A Markov Decision Process Solution for Energy-Saving Network Selection and Computation Offloading in Vehicular Networks

Swapnil Sadashiv Shinde, *Student Member, IEEE*, and Daniele Tarchi, *Senior Member, IEEE*

***Abstract*—Vehicular Edge Computing (VEC) enables the integration of edge computing facilities in vehicular networks (VNs), allowing data-intensive and latency-critical applications and services to end-users. Though VEC brings several benefits in terms of reduced task computation time, energy consumption, backhaul link congestion, and data security risks, VEC servers are often resource-constrained. Therefore, the selection of proper edge nodes and the amount of data to be offloaded becomes important for having VEC process benefits. However, with the involvement of high mobility vehicles and dynamically changing vehicular environments, proper VEC node selection and data offloading can be challenging. In this work, we consider a joint network selection and computation offloading problem over a VEC environment for minimizing the overall latency and energy consumption during vehicular task processing, considering both user and infrastructure side energy-saving mechanisms. We have modeled the problem as a sequential decision-making problem and incorporated it in a Markov Decision Process (MDP). Numerous vehicular scenarios are considered based upon the users' positions, the states of the surrounding environment, and the available resources for creating a better environment model for the MDP analysis. We use a value iteration algorithm for finding an optimal policy of the MDPs over an uncertain vehicular environment. Simulation results show that the proposed approaches improve the network performance in terms of latency and consumed energy.**

***Index Terms*—Vehicular edge computing, network selection, computation offloading, energy saving mechanisms, markov decision process.**

I. Introduction

With the rapid growth of the automotive industry, fueled by the demands from the end-users and the integration of innovative technologies like the Internet of Things (IoT), modern wireless communication technologies, automated vehicles with advanced communication and computation technologies are now part of the vehicular networks (VNs) [1]. These new vehicles are capable of providing new services and applications to vehicular users (VUs) aiming at increasing road safety, avoiding traffic congestion, reducing the pollution level, providing new infotainment services, etc. However, modern applications and services come with stringent requirements in terms of high data processing and critical latency bounds. With limited onboard resources, vehicles alone cannot cope with such requirements and need support from additional platforms, e.g., cloud and edge computing [2].

Though cloud computing facilities have enormous computing resources, since they are located deep inside the core networks, high transmission delays often limit their uses for latency-critical VNs. Edge Computing (EC) technology can address the cloud computing problems by bringing the cloud resources in the proximity of end-users. EC has achieved a great success in the wireless networks when serving users with new innovative services [3]. In VNs, EC facilities can be enabled through the deployment of Road Side Units (RSUs) along the road facilitating several EC servers [4]. This approach, known as vehicular edge computing (VEC), has the potential to serve VUs with reduced transmission delays and energy requirements. The importance of VEC in the VN scenarios is highlighted by several works in the recent past, mainly for enabling latency-critical applications [5], [6].

VEC technology provides a computation environment to VUs for processing their tasks. VUs can transmit a portion of their computation load to the nearby VEC servers while performing the remaining computation locally. VEC servers perform the processing operations on behalf of the VUs and return the results. This approach is known as partial computation offloading, which allows VUs to complete a task processing operation in collaboration with VEC servers to reduce the overall latency and energy requirements during processing [4]. However, when coping with a large number of VUs demanding computation offloading services from VEC servers having limited computation/communication resources, energy limitations, storage capabilities, and coverage range, several new challenges arise into VEC-enabled VNs. These challenges are mainly characterized by a proper selection of when, where, and how much data need to be offloaded to the VEC servers for having adequate performance. This problem is also known as joint network selection and computation offloading, which aims to find a proper VEC server and the amount of data to be offloaded over dynamic vehicular environments [4].

Manuscript received 17 December 2022; revised 6 February 2023 and 24 March 2023; accepted 31 March 2023. Date of publication 5 April 2023; date of current version 19 September 2023. This work was supported in part by European Union under the Italian National Recovery and Resilience Plan (NRRP) of NextGenerationEU, partnership on "Telecommunications of the Future" (PE00000001 - Program "RESTART"). The review of this article was coordinated by Dr. Chau Yuen. *(Corresponding author: Daniele Tarchi.)*

The authors are with the Department of Electrical, Electronic and Information Engineering "Guglielmo Marconi", University of Bologna, 40126 Bologna, Italy (e-mail: swapnil.shinde2@unibo.it; daniele.tarchi@unibo.it).

Digital Object Identifier 10.1109/TVT.2023.3264504

In recent times, Machine Learning (ML) algorithms, especially Reinforcement Learning (RL), have gained lots of popularity for solving such highly complex problems over uncertain vehicular environments [5], [7], [8], [9], [10], [11]. The decision of selecting a proper RSU node for computation offloading depends upon several factors such as VUs position, speed, nearby VUs, the available number of RSUs for offloading, etc. The exact amount of tasks to be offloaded towards the selected RSU can be impacted by these factors, which often change over time. If a VU is under the coverage of multiple RSUs, the most suitable RSU can be selected by sequentially choosing one after another. Also, VU can make sequential decisions for finding a proper amount of data to be offloaded towards the RSU server. Every decision made by a VU can alter the surrounding environment's state, and can be characterized by some rewards (i.e., an increase or decrease in the task processing time and energy). Therefore, finding the proper edge node (EN) and the corresponding data to be offloaded in the dynamic vehicular environment can be considered as a sequential decision-making problem, and RL-based solution methods can be used to solve it [12].

Among several RL-based techniques, the Markov Decision Processes (MDP) can be effectively used to solve the VEC offloading problem [13]. The main components of the MDP models are state space, action space, reward function, and environment dynamics [14]. In a particular instance, the MDP agent, being into a current state, performs a particular action and receives the observations in terms of a state change and possible reward based upon the current state and the action performed. Over time, the MDP agent aims to learn an optimal policy function that maps the states over optimal actions for maximizing the reward. A proper environment dynamics, modeling the state transition probabilities from one state to another based upon the taken action, is required.

In recent times, with the integration of modern technologies, such as IoT, and new communication modes, including Vehicle-to-Vehicle (V2V), Vehicle-to-Road Side Units (V2R), Vehicle-to-Infrastructure of Cellular Networks (V2I), Vehicle-to-Sensors (V2S), and Vehicle-to-Person (V2P), VUs can learn several environment parameters and also share important information [15]. VUs can learn about the nearby competing VUs and their offloading experiences, RSU locations, and availability, etc. This information can be utilized to create better environment dynamics and, in the MDP-based approaches, for solving the VEC offloading problem. With this in mind, in this work, we propose an MDP-based model for the computation offloading problem over the VEC environment. In particular, considering a proper mobility model and different communication technologies, we identify several VUs scenarios. With the help of these scenarios, we define a multidimensional MDP model, where time-dependent transition probabilities equations are proposed.

The scenario under consideration is a VEC-enabled VN with a set of VUs, RSUs, and one macro base station (MBS). VUs generate tasks, and, with limited onboard resources, they request services from the nearby RSUs and MBS. Additionally, VUs are characterized by high mobility with varying speeds. Each VU is covered by multiple RSUs, while each RSU can serve multiple VUs with its limited resources. RSUs are also supposed to operate in different power-saving modes (i.e., standby, active, etc.) for reducing energy consumption during operations. With limited RSU resources and VUs mobility, finding a proper VU-RSU pair can improve the performance of a resource-constrained VN. At the same time, offloading an optimal amount of data to the selected RSU can further increase the performance in terms of energy and latency reduction.

A. Related Work

The problem of network selection and offloading parameters definition aiming at minimizing the latency and energy costs have been addressed in the literature mostly separately.

In [16], authors have considered a computation offloading problem for a vehicular scenario where the aim is to minimize a priority weighted delay performance during the task processing operations. However, the energy performance is neglected. In another case, in [17], authors have proposed a learning-based, energy-efficient task offloading strategy for the vehicular case without optimizing the delay performance. In many such cases, authors have either minimized the latency or energy costs individually and often neglected the cost of the edge-based servers. With the new services and applications in VNs, having critical latency requirements along with larger computations with higher energy costs, minimizing both latency and energy costs together is highly important. In recent times, several researchers have tried to optimize the latency and energy costs in mobile vehicular networks for different settings [10], [11], [18], [19], [20], [21]. In [10], authors have performed the joint optimization of latency and energy costs for VEC-based systems by finding the optimal portion of the offloading data. However, it is assumed that the VUs select the nearest ENs for offloading their data. In [11], authors have proposed a dynamic offloading strategy for the vehicular scenario based upon the Imitation Learning techniques. An energy-efficient strategy is proposed for latency-constrained vehicular tasks. However, the authors have only optimized the energy performance while considering the latency as a constraint. Moreover, the authors have performed the binary offloading operations without considering any energy-saving mechanism on edge node sides. In [18], authors have studied the energy-latency tradeoff for computation offloading operations in dynamic vehicular environments. However, this work is based on binary offloading operations where vehicular tasks are processed either by VUs or edge servers only. In [19], authors have studied the delay and energy performance of VEC systems for federated learning-based applications by neglecting the edge node side energy costs. In [20], [21], authors have addressed the joint network selection and offloading problem, for the minimization of latency and energy costs. However, the edge node side energy cost is not taken into account. In [22], authors have studied the computation offloading problem in heterogeneous VEC scenarios, where multi-armed bandit theory is applied, and online and off-policy learning algorithms are proposed for the network selection problem. However, the authors have performed a network selection operation by assuming a complete task migration toward the selected node. Such solutions have a limited performance since computation offloading and network selection can impact each other. Offloading the optimal amount of data to the incorrect edge server can reduce the performance of a system. Selecting a proper EN without optimizing the amount to be offloaded can not be considered an optimal solution. Recently, in [23], authors have proposed a multi-task learning-based solution for solving the computation offloading problem over a resource-limited EC scenario. The problem is formed as a mixed-integer nonlinear programming problem, and a multi-task learning-based feed-forward neural network model is proposed for optimizing the offloading and resource allocation decisions. In another case, in [24], a two-tier edge intelligence-empowered autonomous driving framework is proposed for assisting VUs with proper offloading decisions, resource allocations, etc.

In the past, some works have considered joint optimization approaches over vehicular networks for different cases, e.g., joint computation offloading and user association [25], joint computation offloading and task scheduling over VEC [26], joint resource allocation and computation offloading [27], and joint computation offloading and caching [28]. In some works, authors have considered a joint network selection and computation offloading problem over a dynamic vehicular scenario [29] limited to the optimization of the delay performance. In [30], authors have developed an efficient partial computation offloading and adaptive task scheduling algorithm for vehicular services. A two-sided matching algorithm is proposed for transmission scheduling, and convex optimization is used for finding a partial offloading ratio. However, most of these works considered traditional heuristic or meta-heuristic approaches and result in limited performance. In addition to this, most of the works are mainly concerned about the VUs energy and completely neglect the RSU energy performance.

B. Motivation

In the current vehicular literature, several works have either optimized the performance in terms of latency or energy costs. In some cases authors have considered the joint minimization of latency and energy costs, however, studies are limited to the network selection or computation offloading processes only. Additionally, in some cases, the energy costs only include the vehicular side energy costs with some assumptions on the edge node energy resources. Also, several of these works have used the traditional energy consumption models while analyzing the energy costs at the edge facilities. This motivates us to form a joint network selection and computation offloading problem with latency and energy cost minimization with an advanced energy consumption model at the EN.

In some cases, authors have considered heuristic or meta-heuristic approaches with limited performance over complex vehicular scenarios. In addition to this, in some cases, authors have considered advanced ML-based frameworks such as RL or DRL methods with model-free solution approaches. Such methods can suffer from a higher convergence time, computation complexities, and unstable behaviors mainly due to the high dynamicity of vehicular environments. This motivates us to propose a novel multi-dimensional MDP model based on local vehicular data with time-dependent state transition probabilities.

Therefore, in this work, we have proposed a joint network selection and computation offloading strategy over a mobile vehicular network for overall latency and energy minimization of both vehicular and infrastructure nodes with additional energy saving mechanism at the edge infrastructure. An original MDP-based RL framework with time-dependent state transition probabilities is proposed, where local vehicular environment parameters are used effectively.

C. Contributions

The main contributions of this work are:

- *Joint network selection and computation offloading problem formulation:* We define a joint network selection and computation offloading problem for minimizing the overall latency and energy consumption over VN as a constrained optimization problem, where ENs can be in different energy-saving states, i.e., standby or active, for a more efficient energy-saving behavior.
- *MDP model with time-dependent state transition probabilities:* The problem is modeled as a sequential decision-making problem and incorporated into an MDP-based model. Various elements of the MDP process including state space, action-space, reward function, and environment dynamics with time-dependent state-transition probabilities are considered.
- *V2X-based on-road scenarios:* Exploiting V2X communication technologies and a proper mobility model, various on-road VUs scenarios are defined for solving the burden of the higher dimensional MDP process without hindering its performance.
- *Value iteration method for MDP policy:* A value iteration-based approach is used for finding the optimal policy for the MDP process. In addition, a set of benchmark methods are considered to analyze the performance of the proposed scheme.

The remaining parts of this paper are composed as follows. Section II introduces the system model and defines the optimization problem to be solved. In Section III, we define different vehicular scenarios based on the nearby environment and design the MDP elements for the considered problem. A set of time-dependent state transition probability equations are defined. In Section IV, a value iteration algorithm and the corresponding elements are detailed for finding an optimal policy for the considered MDP model. Additionally, two other benchmark methods are proposed for comparison purposes. In Section V, the numerical results obtained through computer simulations are provided and analyzed. Finally, in Section VI, the conclusions are drawn.

