Mobility Aware and Dynamic Migration of MEC Services for the Internet of Vehicles

Ibtissam Labriji, Francesca Meneghello, *Graduate Student Member, IEEE*, Davide Cecchinato, Stefania Sesia, Eric Perraud, Emilio Calvanese Strinati, and Michele Rossi, *Senior Member, IEEE*

***Abstract*—Vehicles are becoming connected entities, and with the advent of online gaming, on demand streaming and assisted driving services, are expected to turn into data hubs with abundant computing needs. In this article, we show the value of estimating vehicular mobility as 5G users move across radio cells, and of using such estimates in combination with an online algorithm that assesses when and where the computing services (virtual machines, VM) that are run on the mobile edge nodes are to be migrated to ensure service continuity at the vehicles. This problem is tackled via a Lyapunov-based approach, which is here solved in closed form, leading to a low-complexity and distributed algorithm, whose performance is numerically assessed in a real-life scenario, featuring thousands of vehicles and densely deployed 5G base stations. Our numerical results demonstrate a reduction of more than 50% in the energy expenditure with respect to previous strategies (full migration). Also, our scheme self-adapts to meet any given *risk* target, which is posed as an optimization constraint and represents the probability that the computing service is interrupted during a handover. Through it, we can effectively control the trade-off between seamless computation and energy consumption when migrating VMs.**

***Index Terms*—5G, Internet of Vehicles (IoV), multi-access edge computing (MEC), virtual machine (VM), service migration, mobility estimation, Lyapunov optimization, recurrent neural network, convolutional neural network, Markov chain.**

Manuscript received June 1, 2020; revised October 28, 2020 and December 22, 2020; accepted December 24, 2020. Date of publication January 19, 2021; date of current version March 11, 2021. This work has received funding from the Italian Ministry of Education, University and Research (MIUR) through the PRIN project no. 2017NS9FEY entitled "Realtime Control of 5G Wireless Networks: Taming the Complexity of Future Transmission and Computation Challenges", and has also been supported, in part, by MIUR through the initiative "Departments of Excellence" (Law 232/2016). A preliminary version of the mobility prediction algorithm in Section V was presented at WiMob2020 in the paper "Mobility Prediction via Sequential Learning for 5G Mobile Networks." The associate editor coordinating the review of this article and approving it for publication was H. Zhang. *(Corresponding author: Francesca Meneghello.)*

Ibtissam Labriji and Eric Perraud are with the DEA-L Department, Renault Software Labs, 06560 Valbonne, France (e-mail: ibtissam.labriji@renault.com; eric.perraud@renault.com).

Francesca Meneghello and Davide Cecchinato are with the Department of Information Engineering, University of Padova, 35131 Padua, Italy (e-mail: meneghello@dei.unipd.it; cecchin4@dei.unipd.it).

Stefania Sesia was with the DEA-L Department, Renault Software Labs, 06560 Valbonne, France. She is now with the Marketing and Sale Department, u-blox AG, 8800 Thalwil, Switzerland (e-mail: stefania.sesia@u-blox.com).

Emilio Calvanese Strinati is with the CEA, LETI (Minatec Campus), 38054 Grenoble, France (e-mail: emilio.calvanese-strinati@cea.fr).

Michele Rossi is with the Department of Information Engineering, University of Padova, 35131 Padua, Italy, and also with the Department of Mathematics "Tullio-Levi Civita," University of Padova, 35131 Padua, Italy (e-mail: rossi@dei.unipd.it).

Digital Object Identifier 10.1109/TNSM.2021.3052808

I. Introduction

PIONEERED by the Google car, the Internet of vehicles (IoV) is becoming the new fabric where vehicles will interact with one another, while communicating with the network infrastructure and with the edge/cloud, where services are provided. Examples of such services are assisted driving, traffic management (i.e., prevention of accidents), infotainment, etc. The fifth-generation (5G) of mobile networks is expected to provide broadband access in dense areas, delivering high communication capacity and ultra-low latency [1], [2]. In addition, with 5G, connectivity, computing and caching will converge into the so called (multi-user) multi-access edge computing (MEC), which will offer computing power right at the network edge.

In this article, we are concerned with location and mobility aware computing services [3], [4] which will play a key role in the IoV. We tackle the problem of ensuring *continuity of computing tasks* as mobile users (vehicles) move within a city-wide 5G network, and we do so by migrating their services as the users change their point of attachment to the network. Past work dealt with this by exploiting *reactive* approaches [4]–[8], where the Virtual Machines (VM) hosting the services are migrated *after* the users have moved to the new radio cell. In the present work, we advocate the use of mobility predictions to *proactively* allocate computing resources, by migrating the VM prior to the execution of the handover. If carefully orchestrated and implemented, the proposed proactive allocation strategies lead to relevant energy savings, by keeping (computation) service discontinuity to a minimum. Our work is enabled by recent advances in high precision *network-based positioning* within 5G systems [9]–[11]. Specifically, the use of two dimensional antenna arrays, with the introduction of the massive multiple-input multiple-output (MIMO) physical layer technology, allows terminal positioning via angle-of-arrival (AoA) measurements without the burden of performing triangulation [12].

Article Contribution: we ensure IoV MEC service continuity when the mobility is constrained to the physical road topology. For each vehicle, the best virtual machine (VM) migration strategy is computed at each handover event, proactively reallocating its MEC resources to ensure MEC *service continuity*, while concurrently *minimizing the energy consumption* at network level.

The proposed VM migration strategy operates independently at each eNB (MEH). The key aspects are:

- *Tracking Correlated User's Trajectories:* the vehicular mobility process is highly correlated as it is constrained to the underlying road link topology. Such correlation is utilized to devise proactive MEC allocation strategies prior to the execution of handovers.
- *Multi-Modal Mobility Estimation:* the designed mobility estimation algorithm produces high-quality estimates by combining mobility information inside and across evolved node B (eNB) sites. **Inside eNBs:** position information is inferred from radio link measurements and is processed using convolutional and recurrent neural networks (NNs). **Across eNBs:** the sequence of previously visited eNBs is statistically characterized using Markov chains (MCs). An original (online) learning architecture is put forward, combining NN and MC predictors and returning soft-predictions in the form of a probability distribution over the set of nearby eNBs (MEHs) towards which each vehicle may handover.
- *Joint Mobility Estimation and MEC Service Migration:* mobility predictions are used within a decentralized and online Lyapunov-based optimization technique to decide *where* and *how many* computation resources (VMs) are to be (proactively) migrated for each vehicle. The optimal Lyapunov-based VM migration strategy is obtained in closed form, leading to a low complexity and online technique. The algorithm outputs the number of VM replicas (*how many*) and the eNBs (MEHs) *where* these VMs are to be migrated prior to the handover execution.

The proposed solution is numerically assessed in a real-life vehicular mobility scenario, using the "simulation of urban mobility" (SUMO) [13] to emulate thousands of vehicles moving within the city of Cologne, and simulating a realistic 5G network with densely deployed eNBs (and co-located MEHs). Our numerical results reveal that the proposed mobility-aware VM migration strategy achieves a low energy expenditure, i.e., close to the lower bound where each VM is replicated to a single next MEH, by granting a service discontinuity risk close to that of the *full replication* strategy, i.e., where VMs are replicated to all the available next MEHs. We also assess the impact of the predictor's accuracy on the VM migration decisions, finding a large performance improvement of NN + MC with respect to prior techniques.

The rest of this article is organized as follows. The related work is reviewed in Section II. The system model is detailed in Section III. In Section IV, the optimization problem's formulation is presented along with the online closed form solution. The NN + MC mobility prediction algorithm is put forward in Section V, whereas the numerical results are presented in Section VI. Concluding remarks are drawn in Section VII.

II. Related Work

In wireless mobile networks, mobile users that change their serving eNB experience service connectivity continuity thanks to handover procedures. The same principles apply to MEC services, where not only the connectivity, but also the MEC offloading service process is migrated from the current serving MEH to the MEH in the next connectivity domain. There are two main techniques to handle MEC offloading service continuity in mobile networks during handovers: (i) The VM that hosts the ongoing computation process is kept on the MEH where the MEC service was initiated, and where resources were allocated in the first place [14]. In this case, standard communication handover procedures ensure that the UE remains connected to the serving MEH while handing off from one eNB/MEH to a new one [15]. Such approach was proven to be inefficient in the case of frequent handovers due to the latency of and the energy drained at the backhaul links. (ii) A second option consists of *migrating* the VM to a new MEH in proximity of the UE, leading to a reduction in the communication delay and in the UE uplink transmission power. In this article, we focus on this second option, although the combination of the two approaches is also possible, see, e.g., [16]. We leave this extension for future work.

In the literature, several migration mechanisms have been proposed to determine *when* and *where* to migrate VMs. The majority of the approaches rely on the *reactive* migration of VMs, i.e., applied as UEs exit the coverage area of the eNB/MEH where the VM is currently hosted [16]–[18]. Although this can be effective in handling delay tolerant services, it is inapt to manage delay-sensitive tasks, for which *proactive* approaches represent a better solution [5]. Proactive VM migration decisions can be made based on the UE-MEH distance as the sole metric [6], on mobility predictions [7] and on joint mobility prediction and MEH availability estimates [4]. The currently proposed proactive solutions for VM migration rely on probabilistic mobility models and, in the case of poor mobility estimates, the VM migration process can experience a major performance degradation. To take this into account, the authors of [8] propose to proactively replicate the VM into different neighboring MEHs, restoring service in a new MEH as the user moves from the current eNB to the next one. However, the problem formulation in [8] entails a computationally expensive integer linear programming optimization task that is difficult to deploy in practice. Moreover, the mobility prediction model relies on historical data about user's movements between the radio cells, disregarding the online evolution of the user's position *inside* the serving cell. In our work, we propose to jointly (i) improve the reliability of mobility predictions, by also including in the estimates the user's movement inside the radio cells, and, (ii) dynamically adapt the VM replication strategy (its overhead, intended as the number of VM copies) based on mobility prediction estimates. This makes it possible to dynamically evaluate the risk of service discontinuity given an identified set of target MEHs, by concurrently minimizing the cost incurred in the VM migration. In addition, we tackle the complexity issue via a lightweight optimization framework that is solved in closed form, entailing low complexity calculations.

We advocate that accurate and online mobility models are key to the design of effective MEC migration strategies during handovers. In the literature, such models are often obtained exploiting MCs and support vector machines, using the sequence of *previously visited eNBs* [19]–[21] as input. Although this approach may be effective in simple scenarios, the sole historical information about the sequence of serving eNBs does not provide accurate prediction estimates as the number of physical trajectories increases, and this is especially true in urban environments served by densely deployed

networks. In these cases, more accurate models would be desirable. In fact, as we quantify in the numerical results of Section VI, better mobility estimates lead to the replication of fewer VM instances for the same reliability target, with a subsequent reduction in the amount of resources (memory, computation and energy) that are employed to support the MEC services during handovers.

III. System Model

We consider a vehicular network consisting of a set $\mathcal{S}$ of eNBs, each co-located with a MEH, and a set $\mathcal{V}$ of vehicles which move following the morphological road constraints. Vehicles send offload requests for intensive computation services to the MEHs. The computation services can experiences a large number of handovers due to vehicle mobility and to the radio coverage range of eNBs.

