MIME: Mobility-Induced Dynamic Matching for Partial Offloading in Vehicular Edge Computing

Mahmuda Akter, Debjyoti Sengupta, Anurag Satpathy, and Sajal K. Das
Department of Computer Science
Missouri University of Science and Technology, Rolla, USA
E-mail: {makter, dsengupta, anurag.satpathy, sdas}@mst.edu

2024 IEEE 49th Conference on Local Computer Networks (LCN) | 979-8-3503-8800-8/24/$31.00 ©2024 IEEE | DOI: 10.1109/LCN60385.2024.10639795

***Abstract*—Autonomous vehicles (AVs) execute compute-intensive control operations like adjusting speed and steering, causing significant energy dissipation and latency due to resource-limited onboard units (OBUs). Offloading these tasks to Roadside Units (RSUs) is a solution, but it faces challenges. First, the stringent latency requirements are impacted by the vehicle's stochastic velocity. Second, allocating limited RSU resources to numerous vehicles within its coverage area is difficult. This paper proposes the MIME framework to address these issues. We use Discrete Fourier transform (DFT) that computes the average velocity over an aperiodic velocity signal extracted from a real-world dataset. For resource allocation, we model it as matching with externalities, using reactive preferences based on vehicle speed and location. We present an efficient, scalable, stable solution, showing a 24.61% and 11.2% reduction in offloading latency and energy for the inD dataset, and a 5.6% and 5.52% reduction for the SUMO dataset.**

***Index Terms*—Vehicular Edge, Road Side Units, Partial Offloading, Matching Theory, Deadline, Latency, Energy Awareness**

I. Introduction

Autonomous Vehicles (AVs) execute computation-intensive control operations like adaptive cruise control and emergency braking [1]. These critical operations rely heavily on the vehicles' real-time velocity and demand strict adherence to quality-of-service (QoS) standards [2]. Failing to maintain these standards severely threatens the safety of passengers and the surrounding environment. AVs' onboard units (OBUs) often lack sufficient resources to execute critical operations in real-time, which can be problematic [3] and, in some cases, catastrophic. To address this issue, roadside units (RSUs) with sufficient computing capabilities have been strategically positioned to aid AVs in conducting real-time computations [4], [5]. Consequently, a viable solution to alleviate the resource constraints of AVs is to collaboratively offload some or all computations to the RSU for real-time processing [6].

At the outset, cooperative offloading appears trivial, yet poses numerous research challenges. This study focuses on addressing three pivotal challenges in vehicular edge. *Firstly*, we integrate a vehicle's velocity variability to estimate its traversal dynamics, such as *sojourn time* [7]. The dynamic nature of a vehicle's velocity is critical in determining the *sojourn time*, which approximates the time window for executing computations. *Secondly*, RSUs possess finite compute resources [8], making resource allocation particularly daunting in urban road networks with dynamically arriving and departing vehicles,

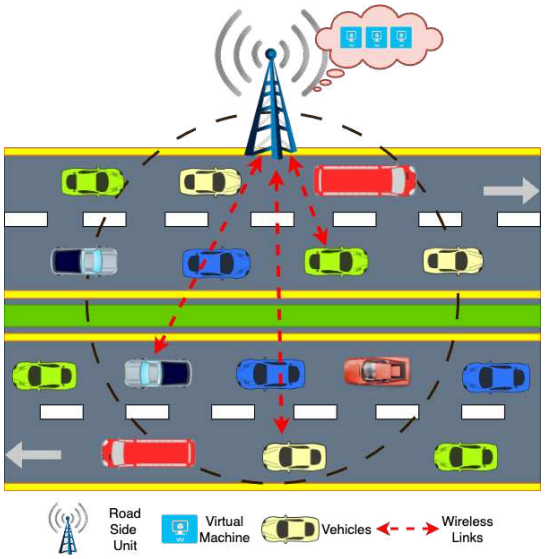


Fig. 1: Vehicular Edge Computing setup with multiple vehicles requiring computations served by the roadside unit (RSU).

proven to be $\mathcal{NP}$-Hard [9]. *Thirdly*, the cooperation between OBU and RSU necessitates an efficient partitioning scheme to execute vehicular operations effectively. Therefore, developing a *polynomial-time*, *scalable*, and *efficient* partial offloading strategy that captures vehicle dynamics and is responsive to the latency requirements of vehicular operations is extremely challenging [8], [10].

This paper presents MIME, an RSU-assisted partial offloading framework with energy awareness to tackle these challenges. The key contributions of this work include:
(1) Propose MIME framework to partially offload vehicular computations with a partition logic that efficiently selects tasks for execution at OBU while offloading the remainder to RSU; and optimally utilize resources from OBU and RSU to manage deadline-specific computations without overburdening RSUs.
(2) Validate MIME realistically by incorporating average variability in vehicular velocities using Discrete Fourier Transform (DFT) analysis on real-world data. Resource assignment employs a matching game with dynamic preferences, ensuring a stable two-sided exchange solution that captures vehicle dynamics, reducing offloading latency and energy consumption.
(3) Evaluate MIME with (i) synthetic traffic data generated using SUMO (Simulation of Urban Mobility), and (ii) real-world traffic data from the inD dataset; and compare its perfor-

mance with three distinct baselines to demonstrate noteworthy reductions in offloading latency (24.61% for inD and 5.6% for SUMO dataset) and energy consumption (11.2% for inD and 5.52% for SUMO dataset).

The remainder of this paper is organized as follows. Section II reviews the literature concerning vehicular edge computing. Section III presents the model and formulates the offloading problem. The proposed MIME approach is detailed in Section IV. Section V presents the experimental results. Lastly, Section VI concludes the paper.

II. Literature Survey

The significance of Vehicular Edge Computing (VEC) is well-established, with many surveys highlighting its critical role. Authors in [11], [12] identify significant challenges within VEC, particularly emphasizing the difficulties associated with offloading for AVs performing real-time computations under time bounds. Given these challenges, we present the literature reviewed from two perspectives: full and partial offloading strategies.

A. Full Offloading Schemes

In this regard, works centered on full offloading are discussed in [13]–[16], focused on minimizing offloading delays, considering both communication and computation overhead. In [13], the authors explored a scenario rife with obstacles where vehicles frequently switched networks due to high mobility and network diversity. They emphasize minimizing communication latency by proactively selecting the most suitable interface. However, given vehicles' dynamic communication resource requirements, relying on a single communication technology proved inadequate. To tackle this challenge, [14] introduced fiber-wireless (FiWi) technology, capable of supporting multiple communication techniques to cater to more vehicles. Similarly, the variation in computational resources among vehicles and RSUs necessitates decision-making to mitigate computational overhead, as discussed in [15]. Here, the authors approached offloading as a multi-user computation offloading game, employing a distributed computation offloading algorithm to determine the equilibrium. Furthermore, the study in [16] aimed at proactively managing road traffic, reshaping the distribution of resource demands to prevent service delays with minimal migration overhead.