II. System Model and Problem Formulation

In this work, an urban Internet of Vehicles (IoV) scenario for intelligent transportation systems with connected and intelligent VUs is considered, where a set of randomly distributed VUs over

TABLE I
LIST OF IMPORTANT NOTATIONS

Notation	Description
M	No. of VUs
N	No. of RSUs
τ_i	ith time instance
$\{x_m(\tau_i), y_m(\tau_i)\}$	mth VUs location
$\{x_n^R, y_n^R\}$, h_n^R	nth RSUs Location and Height
$\vec{v}_m(\tau_i)$	mth VUs location
c_m, f_m	mth VUs Computation Resources
B_n^R	RSU Bandwidth
$\mathcal{L}_n$, c_n^R , f_n^R	CPU cores, FLOPS, frequency
$\{x^{\bar{M}}, y^{\bar{M}}\}$, $h^{\bar{M}}$	MBS Location and Height
d_n	RSU Coverage Range
d^M	MBS Coverage Range
P_a	Task Generation Probability
$\rho_m(\tau_i)$	mth VUs Task Request
D_{ρ_m}	Computation Task Size
$D^r_{\rho_m}$	Computation Task Results
Ω_{ρ_m}	Task Computation Requirements
T_{ρ_m}	Task latency Requirements
$D_{m,n}(\tau_i)$	RSU Coverage Distance Available
$T^{soj}_{m,n}(\tau_i)$	Sojourn Time
$\mathbf{A}(\tau_i)$	VU-EN assignment matrix
$a_{m,n}(\tau_i)$	VU-EN (m, n) assignment Variable
K^{max}	Maximum VUs Server by RSU
$\alpha_{\rho_m}(\tau_i)$	Offloading Parameter
$r_{kl}(\tau_i)$	Transmission Rate
Pt_k	Transmission power
$E_{sw,n}$	RSU Switching Energy
$E^{st}_{en}(\tau_i)$	RSU Standby Mode Energy
γ_1, and γ_2	Weight Coefficients
G	Number of Grid Segments
$\mathcal{E}_m(\tau_i)$	No. of ENs available for Offloading
$d_{m,n}(\tau_i)$	Distance between VU and EN
$\mathcal{V}_m(\tau_i)$	mth VUs Scenario
$\mathcal{ST}$, $\mathcal{AS}$	MDP State, Action Spaces
$P^{F^1}_{(\bar{i},\bar{j})}(a_\nu(\tau_i))$	State Transision Prob. from State $\bar{i}$ to $\bar{j}$
$\exp(\cdot)$	Exponential Function
$\mathbb{E}(\cdot)$	Expected value
Δ	No. of MDP Steps

the road network can communicate with the edge computing servers enclosed by RSUs and a Macro Base Station (MBS). In recent times, such urban IoV scenarios have gained a lot of attention from the vehicular research community [18], [19]. Table I, includes the important notations considered in the following parts. We refer to $\mathcal{V} = \{VU_1, \ldots, VU_m, \ldots, VU_M\}$ as the set of M VUs, and $\mathcal{R} = \{RSU_1, \ldots, RSU_n, \ldots, RSU_N\}$ as the set of N RSUs in the area.

The system is modeled in a time-discrete manner, and the network parameters are supposed to be constant over each time interval τ, where τ_i identifies the ith time interval, i.e., $\tau_i = \{\forall t | t \in [i\tau, (i+1)\tau]\}$ [10]. By focusing on the ith time interval, the mth VU is located in the position $\{x_m(\tau_i), y_m(\tau_i)\}$, while it moves at a speed $\vec{v}_m(\tau_i)$ along the multi-lane road-path in either direction and equipped with a processing capability equal to c_m Floating Point Operations per Second (FLOPS) per CPU cycle, while its CPU frequency is f_m. We have assumed resource-limited edge computing nodes equipped with muli-core computing hardware with restricted capacities and limited bandwidth resources [20], [21], [31]. Each RSU can be

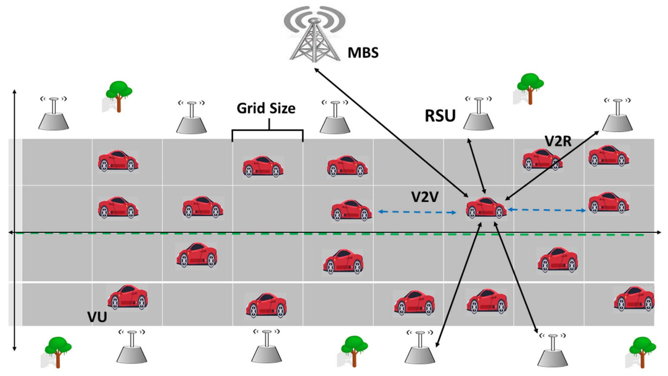

Fig. 1. System Architecture.

identified through a set of parameters where the nth RSU is located at the fixed position $\{x_n^R, y_n^R\}$ having height h_n^R, able to provide communication with a maximum bandwidth B_n^R, and having a multi-core CPU processor with $\mathcal{L}_n$ cores with c_n^R FLOPS per CPU cycle, while its CPU frequency is f_n^R. Similarly, the MBS can be identified through its position $\{x^{\bar{M}}, y^{\bar{M}}\}$, its height $h^{\bar{M}}$, maximum bandwidth $B^{\bar{M}}$, supposed to be equipped with a multi-core processor, where each core has a processing capability equal to $c^{\bar{M}}$ FLOPS per CPU cycle, while its CPU frequency is $f^{\bar{M}}$. Here, we do not put any limitation over the MBS CPU cores and assume that each VU can have access to only one CPU core.

The nth RSU has a limited coverage range d_n, whose value depends on the communication technology and radio-propagation environment, and it is supposed to provide VEC services to the vehicles within the coverage area. Similarly, for the MBS, the coverage range d^M stands. Thus, VUs can offload data up to $N+1$ ENs, i.e., N RSUs (i.e., $EN_1, \ldots, EN_N$) and one MBS (i.e., EN_0). Each $VU_m \in \mathcal{V}$ is supposed to be active in each time interval with a probability p_a within which it generates a computation task request $\rho_m(\tau_i)$ identified through the tuple $\langle D_{\rho_m}, D^r_{\rho_m}, \Omega_{\rho_m}, T_{\rho_m}\rangle$ corresponding to a task with size D_{ρ_m} Byte, expected to give in output a result with size $D^r_{\rho_m}$ Byte, requesting Ω_{ρ_m} CPU execution cycles and a maximum execution latency T_{ρ_m}.

In Fig. 1, a possible IoV scenario is depicted, where randomly distributed VUs are able to offload their computation tasks to the nearby ENs. Also, each VU is covered by multiple RSUs along with one MBS. VUs can communicate with ENs over V2R links and with each other through V2V links for information sharing.

VU Mobility and Sojourn Time: Due to the VUs mobility, each offloading operation should be completed by the VU sojourn time, corresponding to the amount of time it remains under the coverage of the selected EN [32], for avoiding additional latency due to, e.g., vehicle handover, service migration, additional signaling for managing vehicles and service mobility. RSU handover process involves transferring the management of active communication from one RSU to another [33]. Such handover situations can occur if VU fails to get back the offloaded task results before it passes through the RSU coverage. The handovers can degrade the network-wide performance in terms of latency.

Individual VUs mobility parameters often depend upon the nearby VUs decisions. One of the most often considered mobility models for vehicular scenarios is based on the preceding car dynamics [34]. Here, we adopt a similar model for analyzing the VUs mobility. If $\vec{v}_{v_m}(\tau_i)$ and $a_{v_m}(\tau_i)$ represent the speed and acceleration parameters at the ith interval for the mth VU, the model consider that the mth VU mobility parameters depend on the motion and dynamics of the preceding VUs, i.e.,

$$a_{v_m}(\tau_i) = a_{\max}\left[1 - \left(\frac{\vec{v}_{v_m}(\tau_i)}{\vec{v}_{\max}}\right)^{\delta} - \left(\frac{s^*(\vec{v}_{v_m}, \Delta\vec{v}_{v_m})}{s_{v_m}}\right)^2\right] \forall m$$

where $a_{\max}$ is the maximum acceleration value, $\vec{v}_{\max}$ is the desired velocity required for the steady traffic flow, $\Delta\vec{v}_{v_m} = \vec{v}_{v_m} - \vec{v}_{v,m-1}$ and $s_{v_m} = x_{v,m-1} - x_{v_m} - l_o$ are the relative velocity and inter-vehicular distance between m and $m-1$ with l_o being the VUs length. $\delta \in \{1, 5\}$ is the sensitivity of driver, and s^* is the desired space given as:

$$s^*(\vec{v}_{v_m}, \Delta\vec{v}_{v_m}) = s_{\min} + t_r\vec{v}_{v_m} + \frac{\vec{v}_{v_m}\Delta\vec{v}_{v_m}}{2\sqrt{a_{\max}b_{\max}}} \forall m$$

Here, $s_{\min}$ is the desired safe space between consecutive VUs, t_r is the minimum reaction time headway based upon the safe distance, and $b_{\max} > 0$ is the comfortable braking deceleration. In this work, the safety distance between VUs is considered as a design parameter similar to the [34]. However, the safety distance between VUs can be based upon several parameters and tradeoffs i.e., traffic flow characteristics, VUs safety demands, communication capabilities, V2V delays, etc. Interested readers can follow [35], [36] for more information. Therefore, at the ith interval, the mth VU speed and position are:

$$\vec{v}_{v_m}(\tau_i) = \vec{v}_{v_m}(\tau_{i-1}) + a_{v_m}(\tau_{i-1})\tau \quad (1)$$

$$x_{v_m}(\tau_i) = x_{v_m}(\tau_{i-1}) + \vec{v}_{v_m}(\tau_{i-1})\tau + a_{v_m}(\tau_{i-1})\tau^2 \quad (2)$$

The distance in which the mth VU remains under the coverage of nth EN is $D_{m,n}(\tau_i)$ and is given by:

$$D_{m,n}(\tau_i) = \sqrt{d_n^2 - (y_n^{EN} - y_m(\tau_i))^2} \pm (x_n^{EN} - x_m(\tau_i)) \quad (3)$$

where (x^{EN}, y^{EN}) is the location of nth EN, i.e., either an RSU or the MBS. The available sojourn time for the mth VU can be written as:

$$T^{soj}_{m,n}(\tau_i) = \frac{D_{m,n}(\tau_i)}{|\vec{v}_m(\tau_i)|} \qquad \forall i, \quad n = 0, 1, \ldots, N \quad (4)$$

VU-EN Assignment, Offloading Process and Resource Allocation: We define a binary VU-EN assignment matrix $\mathbf{A}(\tau_i) = (a_{m,n}(\tau_i)) \in \{0, 1\}$ with size $M \times (N+1)$. If mth VU is assigned to nth EN in the interval τ_i then $a_{m,n}(\tau_i) = 1$, and $\sum_{n=0}^{N}\sum_{m=1}^{M} a_{m,n}(\tau_i) = M$, where it is supposed that each VU is able to offload data to only one EN. It should be noted that the first column ($n = 0$) represents the assignments towards MBS, while the remaining columns, from n equal to 1 to N, are considered for RSUs. The number of VUs requesting services from the nth EN is given by $K_n(\tau_i) = \sum_{m=1}^{M} a_{m,n}(\tau_i)$. With their limited resources, RSUs can provide services to the VUs before task communication and computation costs become unbearable. We consider that $K^{\max}$ is the maximum number of VUs that can access to the services of each RSU node. However, with rich resource sets, MBS can provide services to several VUs without such limits.

We assume to perform partial offloading, where tasks can be split and processed remotely while the remaining portion is processed locally [20], [21], [32]; the offloaded portion by the mth VU at τ_i is identified as $\alpha_{\rho_m}(\tau_i) \in \{0, 1\}$. With multiple VUs requesting services from the same EN, during the offloading process, the following constraints need to be taken into account $\forall i, n = 1, \ldots, N$:

$$\begin{cases} K_n(\tau_i) \leq K^{\max} & \text{(5a)} \\ \sum_{m=1}^{K_n(\tau_i)} c_n^{\rho_m}(\tau_i) \cdot f_n^{\rho_m}(\tau_i) \leq (\mathcal{L}_n \cdot c_n^R \cdot f_n^R) & \text{(5b)} \\ \sum_{m=1}^{K_n(\tau_i)} b_n^{\rho_m}(\tau_i) \leq B_n^R & \text{(5c)} \end{cases}$$

where $c_n^{\rho_m}(\tau_i) \cdot f_n^{\rho_m}(\tau_i)$ is the processing capacity of nth EN assigned to the mth VUs task, $b_n^{\rho_m}(\tau_i)$ is the communication resources assigned to the VU for communicating with the nth EN. Eqs (5) model an upper bound on the number of users connected, processing capacity, and the communication resources of the RSUs. The constraint (5a) refers to a system constraint for limiting the complexity of the system model. The edge infrastructure manager can define a strategy for the scenarios where the number of VUs requesting the services from the same RSU node becomes higher than $K^{\max}$. In the considered vehicular scenarios, VUs are forced to perform the local computation of their whole tasks in case the limit is violated.[1] It is worth to be noticed that the capacity of each link depends on the specific communication technology and it is out of the scope of this paper. Also, we consider that MBS has abundant resources and is able to serve a large number of VUs without limitations.

