Dutta, S. Rangan, and M. Zorzi, "Achieving ultra-low latency in 5G millimeter wave cellular networks," *IEEE Commun. Mag.*, vol. 55, no. 3, pp. 196–203, Mar. 2017.
[46] R. Morabito, "Virtualization on Internet of Things edge devices with container technologies: A performance evaluation," *IEEE Access*, vol. 5, pp. 8835–8850, 2017.
[47] X. Xu, X. Liu, Z. Xu, F. Dai, X. Zhang, and L. Qi, "Trust-oriented IoT service placement for smart cities in edge computing," *IEEE Internet Things J.*, vol. 7, no. 5, pp. 4084–4091, May 2019.
[48] P. Mach and Z. Becvar, "Mobile edge computing: A survey on architecture and computation offloading," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 3, pp. 1628–1656, 3rd Quart., 2017.
[49] L. Atzori *et al.*, "Smart devices in the social loops: Criteria and algorithms for the creation of the social links," *Future Gener. Comput. Syst.*, vol. 97, pp. 327–339, Aug. 2019.

Olga Chukhno (Graduate Student Member, IEEE) received the B.Sc. degree in business informatics and the M.Sc. degree in fundamental informatics and information technologies from RUDN University, Moscow, Russia, in 2017 and 2019, respectively. She is currently pursuing the Ph.D. degree with the Mediterranea University of Reggio Calabria, Reggio Calabria, Italy, and Tampere University, Tampere, Finland.

She is an Early Stage Researcher within the H2020 MCSA ITN/EJD A-WEAR Project with the Mediterranea University of Reggio Calabria and Tampere University. Her current research interests include wireless communications, social networking, and edge computing.

Nadezhda Chukhno (Graduate Student Member, IEEE) received the B.Sc. degree in business informatics and the M.Sc. degree in fundamental informatics and information technologies from RUDN University, Moscow, Russia, in 2017 and 2019, respectively.

She is an Early Stage Researcher with A-WEAR and a Doctoral Researcher with the Mediterranea University of Reggio Calabria, Reggio Calabria, Italy, and Jaume I University, Castellón de la Plana, Spain. Her current research activity mainly focuses on wireless communications, 5G/6G networks, D2D, and wearable technologies.

Giuseppe Araniti (Senior Member, IEEE) received the Laurea degree and the Ph.D. degree in electronic engineering from the University Mediterranea of Reggio Calabria, Reggio Calabria, Italy, in 2000 and 2004, respectively.

He is currently an Associate Professor of Telecommunications with the University Mediterranea of Reggio Calabria. His major area of research is on 5G/6G networks and it includes personal communications, enhanced wireless and satellite systems, traffic and radio resource management, multicast and broadcast services, device-to-device, and machine-type communications (M2M/MTC).

Claudia Campolo (Senior Member, IEEE) received the master's degree and the Ph.D. degree in telecommunications engineering from the University Mediterranea of Reggio Calabria, Reggio Calabria, Italy, in 2007 and 2011, respectively.

She is an Associate Professor of Telecommunications with the University Mediterranea of Reggio Calabria. Her main research interests are in the field of vehicular networking, 5G, and future Internet architectures.

Antonio Iera (Senior Member, IEEE) received the Laurea degree in computer engineering from the University of Calabria, Arcavacata, Italy, in 1991, the master's degree in IT from the CEFRIEL/Politecnico di Milano, Milan, Italy, in 1992, and the Ph.D. degree from the University of Calabria in 1996.

From 1997 to 2019, he was with the University Mediterranea of Reggio Calabria, Reggio Calabria, Italy. He is currently a Full Professor of Telecommunications with the University of Calabria. His research interests include next-generation mobile and wireless systems, and the Internet of Things.

Antonella Molinaro (Senior Member, IEEE) received the Laurea degree in computer engineering from the University of Calabria, Arcavacata, Italy, in 1991, the master's degree in information technology from the CEFRIEL/Politecnico di Milano, Milan, Italy, in 1992, and the Ph.D. degree in multimedia technologies and communications systems from the University of Calabria in 1996.

She is currently an Associate Professor of Telecommunications with the University Mediterranea of Reggio Calabria, Reggio Calabria, Italy, and with CentraleSupélec, Université Paris-Saclay, Gif-sur-Yvette, France. Her research activity mainly focuses on wireless and mobile networking, vehicular networks, and future Internet.