Taking into account both communication and computation latencies, [17] proposed a Genetic Algorithm (GA)-driven cooperative three-tier offloading framework involving vehicles, the cloud, and Mobile Edge Cloud (MEC). Meanwhile, the energy consumption of mobile applications during offloading to the RSU remains a substantial concern. In this context, the work in [18] employed evolutionary algorithms to formulate solutions for a multi-objective optimization problem, identifying optimal trade-offs.

B. Partial Offloading Schemes

While full offloading offers advantages, it can become inefficient with high data volumes, resulting in increased latency during data transfer and reduced effectiveness for real-time computational tasks. In such scenarios, partial offloading emerges as a viable solution for minimizing latency. For instance, to alleviate computational overhead, authors in [19] proposed a Non-Orthogonal Multiple Access (NOMA)-Assisted MEC system facilitating partial computation offloading, considering the offloading policy and channel resources of the wireless network. Partial computation offloading also enables the utilization of idle resources in parked vehicles, as suggested in [20]. The authors employed game theory to devise the offloading strategy, while a sub-gradient method was adopted to determine the optimal offloading and resource allocation ratio. Alternatively, the work in [21] utilized computation resources from neighboring vehicles for partial offloading to meet the stringent latency requirements in a 5G environment. The authors addressed interference management using Orthogonal Frequency Division Multiple Access (OFDMA), where data segmentation was iteratively adjusted, and resource allocation was optimized for cellular and Device-to-Device (D2D) links while considering QoS requirements.

The literature review highlights a gap in addressing the impact of varying vehicle mobility on service quality, especially in partial offloading. Additionally, there is a scarcity of research dedicated to simultaneously minimizing latency and energy consumption within a partial offloading framework. Thus, this study introduces a comprehensive technique aimed at addressing these limitations.

III. System Model and Problem Formulation

This section introduces the system model and formalizes the offloading problem, highlighting the overall system goal.

A. Overview of the System

As depicted in Fig. 1, we consider a set of AVs captured as $\mathcal{V} = \{v_1, v_2, v_3, \ldots v_m\}$ moving in a two-lane road network of a city. These vehicles frequently perform compute-intensive operations to provide a seamless experience to the users. A vehicle $v_i \in \mathcal{V}$ has an on-board unit (OBU) with computational capacity c_i (*cycles/bit*) to perform such operations. The OBU is also characterized by a processing frequency f_i (*cycles/sec*). In MIME, we consider a RSU, which is virtualized and comprises 'n' VMs, captured as $\mathcal{M} = \{vm_1, vm_2, \ldots, vm_n\}$. Each $vm_j \in \mathcal{M}$ has a computing capacity uniquely identified as c_j (*cycles/bit*). Similar to the vehicles, each $vm_j \in \mathcal{M}$ operates at a frequency of f_j (*cycles/sec*). Considering heterogeneous VMs is motivated by the operations' diversity, requiring disparate computing resources at the RSU. To leverage the heterogeneous resources of vehicles and VMs, we consider that the amount of data, denoted as d_i (for vehicle v_i), needed to compute a task is two-way splittable, allowing simultaneous computations by the OBU and an assigned VM at the RSU. Additionally, for a vehicle v_i, the tolerable time to perform an action based on local and/or offloaded computations is bounded by $s_i(t)$, which denotes the time a vehicle is expected to remain in the RSU coverage. Therefore, vehicles cooperatively perform computations with the RSU (specifically a

VM). The allocation vector, in this partial offloading setup, is represented by $\boldsymbol{\alpha_{i,j}} = \{\alpha^l_{i,j}, \alpha^o_{i,j} \in \mathbb{R}^+\}$, where the former depicts the fraction executed locally and the latter is the offloaded fraction to RSU. Note that $\alpha^l_{i,j} + \alpha^o_{i,j} = 1$ (refer to Section IV-B).

B. Capturing Vehicular Dynamics: Mobility Model

The vehicular motion in an urban two-lane, bidirectional highway setup follows specific trajectories as depicted in Fig. 2. The mobility model for the vehicles in MIME is motivated by the work in [22]. Note that the RSU is assumed to be stationary and located at coordinates (x_r, y_r) having a coverage radius $\mathcal{R}$. At a specific time instant 't,' the position of a vehicle v_i in the coverage area is captured by a tuple in the Cartesian coordinate system $< x_i(t), y_i(t) >$.

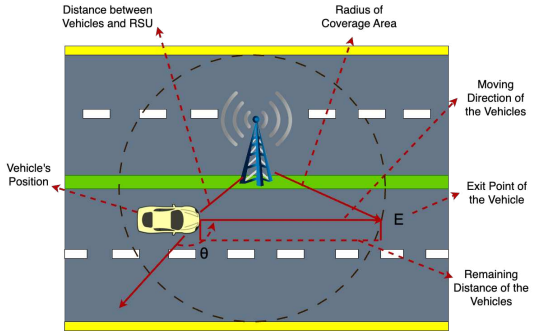


Fig. 2: Vehicle Mobility Model and Corresponding Distance Metrics.

Let E be the exit point of v_i (see Fig. 2), and the vehicles traverse in that direction at an angle θ. The distance of v_i from the RSU, denoted by $\Delta_i(t)$ is computed as

$$\Delta_i(t) = \sqrt{(y_r - y_i(t))^2 + (x_r - x_i(t))^2}. \tag{1}$$

Let $\Omega_i(t)$ be the distance between the vehicle's current position and E. The vehicles are assumed to be uniformly distributed within the RSU coverage having a radius ($\mathcal{R}$). For ease of representation, the vehicle's coordinates are converted to polar $(\Delta_i(t), \theta_i(t))$, and its position is considered to be independent of the traversal direction $\theta_i(t)$. For symmetry, we consider $\theta_i(t) \in [0, \pi]$, and the joint PDF of a vehicle's presence at a specific distance $\Delta_i(t)$ is obtained as

$$f(\Delta_i(t), \theta_i(t)) = \frac{1}{\pi} \times \frac{2\Delta_i(t)}{\mathcal{R}^2} \tag{2}$$

Next, we compute the joint PDF of a vehicle v_i's presence at a distance $\Delta_i(t)$ from the RSU having a remaining distance $\Omega_i(t)$ to exit the coverage area. It can be derived as in Eq. (3). Without loss of generality and for simplicity in the representation, in the rest of the paper, we consider $\Delta_i(t)$, $\theta_i(t)$, $\Omega_i(t)$ as Δ_i, θ_i, and Ω_i, respectively. Thus,

$$f(\Omega_i, \Delta_i) = \frac{\mathcal{R}^2 - \Delta_i^2 + \Omega_i^2}{\pi \mathcal{R}^2 \Omega_i^2 \sqrt{1 - \left(\frac{\mathcal{R}^2 - \Delta_i^2 + \Omega_i^2}{2\Delta_i \Omega_i^2}\right)^2}} \tag{3}$$