With limited EN communication and computation resources, proper scheduling is required when multiple users access. Here, we use the following model for assigning EN resources to the VUs for computation offloading:

$$c_n^{\rho_m}(\tau_i) \cdot f_n^{\rho_m}(\tau_i) = \begin{cases} c_n^R \cdot f_n^R & K_n(\tau_i) \leq \mathcal{L}^n \\ \frac{c_n^R \cdot f_n^R}{\left\lceil \frac{K_n(\tau_i)}{\mathcal{L}^n} \right\rceil} & \mathcal{L}^n \leq K_n(\tau_i) \leq K^{\max} \end{cases} \quad (6)$$

$$b_n^{\rho_m}(\tau_i) = \frac{B_n^R}{K_n(\tau_i)} \qquad \text{for } n = 1, \cdots N \quad (7)$$

(6) and (7) show the EN resource allocation in terms of computation capacity and bandwidth to the VUs' tasks. According to (6), if the number of VUs requesting services from the nth EN are less than $\mathcal{L}_n$, each can have access to the single CPU core with capacity $(c_n^R \cdot f_n^R)$. In case the number of users becomes higher than $\mathcal{L}_n$, multiple VUs share CPU core resources. Here $\lceil x \rceil$ is the ceiling function applied over x for rounding it to the

[1]Note that this is just one possible approach that can be adapted by the RSU nodes. Though it is beyond the scope of this work, these decisions can further be optimized based on specific load-balancing techniques.

nearest integer value higher than or equal to x. According to (7), bandwidth resources will be equally shared among all requesting VUs.

If the mth VU is assigned to the MBS, i.e., $n = 0$, it can have access to the single CPU core, and equally shares bandwidth resources with the other connected VUs. Thus:

$$c_n^{\rho_m}(\tau_i) \cdot f_n^{\rho_m}(\tau_i) = c^{\bar{M}} \cdot f^{\bar{M}}, \qquad b_n^{\rho_m}(\tau_i) = \frac{B^{\bar{M}}}{K_n(\tau_i)} \tag{8}$$

In the following, we model the delay and energy requirements of various operations involved during the partial computation offloading enabled vehicular task processing.

Task Computation Model: The generic expression for the time and energy spent for the ρ_mth task computation on any device is given by [37]:

$$T_{c,l}^{\rho_m} = \frac{\Omega_{\rho_m}}{c_l f_l}, \quad E_{c,l}^{\rho_m} = T_{c,l}^{\rho_m} P_{c,l} \tag{9}$$

where c_l and f_l are the number of FLOPS per CPU cycle and CPU frequency, respectively, whether l identifies a VU (m), an RSU (n) or the MBS $(\bar{M})$. In (9), $P_{c,l}$ is the computation power used by the generic lth device.

Task Communication Model: Since we assume to perform a partial computation offloading, each VU transmits a portion of its task to the assigned EN and receive back the result. Similarly, ENs receive tasks from VUs and send back the results. In general, the transmission time and energy between a generic node k and a generic node l for task ρ_k is given by[2]:

$$T_{tx,kl}^{\rho_k}(\tau_i) = \frac{D_{\rho_k}}{r_{kl}(\tau_i)}, \qquad E_{tx,kl}^{\rho_k}(\tau_i) = T_{tx,kl}^{\rho_k}(\tau_i) Pt_k \tag{10}$$

where $r_{kl}(\tau_i)$ is the data-rate of the link between the two nodes, while Pt_k is the transmission power of kth node. Similarly, the reception time and energy to receive the task of size $D_{\rho_k}^r$ from lth EN by the kth node are:

$$T_{rx,lk}^{\rho_k}(\tau_i) = \frac{D_{\rho_k}^r}{r_{kl}(\tau_i)}, \qquad E_{rx,lk}^{\rho_k}(\tau_i) = T_{rx,lk}^{\rho_k}(\tau_i) Pr_k \tag{11}$$

where Pr_k is the power spent for receiving data.

The channel transmission rate between a generic node k and l at the ith interval can be modeled as [22], [38]:

$$r_{kl}(\tau_i) = b_l^{\rho_k}(\tau_i) \log_2\left(1 + \frac{Pt_k \cdot h_{k,l}(\tau_i)}{\sigma^2 + I_{kl}(\tau_i)}\right) \quad \forall k, l$$

where Pt_k is the transmission power of node k, $b_l^{\rho_k}(\tau_i)$ is the communication bandwidth, σ^2 is the noise power, and $I_{kl}(\tau_i)$ is the interference due to any transmitting node, except k, towards node l, where the total interference during the uplink communication (i.e., VU to RSU) can be calculated as

$$I_{kl}(\tau_i) = \sum_{\forall k' \in K_l(\tau_i) \setminus \{k\}} (Pt_{k'} \cdot h_{k',l}(\tau_i)).$$

For the downlink, instead, we assume to neglect the interference by assuming an orthogonal frequency assignment among RSUs, as well orthogonal RSU to VU transmissions.

[2]In the following we identify with l and k the indexes of any generic node. Hence, l and k can have any index among m, n, and $\bar{M}$.

EN Operating Modes: For improving the overall energy efficiency, we assume that ENs can be either in a stand-by or an active state. ENs in a standby state will not be able to serve any VU and effectively will reduce the overall energy consumption. A switching process is assumed for switching ENs from standby to active state with additional switching time and energy. The amount of energy consumed for switching the nth EN from standby to active state is [39]:

$$E_{sw,n} = P_{sw,n} \cdot T_{sw,n} \tag{12}$$

where, $P_{sw,n}$ is the consumed switching power and $T_{sw,n}$ is the switching time. The amount of time consumed by nth EN for providing offloading services for VU m is given by[3]:

$$T_{en,n}^{\rho_m}(\tau_i) = \frac{T_{sw,n}}{\alpha_{\rho_m}(\tau_i)} + \left(T_{c,n}^{\rho_m} + T_{tx,nm}^{\rho_m}(\tau_i) + T_{rx,mn}^{\rho_m}(\tau_i)\right) \tag{13}$$

where $T_{c,n}^{\rho_m}$, $T_{tx,nm}^{\rho_m}(\tau_i)$ and $T_{rx,mn}^{\rho_m}(\tau_i)$ are the time required for the task computation, transmission and reception between nth EN and mth VU, respectively.

The amount of energy consumed will be based on the operating modes. The nth EN will go into standby mode if no service request from any VU in its coverage area is mapped to it, i.e., $a(m, n) = 0, \forall m$. The total energy consumption of all ENs operating in the standby mode is given by:

$$E_{en}^{st}(\tau_i) = \sum_{n=1}^{N^{st}(\tau_i)} E_{en,n}(\tau_i) \quad \text{with} \quad E_{en,n}^{st}(\tau_i) = \tau_i \cdot P_{sd,n} \tag{14}$$

where $N^{st}(\tau_i) = \{n \mid K_n(\tau_i) = 0, \forall n\}$ gives the total number of ENs operating in the standby mode. Also, $E_{en,n}^{st}(\tau_i)$ is the amount of energy consumed by the nth EN, where $P_{sd,n}$ is the power consumed during standby mode that depends upon the computation hardware on the nth EN. Similarly, the amount of energy consumed by the nth EN while serving the mth VU is given by[4]:

$$E_{en,n}^{\rho_m}(\tau_i) = \frac{\tau_i \cdot P_{0,n} + \frac{E_{sw,n}}{K_n(\tau_i)}}{\alpha_{\rho_m}(\tau_i)} + E_{c,n}^{\rho_m} + E_{tx,nm}^{\rho_m}(\tau_i) + E_{rx,mn}^{\rho_m}(\tau_i) \tag{15}$$

where $P_{0,n}$ is the power consumed for the basic circuit operations, and $E_{sw,n}$ is the switching energy required. It should be noted that, as the switching operation occurs only once, if the number of VUs requesting services (i.e., $K_n(\tau_i)$) from a particular EN increases, the switching energy per VU scales down. $E_{c,n}^{\rho_m}$, $E_{tx,nm}^{\rho_m}(\tau_i)$ and $E_{rx,mn}^{\rho_m}(\tau_i)$ are the energy required during task computation, transmission, and reception of data between nth EN and mth VU, respectively.

Task Offloading Process: If mth VU is assigned to nth EN, then the time and energy required to offload the portion of the

[3]Division by $\alpha_{\rho_m}(\tau_i)$ is merely for equation balancing purposes whose effect will be nullified later in (17a).

[4]Division by $\alpha_{\rho_m}(\tau_i)$ is merely for equation balancing purposes whose effect will be nullified later in (17b).

task with offloading parameter α_{ρ_m} to the selected EN and to get back the result in the ith interval is (from (10) and (11)),

$$\hat{T}^{off}_{m,n}(\alpha_{\rho_m}(\tau_i)) = \alpha_{\rho_m}(\tau_i)\left(T^{\rho_m}_{tx,mn}(\tau_i) + T^{\rho_m}_{rx,nm}(\tau_i)\right) \tag{16a}$$

$$\hat{E}^{off}_{m,n}(\alpha_{\rho_m}(\tau_i)) = \alpha_{\rho_m}(\tau_i)\left(E^{\rho_m}_{tx,mn}(\tau_i) + E^{\rho_m}_{rx,nm}(\tau_i)\right) \tag{16b}$$

Also, the amount of time and energy consumed on the nth EN for providing services to the mth VU is given by (from (13) and (15)):

$$\hat{T}^{off}_{n}(\alpha_{\rho_m}(\tau_i)) = \alpha_{\rho_m}(\tau_i)\left(T^{\rho_m}_{en,n}(\tau_i)\right) \tag{17a}$$

$$\hat{E}^{off}_{n}(\alpha_{\rho_m}(\tau_i)) = \alpha_{\rho_m}(\tau_i)\left(E^{\rho_m}_{en,n}(\tau_i)\right) \tag{17b}$$

Thus, the total time and energy cost required for the offloading process is given by:

$$T^{off}_{m,n}(\alpha_{\rho_m}(\tau_i)) = \hat{T}^{off}_{m,n}(\alpha_{\rho_m}(\tau_i)) + \hat{T}^{off}_{n}(\alpha_{\rho_m}(\tau_i)) \tag{18a}$$

$$E^{off}_{m,n}(\alpha_{\rho_m}(\tau_i)) = w_1\hat{E}^{off}_{m,n}(\alpha_{\rho_m}(\tau_i)) + (1-w_1)\hat{E}^{off}_{n}(\alpha_{\rho_m}(\tau_i)) \tag{18b}$$

where (18b) is constituted by two parts (i.e., EN and VUs energy) that can be based upon different energy sources and can have different utility costs. Therefore, for having a properly balanced energy cost over the offloading process, we introduce w_1 as a weighting coefficient in the range between 0 and 1.

Local Computation: From (9), the amount of time and energy required for the local computation of the remaining task in the ith interval is:

$$T^{loc}_{m}(\alpha_{\rho_m}(\tau_i)) = (1-\alpha_{\rho_m}(\tau_i))\, T^{\rho_m}_{c,m} \tag{19a}$$

$$E^{loc}_{m}(\alpha_{\rho_m}(\tau_i)) = w_1\,(1-\alpha_{\rho_m}(\tau_i))\, E^{\rho_m}_{c,m} \tag{19b}$$

Partial offloading Computation: From (18)-(19), the delay and the energy consumed during the task processing phases when partial offloading is performed (in the ith interval) can be written as:

$$T^{\rho_m}_{m,n}(\alpha_{\rho_m}(\tau_i)) = \max\left\{T^{off}_{m,n}(\alpha_{\rho_m}(\tau_i)), T^{loc}_{m}(\alpha_{\rho_m}(\tau_i))\right\} \tag{20a}$$

$$E^{\rho_m}_{m,n}(\alpha_{\rho_m}(\tau_i)) = E^{off}_{m,n}(\alpha_{\rho_m}(\tau_i)) + E^{loc}_{m}(\alpha_{\rho_m}(\tau_i)) \tag{20b}$$

where the local and offloading processing are supposed to be performed in parallel. Each vehicle should finish the offloading process and receive the result back within the sojourn time, hence:

$$T^{off}_{m,n}(\alpha_{\rho_m}(\tau_i)) \le T^{soj}_{m,n}(\tau_i) \quad \forall i \tag{21}$$

Problem Formulation: The main aim of this work is to optimize the network-wide performance of the VEC-enabled vehicular network. We aim to optimize the performance in terms of overall latency and energy consumed during the offloading process towards edge servers by selecting proper ENs and offloading amounts. The latency and energy requirements of both sides (i.e., VUs and RSU-based edge servers) are considered during the offloading process. The joint latency and energy minimization problem is defined as:

$$\mathbf{P1}: \min_{\mathcal{A},\mathbf{A}} \left\{ \sum_{n=0}^{N}\sum_{m=1}^{M} \left[\gamma_1 T^{\rho_m}_{m,n}(\alpha_{\rho_m}(\tau_i)) + \gamma_2 E^{\rho_m}_{m,n}(\alpha_{\rho_m}(\tau_i))\right] + \gamma_2(1-w_1)E^{st}_{en}(\tau_i) \right\} \forall i \tag{22}$$

s.t.

$$\mathbf{C1}: \sum_{n=1}^{N} a_{m,n}(\tau_i) = 1, \quad \forall m \in M \tag{23}$$

$$\mathbf{C2}: \text{Eqs. } (5a), (5b) \text{ and } (5c) \tag{24}$$

$$\mathbf{C3}: T^{\rho_m}_{m,n}(\alpha_{\rho_m}(\tau_i)) \le T_{\rho_m} \quad \forall \mathcal{V}, \forall i \tag{25}$$

$$\mathbf{C4}: \text{Eq. } (21) \tag{26}$$

$$\mathbf{C5}: E^{off}_{m,n}(\alpha_{\rho_m}(\tau_i)) < w_1 E^{\rho_m}_{c,m} \tag{27}$$

$$\mathbf{C6}: 0 \le \gamma_1, \gamma_2, w_1 \le 1;\ \gamma_1 + \gamma_2 = 1 \tag{28}$$

where $\mathcal{A} = \{\alpha_{\rho_m}\}^M$ is the computation offloading matrix, $\mathbf{A}$ is the VU-EN assignment matrix defined previously, and γ_1, and γ_2 are weighting coefficients for balancing latency and energy consumption. The objective function in $\mathbf{P1}$ includes the overall latency, VU, and the RSU side energy costs including both active and standby modes costs. $\mathbf{C1}$ stands that each VU can select at most one RSU for the computation offloading. $\mathbf{C2}$ provides the limits over the number of user requests, processing capacity, and bandwidth resource blocks requested by VUs towards ENs, while $\mathbf{C3}$ puts a limit on the maximum processing time as one of the task requirements. According to $\mathbf{C4}$, for avoiding handover phenomena and related latency, each VU should complete the offloading process before it passes through the selected RSUs coverage. In order to have a valid offloading process, according to $\mathbf{C5}$, the weighted energy consumed on VU for processing a complete task should be lower than the total weighted energy required to compute a complete task locally. $\mathbf{C6}$ stands that the two weighting coefficients (γ_1, γ_2) should be between 0 and 1 with a sum equal to 1. Additionally, the energy coefficient w_1 can take a value between 0 and 1.