To handle the computation offloading requests and guarantee service continuity, each MEH includes: (i) several VMs to execute the users' services, (ii) a MEC platform, (iii) a mobility prediction unit, (iv) a *VM migration control unit* (including admission control), and (v) a *virtualization infrastructure*. Each vehicle is associated with a specific VM in the serving MEH, which handles the required services and that can be migrated to new MEHs tracking the vehicle's movements. For that, as a vehicle approaches the edges of a cell, the radio network interface service (RNIS) running on the MEC platform informs entities (iii) and (iv) that the vehicle is about to perform the handover to a new radio cell. At this point, the unit running the mobility prediction algorithm starts tracking the vehicle's position to estimate the next point of attachment. The information is then used by the VM migration control unit to decide whether to migrate the user's VM to a new location, and in that case, in which MEHs the replication is to be performed. The VM migration control unit has an overview of the complete mobile edge system and serves as the orchestrator. Finally, the virtualization infrastructure provides computation, storage, and network resources to instantiate VMs, execute offloaded services, and manage their migration process by booking memory and computing capacities at MEHs identified by the VM migration control unit. The MEH architecture detailed above is compliant with [22], [23].

While our system model can be considered for both live and static VM migration, in our investigation we focus on live VM migration and *stateful* application processes. VM migration is ensured between MEHs through backhaul links whose data rate is assumed to be sufficiently high so that the time required for the VM migration can be neglected. This assumption can be relaxed, but in the following of our article, we restrict the analysis to this case.

Our system model is time-dependent and operates in an online fashion. Time is slotted, with time slots of fixed duration τ. At each time slot, a cost function representing the *energy expenditure* is minimized subject to a risk constraint, seeking a good balance between energy minimization and service continuity.

IV. Problem Formulation

Hereafter, we present the proposed online optimization algorithm to dynamically control the number of VM replication instances that are migrated prior to a handover event and also the MEHs destinations of such migrations. Specifically, we address the issue of VM migration due to *off-board* service computation requested by vehicles that are exiting the serving eNBs (radio) coverage area and that are at a distance smaller than δ meters from the edge of the cells, i.e., vehicles that are about to perform a handover to a new eNB. The investigated problem is formulated for the system model described in Section III under the assumption that the considered computing processes cannot be processed on-board on the vehicle, i.e., they must be offloaded to a MEH in the network for their execution. This assumption is especially realistic for those applications that require data that is not stored in the vehicle or software not installed on it. The extension of the present analysis to the local computation of tasks is left for future work.

The set of vehicles that are about to leave the cell, i.e., that are a distance smaller than δ meters from each edge, is here referred to as $\mathcal{A}(t)$. We identify as $r \in \mathcal{A}(t)$ (any) one of such vehicles. When the VM migration control unit decides to migrate the VM handling services for vehicle *r*, $M_r(t)$ parallel migrations are performed, replicating the VM that is serving vehicle *r* towards the $M_r(t)$ distinct MEHs selected among the available next eNBs. This will allow restoring the service computation in a new MEH in case the handover is performed. Hence, $M_r(t)$ is an integer indicating the number of selected MEH sites where the VM serving vehicle *r* will be migrated to in time slot *t*. Note that, for any vehicle at time *t*, there are N_{MEH} available eNBs towards which it can potentially hand over to and, each of them has an associated MEH. N_{MEH} does not depend on the position of the user inside the cell: it is fixed and specific for each eNB, as it reflects the underlying mobility process, i.e., the road topology and the way eNBs are deployed. So, at time *t* there are N_{MEH} MEH candidates for VM migration and $M_r(t) \leq N_{\mathrm{MEH}}$ of these are selected. Specifically, $M_r(t)$ is chosen from the set $\mathcal{N}_{\mathrm{MEH}} = \{1, \ldots, N_{\mathrm{MEH}}\}$ that contains the possible numbers of VM replicas to be performed, $M_r(t) \in \mathcal{N}_{\mathrm{MEH}}$. Most importantly, using the mobility prediction framework of Section III, each vehicle in the considered area (i.e., less than δ meters apart from the cell's edge), obtains a mobility-prediction vector $\mathbf{p}_r = (p_{r,1}, \ldots, p_{r,N_{\mathrm{MEH}}})$ of size N_{MEH}, with $\sum_i p_{r,i} = 1$. The entries in the probability vector $\mathbf{p}_r$ are arranged in decreasing order, namely,

$$p_{r,1} > p_{r,2} > \cdots > p_{r,N_{\mathrm{MEH}}}. \tag{1}$$

Vector $\mathbf{p}_r$ provides a probabilistic estimate of the next visited eNB site when leaving the current serving one, by ranking the N_{MEH} candidate eNBs according to their probability of being visited. $p_{r,i}$ represents the probability that the vehicle will hand over to the eNB ranked as the *i*-th most probable among the neighboring ones. This information is obtained by learning (at runtime) the underlying vehicular mobility dynamics via the combined NN + MC mobility prediction model of Section III. When we select the $M_r(t)$ MEHs for VM replication, we pick eNB indices according to the order of the corresponding probabilities $p_{r,i}$, i.e., we select the most probable ($p_{r,1}$) first and add further eNBs until $M_r(t)$ sites are picked, in the order expressed by Eq. (1). This means that once $\mathbf{p}_r$ is obtained, the number $M_r(t)$ of eNBs that we select is all we need to provide a complete description of the VM migration process. In fact, we may alternatively use a state vector

$(b_{r,1},\ldots,b_{r,N_{\mathrm{MEH}}})$, where $b_{r,i}\in\{0,1\}$ is a binary number indicating whether eNB i is selected or not. However, this vector can be fully replaced (and obtained) by the information contained in vector $\mathbf{p}_r$ (providing the ranking among eNBs) and $M_r(t)$ (the number of eNBs that are picked).

We define the binary function $o_r(t)$, indicating whether the VM migration control unit decides to migrate or not the VM serving vehicle *r* that is leaving the cell

$$o_r(t)=\begin{cases}1, & \text{VM will be migrated}\\ 0, & \text{VM will not be migrated}\end{cases}. \tag{2}$$

If $o_r(t)=1$, the VM serving vehicle *r* will undergo the replication process to guarantee service continuity as the vehicle moves out of the eNB coverage area. If instead $o_r(t)=0$, the VM will not be migrated in any MEH and the handled service will be interrupted when the handover is performed. Being $\psi_r(t)$ the energy required to perform the VM migration towards a single eNB site, the energy consumed in the current MEH to replicate the VM to the $M_r(t)$ selected sites is

$$E_r(t)=M_r(t)\psi_r(t), \tag{3}$$

and it follows that the total energy consumed in time slot *t* for the VMs replications is

$$E(t)=\sum_{r\in\mathcal{A}(t)}o_r(t)E_r(t). \tag{4}$$

Let $N(t)=|\mathcal{A}(t)|$ be the number of vehicles that are about to leave the cell, i.e., the serving MEH, in time slot *t*. We define a control action vector for the VM replication process as follows,

$$\mathbf{\Omega}(t)=\Big(\mathbf{\Omega}_1(t),\ldots,\mathbf{\Omega}_{N(t)}(t)\Big), \tag{5}$$

where $\mathbf{\Omega}_r(t)$ is the control action taken in time slot *t* for vehicle $r\in\mathcal{A}(t)$,

$$\mathbf{\Omega}_r(t)=(o_r(t),\,M_r(t)). \tag{6}$$

The objective function that we want to minimize is the long-term average energy expenditure associated with the migration of VMs,

$$\lim_{T\to+\infty}\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}[E(t)]. \tag{7}$$

Note that only accounting for the energy cost, through Eq. (7), would lead to a solution where no VM replication is performed, as the energy is trivially minimized by setting $o_r(t)=0$, for all *r* and *t*. Therefore, for a proper formulation, we need to account for an additional cost encoding the fact that computing tasks may be discontinued during handovers. To this end, we define a *risk metric* $\zeta_r(t)$ for each vehicle *r*, which encodes the total probability that the associated VM is not correctly migrated, i.e., that is migrated to MEHs which will not be visited after the handover. It follows that $0\le\zeta_r(t)\le 1$ and for any vehicle *r* at time *t*, we define

$$\zeta_r(t)=1-o_r(t)\sum_{i=1}^{M_r(t)}p_{r,i}. \tag{8}$$

The rationale behind Eq. (8) is as follows. If the VM will not undergo the migration process, i.e., $o_r(t)=0$, the risk is $\zeta_r(t)=1$ as the vehicle *r* will suffer from service discontinuity. On the other hand, when the VM migration control unit decides to migrate the VM to all the available MEHs, i.e., to N_{MEH} eNBs, the risk is zero, as the vehicle will hand over with probability one to one of the available MEHs and the computing process will be resumed there. However, it may happen that the VM migration control unit establishes to replicate the VM in the first $M_r(t)<N_{\mathrm{MEH}}$ MEHs (first $M_r(t)$ probabilities from vector $\mathbf{p}_r$), but the vehicle actually hands over to any one of the $N_{\mathrm{MEH}}-M_r(t)$ remaining MEHs, where the VM was not replicated. This occurs with probability $\zeta_r(t)$ and corresponds to the case in which migration is performed to the wrong servers, i.e., after the handover, the computing task will not be resumed by the new serving MEH, as it was not migrated there. This risk shall be small and is here used as a constraint to counterbalance the minimization of the energy consumption in Eq. (7).

We define the average risk across all tasks in the current time slot as

$$\overline{\zeta}(t)=\frac{1}{N(t)}\sum_{r\in\mathcal{A}(t)}\zeta_r(t). \tag{9}$$

Informally, the proposed optimization framework is constrained on the fact that the average (long-term) risk, obtained by averaging Eq. (9) across multiple time slots, must be smaller than a pre-defined threshold $0<\xi\le 1$. ξ is a constant that we use to control the so-called *violation probability* for a processing task, i.e., the probability that the task computation will not be resumed after handing over to a new MEH.

To enforce this constraint, we define a virtual queue *Z*(*t*) with the following update equation

$$Z(t+1)=\max\big\{Z(t)+\overline{\zeta}(t)-\xi,0\big\}, \tag{10}$$

which is utilized to transform the long-term inequality constraint into a corresponding queue mean rate stability problem. Hence, we must ensure that the virtual queue *Z*(*t*) is mean rate stable, which is mathematically expressed as [24, Ch. 2]

$$\lim_{t\to+\infty}\frac{\mathbb{E}[Z(t)]}{t}=0. \tag{11}$$

The optimization problem in Eq. (7) is now changed to account for service priorities through an additional parameter $\chi_r(t)$ for each service class. For example, a service that is computationally (and energy) expensive will be discarded more often by the optimizer due to its high energy consumption. The priority parameter compensates for this: setting $\chi_r(t)=0$ means not to assign any priority to the associated VM, while setting $0<\chi_r(t)\le 1$ allows controlling the migration process in case of VMs with different migration costs. It follows that Eq. (7) should be modified by considering the weighted energy expenditure

$$F(t)=\sum_{r\in\mathcal{A}(t)}o_r(t)E_r(t)(1-\chi_r(t)), \tag{12}$$

instead of the pure energy expenditure *E*(*t*), see Eq. (4).

The optimization problem for VM migration is then formulated as follows

$$\lim_{T\to+\infty} \min_{\substack{\mathbf{\Omega}(t),\\ t\in\{0,\dots,T-1\}}} \frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}[F(t)]$$
$$\textbf{Subject to: } (a)\ \lim_{T\to\infty}\frac{\mathbb{E}[Z(T)]}{T}=0,$$
$$(b)\ o_r(t)\in\{0,1\},\quad \forall\, r,t,$$
$$(c)\ M_r(t)\in\mathcal{N}_{\mathrm{MEH}},\ \forall\, r,t, \tag{13}$$

which amounts to minimizing the overall weighted energy drained by the VM migration process (the objective function in the first line), while maintaining the average risk below constant ξ (queue stability constraint (*a*)).

1) Online Optimization Algorithm: To design a stable and online control algorithm, we use the *Lyapunov drift-plus-penalty* framework, see, e.g., [24]–[26]. This allows obtaining a dynamic decision making system, that controls the VM replication process in each time slot.