By integrating Δ_i the marginal PDF of the remaining distance $\Omega_i \in [0, 2\mathcal{R}]$ to be traversed by v_i is given by

$$f(\Omega_i) = \int_{|\mathcal{R}-\Omega_i|}^{\mathcal{R}} f(\Omega_i, \Delta_i)\, d\Delta_i = \frac{1}{\pi R^2} \sqrt{4\mathcal{R}^2 - \Omega_i^2} \tag{4}$$

The mobility model considers a vehicle traversing at an instantaneous velocity u_i having an average velocity ϑ_i. Thus, the PDF of $s_i(t)$ is derived as

$$f(s_i) = \frac{\vartheta_i}{\pi \mathcal{R}^2} \sqrt{4\mathcal{R}^2 - (\vartheta_i t)^2} \tag{5}$$

Therefore, in MIME, an offloading failure occurs for v_i if the completion time $\mathbb{T}^o_i(\alpha_{i,j}, \Psi_k)$ (see the details in Section III-C), which encompasses communication and computation delays, exceeds the sojourn time. This probability $\mathbb{P}^f_i$ is obtained as

$$\mathbb{P}^f_i = \mathbb{P}(s_i < \mathbb{T}^o_i(\alpha_{i,j}, \Psi_k)) = \int_0^{\mathbb{T}^o_i(\alpha_{i,j}, \Psi_k)} f_{s_i}(t)\, dt$$
$$= \begin{cases} 1 & \mathbb{T}^o_i(\alpha_{i,j}, \Psi_k) \geq \frac{2\mathcal{R}}{\vartheta_i}, \\ \frac{2arcsin(\mathbb{A}) + \mathbb{T}^o_i(\alpha_{i,j}, \Psi_k) u_i \sqrt{1-(\mathbb{B})^2}}{\pi} & 0 \leq \mathbb{T}^o_i(\alpha_{i,j}, \Psi_k) < \frac{2\mathcal{R}}{\vartheta_i} \end{cases} \tag{6}$$

where $\mathbb{A} = \left(\frac{\mathbb{T}^o_i(\alpha_{i,j}, \Psi_k) u_i}{2\mathcal{R}}\right)$ and $\mathbb{B} = \left(\frac{\mathbb{T}^o_i(\alpha_{i,j}, \Psi_k) \vartheta_i}{2\mathcal{R}}\right)$.

C. Communication and Computation Models

As the vehicle enters the coverage area of an RSU, it registers with an information tuple $< x_i, y_i, u_i, \vartheta_i, c_i, f_i, d_i >$. Next, we discuss the communication and computation models.

1) Communication Model: The vehicles are considered to communicate with the RSU using Long Term Evolution (LTE) [23]. Let b capture the channel bandwidth, $p_{i,j}$ and $g_{i,j}$ denote the vehicle's transmission power and channel gain, respectively. The latter is the Rayleigh channel coefficient with complex Gaussian distribution. We do not consider the co-channel interference in this work but consider a free-space propagation path loss model. The experienced data rate $(r_{i,j})$ for a vehicle v_i transmitting to the vm_j can be derived as

$$r_{i,j} = b \log_2 \left[1 + \frac{p_{i,j} \Delta^{-\alpha}_{i,j}(t) |g_{i,j}|^2}{N_0}\right] \tag{7}$$

Then, the communication delay is obtained as

$$\mathbb{T}^c_i(\alpha_{i,j}, \Psi_k) = \left[\alpha^o_{i,j} \times \frac{d_i}{r_{i,j}}\right] \tag{8}$$

where Ψ_k refers to the obtained assignment at a particular iteration 'k.' The procedure to obtain the assignment is detailed in Section IV-B.

2) Local Processing Delay: The local processing delay $\mathbb{T}^l_i(\alpha_{i,j}, \Psi_k)$ is the time expended in processing α^l_i fraction of the data at the OBU and is computed as

$$\mathbb{T}^l_i(\alpha_{i,j}, \Psi_k) = \left[(\alpha^l_{i,j} \times d_i) \times \frac{c_i}{f_i}\right] \tag{9}$$

3) Offloaded Processing Delay: The processing delay $\mathbb{T}_i^p(\alpha_{i,j}, \Psi_k)$ is the time required by v_i to process the offloaded computations at vm_j of the RSU. It is derived as

$$\mathbb{T}_i^p(\alpha_{i,j}, \Psi_k) = \left[(\alpha_{i,j}^o \times d_i) \times \frac{c_j}{f_j} \right] \tag{10}$$

4) Queueing Delay: The vehicular computation may be stalled at a VM due to the following two delays: (1) residual execution of a previously assigned vehicle $v_{i'}$ at the k' iteration, such that $k' < k$, and is reflected as the first term of the following equation; and (2) cumulative offloaded processing delay of vehicles assigned to vm_j from iterations $k'+1$ through $k-1$ with pending execution.

$$\mathbb{T}_i^q(\alpha_{i,j}, \Psi_k) = \max(\mathbb{T}_{i'}^p(\alpha_{i',j}, \Psi_{k'}) - (k-k')T_s, 0) + \sum_{x=k'+1}^{k-1} \mathbb{T}_{i'}^p(\alpha_{i'',j}, \Psi_x) \tag{11}$$

where T_s refers to the sampling interval computed via the Fourier Transform. Refer to Section IV-A for details.

5) Offloading Delay: It is the total time elapsed in communication and computation, derived as

$$\mathbb{T}_i^o(\alpha_{i,j}, \Psi_k) = \mathbb{T}_i^c(\alpha_{i,j}, \Psi_k) + \mathbb{T}_i^p(\alpha_{i,j}, \Psi_k) + \mathbb{T}_i^q(\alpha_{i,j}, \Psi_k) \tag{12}$$

Decision Variable: Before delving into the detailed formulation of the overall problem, we introduce an indicator variable that captures the assignment as follows.

$$\mathcal{I}(v_i, vm_j) = \begin{cases} 1 & \text{If } v_i \text{ offloads } (\alpha_{i,j}^o * d_i) \text{ to } vm_j \\ 0 & \text{Otherwise} \end{cases}$$

D. Energy Model

1) Energy Model of the VMs: Given the power rating ζ_j of vm_j, the energy consumption $\mathbb{E}_j$ at vm_j for processing the offloaded data is obtained as

$$\mathbb{E}_j = \sum_{j=1}^{m} \mathcal{I}(v_i, vm_j) \times \zeta_j \times \mathbb{T}_i^p(\alpha_{i,j}, \Psi_k) \tag{13}$$