III. MDP Formation

When solving the problem in (22), we aim to minimize the overall latency and energy consumed by finding the combination of proper EN and the amount of data to be offloaded by each VU in the MBS service area. In this work, we consider the MDP-based RL approach to solve the problem at hand. The basic elements of the MDP model include the state-space, action-space, reward function, and environment dynamics. However, modeling environment dynamics (i.e., state transition probabilities of MDP states) over a highly uncertain vehicular environment can be a challenging task. Fig. 2, provides an overview of different elements discussed in the following parts. In the following, we first model several possible vehicular scenarios in which a reference VU can find itself over its course. This scenario set can be used to form a proper MDP model aimed at reducing uncertainty

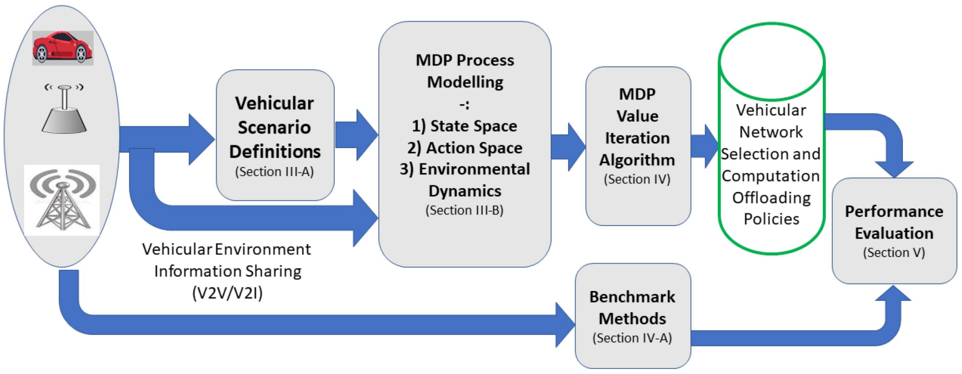
Vehicular Scenario Definitions (Section III-A)
MDP Process Modelling -: 1) State Space 2) Action Space 3) Environmental Dynamics (Section III-B)
MDP Value Iteration Algorithm (Section IV)
Vehicular Network Selection and Computation Offloading Policies
Performance Evaluation (Section V)
Vehicular Environment Information Sharing (V2V/V2I)
Benchmark Methods (Section IV-A)

Fig. 2. Proposed MDP Model.

over the environment. For avoiding any possible mistakes during the network selection and computation offloading process, each VU scenario needs to be treated separately. After that, we present the main MDP elements (i.e., state-space, action-space, reward function, and environment dynamics) for the considered problem. After a detailed analysis of the state transition probability matrix, we propose generic time-dependent expressions for finding the state transition probability values in different scenarios based on the VUs state and the action performed.

A. VU Scenarios Defintion

Different VU scenarios are formed, based upon VUs physical locations, number of ENs available for offloading, and the number of nearby competing VUs, aiming at creating a more reliable MDP model with reduced uncertainty. VUs can use V2X communication technologies for acquiring useful information about the number of nearby competing VUs and available EN servers. As shown in Fig. 1, we have used a grid-based approach for limiting the number of possible scenarios that depend upon the actual VUs position. In the considered grid-based approach, a section of the road is divided into G segments of length l_g, within which each VU is placed, considering its location parameters. Thus, each VU can have associated a specific section number given by $g_m^{id}(\tau_i) = \{1, 2, \ldots, G\}$. Each VU can exploit a different number of ENs for offloading, where $\mathcal{E}_m(\tau_i) = \{EN_n | D_{m,n}(\tau_i) > 0, \forall n\}$ is the set of available ENs for the mth VU to perform the offloading operation in the interval τ_i. Also, we define $\bar{V}_m(\tau_i) = \sum_{n=1}^{\mathcal{E}_m(\tau_i)} K_n(\tau_i)$ as the number of nearby competing VUs, ranging between 0 to $\mathcal{NV}_{\max}$, requesting offloading services from the ENs in the set $\mathcal{E}_m(\tau_i)$.

In the considered multi-user vehicular network, moving VUs can impact each other's network selection and offloading strategies. Each VU should analyze the surrounding environment by finding the competing VUs and their offloading decisions, selected ENs, etc. Since all VUs are supposed simultaneously generate the task requests (i.e., at each ith interval), it is impossible to have such information in advance. In that case, VUs can make offloading decisions by assuming that no other VU is requesting a service leading to a selfish approach. However, this may lead to incorrect node selection and offloading decisions. Another way to tackle this problem is by defining an MDP process that provides a joint solution for all the participating VUs. The presence of a large number of VUs can quickly lead to unbearable complexity and computation requirement. Thus both of these utmost approaches are not suitable for solving the given problem and some sort of assumption is needed for modeling the VUs surrounding environment for avoiding the incorrect offloading strategy/additional complexity. In the following, we consider four strategies supposed by VUs regarding the surrounding environment.

- *Minimum distance-based VU-EN assignment:* In this case, the mth VU considers that all the $\bar{V}_m(\tau_i)$ VUs are offloading their data to the nearest ENs based upon their physical locations. Thus, $\forall VU_{m'} \in \bar{V}_m(\tau_i)$:

$$a_{m',n}(\tau_i) = 1 \iff n = \underset{EN_{n'} \in \mathcal{E}_m(\tau_i)}{\text{argmin}} \{d_{m',n'}(\tau_i)\} \quad (29)$$

- *Maximum sojourn time-based VU-EN assignment:* In this case, the mth VU considers that all $\bar{V}_m(\tau_i)$ VUs are offloading their data to the ENs with maximum available sojourn time. Thus, $\forall VU_{m'} \in \bar{V}_m(\tau_i)$:

$$a_{m',n}(\tau_i) = 1 \iff n = \underset{EN_{n'} \in \mathcal{E}_m(\tau_i)}{\text{argmax}} \{T_{m',n'}^{soj}(\tau_i)\} \quad (30)$$

  It should be noted that this approach only considers the assignment towards RSU nodes (since MBS always have high sojourn time). If VUs are not able to find any nearby RSU nodes, they will be assigned to the MBS.
- *Probabilistic VU-EN assignments:* In this approach $\forall VU_{m'} \in \bar{V}_m(\tau_i)$ we select the $EN_{n'} \in \mathcal{E}_m(\tau_i)$ randomly. The probability of m'th VU selecting n'th EN is given by:

$$Pr\{a_{m',n}(\tau_i) = 1\} = \frac{1}{\mathcal{E}_m(\tau_i)} \quad (31)$$

- *Position-based VU-EN assignments:* In this case, each nearby competing VU is allocated to the ENs based on the available distance before it passes through the ENs coverage range and the distance between VU and EN. Thus $\forall VU_{m'} \in \bar{V}_m(\tau_i)$:

$$a_{m',n}(\tau_i) = 1 \Leftrightarrow \frac{D_{m',n}(\tau_i)}{d_{m',n}(\tau_i)} = \max_{EN_{n'} \in \mathcal{E}_m(\tau_i)} \left\{ \frac{D_{m',n'}(\tau_i)}{d_{m',n'}(\tau_i)} \right\}. \quad (32)$$

Based upon the above discussion for the mth VU, a vector $\hat{V}_m(\tau_i)$ corresponding to the number of nearby VUs assigned to each $EN_n \in \mathcal{E}_m(\tau_i)$ is formed as:

$$\hat{V}_m(\tau_i) = \{V_m^n(\tau_i)\}_{1\times\mathcal{E}_m(\tau_i)},$$
$$\text{with } V_m^n(\tau_i) = \sum_{m'=1}^{\bar{V}_m(\tau_i)} a_{m',n}(\tau_i), \quad \forall n \in \mathcal{E}_m(\tau_i) \tag{33}$$

where, VU-EN assignment (i.e., $a_{m',n}(\tau_i)$) is based upon any of the four methods presented above.

In the end, for the mth VU, a scenario vector can be defined as $\mathcal{V}_m(\tau_i) = \{g_m^{id}(\tau_i), \mathcal{E}_m(\tau_i), \hat{V}_m(\tau_i)\}$. The number of possible scenarios is limited by the parameters G, $\mathcal{E}^{\max}$, and $\mathcal{NV}_{\max}$. Each scenario needs to be treated separately for finding proper EN and offloading amounts. Every vehicular scenario may have an independent optimal policy that needs to be determined through proper analysis. In the end, $\bar{N}$ is the set of all possible VU scenarios.

In the next part, we define the State Space, Action Space, Environment Dynamics or State Transition Probabilities, and Reward Function, as basic elements of an MDP approach for the problem at hand.

B. MDP Elements

The MDP is a stochastic process that evolves over time and is characterized by the state space ($\mathcal{ST}$), action space ($\mathcal{AS}$), reward function (R), and environment dynamics ($\mathcal{P}$). The MDP model can be defined as a tuple $\langle \mathcal{ST}, \mathcal{AS}, R, \mathcal{P}\rangle$.

1) State-Space ($\mathcal{ST}$): In a multi-user vehicular environment, the available resources for the computation offloading process change continuously over time and are a function of the offloading and network selection decisions taken by individual vehicles. Therefore, we define a discrete state-space set function of resources available for computation offloading. For each scenario ν, the related state-space is a function of the sojourn time, the required latency, VU resources, and the resources of the available RSUs; thus, each state s_ν at time τ_i is defined as:

$$s_\nu(\tau_i) = f\left(\alpha_{\rho_m}(\tau_i), T_{m,n}^{soj}(\tau_i), B_n, c_n^R, f_n^R, \mathcal{L}^n, D_{\rho_m}, D_{\rho_m}^r, \Omega_{\rho_m}, T_{\rho_m}\right). \tag{34}$$

We suppose to limit the multi-dimensional state space to $\bar{N}$ scenarios, hence, $\nu = 1, \dots, \bar{N}$. Moreover, we assume that the environment states observed by each VU during the joint network selection and computation offloading process can be modeled through proper binary functions. If the mth VU is assigned to the nth EN and performs offloading operation with offloading parameter α_{ρ_m}, the environment can be modeled through three proper binary functions, as:

$$F_{\rho_m,n}^1(\tau_i) = \begin{cases} 0 & T_{m,n}^{off}(\alpha_{\rho_m}(\tau_i)) \le T_{m,n}^{soj}(\tau_i) \\ 1 & \text{else} \end{cases} \tag{35}$$

$$F_{\rho_m,n}^2(\tau_i) = \begin{cases} 0 & T_{m,n}^{\rho_m}(\alpha_{\rho_m}(\tau_i)) \le T_{\rho_m} \\ 1 & \text{else} \end{cases} \tag{36}$$

$$F_{\rho_m,n}^3(\tau_i) = \begin{cases} 0 & E_{m,n}^{off}(\alpha_{\rho_m}(\tau_i)) < w_1 E_{c,m}^{\rho_m} \\ 1 & \text{else} \end{cases} \tag{37}$$

where $F_{\rho_m,n}^1(\tau_i)$, $F_{\rho_m,n}^2(\tau_i)$ and $F_{\rho_m,n}^3(\tau_i)$ are the binary functions depending upon the sojourn time constraint (21), application latency requirement (25) and the energy constraint (27), respectively, and $F_{\rho_m,n}^3(\tau_i)$ includes both active and standby mode energy costs of RSU nodes. Thus, at τ_i, the state of mth VU in scenario ν is given by,

$$s_\nu^{m,n}(\tau_i) = \left\{F_{\rho_m,n}^1(\tau_i), F_{\rho_m,n}^2(\tau_i), F_{\rho_m,n}^3(\tau_i)\right\} \in S_\nu$$

where, $S_\nu = \mathbb{Z}_2^3$ is the complete state space for the *scenario* ν containing all possible binary combinations of $F_{\rho_m,n}^1(\tau_i)$, $F_{\rho_m,n}^2(\tau_i)$ and $F_{\rho_m,n}^3(\tau_i)$.

2) Action-Space ($\mathcal{AS}$): The action space defines all the possible actions available during the learning process. If mth VU belongs to the scenario ν, it can explore the available ENs ($\mathcal{E}_m(\tau_i)$), by properly setting a binary vector $\mathbb{EN}_\nu(\tau_i) = \{0,1\}^{\mathcal{E}_m(\tau_i)}$ mapping the RSUs selection among the $\mathcal{E}_m(\tau_i)$ available in the given scenario. At the same time, the offloaded amount can be selected from a discrete set of values given by $\alpha_{\rho_m}(\tau_i) \in \{0, \Lambda, 2\Lambda, \dots, 1\}$ where $0 < \Lambda < 1$ is a step change of offloading amount.