Given that *Z(t)* represents the virtual queue backlog process, the *Lyapunov function* is defined as

$$L(Z(t))=\frac{1}{2}Z(t)^2 \tag{14}$$

and the *one-step Lyapunov drift* is computed as

$$\Delta L(Z(t)) \stackrel{\mathrm{def}}{=} L(Z(t+1))-L(Z(t)). \tag{15}$$

The expression for the *Lyapunov drift-plus-penalty* is then obtained by combining the average weighted energy consumption in Eq. (12) and the average queue backlog in Eq. (15) through a parameter $V>0$ that weighs the importance of the two quantities in the minimization problem, i.e.,

$$\Delta(t)=\Delta L(Z(t))+VF(t). \tag{16}$$

Theorem 1 (Upper Bound on the Drift-Plus-Penalty): Eq. (16) can be upper bounded as

$$\Delta(t)\le\frac{1}{2}+Z(t)\big(\overline{\zeta}(t)-\xi\big)+VF(t). \tag{17}$$

Proof: See Appendix A. ∎

The *drift-plus-penalty* optimization algorithm observes *Z(t)* in every time slot *t* and chooses an action $\mathbf{\Omega}(t)$ that minimizes the right-hand-side of the inequality in Eq. (17). To do so, we use the concept of *opportunistically minimizing an expectation* [24, Section 1.8]. This is accomplished by greedily minimizing Eq. (16), neglecting the constant term 1/2. Hence, the online optimization problem is written as,

$$\min_{\mathbf{\Omega}(t)} \Bigg[Z(t)\big(\overline{\zeta}(t)-\xi\big) + V\sum_{r\in\mathcal{A}(t)} o_r(t)M_r(t)\psi_r(t)(1-\chi_r(t))\Bigg]$$
$$\textbf{Subject to: } (b)\ o_r(t)\in\{0,1\},\ \forall\, r,$$
$$(c)\ M_r(t)\in\mathcal{N}_{\mathrm{MEH}},\ \forall\, r. \tag{18}$$

In the following proposition, we show how the optimization problem in Eq. (18) can be solved in closed form, leading to an efficient implementation of the solver.

Proposition 1 [Closed Form Solution of Eq. (18)]: Defining function $g_r(M_r(t))$ as

$$g_r(M_r(t)) \stackrel{\mathrm{def}}{=} VM_r(t)\psi_r(t)(1-\chi_r(t))-\frac{Z(t)}{N(t)}\sum_{i=1}^{M_r(t)}p_{r,i}, \tag{19}$$

the optimum control action at time *t* for vehicle $r\in\mathcal{A}(t)$ is $\mathbf{\Omega}_r^*(t)=(o_r^*(t),M_r^*(t))$, where

$$M_r^*(t)=\operatorname*{argmin}_{M_r(t)\in\mathcal{N}_{\mathrm{MEH}}} g_r(M_r(t)), \tag{20}$$

and

$$o_r^*(t)=\begin{cases}1, & \text{if } g_r(M_r^*(t))<0\\ 0, & \text{otherwise}\end{cases}. \tag{21}$$

Proof: See Appendix B. ∎

A. Dealing With Computation Resource Limitation in the Neighboring MEHs

Exploiting the side information gathered from the network, the VM migration control unit at one eNB can avoid migrating the VM associated with vehicle *r* when no computation resources are left on its neighboring MEHs. This requires MEHs to periodically send their load status to their neighbors. With this information, the probability vectors $\mathbf{p}_r$ of all vehicles at time *t* can be modified to consider possible resource limitations. Let C_i be the number of additional VMs that can be instantiated on the neighboring MEH *i*. This MEH ranks the vehicles in the current cell according to the probability of migrating towards it, from highest to lowest. The first C_i migration requests (vehicles) are accepted, whereas the remaining ones are not admitted by masking their probability vector, i.e., by assigning a zero to their probability of migrating towards MEH *i*, i.e., $p_{r,i}\leftarrow 0$ if vehicle *r* is not admitted. This is a form of *admission control* which is implemented prior to computing Eq. (8) and running the optimization, to avoid a later rejection due to insufficient computation resources at the MEH.

V. Mobility Estimation

In this section, we detail the proposed mobility estimation approach. The algorithm returns a new estimate every time a new vehicle position is gathered, with an accuracy increasing with the number of samples. Estimation on mobility is given as the probability of performing the handover to each of the neighboring eNBs. This probability vector is then used within the proposed VM replication strategy to compute the risk of not guaranteeing service continuity in (8) (see Section VI). Note that the prediction of the next point of attachment and the optimization algorithm for VM migration are independently executed on the MEH by two different entities. At the time the VM migration control unit has to decide which VMs to replicate and where, the mobility prediction unit sends the most updated probability vector for each of the tracked users.

The location of a vehicle is collected through the *azimuth* (α) and *elevation* (β) angles that describe its relative position with respect to the serving eNB. The usage of two-dimensional antenna arrays, with the introduction of the massive MIMO technology, allows acquiring such AoA information from a single eNB, without the need for triangulation [12]. In the

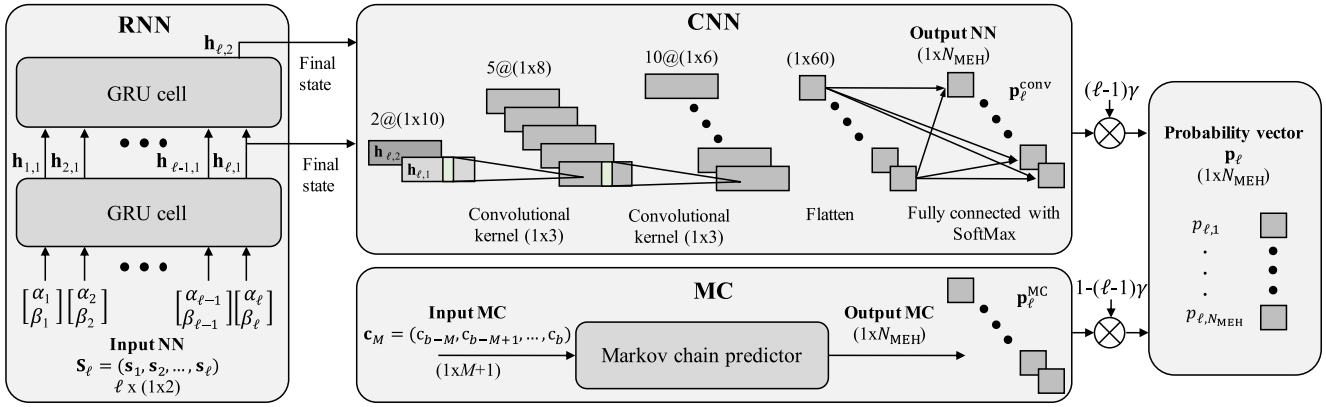


Fig. 1. Processing pipeline. The classification is performed in parallel by the NN block (GRU-based RNN + CNN), using as input the (α, β) angles describing the trajectories, and the MC block, exploiting the sequence of previous visited cells. The output vector of probabilities for each adjacent eNB ($\mathbf{p}_\ell$) is obtained as a combination of the two classification outputs.

following, we refer to a *trajectory* as the sequence of azimuth and elevation angles that describes the positions of the vehicle from the time it connects to a MEH ($t = 0$) to the instant in which it hands over to another MEH ($t = LT_s$), where L is the number of samples of the complete trajectory, collected with a sampling period of T_s seconds. Formally, an L-long trajectory is denoted by $\mathbf{S}_L \triangleq (\mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_L)$, with $\mathbf{s}_\ell = (\alpha_\ell, \beta_\ell)^T$.

Considering that the MEH to which the user will successfully hand over must be one among the N_{MEH} adjacent MEHs, we formulate a classification problem and, we design a supervised learning algorithm to carry it out. The classification is executed in parallel by two blocks: (i) a NN block, consisting of a recurrent neural network (RNN) (see Section V-A) and a convolutional neural network (CNN) (see Section V-B), that uses as input the user trajectory samples, and (ii) a final MC block (as proposed by [20]) that exploits the sequence of previously visited cells (see Section V-C). The classification results from (i) and (ii) are then combined to obtain the final estimates using a parameter that is learned as part of the training (see Section V-D). The blocks are shown in Fig. 1, while the training process is detailed in Section V-F.

To complete our approach, we propose a pre-processing algorithm to deal with possible imbalanced datasets (see Section V-E), which is key to also learn those trajectories for which there are fewer training examples.

The benefits of adopting an AoA and learning-based mobility predictor are two-fold. First, this approach allows preserving the user's privacy, as we estimate the positions of the vehicles on the MEHs without requiring them to send information such as position, speed, and direction. This also avoids the additional network traffic that would be caused by regular exchanges of such information between the vehicles and the MEHs. The second reason is that learning is key to automatically capture complex road topologies and mobility behaviors, which would hardly be represented via, e.g., first order models entailing straight motion and constant velocity.

A. GRU-Based RNN for Mobility Feature Extraction

RNN is the most used learning tool when dealing with sequential data, as RNNs are able to track temporal correlation among input samples and allow handling inputs with different numbers of samples [27]. An RNN is composed of one or more *memory cells* whose number is indicated by the number of layers, L_{rec}. The temporal correlation between subsequent input samples, from $\mathbf{s}_1$ to $\mathbf{s}_\ell$, is recorded in the values of the neurons constituting the cell *internal state*, $\mathbf{h}_\ell$, where ℓ is the sample index. To that end, the input sequence $\mathbf{S}_\ell \triangleq (\mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_\ell)$, is fed one sample at a time into the first layer of the RNN. In case of multiple layers, the internal state of the first layer cell becomes the input of the second layer cell, and so on. Every time a new sample arrives, the internal states of the RNN cells are updated based on the values of the previous states and on the new sample. The update procedure is described by the *cell update function*, that defines the memory cell, whose parameters are adjusted to minimize a *loss function*, i.e., a user-defined distance between the expected and the actual network output. Dropout layers can be inserted between recurrent layers, providing network regularization by randomly zeroing some of the elements of the input, following a Bernoulli distribution. In our implementation, we use $L_{\text{rec}} = 2$ gated recurrent units (GRUs) [28] memory cells with 20 neurons per layer, and a dropout layer with a retain probability of 0.9. For each input sequence, we collect the internal state values of the two stacked RNN cells, each time a new trajectory sample is processed. Note that the RNN internal state is a *feature vector* capturing the most representative traits of the input time series. After applying layer normalization, this feature vector is used by the subsequent CNN network that acts as a classifier.

B. CNN for Next Serving eNB Prediction

A CNN consists of the cascade of layers (i.e., functions), where all or some of them use the *convolution operator*, i.e., small-size kernels (matrices of weights), which are convolved with the entire input data volume [27]. The result of such convolution is passed through a non-linearity, called *activation function*, to produce an output *activation map*. The number of kernel filters employed at one layer defines the *depth* of the layer output, i.e., the number of activation maps produced for a single input, each of which captures different aspects of the input. One interesting feature of CNNs is their *parameter sharing*: the kernel filters are applied to each position of the input, allowing one to capture specific input peculiarities, no matter their specific location. As for RNNs, dropout layers can be interposed between convolutional layers. We use a one dimensional CNN, where the kernels are vectors, with $L_{\text{conv}} = 2$ convolutional layers with `ReLu` activation

functions, and two dropout layers with a retain probability of 0.9. Batch normalization is used after each convolutional layer to reduce the training time. The CNN input corresponds to the two (stacked) final internal states extracted from the GRU-based RNN. The output of the last convolutional layer is then passed through a fully connected layer with `SoftMax` activation function that produces an N_{MEH}-dimensional vector, $\mathbf{p}_\ell^{\mathrm{conv}} \triangleq (p_{\ell,1}^{\mathrm{conv}}, p_{\ell,2}^{\mathrm{conv}}, \ldots, p_{\ell,N_{\mathrm{MEH}}}^{\mathrm{conv}})$, containing the probability that the user will hand over to each of the N_{MEH} neighboring cells after leaving the current one.