2) Energy Model of the Vehicles: A vehicle v_i with a power rating of ζ_i will have the local energy consumption, also obtained as

$$\mathbb{E}_i = \zeta_i \times \mathbb{T}_i^l(\alpha_{i,j}, \Psi_k) \tag{14}$$

E. Problem Formulation

The overall objective of MIME is represented as per Eq. (15a). Constraint (15b) states that a vehicle can offload to at most one VM. Constraints (15c), (15d), and (15e) capture the feasibility of the partitioning logic and the probabilistic expressions. Finally, Constraint (15e) captures the values for the decision variables.

$$\min \max_{\forall i \in [1,m]} \{\mathbb{T}_i^o(\alpha_{i,j}, \Psi_k)\} \tag{15a}$$

$$s.t. \quad \mathcal{I}(v_i, vm_j) \leq 1 \tag{15b}$$

$$\boldsymbol{\alpha_{i,j}} \in \mathbb{R}^+ \tag{15c}$$

$$\mathbf{1}^T \boldsymbol{\alpha_{i,j}} \leq 1 \tag{15d}$$

$$\mathbb{P}_i^s = 1 - \mathbb{P}_i^f \tag{15e}$$

$$\forall\, i \in [1,m],\ \forall\, j \in [1,n] \tag{15f}$$

$\mathcal{NP}$-Hardness: The above optimization problem expressed is proven to be $\mathcal{NP}$-Hard [9], implying that obtaining an optimal solution will be highly computationally intensive. In this paper, we propose an efficient sub-optimal solution that not only generates a *stable* and *scalable* assignment in *polynomial-time* but also considers individual *preferences* of stakeholders.

IV. Proposed MIME Framework

We elaborate on the components and roles of the RSU Controller, illustrated in Fig. 3, which essentially serves as the system's central intelligence. It includes (1) Registry Module comprising VM and Vehicle registries. The former tracks the VMs, including configuration details and availability status. The latter logs the vehicles in the RSU coverage. (2) Preference Generator Module that computes the preferences of vehicles and VMs and feeds them to the matcher. (3) Assignment Module that matches VMs to vehicles. It consists of three sub-components, dynamic virtual machine assignment algorithm (DVMA), which invokes the *matching algorithm* (MA) that assigns vehicles to VMs. Once matched, the partition algorithm (PA) assigns the fraction of computations for the OBU and VM.

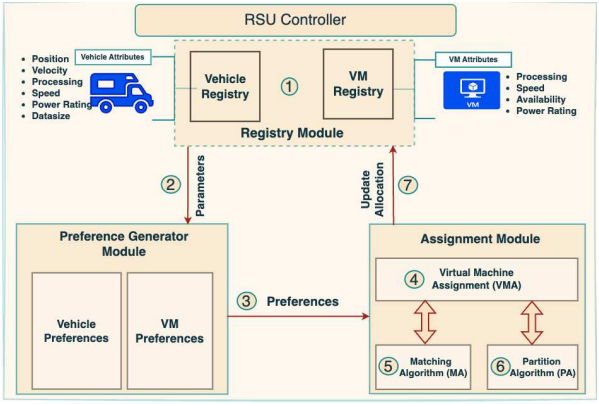


Fig. 3: Workflow of the proposed MIME framework.

A. Mobility-Induced Determination of Sampling Period

Most models in vehicular edge focus on a static snapshot of the vehicle with the assumption of a consistent velocity. However, for a vehicle to maintain a consistent velocity while traversing the coverage area is an unrealistic assumption and depends on various factors. Moreover, the dynamic variation of the velocity introduces many challenges that need to be addressed accurately to predict the vehicle trajectory precisely. (1) Effectively capture the variation in the velocity of the vehicles and their position, which directly impacts the *sojourn time* for the undertaken computations. (2.) The resources at the RSU are limited; it is safe to assume that the number of vehicles is much more than the available VMs, and their *waiting and sojourn time* variation at different instants must be captured in the preferences of vehicles. We use FT (defined below) to extract the frequency distribution of the aperiodic velocity $v(t)$, facilitating the determination of an effective sampling rate and precisely capturing the variable velocity.

Definition 1. ***(Fourier Transform)*** [24]*: The FT of the time-varying velocity signal $v(t)$ captured as $V(\omega)$ is computed as $\int_{-\infty}^{+\infty} v(t)e^{-j\omega t}dt$, where $V(\omega)$ is the frequency counterpart.*

Definition 2. ***(Dirichlet Conditions)*** [24]*: For $V(\omega)$ to exist, $v(t)$ should have a finite number of finite discontinuities and finite maxima and minima. It should also be integrable over time, i.e., $\int_{-\infty}^{+\infty} |v(t)|dt < \infty$.*

For our implementation, we identify discrete data points on the curve $v(t)$ akin to real-world capturing of velocities at specific time instants considering a finite number of samples often termed as the *data window*. Therefore, we adopt the Discrete Fourier Transform (DFT).

Definition 3. ***(Discrete Fourier Transform)*** [24]*: The Discrete Fourier Transform (DFT) of a time signal can be computed as $V(k) = \sum_{n=0}^{N-1} v(n)e^{\frac{-j2\pi nk}{N}} (k = 0, 1, 2, ..., N-1)$*

To find the appropriate sampling rate of a vehicle's velocity-time curve, we (1) Apply the FT on velocity $v(t)$ over a certain interval; (2) Obtain the frequency spectrum $V(\omega)$ of the velocity $v(t)$ over that interval; (3) Select the highest frequency component f_m, where $f_m = \frac{\omega_m}{2\pi}$; and (4) Get the optimum sampling rate as per the *Nyquist Theorem*, which gives the ideal sampling rate (that preserves the fidelity of a time-varying signal) as $f_s \geq 2f_m$.

The RSU controller module will receive the updated position and velocity values from the vehicles after each sampling interval of $T_s = \frac{1}{f_s}$, and thus generate the updated preferences. Thus, the dynamic nature of our model due to the mobility of the vehicles is accounted for in this manner while getting an accurate representation of the velocity variation with time.

B. Resource Assignment as a Matching with Externalities

Due to variability in the velocity and positioning of the vehicle, its preferences are dynamic; modeling it as a game with externalities is an apt choice [25]–[30]. Formally, we define the matching game as follows.

Definition 4. ***(Preferences of Agents)****: A matching is an association between two sets $\mathcal{M}$ and $\mathcal{V}$. An agent $a \in \mathcal{M} \cup \mathcal{V}$ has a preference relation $\succ_a$ that indicates its priorities over any two agents in the opposite set* [31].

Definition 5. ***(Offloading Game)****: Matching between sets $\mathcal{M}$ and $\mathcal{V}$ can be inferred as a many-to-one association $\Psi : \mathcal{M} \cup \mathcal{V} \rightarrow 2^{\mathcal{M} \cup \mathcal{V}}$ adhering to the conditions* (1) $\forall\, vm_j \in \mathcal{M}$, $|\Psi(vm_j)| \leq q_j$, $\Psi(vm_j) \subseteq \mathcal{V}$, (2) $\forall\, v_i \in \mathcal{V}$, $|\Psi(v_i)| = 1$, $\Psi(v_i) \subseteq \mathcal{M}$, *and* (3) $v_i \in \Psi(vm_j) \iff vm_j \in \Psi(v_i)$ [32].