The generic action a_ν for the νth scenario at time τ_i can be defined as $a_\nu(\tau_i) = \{\mathbb{EN}_\nu(\tau_i), \alpha_{\rho_m}(\tau_i)\}$ where $\mathbb{EN}_\nu(\tau_i)$ is a binary vector with length ν, where 1 in the nth position corresponds to the selected EN. The complete action space for scenario ν is given by $A_\nu = \{a_\nu(\tau_i)\}$.

Once selected, action $a_\nu(\tau_i)$ can change the state of function $F^1(\tau_i)$, $F^2(\tau_i)$ and $F^3(\tau_i)$ with certain probability[5]. Such probabilistic transitions can be defined through:

$$P_{(\bar{i},\bar{j})}^{F^1}(a_\nu(\tau_i)) = Pr\left\{F^1(\tau_{i+\delta}) = \bar{j} \mid F^1(\tau_i) = \bar{i}, a_\nu(\tau_i)\right\} \quad \bar{i},\bar{j} \in \{0,1\} \tag{38}$$

where $P_{(\bar{i},\bar{j})}^{F^1}(a_\nu(\tau_i))$ is the transition probability of $F^1(\cdot)$ from state $\bar{i}$ to state $\bar{j}$ at τ_i through the action $a_\nu(\tau_i)$. Here, δ is the time step of the MDP process. Similarly for $F^2(\cdot)$ and $F^3(\cdot)$ the transition probability expressions are given by:

$$P_{(\bar{i},\bar{j})}^{F^2}(a_\nu(\tau_i)) = Pr\left\{F^2(\tau_{i+\delta}) = \bar{j} \mid F^2(\tau_i) = \bar{i}, a_\nu(\tau_i)\right\}$$
$$P_{(\bar{i},\bar{j})}^{F^3}(a_\nu(\tau_i)) = Pr\left\{F^3(\tau_{i+\delta}) = \bar{j} \mid F^3(\tau_i) = \bar{i}, a_\nu(\tau_i)\right\}$$

In general, $F^1(\cdot)$, $F^2(\cdot)$ and $F^3(\cdot)$ can have different probabilistic transitions for any given action $a_\nu(\tau_i)$. Here, we introduce three transition matrices by considering all the possible transitions of $F^1(\cdot)$, $F^2(\cdot)$, and $F^3(\cdot)$. For $F^1(\cdot)$, the transition matrix $P^{F^1}(a_\nu(\tau_i))$ is given by,

$$P^{F^1}(a_\nu(\tau_i)) = \begin{bmatrix} P_{(0,0)}^{F^1}(a_\nu(\tau_i)) & P_{(0,1)}^{F^1}(a_\nu(\tau_i)) \\ P_{(1,0)}^{F^1}(a_\nu(\tau_i)) & P_{(1,1)}^{F^1}(a_\nu(\tau_i)) \end{bmatrix}, \forall a_\nu(\tau_i) \tag{39}$$

[5]For the simplicity of notations hereafter we omit, $(\rho_m/m,n)$ from $s_\nu^{m,n}(\tau_i)$, $F_{\rho_m,n}^1(\tau_i)$, $F_{\rho_m,n}^2(\tau_i)$, and $F_{\rho_m,n}^3(\tau_i)$.

with, $P^{F^1}_{(0,0)}(a_\nu(\tau_i)) + P^{F^1}_{(0,1)}(a_\nu(\tau_i)) = 1$ and $P^{F^1}_{(1,0)}(a_\nu(\tau_i)) + P^{F^1}_{(1,1)}(a_\nu(\tau_i)) = 1$. Similarly, for F^2, the transition matrix $P^{F^2}(a_\nu(\tau_i))$ is given by,

$$P^{F^2}(a_\nu(\tau_i)) = \begin{bmatrix} P^{F^2}_{(0,0)}(a_\nu(\tau_i)) & P^{F^2}_{(0,1)}(a_\nu(\tau_i)) \\ P^{F^2}_{(1,0)}(a_\nu(\tau_i)) & P^{F^2}_{(1,1)}(a_\nu(\tau_i)) \end{bmatrix}, \forall a_\nu(\tau_i) \tag{40}$$

with, $P^{F^2}_{(0,0)}(a_\nu(\tau_i)) + P^{F^2}_{(0,1)}(a_\nu(\tau_i)) = 1$ and $P^{F^2}_{(1,0)}(a_\nu(\tau_i)) + P^{F^2}_{(1,1)}(a_\nu(\tau_i)) = 1$. Also, for the case of $F^3(\cdot)$, the transition matrix $P^{F^3}(a_\nu(\tau_i))$ is given by,

$$P^{F^3}(a_\nu(\tau_i)) = \begin{bmatrix} P^{F^3}_{(0,0)}(a_\nu(\tau_i)) & P^{F^3}_{(0,1)}(a_\nu(\tau_i)) \\ P^{F^3}_{(1,0)}(a_\nu(\tau_i)) & P^{F^3}_{(1,1)}(a_\nu(\tau_i)) \end{bmatrix}, \forall a_\nu(\tau_i) \tag{41}$$

with $P^{F^3}_{(0,0)}(a_\nu(\tau_i)) + P^{F^3}_{(0,1)}(a_\nu(\tau_i)) = 1$ and $P^{F^3}_{(1,0)}(a_\nu(\tau_i)) + P^{F^3}_{(1,1)}(a_\nu(\tau_i)) = 1$.

3) Reward Function $(R(s_\nu(\tau_i), a_\nu(\tau_i))$*:* The reward function $(R(s_\nu(\tau_i), a_\nu(\tau_i))$ is defined as the joint objective function of time and energy consumed for complete task processing (22). At the ith interval, if the mth VU is in state $s_\nu(\tau_i)$, and decides to take an action $a_\nu(\tau_i)$ by selecting the nth RSU and $\alpha_{\rho_m}(\tau_i)$ as an offloading amount, the instant reward received by it is given by,

$$R(s_\nu(\tau_i), a_\nu(\tau_i)) = \left[\gamma_1 T^{\rho_m}_{m,n}(\alpha_{\rho_m}(\tau_i)) + \gamma_2 E^{\rho_m}_{m,n}(\alpha_{\rho_m}(\tau_i))\right] \tag{42}$$

4) State Transition Matrix $(\mathcal{P})$*:* For the MDP process, the state transition matrix characterizes the environment dynamics through the probabilistic transitions between the present states to the next state. Thus, for scenario ν, the state transition probability at τ_i is given by

$$\begin{aligned} &Pr\left\{s_\nu(\tau_{i+\delta}) \middle| s_\nu(\tau_i), a_\nu(\tau_i)\right\} \\ &= Pr\left\{\left\{F^1(\tau_{i+\delta}), F^2(\tau_{i+\delta}), F^3(\tau_{i+\delta})\right\} \right| \\ &\quad \left(\left\{F^1(\tau_i), F^2(\tau_i), F^3(\tau_i)\right\}, a_\nu(\tau_i)\right)\Big\} \end{aligned} \tag{43}$$

where $\{F^1(\tau_i), F^2(\tau_i), F^3(\tau_i)\}$ is the current state of VU at τ_i that takes action $a_\nu(\tau_i)$.

We assume that the state transition probability expression based on $F^1(\tau_i)$, $F^2(\tau_i)$, and $F^3(\tau_i)$ can be considered as independent events, hence (43) can be rewritten as:

$$\begin{aligned} &Pr\{s_\nu(\tau_{i+\delta}) | s_\nu(\tau_i), a_\nu(\tau_i)\} \\ &= Pr\left\{F^1(\tau_{i+\delta}) \middle| F^1(\tau_i), a_\nu(\tau_i)\right\} \\ &\quad \cdot Pr\left\{F^2(\tau_{i+\delta}) \middle| F^2(\tau_i), a_\nu(\tau_i)\right\} \\ &\quad \cdot Pr\left\{F^3(\tau_{i+\delta}) \middle| F^3(\tau_i), a_\nu(\tau_i)\right\} \end{aligned} \tag{44}$$

where each term is based upon (39)-(41). For example if $F^1(\tau_i) = 0$ and $F^1(\tau_{i+\delta}) = 1$, then $Pr\{F^1(\tau_{i+\delta}) \mid F^1(\tau_i), a_\nu(\tau_i)\} = P^{F^1}_{(0,1)}(a_\nu(\tau_i))$. Detailed analysis of this probability values is given below.

In (39) the four probabilistic transitions for the binary-valued function $F^1(\cdot)$ are set. As shown in (35), $F^1(\cdot)$ becomes 1, if the computation offloading process fails to follow the sojourn time constraint; on the other hand, it becomes 0, if the process follows the constraint. Two probability values $P^{F^1}_{(0,1)}(a_\nu(\tau_i))$ and $P^{F^1}_{(1,0)}(a_\nu(\tau_i))$ model the behavior of $F^1(\cdot)$ based upon the action taken. These transitions can depend upon several factors, including the number of VUs assigned to the selected ENs, the available sojourn time value, which differs for different ENs, the offloading amount, etc. Modeling the exact nature of these transitions can be hard; we resort to exponential distribution functions for modeling the behavior of $F^1(\cdot)$.

In case the mth VU in scenario ν selects the nth EN, through the action $a_\nu(\tau_i)$ we define:

$$P^{F^1}_{(0,1)}(a_\nu(\tau_i)) = \begin{cases} 0 & \text{if } \alpha_{\rho_m}(\tau_i) = 0 \\ 1 - \exp(-\lambda_1(\tau_i)) & \text{else} \end{cases} \tag{45}$$

where $\lambda_1(\tau_i) = K_{11} \cdot \alpha_{\rho_m}(\tau_i) + K_{12} \cdot V^n_m(\tau_i) + K_{13}/T^{soj}_{m,n}(\tau_i)$ is a parameter modeling the slope of the exponential function and is determined from the action $a_\nu(\tau_i)$. According to (45), if the selected action is characterized by $\alpha_{\rho_m}(\tau_i) = 0$, the possibility of the failure of offloading constraint becomes zero. The value of $\lambda_1(\tau_i)$ depends upon several factors. In particular, if the action performed by the mth VU is characterized by high $\alpha_{\rho_m}(\tau_i)$, if VU selects the EN having a higher number of VUs requesting services (i.e., large $V^n_m(\tau_i)$), or VU-EN pair is characterized by the low sojourn time, the value of $\lambda_1(\tau_i)$ can increase. As a result, the probability that $F^1(\cdot)$ changes its state from 0 to 1 (i.e., failure of offloading constraint) becomes high, which can be emphasized in (45). K_{11}, K_{12} and K_{13} are weighting coefficients assigning proper weights to each of these parameters.

$P^{F^1}_{(1,0)}(a_\nu(\tau_i))$ models the case where, with the selected action $a_\nu(\tau_i)$, the VU is able to satisfy the offloading time constraint where:

$$P^{F^1}_{(1,0)}(a_\nu(\tau_i)) = \begin{cases} 1 & \text{if } \alpha_{\rho_m}(\tau_i) = 0 \\ \exp(-\lambda_2(\tau_i)) & \text{else} \end{cases} \tag{46}$$

where $\lambda_2(\tau_i) = K_{21} \cdot \alpha_{\rho_m}(\tau_i) + K_{22} \cdot V^n_m(\tau_i) + K_{23}/T^{soj}_{m,n}(\tau_i)$ is a parameter modeling the slope of the exponential function and is determined from the action $a_\nu(\tau_i)$. This corresponds to say that, if the selected action is characterized by $\alpha_{\rho_m}(\tau_i) = 0$, VUs offloading time becomes zero and, as a result, it will satisfy the sojourn time constraint. Also from the expression of $\lambda_2(\tau_i)$ and (46), it can be seen that, when increasing $\alpha_{\rho_m}(\tau_i)$ and $V^n_m(\tau_i)$, the probability that the mth VU respects the sojourn time constraint is reduced. The reduced value of $T^{soj}_{m,n}(\tau_i)$ between the VU-EN pair can also reduce the chances that VU respects the sojourn time constraint. Here, K_{21}, K_{22} and K_{23} are weighting coefficients.

The second function $F^2(\cdot)$ models the VUs behavior with respect to the task latency constraint, where each VU needs to perform the task processing within the task latency requirements. In this case, $P^{F^2}_{(0,1)}(a_\nu(\tau_i))$ defines the probability that VU fails to satisfy the task latency constraint for a selected action $a_\nu(\tau_i)$

and is given by:

$$P^{F^2}_{(0,1)}(a_\nu(\tau_i)) = \begin{cases} 1 & \text{if } \alpha_{\rho_m}(\tau_i) = 0 \\ \exp(-\lambda_3(\tau_i)) & \text{else} \end{cases} \quad (47)$$

where $\lambda_3(\tau_i) = K_{31} \cdot T_{\rho_m} + \frac{1}{K_{32}+\alpha_{\rho m}(\tau_i)(1-\alpha_{\rho m}(\tau_i))} + \frac{K_{33}}{V^n_m(\tau_i)}$ is a parameter modeling the slope of the exponential function and is determined from the action $a_\nu(\tau_i)$. This corresponds to say that, with its limited resources, if a VU performs the task processing by itself without offloading any data towards ENs, it always fails to satisfy the task latency requirements. In addition, if we have a strict task latency requirement (T_{ρ_m}), and the selected EN has already a large number of VUs ($V^n_m(\tau_i)$) requesting services, this results in increasing the failure probability of the task latency constraint.