C. MC for Next Serving eNB Prediction

MC is here used to capture the correlation in the sequence of MEHs that are visited by a user. Indicating with $c_{b+1,i}$ the i-th neighbouring MEH to the current (serving) MEH, $i \in \{1, \ldots, N_{\mathrm{MEH}}\}$, and $\mathbf{c}_M = (c_{b-M}, c_{b-M+1}, \ldots, c_b)$ the sequence of $M+1$ eNBs visited up to time $t = \ell T_s$, each element of the N_{MEH}-dimensional vector $\mathbf{p}_\ell^{\mathrm{MC}} \triangleq (p_{\ell,1}^{\mathrm{MC}}, p_{\ell,2}^{\mathrm{MC}}, \ldots, p_{\ell,N_{\mathrm{MEH}}}^{\mathrm{MC}})$ is computed as follows:

$$p_{\ell,i}^{\mathrm{MC}} = \frac{\sum_{\mathrm{dataset}} \mathtt{Number}\left(c_{b+1,i}, \mathbf{c}_M\right)}{\sum_{\mathrm{dataset}} \mathtt{Number}(\mathbf{c}_M)}, \tag{22}$$

where `Number` indicates the times the sequence inside the parenthesis appears in the training dataset. The value of *M*, i.e., the MC order, used in this work results from a performance assessment reported in Section VI-B.

D. Combination of NN and MC Predictors

The final N_{MEH}-dimensional probability vector $\mathbf{p}_\ell \triangleq (p_{\ell,1}, p_{\ell,2}, \ldots, p_{\ell,N_{\mathrm{MEH}}})$ is obtained by combining the NN and MC estimations using a parameter $\gamma \in \{0, 1\}$ as follows:

$$k = \min\{(\ell - 1)\gamma, 1\}, \tag{23}$$

$$\mathbf{p}_\ell = k\mathbf{p}_\ell^{\mathrm{conv}} + (1 - k)\mathbf{p}_\ell^{\mathrm{MC}}. \tag{24}$$

This combination strategy allows the network to weight differently the NN and MC estimations as the number of gathered trajectory samples ℓ changes. Note that γ is fixed for each NN-MC predictor and is automatically learned during training. Equation (13) ensures that the combination parameter *k* remains inside the range [0, 1].

E. Dealing With Imbalanced Datasets

In an urban scenario, the number of examples available for each of the possible trajectories inside a MEH's coverage area usually differs. Using such an imbalanced dataset for training would lead to a neural network with poor mobility prediction capabilities for those users that follow under represented trajectories. To cope with this, we propose a method to artificially generate new trajectories, so that they resemble the ones belonging to the under represented classes. This oversampling process is only performed on the training dataset, whose trajectories are labeled with the indication of the next MEH towards which the user will be moving after the handover. Based on this label, the input data is split into different *groups*. The groups with cardinality less than 70% that of the most populated one undergo an oversampling process, while the others remain unchanged. The trajectories belonging to each of these groups are further split into a number of *classes* reflecting a quantization on the trajectory space, for the trajectories that lead to the same final MEH (same group). To do this, we first obtain fixed-length trajectory representations (*codes*), as standard clustering algorithms do not deal with variable length inputs. A simple and common coding method in machine learning is *padding*, i.e., to concatenate each input vector with as many zeros as needed to reach a common (fixed) length. However, in our case, this is not the best choice as the inputs have a specific physical meaning (city roads). Instead, we interpolate the trajectories with a degree 3 polynomial function, and use the interpolated versions, i.e., the codes, as inputs for the (unsupervised) "hierarchical density-based spatial clustering of applications with noise" (HDBSCAN) algorithm [29]. For each trajectory class, we then produce artificial trajectories picking at random trajectories from the class, and adding white Gaussian noise with standard deviation $\sigma = 5 \times 10^{-4}$ rad to each sample. The new trajectories are assigned to the proper label to be used in the training process.

F. Training Process

Next, we detail how the building blocks composing our solution are combined and jointly trained, as summarized in Algorithm 1. First, the training trajectories are used to generate artificial examples to balance the training dataset. Hence, from each (*L*-long) trajectory we extract all the possible $L - 1$ sub-trajectories of size greater than two, i.e., $([\mathbf{s}_1, \mathbf{s}_2], [\mathbf{s}_1, \mathbf{s}_2, \mathbf{s}_3], \ldots [\mathbf{s}_1, \boldsymbol{s}_2, \ldots, \mathbf{s}_L])$. Each of these sub-trajectories is forwarded through the NN block to obtain the probability vector $\mathbf{p}_\ell^{\mathrm{conv}}$ for the next MEH. The sequence of previous visited cells $\mathbf{c}_M$ is used to compute the probability vector $\mathbf{p}_\ell^{\mathrm{MC}}$ that is then combined with $\mathbf{p}_\ell^{\mathrm{conv}}$ to obtain $\mathbf{p}_\ell$. $\mathbf{p}_\ell$ is then compared with the ground truth to determine the prediction loss. (The ground truth is a N_{MEH}-dimensional "one hot vector" with the element corresponding to the actual next MEH being equal to 1, and the other ones to 0). The loss is computed by multiplying the negative log-likelihood by an exponential penalty term that depends on the sub-trajectory length (ℓ) and on the total length (*L*) of the trajectory as follows:

$$\mathrm{penalty}(\ell) = 2^{\ell/L} - 1. \tag{25}$$

The backpropagation of sub-trajectory prediction losses allows the network to adjust its parameters to correctly estimate the next MEH even when the complete *L*-long trajectory is not yet available. Through (25), higher penalties are assigned as the number of collected samples, ℓ, approaches *L*. This forces the network to become more accurate as the number of samples increases. Every 20 epochs, the performance of the framework is assessed using the validation data. The RNN + CNN weights and the combination parameter (γ) are extracted and saved if they outperform the previously saved ones in terms of validation accuracy. Otherwise, an ε-greedy approach is adopted, see, e.g., [30]: with probability $\varepsilon = 0.3$ the training continues from the weights computed in the last iteration, while with probability $1 - \varepsilon$ it restarts from the previously saved network parameters. Every 20 epochs the artificial training examples used to balance the dataset are re-computed and the ε-greedy approach prevents the network from learning from artificial trajectories that are not representative of the real ones. These combined actions allow increasing the generalization capacity of the framework as during each training round (20 epochs) the dataset is re-populated maintaining all the real training data, while adding *different* artificial examples.

Algorithm 1 Deep Learning Framework: Training Process

- `num_epochs` $\leftarrow 100$, $\varepsilon \leftarrow 0.3$, `max_val_accuracy` $\leftarrow 0$;
- **for** $j \leftarrow 1$ to (`num_epochs`/20) **do**
  - add training trajectory examples – Section V-E;
  - extract training sub-trajectories;
  - **for** $jj \leftarrow 1$ to 20 **do**
    - split the set of sub-trajectories in batches with `batch_size` $= 64$ examples each;
    - **for all** batches **do**
      - forward propagate the input – Section V-F;
      - compute the error using penalty in Eq. (25);
      - backpropagate the error through the network;
  - assess prediction performance with validation set;
  - `val_accuracy` $\leftarrow$ accuracy on the validation set;
  - **if** `val_accuracy>max_val_accuracy` **then**
    - save network parameters (weights, biases, γ);
    - `max_val_accuracy` $\leftarrow$ `val_accuracy`;
  - **else if** `random`(0, 1)$> \varepsilon$ **then**
    - restore previously saved network parameters;

The network parameters specified in the previous sections have been selected after a prior hyperparameter selection phase.

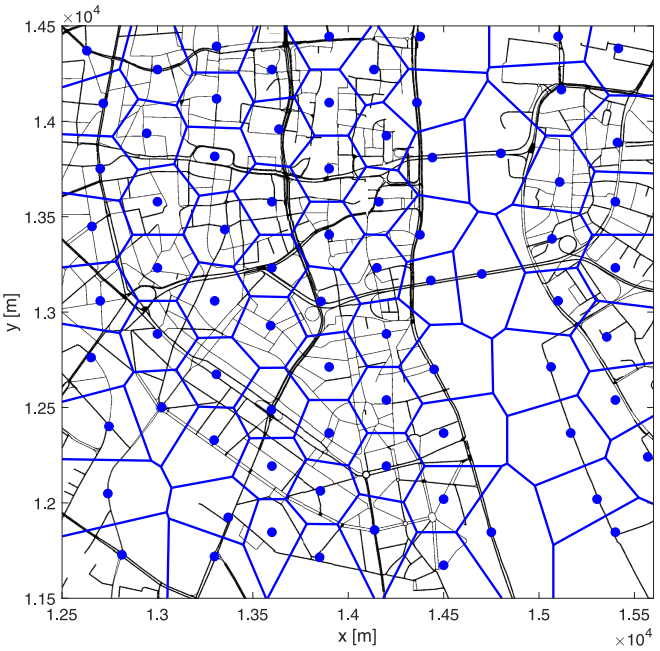


Fig. 2. Deployment of the eNBs over the selected area of the city of Cologne. The eNBs are the centroids of the Voronoi cells. The x and y coordinates are the same used in the TAPAS Cologne simulation scenario.

VI. Numerical Results

We consider an urban 5G scenario in the center of the city of Cologne ($3,000 \times 3,000$ square meters), with 5G enabled vehicles that use the 5G network to offload computation tasks [31]. The mobility is emulated with SUMO [13], an open-source traffic simulator that allows generating the movement of mobile users around a predefined city road map. Specifically, we use the "TAPAS Cologne" simulation scenario, which mimics the vehicular traffic within the city for a whole day on the basis of travelling habits of the city dwellers.[1] The vehicle density changes during the day presenting two peaks, in the time intervals between 8-9 a.m. and 4-6 p.m., reflecting the mobility due to getting to and leaving work.

The eNBs endowed with MEH functionalities are deployed according to the road topology: eNBs are placed at the street crosses or along the roads, ensuring an inter-distance among nodes of around 400 m. The mobility area is covered with convex polygonal cells, each with a 5G eNB and a co-located MEH in its center, see Fig. 2. In detail, the considered city region is partitioned into $|\mathcal{S}|$ Voronoi cells, using the positions of the eNBs as generating points (the centroids), and considering the Euclidean distance to shape the cell edges (the faces of the Voronoi cells). The average number of vehicles within each eNB coverage ranges from 10 to 50 depending on the position of the eNB and on the hour of the day, with velocity ranging between 0 and 45 km/h. The proposed simulation scenario is in line with field trials, e.g., [32], [33].

Twenty-four-hour long SUMO mobility traces with $T_s = 0.25$ s granularity were collected for each of the 77 eNBs in the deployment: 70% of such data was used to train and validate the mobility prediction algorithm, while the remaining 30% served to assess the performance of the proposed VM replication strategy. For such assessment, we considered vehicles $r \in \mathcal{A}(t)$ approaching the edge of the serving eNB coverage area, i.e., that are about to hand over to a new eNB. For the following results, on average this occurs when a user is less than $\delta = 40$ meters apart from the radio cell's edge.