Definition 6. ***(Existence of Swap Matching)****: For an assignment Ψ, and a pairs of assignments (v_i, vm_j), $(v_{i'}, vm_{j'}) \in \Psi$, a swap matching can be defined as $\Psi_{i,i'}^{j,j'}$, and $\Psi_{i,i'}^{j,j'} = \Psi \setminus \{(v_i, vm_j), (v_{i'}, vm_{j'})\} \cup \{(v_{i'}, vm_j), (v_i, vm_{j'})\}$, such that the following conditions are satisfied:* (1) $\mathbb{T}_i^o(\alpha_i, \Psi_{i,i'}^{j,j'}) \leq \mathbb{T}_i^o(\alpha_i, \Psi)$ *and* $\mathbb{T}_{i'}^o(\alpha_i, \Psi_{i,i'}^{j,j'}) \leq \mathbb{T}_{i'}^o(\alpha_i, \Psi)$*, and* (2) $u_{j,i} \leq u_{j,i'}$ *and* $u_{j',i'} \leq u_{j',i}$.

Definition 7. ***(Exchange Stability)*** *A matching Ψ is a two-sided exchange stable (2ES) if there is no swap matching from its current assignment.*

Definitions (6) and (7) capture the notion of swap-stability. It specifically states that a vehicle VM pair $(v_i, vm_j) \in \Psi$ is considered to exhibit exchange stability if there does not exist another pair $(v_{i'}, vm_{j'}) \in \Psi$ such that v_i prefers $vm_{j'}$ over its current assignment v_i and vm_j prefers $v_{i'}$ over its current allocation v_i. Moreover, such a swapping must be beneficial for all the competing agents, i.e., $\{v_i, v_{i'}, vm_j, vm_{j'}\}$, without compromising their benefits. The matching is considered exchange stable if no such pairs are incentivized to swap.

1) Preferences of Agents: Let the preference of agents $a \in \mathcal{M} \cup \mathcal{V}$ be captured as $P(a)$. Note that while computing the preferences, we consider full offloading, wherein $\alpha_{i,j}^l = 0$, and $\alpha_{i,j}^o = 1$. Once the assignment is made, the actual offloaded fractions are computed as per Algorithm. 3.

a) Vehicle Preferences: Vehicle v_i assigns preferences to VMs to minimize communication and processing latency. The preference of v_i over VMs vm_j and $vm_{j'}$ is expressed as

$$vm_j \succ_{v_i} vm_{j'} \iff \mathbb{T}_i^c(\alpha_{i,j}, \Psi_k) + \mathbb{T}_i^p(\alpha_{i,j}, \Psi_k) < \mathbb{T}_i^c(\alpha_{i,j'}, \Psi_k) + \mathbb{T}_i^p(\alpha_{i,j'}, \Psi_k) \tag{16}$$

b) Urgency of Vehicles: We define *urgency score* $u_{j,i}$ for each vehicle $v_i \in \mathcal{V}$ to be offloaded to vm_j in terms of two metrics: (i) execution options, $P(v_i)$, and (ii) failure probability, $\mathbb{P}_i^f$. Assigning a higher score to a vehicle with fewer options increases its likelihood of assignment. In addition, prioritizing a vehicle with a higher probability of failure boosts its chances of being assigned. Consequently, a combined valuation helps MIME achieve reduced latency.

$$u_{j,i} = \frac{\mathbb{P}_i^f}{|P(v_i)|} \tag{17}$$

c) Preferences of VMs: Based on the above definition of urgency, the preference of vm_j is defined as

$$v_i \succ_{vm_j} v_{i'} \iff u_{j,i} > u_{j,i'} \tag{18}$$

2) Dynamic Vehicle to Machine Assignment Algorithm (DVMA): The working of DVMA is depicted in Algorithm 1. The assignment starts by computing the preferences $P(.)$ for all $v_i \in \mathcal{V}$. The status of all the agents in $\mathcal{V} \cup \mathcal{M}$ is initialized to $True$, indicating their willingness to participate in DVMA (Steps 2-6 of Algorithm 1). At each sampling interval $t_k \in \mathcal{T}$, the preference of the VMs is recomputed. Note that 'k' captures the current iteration of DVMA and is initialized to 1. Once the preference is recomputed, the free VMs, i.e., $free[vm_j]$ is $True$, send proposals to their most preferred vehicles. Depending on the status of vm_j, the following conditions are encountered, (1) if vm_j has the capacity, i.e., $q_j \geq \Psi_k(vm_j)$ then v_i and vm_j are matched (Steps $1-7$ of Algorithm 2) or (2) if vm_j is already holding proposals, then appropriate action is taken based on the preference of $P_k(vm_j)$. If vm_j prefers the incoming proposal $v_i \succ_{vm_j} v_{i'}$, then v_i replaces $v_{i'}$; otherwise the proposal from v_i is rejected

(Steps $9-19$ of Algorithm 2). Note that the variable $reject$ retains the rejected proposal, set to Φ if no rejections implying the matching of v_i and vm_j. If an assignment is made,

Algorithm 1: Dynamic Vehicle to Machine Assignment Algorithm (DVMA)

Input: $\mathcal{V}$, $\mathcal{M}$, ζ_i, $\forall\, v_i \in \mathcal{V}$; ζ_j, $\forall\, vm_j \in \mathcal{M}$.
Result: Ψ_k ▷ Final Matching
Initialize: $\Psi_k(v_i) = \Phi, \forall v_i \in \mathcal{V}$;
  $\Psi_k(vm_j) = \Phi, \forall vm_j \in \mathcal{M}, \boldsymbol{\alpha} = \Phi, k = 1$
for *each* $v_i \in \mathcal{V}$ **do**
  Compute the preference $P(v_i)$ as per Eq. (16).
  free[v_i] = $True$
end
for *each* $vm_j \in \mathcal{M}$ **do**
  free[vm_j] = $True$
end
for *each sampling interval* $t_k \in \mathcal{T}$ **do**
  for *each* $vm_j \in \mathcal{M}$ **do**
    Compute the preference $P_k(vm_j)$ as per Eq. (18).
  end
  while $\exists\, vm_j \mid free[vm_j]$ && $P_k(vm_j)! = \Phi$ **do**
    v_i = Highest currently ranked vehicle in $P_k(vm_j)$
    $reject$ = **Matching_Algorithm** (v_i, vm_j)
    if $reject == \Phi \,||\, reject == v_{i'}$ **then**
      $\boldsymbol{\alpha_{i,j}}$ = **Partition_Algorithm** $(v_i, vm_j, \zeta_i, \zeta_j)$
      $\boldsymbol{\alpha} = [\boldsymbol{\alpha} | \boldsymbol{\alpha_{i,j}^T}]$
    end
  end
end
return Ψ_k

Algorithm 2: Matching_Algorithm (MA).