If any VU offloads a very small percentage of data towards an EN, the local computation time required for processing the remaining task can be high. On the other hand, if any VU offloads a larger amount of data toward an EN, it is possible to have a higher offloading time mainly because of unreliable channel conditions, limited EN resources, and other competing VUs. The behavior of $P^{F^2}_{(0,1)}(a_\nu(\tau_i))$ concerning the offloading parameter $\alpha_{\rho_m}(\tau_i)$ is modeled as a square function for accommodating these facts. In the end, K_{31} and K_{33} define the weights assigned to latency and competing vehicles parameters, while K_{32} avoid having infinite in the second term.

$P^{F^2}_{(1,0)}(a_\nu(\tau_i))$ models the VUs' chances of satisfying the task latency requirements and is defined as,

$$P^{F^2}_{(1,0)}(a_\nu(\tau_i)) = \begin{cases} 0 & \text{if } \alpha_{\rho_m}(\tau_i) = 0 \\ 1-\exp(-\lambda_4(\tau_i)) & \text{else} \end{cases} \quad (48)$$

where $\lambda_4(\tau_i) = K_{41} \cdot T_{\rho_m} + \frac{1}{K_{42}+\alpha_{\rho m}(\tau_i)(1-\alpha_{\rho m}(\tau_i))} + \frac{K_{43}}{V^n_m(\tau_i)}$ is a parameter modeling the slope of the exponential function and is determined from the action $a_\nu(\tau_i)$. In case VU does not offload any data, it is not able to satisfy the task latency requirements. On the other hand, the behavior of $P^{F^2}_{(1,0)}(a_\nu(\tau_i))$ will be based upon the offloading parameter, number of competing VUs, and the task latency requirements. K_{41} and K_{43} are weighting coefficients, while K_{42} avoid to have infinite in the second term.

The third function, $F^3(\cdot)$, models the VU behavior in terms of energy constraint. If the overall offloading process energy becomes higher than the energy required to compute the complete task locally, the offloading process becomes inefficient. $P^{F^3}_{(0,1)}(a_\nu(\tau_i))$ gives the probability that the VU fails to satisfy the energy constraint for a selected action $a_\nu(\tau_i)$ and is defined as,

$$P^{F^3}_{(0,1)}(a_\nu(\tau_i)) = \begin{cases} 0 & \text{if } \alpha_{\rho_m}(\tau_i) = 0 \\ 1-\exp(-\lambda_5(\tau_i)) & \text{else} \end{cases} \quad (49)$$

where, $\lambda_5(\tau_i) = K_{51} \cdot \alpha_{\rho_m}(\tau_i) + K_{52} V^n_m(\tau_i)$ is a parameter modeling the slope of the exponential function and is determined from the action $a_\nu(\tau_i)$. If the mth VU offloads a large amount of data towards the EN with more $V^n_m(\tau_i)$, with selected action $a_\nu(\tau_i)$, there is a high chance that the offloading process energy becomes higher than the local computation energy. However, if VU does not offload any data towards EN, it always follows the energy constraint. Here, K_{51} and K_{52} are weighting coefficients.

$P^{F^3}_{(1,0)}(a_\nu(\tau_i))$ models the chances that VU is satisfying the energy constraint based upon the selected action $a_\nu(\tau_i)$:

$$P^{F^3}_{(1,0)}(a_\nu(\tau_i)) = \begin{cases} 1 & \text{if } \alpha_{\rho_m}(\tau_i) = 0 \\ \exp(-\lambda_6(\tau_i)) & \text{else} \end{cases} \quad (50)$$

where $\lambda_6(\tau_i) = K_{61} \cdot \alpha_{\rho_m}(\tau_i) + K_{62} V^n_m(\tau_i)$ is a parameter modeling the slope of the exponential function and is determined from the action $a_\nu(\tau_i)$. The chances that VU satisfies the energy constraint reduce with the increasing of $\alpha_{\rho_m}(\tau_i)$ and $V^n_m(\tau_i)$. K_{61} and K_{62} are weighting coefficients.

By using (45)-(50), the transition probability matrices for $F^1(\cdot)$, $F^2(\cdot)$, and $F^3(\cdot)$ can be determined. In the following Section, we define a value iteration algorithm for solving the MDP.

IV. MDP-Based Joint Network Selection and Computation Offloading

In the previous section, the elements of the MDP model are presented. By solving the proposed MDP model, VUs can find a proper EN and the offloading amount able to minimize the overall latency and the energy consumed during the task processing operations. The solutions set can be defined as a policy function $\pi_\nu = \{\pi_\nu(s_\nu(\tau_i+\delta)), \forall\delta\}$ that maps every state $s_\nu \in \mathcal{ST}$ to action $a_\nu \in \mathcal{AS}$. Selecting different actions can result in different policy functions, where the aim is to find an optimal policy that corresponds to the minimum delay and energy cost during vehicular task processing. For every policy π_ν, a value function $V_{\pi_\nu}(s_\nu(\tau_i))$, corresponding to a state $s_\nu(\tau_i)$ can be defined for analyzing its performance. In general, $V_{\pi_\nu}(s_\nu(\tau_i))$ corresponds to an expected value of a discounted sum of total reward received by following the policy π_ν from state $s_\nu(\tau_i)$, and can be defined as:

$$V_{\pi_\nu}(s_\nu(\tau_i)) = \mathbb{E}\left\{\sum_{\delta=0}^{\Delta} \gamma^\delta R\left(s_\nu\left(\tau_i+\delta\right), \pi_\nu\left(s_\nu(\tau_i+\delta)\right)\right)\right\}$$

where $\gamma \in [0,1]$ is the discount factor, $R(s_\nu(\tau_i+\delta), \pi_\nu(s_\nu(\tau_i+\delta)))$ is the immediate reward received for following the policy π_ν at time $\tau_i+\delta$ from the state $s_\nu(\tau_i+\delta)$, Δ is the maximum number of steps considered during the MDP evaluation, i.e., episode length, and $\mathbb{E}\{\cdot\}$ corresponds to the expected value. Thus, the value function analyzes the particular policy function by assigning a numeric value to each state, and can be utilized to compare the performance of the different policies. In the end, the following optimization problem can be formulated in order to be able to find the best possible policy function associated with state $s_\nu(\tau_i)$:

$$V(s_\nu(\tau_i)) = \min_{\pi_\nu \in \Pi_\nu} V_{\pi_\nu}(s_\nu(\tau_i)) \quad (51)$$

where, Π_ν corresponds to the set of policy functions that can be explored.

As shown by many works (e.g., [13], [40]), the problem defined in (51), can converge into a Bellman optimality equation given by:

$$V(s_\nu(\tau_i)) = \min_{a_\nu(\tau_i)\in A_\nu(\tau_i)} \Bigg\{ R(s_\nu(\tau_i), a_\nu(\tau_i)) + \gamma \sum_{s_\nu(\tau_i+\delta)\in \mathcal{ST}} Pr\left\{s_\nu(\tau_i+\delta) \mid s_\nu(\tau_i), a_\nu(\tau_i)\right\} V(s_\nu(\tau_i+\delta))\Bigg\} \tag{52}$$

Different approaches can be used to solve the problem in (52); however, the value iteration approach is widely known for its fast convergence and easy implementation [41]. Therefore, below we present a value iteration approach aimed at solving the MDP designed in the previous section for finding an optimal policy that corresponds to the minimization of a task processing time and energy during offloading process over VNs.

The value iteration method allows finding an optimal policy and value function for the MDP models. The Algorithm 1 describes the steps involved during the value iteration process. For every scenario ν, the process begins by initializing the values of each state to ∞ and iteration count (it) to 0 (Line 2). For each state-action pair, the state value is determined by using (53) (Line 5). In the end state value and a corresponding optimal policy $(\pi^*_\nu(s_\nu(\tau_i)))$ associated with state s_ν is determined by using (54) and (55) (Lines 7–8). The iterative process continues till the change in all states values becomes less than the predefined convergence parameter ϵ (Lines 10–14). In the end, the algorithm returns the set of optimal policy functions $\{\pi^*_\nu\}$ associated with all possible scenarios in which VUs can find themselves over the road (Line 16).

The time complexity of the traditional value iteration process can be estimated to be equal to $O(\Delta|\mathcal{ST}|\cdot|\mathcal{AS}|)$ with Δ being the maximum number of time steps considered, $|\mathcal{ST}|$ state space dimension, and $|\mathcal{AS}|$ representing the action space. With the involvement of $\bar{N}$ scenarios, the time complexity expression becomes $O(\bar{N}\cdot\Delta|\mathcal{ST}|\cdot|\mathcal{AS}|)$. The considered scenario-based modeling can reduce the state and action space dimensions significantly by limiting the number of VUs per scenario compared to the one-shot approaches where all VUs are considered altogether. Especially for the case of VNs, such an approach can be beneficial given the importance of VUs' local environments in the decision-making process (i.e., nearby VUs can influence the VUs' decision-making compared with the other VUs that are located far away from it). Additionally, time-dependent state transition probabilities can reduce the overall uncertainty in the MDP process. It should be noticed that $\bar{N}$, i.e., the considered number of VUs scenarios, can impact the performance of the MDP process. On one side, a smaller $\bar{N}$, corresponding to a limited set of parameters, can impact the MDP model performance due to additional uncertainties. On the other side, with a bigger $\bar{N}$, the computational complexity can be higher with improved performance.

Algorithm 1: MDP Value Iteration.

```
Input: $\epsilon, \gamma, \bar{N}, S_\nu, A_\nu, Pr$
Output: $\{\pi^*_\nu\}$
1: for $\nu \in \bar{N}$ do
2:   Initialize $it = 0$, $V^0(s_\nu(\tau_i)) = \infty, \forall s_\nu(\tau_i)$
3:   for $s_\nu(\tau_i) \in S_\nu$ do
4:     for $a_\nu(\tau_i) \in A_\nu$ do
5:
         $V^{it+1}(s_\nu(\tau_i), a_\nu(\tau_i)) \leftarrow R(s_\nu(\tau_i), a_\nu(\tau_i)) + \gamma \sum_{s_\nu(\tau_i+\delta)\in S_\nu} Pr(s_\nu(\tau_i+\delta) \mid s_\nu(\tau_i), a_\nu(\tau_i)) v^{it}(s_\nu(\tau_i+\delta))$   (53)
6:     end for
7:
         $V^{it+1}(s_\nu(\tau_i)) = \min_{a_\nu(\tau_i)} V^{it+1}(s_\nu(\tau_i), a_\nu(\tau_i))$   (54)
8:
         $\pi^*_\nu(s_\nu(\tau_i)) = \operatorname{argmin}_{a_\nu(\tau_i)} V^{it+1}(s_\nu(\tau_i), a_\nu(\tau_i))$   (55)
9:   end for
10:  if any $|v^{it+1}(s_\nu(\tau_i) - v^{it}(s_\nu(\tau_i)| > \epsilon$ then
11:    $it = it + 1$
12:  else
13:    return $\pi^*_\nu = \{\pi^*_\nu(s_\nu(\tau_i))\}$
14:  end if
15: end for
16: return $\{\pi^*_\nu\}$
```

A. Benchmark Approaches

For comparing the proposed MDP model performance, the following benchmark methods are considered:

- *Minimum distance VU-EN assignment based approach (MDA):* In this approach, VUs are always assigned to the EN, which is at a minimum distance from them. Also, VUs prefer to offload a complete task towards selected EN. Thus, $\forall m$,

$$a_{m,n}(\tau_i) = 1 \iff n = \operatorname*{argmin}_{n\in N}\{d_{m,n}(\tau_i)\} \tag{56}$$

  Though this approach can potentially reduce the overall task communication delay and energy, high handovers requirements and questionable energy performance can reduce the offloading performance.
- *Maximum sojourn time based VU-RSU assignment approach (MSA):* In this method, VUs prefer to offload their task towards RSUs having the highest sojourn time, hence:

$$a_{m,n}(\tau_i) = 1 \iff n = \operatorname*{argmax}_{n\in N}\{D_{m,n}(\tau_i)\} \tag{57}$$

  This approach can reduce the number of handover requirements however, the computation/communication delay and energy performance might not be optimal.
- *MDP-based network selection with static offloading policy (MDP-NS):* To show the impact of a joint network selection and offloading optimization, here we consider an MDP-based network selection decision optimization with

TABLE II
SIMULATION PARAMETERS

MBS Coverage (d^M)	500 m
RSU Coverage (d_n)	25 m
Task Size ($D_{\rho m}$)	5 MB
Task Results ($D^r_{\rho m}$)	($D_{\rho m}/5$) MB
Required Task Latency ($T_{\rho m}$)	2 s
VU Computation Cap. ($c_m \cdot f_m$)	18 GFLOPS
RSU Computation Cap. ($c^R_n \cdot f^R_n$)	45 GFLOPS
MBS Computation Cap. ($c^M \cdot f^M$)	150 GFLOPS
RSU Height (h^R_n)	4 m
MBS Height (h^M)	10 m
CPU Cores ($\mathcal{L}_n$)	4
Task Proc. Requirements $\Omega_{\rho m}$	8 GFLOPS per MB
RSU Bandwidth (B^R_n)	80 MHz
BS Bandwidth (B^M)	1 GHz
VU Energy ($P_{c,m}, Pt_m, Pr_m$)	(0.9, 1.3, 1.1) W
RSU Energy ($P_{c,n}, Pt_n, Pr_n$)	(1.2, 1.3, 1.1) W
Weighting Coefficients (w_1, γ_1, γ_2)	(0.6, 0.5, 0.5)

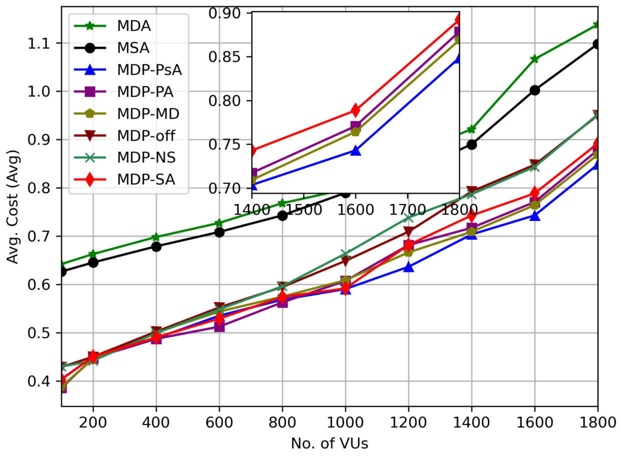

Fig. 3. Cost Function.

static offloading process. In particular, the MDP process (i.e., action space) is adapted to the network selection only while considering a static offloading policy with $\alpha_{\rho_m}(\tau_i) = 0.5, \forall i, m$.