Channel Model for User-eNB Association: For the wireless link between the vehicles and the eNB, we use the mm-wave channel model of [34], describing urban NLoS scenarios. At every point in time, each vehicle connects with the eNB providing the best communication conditions, measured in terms of path loss (PL),

$$\mathrm{PL}(d) = \rho_1 + \rho_2 10\log_{10}(d) + \eta \quad [\mathrm{dB}], \tag{26}$$

$$\eta \sim \mathcal{N}(0, \sigma^2) \quad [\mathrm{dB}], \tag{27}$$

where d is the distance between the transmitter and the receiver and $\mathcal{N}(0, \sigma^2)$ represents a Gaussian r.v. with zero mean and variance σ^2. Considering $d = 200$ m, the channel parameters are set to $\rho_1 = 46.61$, $\rho_2 = 3.63$ and $\sigma = 9.83$ dB, see [34].

For the handover user association, a hysteresis mechanism, with parameter ε_P, is considered (see, e.g., [35]). Specifically, a handover is performed to a new eNB only if the power received from this new eNB exceeds that received from the serving one by an amount greater than $\varepsilon_P = 2$ dB.

Vehicle Location Information Acquisition: In the numerical evaluation, we emulated the AoA positioning service of 5G eNBs by processing the $x - y$ traces extracted from the SUMO simulator, obtaining the angular information along elevation and azimuth. White Gaussian noise was added to the extracted data to account for estimation errors as large as five meters ($\pm 3\sigma_w$), i.e., $\sigma_w = 5/3$ m.

Energy Expenditure Model for VMs Migration: We use the model in [36], where the authors exploit CloudSim, a widely used simulator for cloud computing infrastructures and services, to estimate the energy drained in a MEH to perform

[1] https://sumo.dlr.de/docs/Data/Scenarios/TAPASCologne.html

TABLE I
SERVICE TYPES AND SIMULATION PARAMETERS CONSIDERED IN THE NUMERICAL EVALUATION OF VM MIGRATION STRATEGIES

service type	$\psi_r(t)$
OC	15.91 MJ
GP	79.89 MJ
GPU	221.73 MJ

parameter	value
τ	3 s
ξ	10^{-2}
V	8×10^{-12}

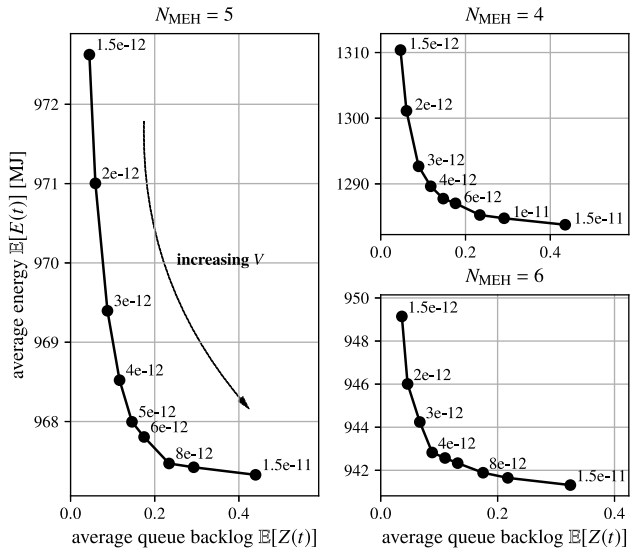


Fig. 3. Energy consumption as a function of the queue backlog by varying V as a free parameter. Three plots are shown, by varying the number of candidate eNBs (MEHs) for the handover, $N_{\text{MEH}} \in \{4, 5, 6\}$.

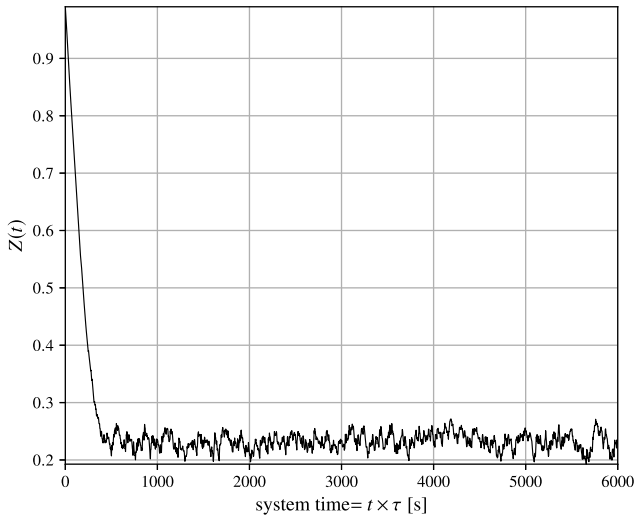


Fig. 4. The evolution of the virtual queue $Z(t)$ depends on the number of times $\overline{\zeta}(t)$ exceeds $\xi = 10^{-2}$. Using our approach, Eq. (11) holds and the queue stabilizes for increasing t.

VM migration. Following their model, $\psi_r(t)$ is computed as a function of the VM memory usage $W_{[\text{MB}]}$ (in megabytes) as

$$\psi_r(t) = 3600 \times \Big(0.512\, W_{[\text{MB}]} + 20.165\Big) \quad [\text{J}], \qquad (28)$$

where $W_{[\text{MB}]}$ depends on service type. In our numerical evaluation, we focus on three different service types (see [37] for their memory use): optimized calculus (OC), general purpose (GP) and hardware accelerated services requiring a GPU (GPU). The energy drained at the MEH to perform the VM migration is computed using Eq. (28) and is reported in Table I.

A. Performance of the Proposed VM Migration Algorithm

1) System Parameters Selection: Table I summarizes the numerical parameters. The duration of a time slot, τ, indicates the time between two consecutive executions of the optimization algorithm for the selection of the proper MEHs, towards which the VM is to be replicated. Note that, the user before moving to a new cell is served by a local VM which runs inside the current (serving) cell, our optimization process only handles when and where the current VM is migrated. Hence, the value of τ does not affect the end-to-end latency of a URLLC bearer, but rather determines how many migration requests are collected before triggering the migration mechanism. The rationale is to proactively trigger the migration before a user moves to a new cell, so that the VM will be already instantiated as the user gets there. The parameter τ is selected considering the (typical) maximum speed of the vehicles $v_{\max}$ for the users approaching the cell's edge, and a distance parameter $\delta = 40$ m. Specifically, the inequality $\tau \leq \delta / v_{\max}$ is to be satisfied. This allows a vehicle requesting a service and entering the border region of a cell (distance smaller than δ with respect to its edge) at the beginning of a time slot τ not to suffer from service discontinuity during the handover. In our setup, being $v_{\max} = 45$ km/h, we fix $\tau = 3$ s.

The value of the risk parameter ξ is connected with the quality of service that we would like to guarantee to the users requiring computation offloading. We set $\xi = 10^{-2}$, i.e., we require the long-term average risk of our VM replication strategy not to exceed 10^{-2} (the VM should be migrated to the wrong set of eNBs once every 100 requests, on average).

In this first part of performance evaluation, we focus on a single service type, considering VMs for optimized calculus that require $\psi = 15.91$ MJ to be migrated, according to [36], [37]. The priority parameter $\chi_r(t)$ is set to zero for every vehicle r and time slot t (see Section VI-A3 for the evaluation in case of multiple service types). We also consider that MEHs have enough computing capacity to accept all the VM migrations from neighboring MEHs, i.e., no blockage can occur (resources limitation is considered in Section VI-A4).

In Fig. 3, we show the long-term average energy consumption over $1,000$ time slots and the average size of the virtual queue $Z(t)$ as a function of the parameter V. Each point in the graph corresponds to a different value of V, which is varied as a free parameter in the interval $V \in [1.5 \times 10^{-12}, 1.5 \times 10^{-11}]$. This evaluation is carried out across several eNBs, for different values of N_{MEH} (three examples, for $N_{\text{MEH}} = 4, 5, 6$, are shown in Fig. 3). In general, $V = 8 \times 10^{-12}$ consistently represents a suitable choice across all the cases, as it provides a good trade-off between energy minimization and queue size. $V = 8 \times 10^{-12}$ may be referred to as the "optimal" point according to the elbow method [38].

The behavior of the queue backlog for $V = 8 \times 10^{-12}$ is plotted in Fig. 4 for $N_{\text{MEH}} = 5$. As shown in this plot, after

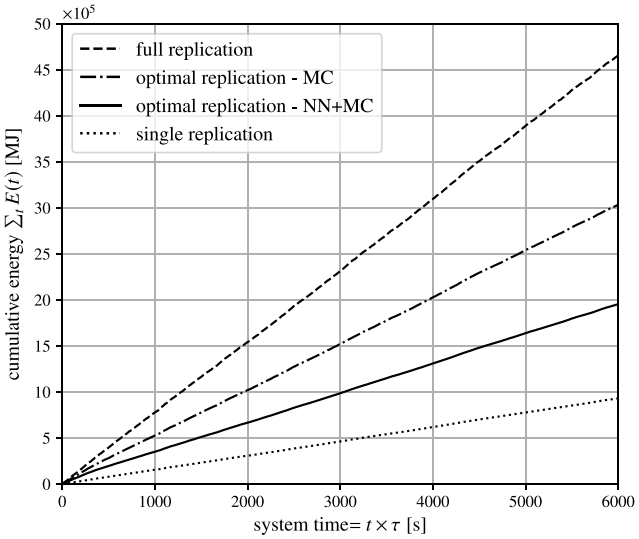


Fig. 5. Total energy consumption *vs* time. The proposed approach considerably reduces the energy consumption as compared to the full replication scheme, by optimizing the number of VM replications for each submitted task.

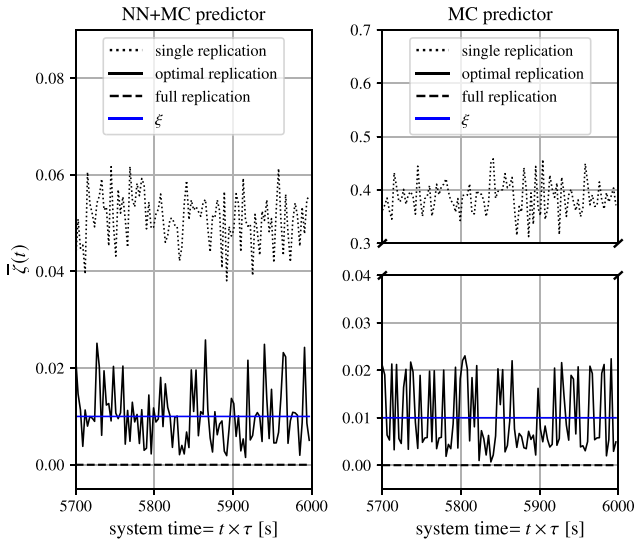


Fig. 6. Temporal evolution of the average risk $\overline{\zeta}$. Our approach is constrained to meet the risk constraint ξ in an average sense (long-term).

a transient phase, the queue $Z(t)$ stabilizes, as imposed by the mean rate stability constraint of Eq. (11).

2) Performance Analysis and Comparison Against Prior Art: To assess the performance of the proposed algorithm, referred to here as *optimal replication*, we compare it with the following commonly adopted strategies,

- *Full Replication:* VM serving vehicle *r* is migrated to all the N_{MEH} neighboring eNBs. This approach has been adopted in, e.g., [5].
- *Single Replication:* VM serving vehicle *r* is migrated only to the MEH that is the most likely to be visited after the handover, i.e., the MEH associated with $p_{r,1}$ in Eq. (1). This approach was considered in, e.g., [7].

Moreover, to assess the impact of the mobility prediction accuracy (i.e., the quality of probability estimates $\mathbf{p}_r$) on the energy and risk performance, we have run simulations using (i) our proposed NN + MC predictor and (ii) a predictor solely based on MCs (see Section V-C), as presented in, e.g., [20].