Input: v_i, vm_j
Result: $reject$ ▷ The rejected proposal
if $q_j > ||\Psi_k(vm_j)|$ **then**
  $\Psi_k(v_i) = \{vm_j\}$
  $\Psi_k(vm_j) = \Psi_k(vm_j) \cup \{v_i\}$ ▷ Match (v_i, vm_j)
  free[v_i] = $False$
  if $q_j == |\Psi_k(vm_j)|$ **then**
    free[vm_j] = $False$
  end
  $reject = \Phi$
end
else
  $v_{i'}$ = The least preferred matched vehicle to VM vm_j
  if $v_i \succ_{vm_j} v_{i'}$ **then**
    $\Psi_k(v_{i'}) = \Phi$
    free[$v_{i'}$] = $True$
    $\Psi_k(vm_j) = \Psi_k(vm_j) \setminus \{v_{i'}\}$ ▷ Un-Match $(v_{i'}, vm_j)$.
    free[v_i] = $False$
    $\Psi_k(v_i) = \{vm_j\}$
    $\Psi_k(vm_j) = \Psi_k(vm_j) \cup \{v_i\}$ ▷ Match (v_i, vm_j).
    $reject = v_{i'}$
  end
  else
    $reject = v_i$
  end
end
return $reject$

Algorithm 3 is triggered, which outputs the allocation vector (Steps 13-15 of Algorithm 1). It captures the fraction of the computation that is to be performed locally ($\alpha_{i,j}^l$) and at vm_j ($\alpha_{i,j}^o$). The fraction is computed as

$$\alpha_{i,j}^l = \frac{1}{1 + \frac{\zeta_i}{\zeta_j}} \tag{19}$$

and the ratio of power ratings gives an energy-aware partition.

Algorithm 3: Partition_Algorithm (PA)

Input: $v_i, \Psi_k(v_i), \zeta_i, \zeta_j$
Result: $\boldsymbol{\alpha_{i,j}}$ ▷ Allocation Vector
Initialize: $\boldsymbol{\alpha_{i,j}} := \Phi, \forall v_i \in \Psi_k$;
Compute $\alpha_{i,j}^l$ as per Eq. (19)
if $\mathbb{T}_i^l(\alpha_{i,j}, \Psi_k) \leq s_i$ **then**
  $\alpha_{i,j}^o := 1 - \alpha_{i,j}^l$
  $\boldsymbol{\alpha_{i,j}} := \{\alpha_{i,j}^l, \alpha_{i,j}^o\}$
end
else
  $\boldsymbol{\alpha_{i,j}} := \{0, 1\}$
end
return $\boldsymbol{\alpha_{i,j}}$

Once the partitions are identified, PA tests if the local delay $\mathbb{T}_i^l(\alpha_{i,j}, \Psi_k)$ overshoots the estimated sojourn time s_i, then the entire computation is offloaded to the VM, otherwise the obtained partition is retained (Steps 3-8 of Algorithm 3). An unassigned vehicle reattempts allocation in the next interval with an updated preference. Note that this procedure terminates when all vehicles are assigned or run out of attempts, upper bounded by the number of sampling intervals.

TABLE I: Simulation Parameters [33], [34].

Parameter	Value
RSU Coverage	$500\ m$
Computation Size of Vehicles	$[100 - 1000]$ *KB*
Computational Requirement of Vehicles	$[500 - 1500]$ *cycles/bit*
VM Processing Frequency	$[0.1 - 1.0]\ GHz$
OBUs Execution Power of Vehicles	$[250 - 400]$ *MW*
VMs Execution power	$[150 - 500]$ *MW*
Bandwidth of RSUs	$20\ MHz$ [23]
Channel Gain between RSU and Vehicles	18
Noise Power	$-114\ dBm$
Path loss	-3.4
Transmission Power of Vehicles	$31\ dBm$ [22]

V. Performance Evaluation

To demonstrate the efficacy of MIME, we perform validations using synthetic and real-world transportation data obtained from SUMO [35] and inD dataset [36]. We extract the velocity and positioning of vehicles from the dataset to emulate a realistic vehicular trajectory.

1) Environmental Setup: Our analysis considers a two-lane road network with multiple vehicles traversing the area. The distribution of the vehicles from the datasets is depicted in Fig. 4. We consider a statically deployed RSU to have a coverage area of $500\ m$. Note that the compute resources at the RSU are logically partitioned into VMs with heterogeneous processing frequencies and execution power. All the vehicles have onboard units (OBUs) with wireless transceivers and receivers that assist the vehicles in communicating with the RSU over a noiseless wireless channel [37]. Table I captures the values of different parameters used for experimentation and are adopted from standard sources [33], [34].

2) Benchmarking Algorithms: To evaluate the proficiency of MIME, we compare its performance against the following.

- Random Assignment (RA): The vehicles are randomly matched to VMs and checked for the satisfaction of all the Constraints in Eq. (15).
- Sequential Optimal Matching (SOM): The vehicles are sequentially matched to the VMs sorted according to their computing capabilities. This assignment should also adhere to the Constraints of Eq. (15).
- A Matching Theory Framework for Task Offloading in Fog Computing for IoT Systems [38] (MTO): This work proposes a one-to-many matching framework for full offloading in densely connected IoT-Fog systems. The overall objective of the proposal was to reduce the latency incurred in offloading by targeting the reduction in the worst-case completion time of tasks executed on the IoTs. For our implementation, we consider the IoTs to be static vehicles and the Fog Nodes (FNs) to be the VMs.

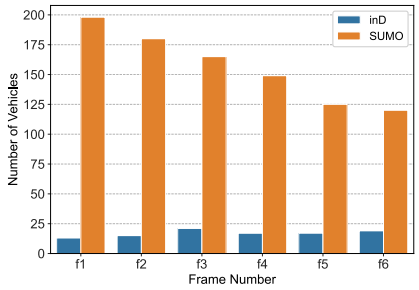


Fig. 4: Number of vehicles at different frames across datasets.

Additionally, we would like to state that we implement the partial offloading versions of the baselines to make the comparison fair and impartial. Note that for all the baselines, the offloading partition is obtained via Algorithm 3.

*3) **inD** Dataset* [36]*:* We have evaluated MIME using the **inD** Dataset. The dataset comprises the trajectories of the road users captured from the bird's eye view of a drone, providing the most naturalistic position. The drone hovered over four German intersections with a speed restriction of $50\ Kmh^{-1}$. Prepossessing the captured drone data, 13500 users were detected, comprising vehicles, pedestrians, and bicyclists. From the data, we considered 400 vehicles that were annotated. The dataset contains many parameters, but we focus on the two-dimensional position, x, and y directional velocity, where the latter was obtained using the Bayesian smoothing and acceleration model.