- *MDP-based offloading with static network selection policy (MDP-Off):* In this case, the offloading decisions are optimized, while a static network selection is considered. In particular, VUs are considered to select the nearest EN while offloading with an optimal policy generated through the MDP process of Algorithm 1.

In the following, MDP-MD, MDP-PA, MDP-SA, and MDP-PsA stand for the MDP with minimum distance assignments, probabilistic assignments, sojourn time based assignments and the position-based assignments of nearby VUs, respectively.

V. Numerical Results

The proposed MDP model and corresponding value iteration algorithm is evaluated over a Python-based simulator, using ML-related libraries such as NumPy, Pandas, and Matplotlib. The main simulation parameters are listed in Table II. In this work, we have considered that 80 RSUs with $h_n^R = 3$ m are located alongside the road network in the MBS coverage area. The number of VUs is between 200 to 1800 with $p_a = 0.2$. Each VU travels with a variable speed based upon the intelligent mobility model, with parameters $\vec{v}_{\max} = 15\, m/s, s_{\min} = 2\, m, a_{\max} = 0.7\, m/s^2, b_{\max} = 1.5\, m/s^2, t_r = 2\, s$. The background noise power $\sigma = -110$ dBm is considered.

Also, each RSU can serve up to $K^{\max} = 12$ VUs. Additionally, the communication channel parameters are $\beta_0 = -25$ dB, and $\theta = 2.5$. The RSU switching parameters include switching time $T_{sw,n} = 25$ ms, and switching power $P_{sw,n} = 0.2$ W. Also, when the nth EN is operating in the standby mode, the standby power is $P_{sd,n} = 0.42$ W. The power consumed for the basic circuit operations is $P_{0,n} = 0.5$ W. The VUs scenarios are based upon $l_g = 3.3$ m, $\mathcal{E}^{\max} = 4$, and $\mathcal{NV}_{\max} = 36$. Other MDP parameters include the set of weighing coefficients given by, $[K_{11}, K_{12}, K_{13}, K_{21}, K_{22}, K_{23}] = [0.5, 0.07, 0.4, 0.5, 0.07, 0.4]$, $[K_{31}, K_{32}, K_{33}, K_{41}, K_{42}, K_{43}] = [0.08, 0.6, 0.5, 0.08, 0.6, 0.5]$, and $[K_{51}, K_{52}, K_{61}, K_{62}] = [0.5, 0.07, 0.5, 0.07]$. During the value iteration process $\gamma = 0.9$, $\epsilon = 0.01$, $\Lambda = 0.1$ and episode length $\Delta = 100$ are used.

Avg. Latency and Energy Cost with Varying VUs: In Fig. 3, we present the average cost value in terms of the total latency and energy requirements of VUs task processing. By varying the number of VUs, we obtain the performance of different MDP schemes defined before and analyze their performance by comparing the results with the benchmark methods. It can be seen that proposed MDP schemes perform better compared with the benchmark approaches. By analyzing the surrounding environment, different MDP schemes are able to find proper EN and the amount to be offloaded. In particular, with a high number of VUs, the MDP-PsA approach having a better knowledge of the surrounding environments in terms of various distance measures (i.e., the distance between VU and ENs and the distance before it passes through the EN coverage range), performs better than the other schemes. The superiority of the MDP-PsA approach can be visualized through the zoomed version of the plot. The two benchmark MDP methods (MDP-Off and MDP-NS) have worse performance compared to the joint optimization-based approaches, mainly due to the static policies. This highlights the importance of simultaneously selecting the proper ENs and offloading the proper amount.

Number of RSU handover required during computation offloading: If VU fails to perform the offloading operation (which includes the transmission of VUs data towards selected EN, EN processing, and receiving back the results from EN), before going out from the coverage of the selected EN, an additional handover process/cost is required. In Fig. 4, we present such handover requirements posed by a different set of VUs in terms of the average number of VUs which fail to complete the offloading operation within time limits. It can be verified from this figure that the proposed MDP schemes (in particular MDP-PsA) are performing better compared to the other benchmark methods in terms of a reduction in the overall handover requirements. Thus by avoiding the number of handover requirements, MDP schemes can reduce the service provisioning costs over vehicular environments. The benchmark MDP methods, in

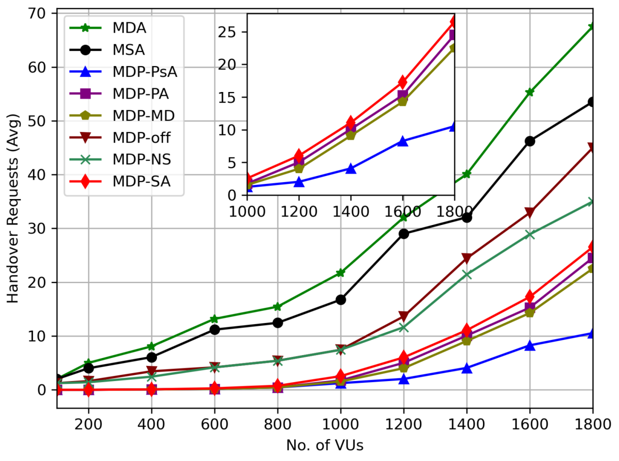

Fig. 4. Percentage of VUs with Handover Requirements.

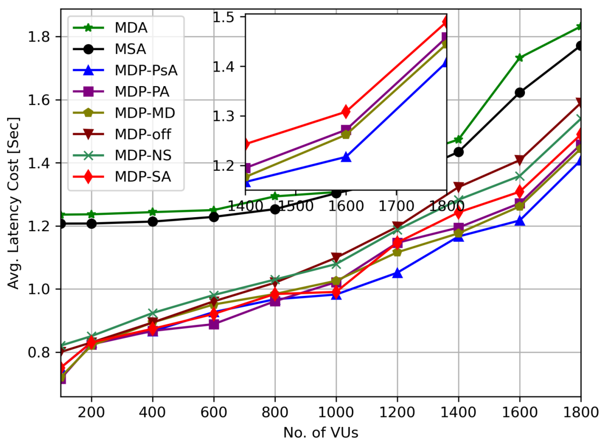

Fig. 6. Avg. Latency Cost.

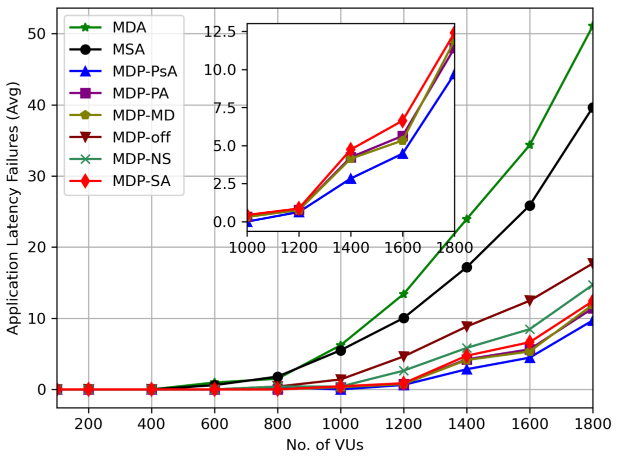

Fig. 5. Percentage of VUs with service time constraint violation.

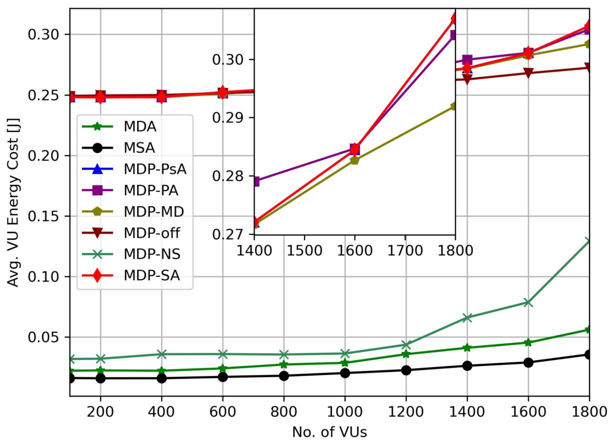

Fig. 7. Avg. VU Energy Cost.

particular MDP-Off, suffer from higher handover requests due to the imperfect offloading decisions compared to the other MDP methods.

Number of service time constraint failures: In general, VUs application latency requirements need to be respected during task processing operations, failure of which can reduce the overall QoS. In Fig. 5, we provide the percentage of VUs that fails to satisfy the application latency requirement constraint in (25). The proposed MDP approaches are able to reduce such failures effectively and can be vital for enabling latency-critical services over VN. Similar to the previous cases, the MDP-PsA approach outperforms the other MDP schemes and can be seen through the zoomed version of the plot. The MDP-NS and MDP-Off methods induce higher latency costs, and their performance suffers with more service latency failures than the other MDP approaches.

Task Completion Latency: To have a better understanding of overall latency requirements, in Fig. 6, we present the performance of different schemes in terms of average latency requirements during the task processing operations. This figure shows the overall reduction of latency cost for VUs task processing operations. Through a proper understanding of the nearby environment parameters (e.g., competing VUs, available RSUs, mobility characteristics), the MDP schemes, in particular MDP-PsA approach, can determine the proper EN and the offloading amount for having better performance. Optimizing only the network selection or offloading decisions through the MDP-NS and MDP-Off methods cannot guarantee optimal performances and suffers from higher latency requirements.

Average Energy Consumption: In the following Figs. 7 and 8, we present the performance of different schemes in terms of average energy requirements. Fig. 7 presents the average amount of energy cost over VUs, which includes the local computation, data transmission, and reception costs. The benchmark approaches do not perform any local computation, due to which they have slightly better performance in terms of VUs energy consumption. However, as shown in Fig. 8, both benchmark methods add large energy costs over ENs. On the other hand, with proper EN selection, and proper offloading decisions, all MDP schemes are having better energy performances over EN. Also, as shown in Fig. 3, the overall performance of the MDP process in terms of joint latency and energy cost is better compared with the benchmark approaches. The joint energy performance of the MDP-Off and MDP-NS methods suffers from imperfect decision makings and impacts the overall costs shown in Fig. 3.

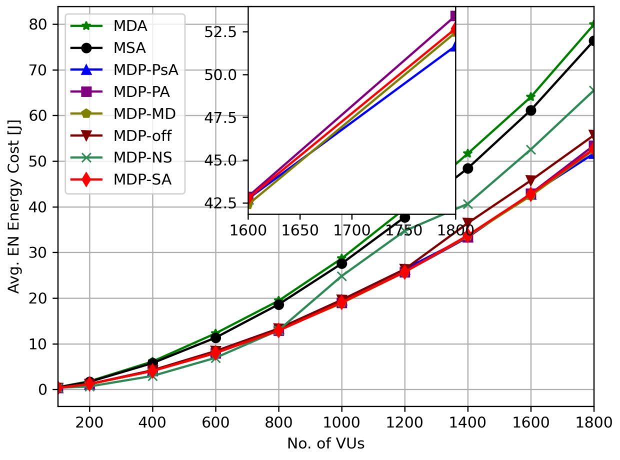

Fig. 8. Avg. EN Energy Cost.

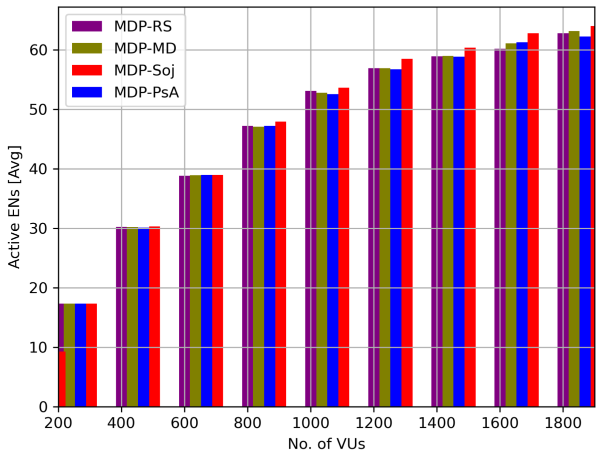

Fig. 9. Avg. Number of Active ENs.

Average Number of Active ENs: In Fig. 9, we have presented the average number of active ENs for varying numbers of VUs. In the beginning with a limited number of VU density, only a limited number of ENs are active. With most of the ENs being inactive, the overall energy cost can be reduced compared with the traditional approaches with all ENs being active. As VU density increases, the active ENs increase for satisfying all VUs service requests. This allows reducing the total number of service failures. With this and previous results, it can be validated that the proposed methods are able to adapt the ENs energy resources according to the VUs demands limiting the EN energy costs along with the potential service failures.