In Fig. 5, we show the time evolution of the total energy drained by the considered migration techniques for $N_{\mathrm{MEH}} = 5$. For a clearer visualization of the differences among them, the cumulative energy is plotted rather than the instantaneous one. As expected, the single replication strategy leads to the lowest energy consumption, as it replicates the VM to a single MEH at all times (for every vehicle). On the contrary, the full replication strategy leads to the highest, as it replicates the VM to *all* the available neighboring N_{MEH} sites. From this graph, we see that our approach (optimal replication) effectively reduces (by more than 50%) the total energy drained as compared to the full replication strategy. This is possible thanks to the careful selection of the number and destination of the replicated VM instances for each computation task. Moreover, we underline the impact of using the combined NN + MC mobility prediction algorithm against the simplest one based on MCs. Note that NN + MC provides more accurate probability estimates $\mathbf{p}_r$ which, in turn, entail the selection of fewer next eNBs for the same risk target. This translates into lower energy consumption. As expected, for both full and single replication strategies, the energy expenditure is independent of the mobility estimation technique. In fact, with the former, mobility estimates are not used, whereas with the latter the energy expenditure amounts to copying the VM to a single next MEH, and the energy cost of this is the same across all the candidate sites.

The risk $\overline{\zeta}(t)$ is shown in Fig. 6 for the NN + MC and MC mobility predictors, with $N_{\mathrm{MEH}} = 5$. For a better visualization, the plot interval is restricted to the last 300 slots towards the end of the simulation. As expected, the risk is always zero in the full replication case. For the single replication strategy, it instead highly depends on the performance of the mobility predictor. For this scheme, a perfect prediction, e.g., carried out by a genie, would lead to a zero risk, and in general the better the prediction is, the lower the risk. From the numerical results, single replication achieves an average long-term risk of 0.463 when using the MC predictor, which is decreased to 0.057 using NN + MC, due to the highest accuracy of the latter. Nevertheless, the average risk of the single replication strategy is still at least five times higher than that of the proposed approach, for which the average risk stabilizes to $\xi = 10^{-2}$.

In more detail, for the proposed technique, the risk is subjected to a constraint that must be met regardless of the accuracy of the predictor. For this reason, the average risk of the proposed optimal technique in the left and right plots of Fig. 6 behave similarly, remaining both bounded around the same target risk $\xi = 10^{-2}$. This is a good property of the proposed algorithm, whose behavior is adjustable as a function of the user-defined average risk constraint ξ, which will be always met by design. Also, as seen from Fig. 5, although the risk performance is unaffected, a better mobility estimator translates into lower energy consumption, as thanks to it fewer replications are performed for the same risk target ξ.

As a last performance metric, we compute the successful migration probability: we say that the VM migration succeeds when the user hands over to an eNB (MEH) where its VM was proactively migrated into. In this case, the user resumes

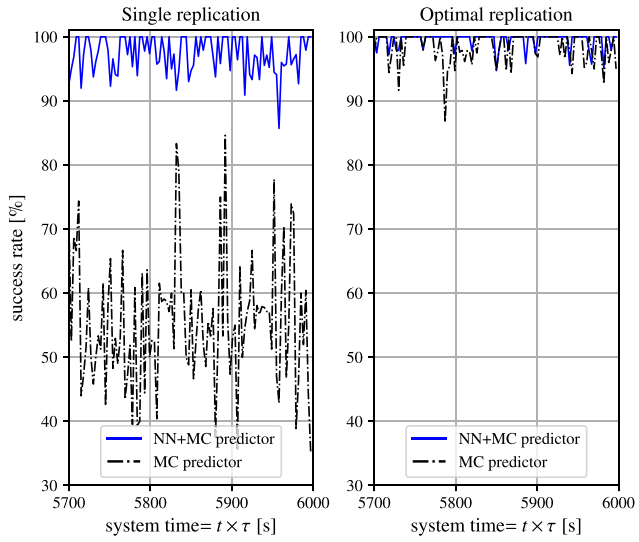


Fig. 7. Success rate in VM migrations. Although the success rate also depends on the mobility predictor's accuracy, the proposed replication approach overtakes the problem by increasing the number of VM replicas, by always meeting the average risk constraint ξ. The metric is obtained only considering the offloaded tasks, i.e., the ones for which $o_r(t) = 1$.

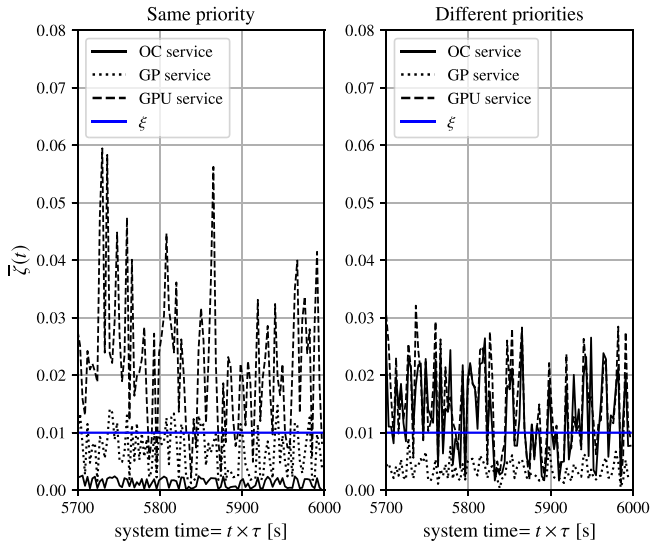


Fig. 8. Temporal evolution of the risk ζ averaged for the different computing services.

TABLE II
AVERAGE NUMBER OF REPLICAS FOR VMS HANDLING DIFFERENT SERVICES

service type	same priority		different priorities	
	$\chi_r(t)$	$\overline{M_r}(t)$	$\chi_r(t)$	$\overline{M_r}(t)$
OC	0	3.2	0	2.7
GP	0	2.4	0.4	2.0
GPU	0	1.7	0.8	2.1

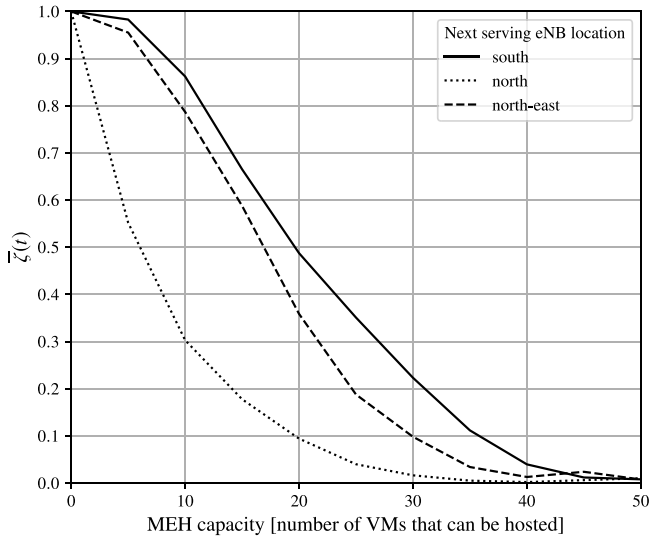


Fig. 9. Evolution of the average risk as a function of the MEHs capacity. The vehicles are grouped based on the eNBs where they are going.

the computation on the new MEH without loss of continuity. In Fig. 7, we compare the successful migration rate for the single and the proposed migration strategies. The success rate depends on the accuracy of the mobility predictor used to estimate the probability vector over the next serving eNBs. The performance gap between MC and NN + MC is evident for the single replication case. Instead, the proposed technique is less sensitive to the adopted mobility estimator, as it compensates for the less accurate prediction accuracy of MC by replicating the VM to more MEHs. As expected, full replication (not shown in the figure) always achieves a success rate of 100%, as it sends the VM to all the neighboring MEH sites.

3) Performance Analysis for Different Service Types: Here, we evaluate the performance of the proposed algorithm in case vehicles request different service types and, in turn, their VMs drain a different amount of energy for their migration. We consider the three services introduced before, i.e., OC, GP and GPU, comparing two different scenarios. In the first one, we set $\chi_r(t) = 0, \ \forall r, t$. Table II ("same priority") details the average number of optimal (see Proposition 1) VMs migrations for this case, by grouping the vehicles requesting the same service. Fig. 8, left plot, shows the evolution of the average risk for each service type. In this first scenario, the VMs with the lowest migration costs are served more often, whereas higher cost VMs are penalized. In the second scenario, we set $\chi_r(t) = 0.8$ and $\chi_r(t) = 0.4$ for the GPU and GP services respectively, while set the priority parameter for OC to zero. This allows counterbalancing the higher energy cost of GPU services, re-equalizing the risk, see the right plot in Fig. 8 and Table II ("different priorities").

4) Impact of Limited Computation Resources at the MEHs: In the previous sections, we have considered that the computation capacity at the MEHs is always sufficient to accept all the incoming computation requests. Instead, in case the MEHs have limited computation capacity, e.g., due to a system overload or to their poor dimensioning, some VM migrations may be blocked. In Fig. 9, we show the average risk as a function of the MEH capacity, i.e., the number of VMs that can be simultaneously hosted at a MEH. In this evaluation, the energy cost and the priority are respectively set to 15.91 MJ (OC) and zero for all the vehicles. Each curve in the plot refers to a different eNB in the neighboring set. For this plot, the MEH capacity

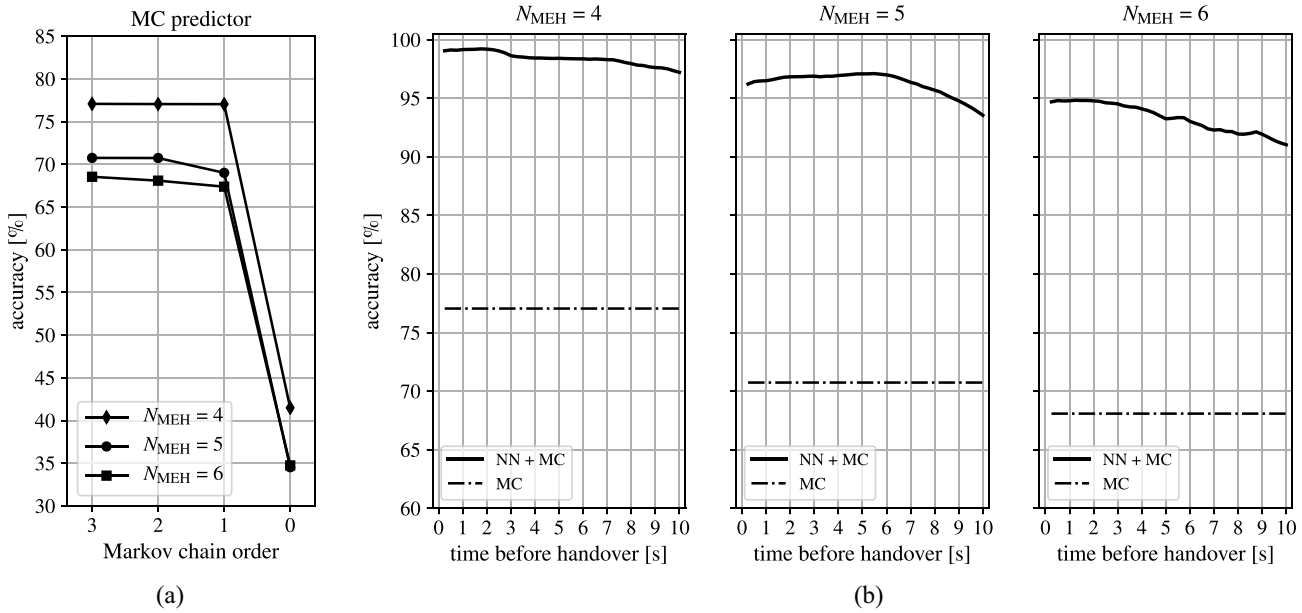


Fig. 10. (a) Next MEH estimation accuracy using different order MCs. (b) Prediction accuracy as a function of the time left before handing over to the next eNB. MC indicates the Markov chain predictor from the literature, whereas NN + MC is the proposed approach.

was varied from 0 to 50 in steps of 5, the risk was averaged over 2,000 time slots by grouping vehicles according to their next eNB. As the capacity approaches zero, the blockage probability and, in turn, the risk, both approach one. Instead, an increasing capacity leads to a decreasing risk, which eventually stabilizes around the target, which is $\xi = 0.01$. A proper dimensioning of the MEH capacity depends on the local amount of requests per unit time, which is in turn linked with the road topology and with the amount of vehicular traffic in the considered area.