4) SUMO Dataset: To generate the synthetic dataset using SUMO, we considered the map of the Jurong area of Singapore, incorporating its road network and traffic regulations. The simulation was run for 200 vehicles, where each vehicle is distributed randomly and follows realistic traffic flow.

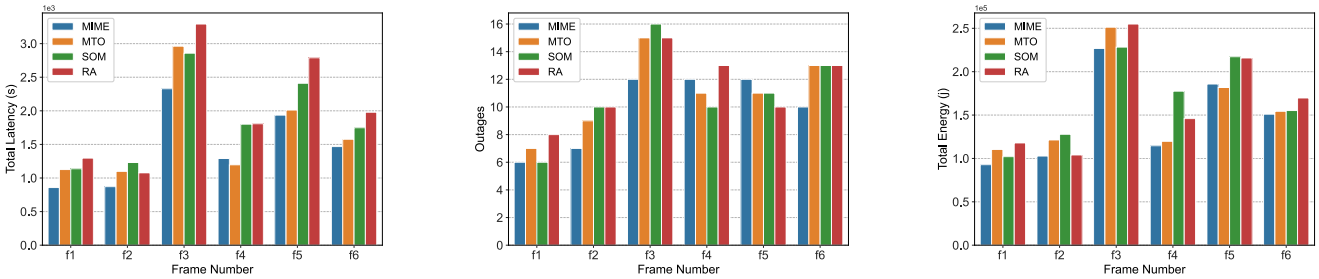
5) Implementation Details: Although the *inD* dataset provides the position and velocities of vehicles, it cannot be directly used in MIME owing to the assumption of a two-lane structure. Moreover, we extract the x-velocity component from the dataset, approximating a realistic variation of v_i with time. We use the dataset's x and y coordinates to determine the vehicle's precise location in a two-dimensional layout. Additionally, the vehicles falling within the radial range of the RSU (which is assumed to be statically positioned) are considered for assignment, whereas those outside the range are ignored. In every iteration of DVMA, we extract the x velocity component and its corresponding position, taken as inputs that realistically capture the dynamics of a vehicle trajectory in an urban road network. Note that the same procedure is followed for the SUMO dataset while extracting the x and y coordinates.

6) Results and Analysis: This section presents the comparative study of different schemes across evaluation metrics.

Total Latency: Figs. 5a and 6a capture the variation in the total latency of different protocols across inD and SUMO datasets. It can be observed from the figure that MIME achieves the least aggregate latency in most cases, which is promising behavior. This condescending performance is attributed to the vehicles ranking the VMs according to their offloading delays (refer to Eq. (16)). However, the performance across frames is directly impacted by the number of vehicles in that frame (refer to Fig. 4). In contrast, MTO grapples with higher latency compared to MIME, as it prioritizes the assignment of vehicles with lesser offloaded data instead of prioritizing the ones with more extensive data. This implies that the vehicles with more significant data are either queued at the computationally efficient VMs or are pushed to less efficient ones. The former elevates the queueing delay, whereas the latter increases the processing delay, which leads to an overall increase in the offloading delay. SOM and RA experience higher delays than MIME and MTO owing to greedy and randomized assignments leading to inferior assignments.

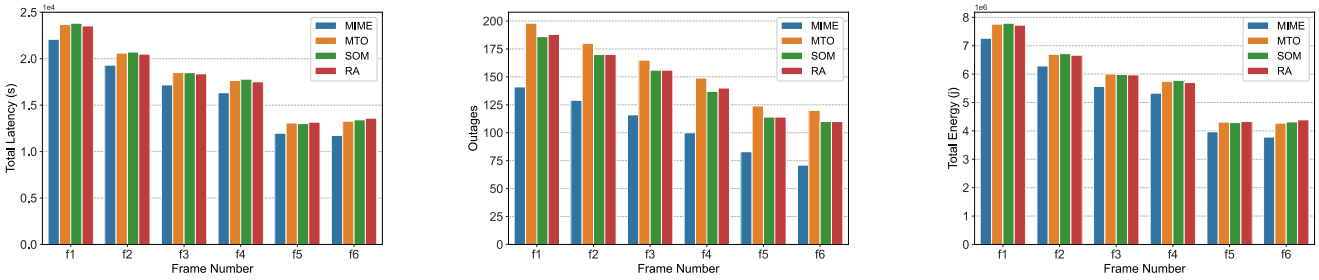
Outages: MIME suffers from minimum outages compared to the benchmarking algorithms in most scenarios, as can be inferred from Figs. 5b and 6b. The outages refer to the vehicular operations that were not performed in the RSU, considering their respective sojourn time. The ranking scheme adopted (refer to Eq. (18)) reasons for the superior performance of MIME. The VM ranking scheme prioritizes assigning vehicles that have (1) lower execution options and (2) higher failure probability. Such a combined valuation boosts the possibility of vehicles with higher data meeting sojourn time constraints, thereby exhibiting superior performance compared to the baselines, which consider neither. On the other hand, in MTO, the ability of the vehicles to reorder their preferences boosts their possibility of meeting the sojourn time. Finally, RA and SOM exhibit a higher number of outages owing to the non-consideration of sojourn time during the assignment decisions.

Energy Consumption: The energy expended in the offloading is captured in Figs. 5c and 6c. The energy expended increases with increased vehicles in the considered frame, which is expected behavior. However, we can infer from the figure that MIME expends minimum energy across frames compared to the benchmarking algorithm. This is because energy is considered during partitioning, which reduces energy consumption. However, none of the baselines take offloading



(a) Total Latency (*s*) *vs.* Frame Number. (b) Outages *vs.* Frame Number. (c) Energy (*J*) *vs.* Frame Number.

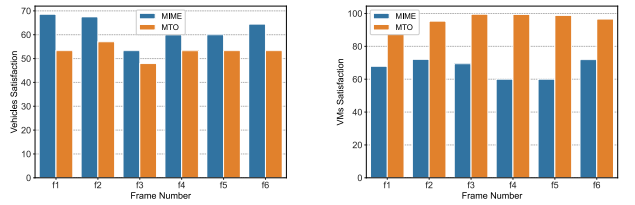
Fig. 5: Performance based on inD Dataset.



(a) Total Latency (*s*) *vs.* Frame Number. (b) Outages *vs.* Frame Number. (c) Energy (*J*) *vs.* Frame Number.

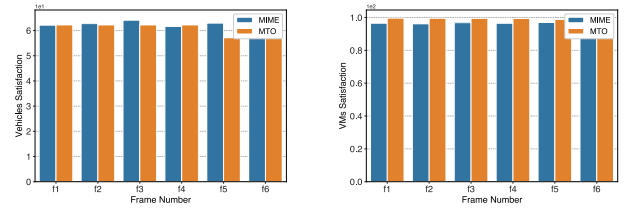
Fig. 6: Performance based on SUMO Dataset.

decisions based on the energy expended, resulting in higher energy dissipation.