VI. Conclusion

In this work, we considered the joint optimization of network selection and task offloading through a proper minimization of delay and energy for a VEC offloading system. For solving such a complex problem over a highly uncertain vehicular environment, we have proposed a MDP approach by analyzing different vehicular scenarios. The proposed MDP model considers the changing vehicular environment while making the decisions of EN selection and offloading portion. A value iteration-based method is used for solving the proposed MDP model by finding the optimal policy to be followed by each VU in the different scenarios. The simulation results show the superiority of the proposed scheme over various benchmark methods. One of the most prominent contributions is that of having considered the joint network selection and computation offloading problem while jointly minimizing latency and energy costs with additional energy-saving mechanisms at the edge infrastructure. Such studies were not present in the current literature and thus can motivate future readers to investigate it further. However, with additional granularities and joint decision-making processes, the problem becomes extremely complex to be solved through the traditional approaches. For this, we have proposed a novel MDP model with time-dependent state transition probabilities reducing the overall instability. However, since the MDP approach could become very complex in case of a large parameter set, in this work, we have exploited the local vehicular communication modes in the MDP process for improving the overall performance. This can motivate future readers to investigate the proposed solution methods in these directions.

References

[1] D. C. Nguyen et al., “6G Internet of Things: A comprehensive survey,” *IEEE Internet Things J.*, vol. 9, no. 1, pp. 359–383, Jan. 2022.

[2] J. Zhao, Q. Li, Y. Gong, and K. Zhang, “Computation offloading and resource allocation for cloud assisted mobile edge computing in vehicular networks,” *IEEE Trans. Veh. Technol.*, vol. 68, no. 8, pp. 7944–7956, Aug. 2019.

[3] Y. C. Hu, M. Patel, D. Sabella, N. Sprecher, and V. Young, “Mobile edge computing-a key technology towards 5G,” ETSI, Sophia Antipolis, France, White Paper 11, Sep. 2015. [Online]. Available: https://www.etsi.org/images/files/etsiwhitepapers/etsi_wp11_mec_a_key_technology_towards_5g.pdf

[4] A. B. De Souza, et al., “Computation offloading for vehicular environments: A survey,” *IEEE Access*, vol. 8, pp. 198214–198243, 2020.

[5] M. Li, J. Gao, L. Zhao, and X. Shen, “Deep reinforcement learning for collaborative edge computing in vehicular networks,” *IEEE Trans. Cogn. Commun. Netw.*, vol. 6, no. 4, pp. 1122–1135, Dec. 2020.

[6] S. Wang, J. Li, G. Wu, H. Chen, and S. Sun, “Joint optimization of task offloading and resource allocation based on differential privacy in vehicular edge computing,” *IEEE Trans. Comput. Social Syst.*, vol. 9, no. 1, pp. 109–119, Feb. 2022.

[7] Y. Sun et al., “Adaptive learning-based task offloading for vehicular edge computing systems,” *IEEE Trans. Veh. Technol.*, vol. 68, no. 4, pp. 3061–3074, Apr. 2019.

[8] Y. Liu, H. Yu, S. Xie, and Y. Zhang, “Deep reinforcement learning for offloading and resource allocation in vehicle edge computing and networks,” *IEEE Trans. Veh. Technol.*, vol. 68, no. 11, pp. 11158–11168, Nov. 2019.

[9] Z. Ning, P. Dong, X. Wang, J. J. Rodrigues, and F. Xia, “Deep reinforcement learning for vehicular edge computing: An intelligent offloading system,” *ACM Trans. Intell. Syst. Technol.*, vol. 10, no. 6, Nov. 2019, Art. no. 60.

[10] S. S. Shinde, A. Bozorgchenani, D. Tarchi, and Q. Ni, “On the design of federated learning in latency and energy constrained computation offloading operations in vehicular edge computing systems,” *IEEE Trans. Veh. Technol.*, vol. 71, no. 2, pp. 2041–2057, Feb. 2022.

[11] X. Wang, Z. Ning, S. Guo, and L. Wang, “Imitation learning enabled task scheduling for online vehicular edge computing,” *IEEE Trans. Mobile Comput.*, vol. 21, no. 2, pp. 598–611, Feb. 2022.

[12] W. Zhan et al., “Deep-reinforcement-learning-based offloading scheduling for vehicular edge computing,” *IEEE Internet Things J.*, vol. 7, no. 6, pp. 5449–5465, Jun. 2020.

[13] X. Zhang, J. Zhang, Z. Liu, Q. Cui, X. Tao, and S. Wang, “MDP-based task offloading for vehicular edge computing under certain and uncertain transition probabilities,” *IEEE Trans. Veh. Technol.*, vol. 69, no. 3, pp. 3296–3309, Mar. 2020.

[14] M. L. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. Hoboken, NJ, USA: John Wiley Sons, Inc., 2005.

[15] L. Liu, C. Chen, Q. Pei, S. Maharjan, and Y. Zhang, “Vehicular edge computing and networking: A survey,” *Mobile Networks Appl.*, vol. 26, no. 3, pp. 1145–1168, Jun. 2021.

[16] W. Fan, J. Liu, M. Hua, F. Wu, and Y. Liu, “Joint task offloading and resource allocation for multi-access edge computing assisted by parked and moving vehicles,” *IEEE Trans. Veh. Technol.*, vol. 71, no. 5, pp. 5314–5330, May 2022.

[17] P. Qin, Y. Fu, G. Tang, X. Zhao, and S. Geng, “Learning based energy efficient task offloading for vehicular collaborative edge computing,” *IEEE Trans. Veh. Technol.*, vol. 71, no. 8, pp. 8398–8413, Aug. 2022.

[18] R. Yadav, W. Zhang, O. Kaiwartya, H. Song, and S. Yu, “Energy-latency tradeoff for dynamic computation offloading in vehicular fog computing,” *IEEE Trans. Veh. Technol.*, vol. 69, no. 12, pp. 14198–14211, Dec. 2020.

[19] H. Xiao, J. Zhao, Q. Pei, J. Feng, L. Liu, and W. Shi, “Vehicle selection and resource optimization for federated learning in vehicular edge computing,” *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 8, pp. 11073–11087, Aug. 2022.

[20] S. S. Shinde and D. Tarchi, “Collaborative reinforcement learning for multi-service internet of vehicles,” *IEEE Internet Things J.*, vol. 10, no. 3, pp. 2589–2602, Feb. 2023.

[21] S. S. Shinde and D. Tarchi, “Network selection and computation offloading in non-terrestrial network edge computing environments for vehicular applications,” in *Proc. IEEE 11th Adv. Satell. Multimedia Syst. Conf. 17th Signal Process. Space Commun. Workshop (ASMS/SPSC)*, 2022, pp. 1–8.

[22] A. Bozorgchenani, S. Maghsudi, D. Tarchi, and E. Hossain, “Computation offloading in heterogeneous vehicular edge networks: On-line and off-policy bandit solutions,” *IEEE Trans. Mobile Comput.*, vol. 21, no. 12, pp. 4233–4248, Dec. 2022.

[23] B. Yang, X. Cao, J. Bassey, X. Li, and L. Qian, “Computation offloading in multi-access edge computing: A multi-task learning approach,” *IEEE Trans. Mobile Comput.*, vol. 20, no. 9, pp. 2745–2762, Sep. 2021.

[24] B. Yang et al., “Edge intelligence for autonomous driving in 6G wireless system: Design challenges and solutions,” *IEEE Wireless Commun.*, vol. 28, no. 2, pp. 40–47, Apr. 2021.

[25] Y. Dai, D. Xu, S. Maharjan, and Y. Zhang, “Joint computation offloading and user association in multi-task mobile edge computing,” *IEEE Trans. Veh. Technol.*, vol. 67, no. 12, pp. 12313–12325, Dec. 2018.

[26] J. Sun, Q. Gu, T. Zheng, P. Dong, A. Valera, and Y. Qin, “Joint optimization of computation offloading and task scheduling in vehicular edge computing networks,” *IEEE Access*, vol. 8, pp. 10466–10477, 2020.

[27] S. Li, S. Lin, L. Cai, W. Li, and G. Zhu, “Joint resource allocation and computation offloading with time-varying fading channel in vehicular edge computing,” *IEEE Trans. Veh. Technol.*, vol. 69, no. 3, pp. 3384–3398, Mar. 2020.

[28] Z. Ning et al., “Intelligent edge computing in internet of vehicles: A joint computation offloading and caching solution,” *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 4, pp. 2212–2225, Apr. 2021.

[29] L. Tang, B. Tang, L. Zhang, F. Guo, and H. He, “Joint optimization of network selection and task offloading for vehicular edge computing,” *J. Cloud Comput.*, vol. 10, 2021, Art. no. 23.

[30] Z. Ning et al., “Partial computation offloading and adaptive task scheduling for 5G-enabled vehicular networks,” *IEEE Trans. Mobile Comput.*, vol. 21, no. 4, pp. 1319–1333, Apr. 2022.

[31] Y. Liu et al., “Dependency-aware task scheduling in vehicular edge computing,” *IEEE Internet Things J.*, vol. 7, no. 6, pp. 4961–4971, Jun. 2020.

[32] A. Bozorgchenani, D. Tarchi, and G. E. Corazza, “Mobile edge computing partial offloading techniques for mobile urban scenarios,” in *Proc. IEEE Glob. Commun. Conf.*, 2018, pp. 1–6.

[33] Z. Zhou, P. Liu, Z. Chang, C. Xu, and Y. Zhang, “Energy-efficient workload offloading and power control in vehicular edge computing,” in *Proc. IEEE Wireless Commun. Netw. Conf. Workshops*, 2018, pp. 191–196.

[34] Y. Liu et al., “Joint communication and computation resource scheduling of a UAV-assisted mobile edge computing system for platooning vehicles,” *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 7, pp. 8435–8450, Jul. 2022.

[35] M. Nekoui and H. Pishro-nik, “Fundamental tradeoffs in vehicular ad hoc networks,” in *Proc. 7th ACM Int. Workshop VehiculAr InterNETworking*, 2010, pp. 91–96.

[36] K. Xiong, S. Leng, X. Chen, C. Huang, C. Yuen, and Y. L. Guan, “Communication and computing resource optimization for connected autonomous driving,” *IEEE Trans. Veh. Technol.*, vol. 69, no. 11, pp. 12652–12663, Nov. 2020.

[37] J. X. Xiaopeng Mo, “Energy-efficient federated edge learning with joint communication and computation design,” *J. Commun. Inf. Netw.*, vol. 6, no. 2, pp. 110–124, Jun. 2021.

[38] J. Zhang, H. Guo, J. Liu, and Y. Zhang, “Task offloading in vehicular edge computing networks: A load-balancing solution,” *IEEE Trans. Veh. Technol.*, vol. 69, no. 2, pp. 2092–2104, Feb. 2020.

[39] L. Cesarano, A. Croce, L. D. C. Martins, D. Tarchi, and A. A. Juan, “A real-time energy-saving mechanism in internet of vehicles systems,” *IEEE Access*, vol. 9, pp. 157842–157858, 2021.

[40] G. Yang, L. Hou, X. He, D. He, S. Chan, and M. Guizani, “Offloading time optimization via Markov decision process in mobile-edge computing,” *IEEE Internet Things J.*, vol. 8, no. 4, pp. 2483–2493, Feb. 2021.

[41] N. Balaji, S. Kiefer, P. Novotnỳ, G. A. Pérez, and M. Shirmohammadi, “On the complexity of value iteration,” *Proc. 46th Int. Colloq. Automata, Lang., Program.*, vol. 132, pp. 102:1–102:15, 2019.

Swapnil Sadashiv Shinde (Student Member, IEEE) was born in Pune, India, in 1991. He received the B.Eng. degree in electronics and telecommunication engineering from Pune University, Pune, India, in 2013, the M.Des. degree in communication systems from the Indian Institute of Information Technology Design and Manufacturing, Kancheepuram, India, in 2015, and the M.S. degree in telecommunication engineering from the University of Bologna, Bologna, Italy, in 2020. He is currently working toward the Ph.D. degree with the University of Bologna. From 2015 to 2017, he was a Project Engineer with the Indian Institute of Technology, Kanpur, India. His research interests include the connected vehicles for beyond 5G scenarios, edge computing, reinforcement learning, distributed machine learning, and non-terrestrial networks.

Daniele Tarchi (Senior Member, IEEE) was born in Florence, Italy in 1975. He received the M.Sc. degree in telecommunications engineering and the Ph.D. degree in informatics and telecommunications engineering from the University of Florence, Florence, Italy, in 2000 and 2004, respectively. From 2004 to 2010, he was a Research Associate with University of Florence. From 2010 to 2019, he was an Assistant Professor with the University of Bologna, Bologna, Italy. He is currently an Associate Professor with the University of Bologna. He is the author of around 150 published articles in international journals and conference proceedings. His research interests include wireless communications and networks, satellite communications and networks, edge computing, distributed learning, smart cities, and optimization techniques. He has been involved in several national and international research projects and collaborates with several foreign research institutes. Since 2012, Dr. Tarchi has been an IEEE Senior Member. He is Editorial Board Member for IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, IEEE Open Journal of the Communication Society and *IET Communications*. He has been symposium Co-Chair for IEEE WCNC 2011, IEEE Globecom 2014, IEEE Globecom 2018 and IEEE ICC 2020, and a workshop Co-Chair at IEEE ICC 2015.