B. Performance of the Proposed Mobility Prediction Algorithm

We independently trained the proposed mobility prediction approach for each of the 77 eNBs (MEHs) in the deployment. The twenty-four hour long mobility traces were labeled according to the next serving eNB identifier and split as follows: 60% for training, 10% for validation, and 30% for the final test. Note that, although the architecture is the same for all the eNBs, the NN weights and the combination parameter γ are independently adjusted during the training processes, and are in turn eNB specific.

The best order for the MC predictor is assessed by evaluating the next cell's prediction accuracy obtained using MC only, for different values of M. As can be seen from Fig. 10(a) the best choice is $M = 2$, as using higher MC orders does not lead to any noticeable increase in the accuracy. Note that if the number of previously visited eNBs, m, at time t is smaller than the MC order M, the prediction is performed by additionally training an MC with order $m < M$.

In Fig. 10(b), we show the accuracy obtained by averaging the results achieved by the 77 NN + MC classifiers, grouped based on the number of MEHs in their handover set. For comparison, we also report the accuracy achieved by the MC classifier. We do not report the cases when the handover set contains fewer than four eNBs because they rarely occurred in our simulations, and hence we could not collect enough data for a reliable training and assessment in such cases. As expected, the NN + MC estimation technique outperforms the state of the art MC classifiers, proving the effectiveness of collecting users' positions inside the serving eNB to estimate the next serving eNB (MEH), rather than only relying on the sequence of previously visited eNBs (MEHs). In all the cases, the average accuracy was found to be greater than 94%, starting from 4 s before the handover and remaining above 91% until 10 s before it.

VII. Concluding Remarks

In this work, we have presented a new proactive technique for the online migration of computation services (VMs) for vehicular 5G networks. Our approach combines a mobility prediction algorithm with an online optimization solver. The mobility estimates are expressed as a probability vector over the neighboring radio cells, i.e., the next vehicle's point of attachment. The mobility prediction algorithm, which combines neural networks and Markov chains, is independently trained and exploited at each radio cell and the output probability vectors are utilized as the input of a Lyapunov based online optimizer. The resulting approach, exploiting online decision making and mobility estimation, allows striking a good balance between the number of VMs that are replicated (representing an energy cost for the system) and the risk of losing service continuity. Numerical results demonstrate the effectiveness of the proposed solution in comparison with two benchmark (proactive) migration strategies from the literature. Our technique effectively reduces, by more than 50%, the total energy drained as compared to replicating the VM to all the neighboring radio cells, by keeping the risk of losing continuity of computation services as vehicles hand over across radio cells at a (user-defined) low value.

Future work includes the extension of the present approach to the cases where (i) computing servers are subject to hardware constraints (e.g., a limited number of CPUs cycles and memory), (ii) computation tasks can be executed both on-board and off-board.

APPENDIX A
PROOF OF THEOREM 1

Proof: Applying inequality $\max(x,0)^2 \leq x^2,\ \forall x \in \mathbb{R}$, to the expression of the queue evolution in Eq. (10), we obtain

$$Z(t+1)^2 \leq Z(t)^2 + \big(\overline{\zeta}(t) - \xi\big)^2 + 2Z(t)\big(\overline{\zeta}(t) - \xi\big), \quad (29)$$

from which it follows that

$$Z(t+1)^2 - Z(t)^2 \leq \big(\overline{\zeta}(t) - \xi\big)^2 + 2Z(t)\big(\overline{\zeta}(t) - \xi\big). \quad (30)$$

Since, by construction, $0 \leq \overline{\zeta}(t) \leq 1$ (see Eq. (8) and Eq. (9)) and $0 < \xi \leq 1$ (by definition), we have $(\overline{\zeta}(t) - \xi)^2 \leq 1$ and

$$Z(t+1)^2 - Z(t)^2 \leq 1 + 2Z(t)\big(\overline{\zeta}(t) - \xi\big). \quad (31)$$

It follows that

$$\begin{aligned}\Delta(t) &= \Delta L(Z(t)) + VF(t)\\ &= \frac{1}{2}\Big[Z(t+1)^2 - Z(t)^2\Big] + VF(t)\\ &\leq \frac{1}{2} + Z(t)\big(\overline{\zeta}(t) - \xi\big) + VF(t). \end{aligned} \quad (32)$$

■

APPENDIX B
PROOF OF PROPOSITION 1

Proof: The minimization problem in Eq. (18) can be simplified by observing that term $Z(t)\xi$ at time *t* does not depend on the decision variables $\mathbf{\Omega}(t)$. Henceforth, Eq. (18) can be equivalently rewritten as

$$\min_{\mathbf{\Omega}(t)} \left[V \sum_{r \in \mathcal{A}(t)} o_r(t) M_r(t) \psi_r(t)(1 - \chi_r(t)) + Z(t)\overline{\zeta}(t) \right]. \quad (33)$$

Using the equations for the risk Eq. (8) and Eq. (9), we obtain

$$\begin{aligned}\min_{\mathbf{\Omega}(t)} \Bigg[& V \sum_{r \in \mathcal{A}(t)} o_r(t) M_r(t) \psi_r(t)(1 - \chi_r(t))\\ & - \frac{Z(t)}{N(t)} \sum_{r \in \mathcal{A}(t)} \left(o_r(t) \sum_{i=1}^{M_r(t)} p_{r,i} \right) \Bigg], \end{aligned} \quad (34)$$

where the terms not depending on the optimization variables $\mathbf{\Omega}(t)$ are omitted. Rearranging the previous equation, we get

$$\begin{aligned}\min_{\mathbf{\Omega}(t)} & \sum_{r \in \mathcal{A}(t)} o_r(t)\\ & \times \left[V M_r(t) \psi_r(t)(1 - \chi_r(t)) - \frac{Z(t)}{N(t)} \sum_{i=1}^{M_r(t)} p_{r,i} \right]. \end{aligned} \quad (35)$$

It follows that Eq. (35) naturally decomposes across variable *r*, so minimizing the sum corresponds to minimizing each term in isolation. Due to this, the optimal control action $\mathbf{\Omega}_r^*(t) = (o_r^*(t), M_r^*(t))$ is obtained from Eq. (35) for each $r \in \mathcal{A}(t)$ as follows.

First, we find the optimal number of VM replications, $M_r^*(t) \in \mathcal{N}_{\text{MEH}}$, to be performed if the VM migration control unit decides to migrate the VM serving vehicle *r*. Let $g_r(M_r(t))$ be

$$g_r(M_r(t)) \stackrel{\text{def}}{=} V M_r(t) \psi_r(t)(1 - \chi_r(t)) - \frac{Z(t)}{N(t)} \sum_{i=1}^{M_r(t)} p_{r,i}, \quad (36)$$

the optimum choice for $M_r(t)$ is

$$M_r^*(t) = \operatorname*{argmin}_{M_r(t) \in \mathcal{N}_{\text{MEH}}} g_r(M_r(t)). \quad (37)$$

Now, Eq. (35) can be re-expressed as,

$$\sum_{r \in \mathcal{A}(t)} \min_{o_r(t)} o_r(t) g_r(M_r^*(t)), \quad (38)$$

given that $g_r(M_r^*(t)) \in \mathbb{R}$ and $o_r(t) \in \{0,1\}$ the minimum in Eq. (38) is achieved setting

$$o_r^*(t) = \begin{cases} 1, & \text{if } g_r(M_r^*(t)) < 0 \\ 0, & \text{otherwise} \end{cases}. \quad (39)$$

■

ACKNOWLEDGMENT

The views and opinions expressed in this work are those of the authors and do not necessarily reflect those of the funding institutions.

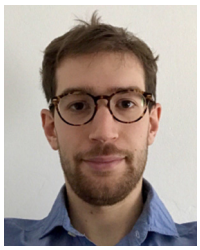
REFERENCES

[1] A. Osseiran *et al.*, "Scenarios for 5G mobile and wireless communications: The vision of the METIS project," *IEEE Commun. Mag.*, vol. 52, no. 5, pp. 26–35, May 2014.
[2] A. Ghosh, A. Maeder, M. Baker, and D. Chandramouli, "5G evolution: A view on 5G cellular technology beyond 3GPP Release 15," *IEEE Access*, vol. 7, pp. 127639–127651, 2019.
[3] M. Koivisto, A. Hakkarainen, M. Costa, P. Kela, K. Leppanen, and M. Valkama, "High-efficiency device positioning and location-aware communications in dense 5G networks," *IEEE Commun. Mag.*, vol. 55, no. 8, pp. 188–195, Aug. 2017.
[4] J. Plachy, Z. Becvar, and E. C. Strinati, "Dynamic resource allocation exploiting mobility prediction in mobile edge computing," in *Proc. IEEE 27th Annu. Int. Symp. Pers. Indoor Mobile Radio Commun. (PIMRC)*, Valencia, Spain, Sep. 2016, pp. 1–6.
[5] P. A. Frangoudis and A. Ksentini, "Service migration versus service replication in multi-access edge computing," in *Proc. 14th Int. Wireless Commun. Mobile Comput. Conf. (IWCMC)*, Limassol, Cyprus, Jun. 2018, pp. 124–129.
[6] S. Wang, R. Urgaonkar, M. Zafer, T. He, K. Chan, and K. K. Leung, "Dynamic service migration in mobile edge-clouds," in *Proc. IEEE IFIP Netw. Conf.*, Toulouse, France, Sep. 2015, pp. 1–9.
[7] F. Zhang, G. Liu, B. Zhao, X. Fu, and R. Yahyapour, "Reducing the network overhead of user mobility-induced virtual machine migration in mobile edge computing," *Softw. Pract. Exp.*, vol. 49, no. 4, pp. 673–693, 2018.
[8] I. Farris, T. Taleb, M. Bagaa, and H. Flick, "Optimizing service replication for mobile delay-sensitive applications in 5G edge network," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Paris, France, May 2017, pp. 1–6.
[9] S. E. Elayoubi *et al.*, "5G innovations for new business opportunities," in *Proc. Mobile World Congr.*, Barcelone, Spain, Feb. 2017, pp. 1–18.
[10] *Feasibility Study on New Services and Markets Technology Enablers (Release 14), V14.1.0*, IEEE Standard 22.891, Sep. 2016.
[11] *Study on Positioning Use Cases (Release 16), V16.1.0*, 3GPP Standard 22.872, Sep. 2018.

[12] E. Björnson, L. Sanguinetti, H. Wymeersch, J. Hoydis, and T. L. Marzetta, "Massive MIMO is a reality—What is next?: Five promising research directions for antenna arrays," *Digit. Signal Process.*, vol. 94, pp. 3–20, Nov. 2019.
[13] D. Krajzewicz, J. Erdmann, M. Behrisch, and L. Bieker, "Recent development and applications of SUMO—Simulation of urban mobility," *Int. J. Adv. Syst. Meas.*, vol. 5, nos. 3–4, pp. 128–138, 2012.
[14] V. Di Valerio and F. Lo Presti, "Optimal virtual machines allocation in mobile femto-cloud computing: An MDP approach," in *Proc. IEEE Wireless Commun. Netw. Conf. Workshops (WCNCW)*, Istanbul, Turkey, Apr. 2014, pp. 7–11.
[15] *Handover Procedures, V12.0.0*, 3GPP Standard TS 23.009, Sep. 2014.
[16] S. Wang, R. Urgaonkar, M. Zafer, T. He, K. Chan, and K. K. Leung, "Dynamic service migration in mobile edge computing based on Markov decision process," *IEEE/ACM Trans. Netw.*, vol. 27, no. 3, pp. 1272–1288, Jun. 2019.
[17] A. Aissioui, A. Ksentini, A. M. Gueroui, and T. Taleb, "On enabling 5G automotive systems using follow me edge-cloud concept," *IEEE Trans. Veh. Technol.*, vol. 67, no. 6, pp. 5302–5316, Jun. 2018.
[18] 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.
[19] S. Michaelis, N. Piatkowski, and K. Morik, "Predicting next network cell IDs for moving users with discriminative and generative models," in *Proc. Mobile Data Challenge Nokia Workshop Conjunction Int. Conf. Pervasive Comput.*, Newcastle, U.K., Jun. 2012, pp. 1–5.
[20] A. Hadachi, O. Batrashev, A. Lind, G. Singer, and E. Vainikko, "Cell phone subscribers mobility prediction using enhanced Markov chain algorithm," in *Proc. IEEE Intell. Veh. Symp.*, Dearborn, MI, USA, Jun. 2014, pp. 1049–1054.
[21] A. A. Hasbollah, S. H. Ariffin, N. E. Ghazali, K. M. Yusuf, and H. Morino, "Handover algorithm based VLP using mobility prediction database for vehicular network," *Int. J. Elect. Comput. Eng.*, vol. 8, no. 4, p. 2477, 2018.
[22] *Mobile Edge Computing (MEC); Framework and Reference Architecture, V1.1.1*, ETSI Standard MEC 003, 2016.
[23] *Multi-Access Edge Computing (MEC); V2X Information Service API, V2.1.1*, ETSI Standard MEC 030, Apr. 2020.
[24] M. J. Neely, *Stochastic Network Optimization With Application to Communication and Queueing Systems*. San Rafael, CA, USA: Morgan & Claypool, 2010.
[25] L. Georgiadis, M. J. Neely, and L. Tassiulas, "Resource allocation and cross-layer control in wireless networks," *Found. Trends Netw.*, vol. 1, no. 1, pp. 1–144, 2006.
[26] L. Tassiulas and A. Ephremides, "Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks," *IEEE Trans. Autom. Control*, vol. 37, no. 12, pp. 1936–1948, Dec. 1992.
[27] I. J. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016.
[28] K. Cho *et al.*, "Learning phrase representations using RNN encoder–decoder for statistical machine translation," in *Proc. Conf. Empirical Methods Nat. Lang. Process. (EMNLP)*, Doha, Qatar, Oct. 2014, pp. 1724–1734.
[29] M. F. Rahman, W. Liu, S. B. Suhaim, S. Thirumuruganathan, N. Zhang, and G. Das, "Density based clustering over location based services," in *Proc. 33rd IEEE Int. Conf. Data Eng. (ICDE)*, San Diego, CA, USA, Apr. 2017, pp. 461–469.
[30] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press, 2011.
[31] S. Redana *et al. 5G PPP Architecture Working Group—View on 5G Architecture, Version 3.0*. Accessed: Jan. 2021. [Online]. Available: http://doi.org/10.5281/zenodo.3265031
[32] J. Kim *et al.*, "Field trial of millimeter-wave-based MHN system for vehicular communications," in *Proc. 12th Eur. Conf. Antennas Propag. (EuCAP)*, London, U.K., Jan. 2018, pp. 1–5.
[33] E. C. Strinati *et al.*, "5GCHAMPION—Disruptive 5G technologies for roll-out in 2018," *ETRI J.*, vol. 40, no. 1, pp. 10–25, 2018.
[34] J. Ko *et al.*, "Millimeter-wave channel measurements and analysis for statistical spatial channel model in in-building and urban environments at 28 GHz," *IEEE Trans. Wireless Commun.*, vol. 16, no. 9, pp. 5853–5868, Sep. 2017.
[35] Z. Becvar and P. Mach, "Adaptive hysteresis margin for handover in femtocell networks," in *Proc. 6th Int. Conf. Wireless Mobile Commun.*, Valencia, Spain, Sep. 2010, pp. 256–261.
[36] M. Zakarya, L. Gillam, A. A. Khan, and I. U. Rahman, "PerficientCloudSim: A tool to simulate large-scale computation in heterogeneous clouds." *J. Supercomput.*, to be published.
[37] Microsoft. *Sizes for Virtual Machines in Azure*. Accessed: Jan. 2021. [Online]. Available: https://docs.microsoft.com/en-us/azure/virtual-machines/sizes
[38] D. J. ketchen and C. L. Shook, "The application of cluster analysis in strategic management: an analysis and critique," *Strategic Manag. J.*, vol. 17, no. 6, pp. 441–458, 1996.

Ibtissam Labriji received the Engineering degree in telecommunication from the National Institute of Posts and Telecommunications of Rabat, Morocco, the M.Sc. degree in advanced wireless communications from Centrale Supélec, Gif-sur-Yvette, France, in 2018. She is currently pursuing the Ph.D. degree with Renault Software Labs, Toulouse, France, in collaboration with the CEA-LETI laboratory of Grenoble, France. Her research interests include edge computing, 5G systems, stochastic optimization, and machine learning applied to vehicular communication systems.

Francesca Meneghello (Graduate Student Member, IEEE) received the B.Sc. degree in information engineering and the M.Sc. degree in telecommunication engineering from the University of Padova, Italy, in 2016 and 2018, respectively, where she is currently pursuing the Ph.D. degree with the Department of Information Engineering. Her current research interests include deep-learning architectures and signal processing with application to remote radio frequency sensing and wireless networks. She was a recipient of the Best Student Paper Award at WUWNet 2016, the Best Student Presentation Award at the IEEE Italy Section SSIE 2019, and received an honorary mention in the 2019 IEEE ComSoc Student Competition.



Davide Cecchinato received the B.Sc. degree (Hons.) in information engineering and the M.S. degree (Hons.) in telecommunication engineering from the university of Padova, Italy, in 2015 and 2017, respectively, where he is currently pursuing the Ph.D degree with the SIGNET Research Group. His research interests are on edge computing and adaptive systems for Internet of Things networks equipped with energy harvesting mechanisms.

Stefania Sesia received the Diploma degree in telecommunication from the Politecnico di Torino, Italy, the M.Sc. degree from the Eurecom Institute, France, in 2001, and the Ph.D. degree in electrical engineering from the Ecole Nationale Supérieure des Télécommunications (Telecom ParisTech, ENST) and the Eurecom Institute, France, in 2005. Her Ph.D. research activity was carried out jointly between Eurecom Institute and Motorola Labs, France. After the Ph.D., she joined the telecommunication industry where she held several technical lead positions (Philips, Ericsson, and Intel) on physical layer algorithms and new concept development, as well as 3GPP standardization activities. In 2015, she has been appointed as a Principal Engineer. In 2017, she joined the automotive industry with Renault Software Labs (Sophia Antipolis, France), where she had a role of Expert on New Communication Technologies and she was responsible for cross-system and SW architecture of features related to the use of vehicular communication for assisted/autonomous vehicles. She has recently joined u-blox, where she is a Senior Director, the Global Head of Application Marketing for the automotive business. She has authored of several IEEE conference and journal papers, and coauthored of book chapters as well as more than 50 patents. She has authored and a Lead Editor of *LTE-the UMTS Long Term Evolution: From Theory to Practice* (Wiley & Sons), with about 5 000 citations. She was an Associate Editor of the IEEE INTERNET OF THINGS JOURNAL, a Guest Editor of the IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS "Ultra-Reliable Low-Latency Communications in Wireless Networks", and a Guest Editor of a special edition of *EURASIP Journal on Wireless Communications and Networking* on 3GPP LTE and LTE Advanced.

Eric Perraud received the master's degree in telecommunication from the École nationale supérieure des télécommunications de Bretagne (specialty microwaves). He carried out his Ph.D. with the ONERA (French Aerospace Research Center) where he worked on new networking techniques of optical fibre sensors. He worked on an Earth Observation Satellite with Aerospatiale, then he moved to Freescale (formerly, Motorola Semi-conductors), initially with the Automotive Division, then with the Wireless Division. Later, he joined Motorola Mobile, as a Chief Architect. At Motorola, he has been in charge of the Hardware/Software partitioning of the 3G, HSPA and 4G platforms; he was also responsible for the performance optimization (to reduce the MIPS footprint and the current drain). He has been the System Architecture Manager, when Motorola shipped its first LTE modem (the second on the market) and redefined its modem software baseline. He joined Intel as a System Architect, leading the development of the first 4G Video Telephony solution at Intel and the first platform on the market enabling the differentiated QoS between the different bearers. Three years ago, he joined Renault around V2X projects and he has been working Research and Development programs about 5G MEC-computing for new automotive services.

Emilio Calvanese Strinati received the Ph.D. degree in engineering science from Telecom Paris, France, in 2005. He worked with Motorola Labs from 2002 to 2006. In 2006, he joints CEA/LETI as a Research Engineer. In 2007, he becomes a Ph.D. supervisor. From 2011 to 2016, he was the Smart Devices & Telecommunications European Collaborative Strategic Programs Director. From December 2016 to January 2020, was the Smart Devices & Telecommunications Scientific and Innovation Director. In February 2018, he directed the first 5G mobile millimeter waves demonstration in realistic operational environments at the 2018 winter Olympic Games, 5G technologies. Since 2018, he has been holding the French Research Director Habilitation. Since February 2020, he has been directs activities at CEA-LETI focusing on future 6G technologies. He has published around 120 papers in international conferences, journals and books chapters, given more than 150 international invited talks, keynotes and tutorials. He is the Main Inventor or Co-Inventor of more than 60 patents.

Michele Rossi (Senior Member, IEEE) is a Full Professor of Telecommunications with the Department of Information Engineering (DEI), University of Padova (UNIPD), Italy, teaching courses within the master's degrees in ICT for Internet and multimedia with DEI and Data Science, offered by the Department of Mathematics with UNIPD. Since 2017, he has been the Director of the DEI/IEEE Summer School of Information Engineering. In recent years, he has been involved in several EU projects on IoT technology (e.g., IOT-A, project no. 257521), and has collaborated with companies, such as DOCOMO (compressive dissemination and network coding for distributed wireless networks) and Worldsensing (optimized IoT solutions for smart cities). His research is currently supported by the European Commission through the H2020 projects SCAVENGE on "green 5G networks," MINTS on "mm-wave networking and sensing" and GREENEDGE on "green edge computing for mobile networks" (project coordinator). His research interests lie in wireless sensing systems, green mobile networks, edge, and wearable computing. In 2014, he was a recipient of the SAMSUNG GRO Award with a project entitled "Boosting Efficiency in Biometric Signal Processing for Smart Wearable Devices." From 2016 to 2018, he has been involved in the design of IoT protocols exploiting cognition and machine learning, as part of INTEL's Strategic Research Alliance (ISRA) Research and Development program. He has been a recipient of seven best paper awards from the IEEE and currently serves on the Editorial Boards of the IEEE TRANSACTIONS ON MOBILE COMPUTING, and the *Open Journal of the Communications Society*.