(a) Vehicles Satisfaction (%) vs. Frame Number. (b) VM Satisfaction (%) vs. Frame Number.

Fig. 7: Vehicular and VM Satisfaction using inD Dataset.



(a) Vehicles Satisfaction (%) vs. Frame Number. (b) VM Satisfaction (%) vs. Frame Number.

Fig. 8: Vehicular and VM Satisfaction using SUMO Dataset.

Agent Satisfaction: The agents' satisfaction reflects their happiness quotient in the matching given a list of preferences. Note that it can only be computed for matching-based solutions requiring agents to have a ranked order list. Figs 7a-8b respectively capture the satisfaction of vehicles and VMs for MIME and MTO. The satisfaction of a vehicle is computed as $(|P(v_i)| - rank(vm_j))/|P(v_i)| * 100$, where, $rank(vm_j)$, captures the position of vm_j in $P(v_i)$ such that $rank(vm_j) \in [0, |P(v_i)| - 1]$. The VMs can be matched to multiple vehicles; the individual satisfaction is computed as $(|P(vm_j)| - rank(v_i))/(|P(vm_j)| - count) * 100$, for each matched v_i, and the aggregate is averaged over $|P(v_i)|$. Note that *count* captures the assigned vehicles before computing the preference of v_i. As VMs are proposers, the VM satisfaction is higher than the vehicles for both MIME and MTO.

VI. Conclusion

This paper proposed the MIME framework for partial offloading of real-time computations from vehicles. Specifically, MIME addresses two fundamental challenges: (1) capturing the realistic variation of the vehicular trajectories and (2) designing an energy-aware, efficient, and scalable partial offloading scheme that meets the QoS demands. We proposed a matching game powered by DFT with dynamic preferences triggered at each sampling interval. Experimental validations and comparisons with three distinct baselines establish the efficacy of the proposed scheme with the reduction of 24.61% and 11.2% offloading latency and energy for the inD dataset and 5.6% and 5.52% reduction in offloading latency and energy for SUMO dataset.

In future work, we plan to explore multiple RSU setups with effective handover and migration capabilities to enhance inter-RSU communication. Additionally, we aim to upgrade MIME to account for co-channel interference, thereby increasing the robustness of the implementation.

Acknowledgements: This work is partially supported by NSF grants under award numbers CNS-2030624, OAC-2104078, and ECCS-2319995; Daniel St. Clair Endowed Chair funds and Kummer Doctoral Fellowship at Missouri S&T.

REFERENCES

[1] J. Wang, J. Steiber, and B. Surampudi, "Autonomous ground vehicle control system for high-speed and safe operation," in *2008 American Control Conference*, 2008, pp. 218–223.

[2] H. Lu, Q. Liu, D. Tian, Y. Li, H. Kim, and S. Serikawa, "The Cognitive Internet of Vehicles for Autonomous Driving," *IEEE Network*, vol. 33, no. 3, pp. 65–73, 2019.

[3] J. Lee and W. Na, "A survey on vehicular edge computing architectures," in *2022 13th International Conference on Information and Communication Technology Convergence (ICTC)*, 2022, pp. 2198–2200.

[4] M. Chen and Y. Hao, "Task Offloading for Mobile Edge Computing in Software Defined Ultra-Dense Network," *IEEE Journal on Selected Areas in Communications*, vol. 36, no. 3, pp. 587–597, 2018.

[5] P. Chennakesavula, J.-M. Wu, and A. Ambikapathi, "Incentive-Driven Fog-Edge Computation Offloading and Resource Allocation for 5G-NR V2X-Based Vehicular Networks," in *2023 IEEE 97th Vehicular Technology Conference (VTC2023-Spring)*, 2023, pp. 1–5.

[6] J. Zhang, Y. Wu, G. Min, and K. Li, "Neural Network-Based Game Theory for Scalable Offloading in Vehicular Edge Computing: A Transfer Learning Approach," *IEEE Transactions on Intelligent Transportation Systems*, 2024.

[7] 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 Transactions on Vehicular Technology*, vol. 71, no. 2, pp. 2041–2057, 2022.

[8] H. Wang, Z. Lin, K. Guo, and T. Lv, "Computation offloading based on game theory in mec-assisted v2x networks," in *2021 IEEE International Conference on Communications Workshops (ICC Workshops)*, 2021, pp. 1–6.

[9] Z. Liu, X. Yang, Y. Yang, K. Wang, and G. Mao, "DATS: Dispersive stable task scheduling in heterogeneous fog networks," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 3423–3436, 2018.

[10] C. Swain, M. N. Sahoo, and A. Satpathy, "SPATO: A student project allocation based task offloading in IoT-fog systems," in *ICC 2021-IEEE International Conference on Communications*. IEEE, 2021, pp. 1–6.

[11] H. Wang, T. Liu, B. Kim, C.-W. Lin, S. Shiraishi, J. Xie, and Z. Han, "Architectural Design Alternatives Based on Cloud/Edge/Fog Computing for Connected Vehicles," *IEEE Communications Surveys & Tutorials*, vol. 22, no. 4, pp. 2349–2377, 2020.

[12] L. Liu, C. Chen, Q. Pei, S. Maharjan, and Y. Zhang, "Vehicular Edge Computing and Networking: A Survey," *Mob. Netw. Appl.*, vol. 26, no. 3, p. 1145–1168, jun 2021.

[13] L. Bréhon–Grataloup, R. Kacimi, and A.-L. Beylot, "Field trial for enhanced v2x multi-rat handover in autonomous vehicle networks," in *2023 IEEE 48th Conference on Local Computer Networks (LCN)*, 2023, pp. 1–8.

[14] J. Zhang, H. Guo, J. Liu, and Y. Zhang, "Task Offloading in Vehicular Edge Computing Networks: A Load-Balancing Solution," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 2, pp. 2092–2104, 2020.

[15] Y. Liu, S. Wang, J. Huang, and F. Yang, "A Computation Offloading Algorithm Based on Game Theory for Vehicular Edge Networks," *2018 IEEE International Conference on Communications (ICC)*, pp. 1–6, 2018. [Online]. Available: https://api.semanticscholar.org/CorpusID:51870833

[16] Q. Yuan, J. Li, H. Zhou, T. Lin, G. Luo, and X. Shen, "A Joint Service Migration and Mobility Optimization Approach for Vehicular Edge Computing," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 8, pp. 9041–9052, 2020.

[17] P. L. Nguyen, R.-H. Hwang, P. M. Khiem, K. Nguyen, and Y.-D. Lin, "Modeling and Minimizing Latency in Three-tier V2X Networks," in *GLOBECOM 2020 - 2020 IEEE Global Communications Conference*, 2020, pp. 1–6.

[18] A. Bozorgchenani, F. Mashhadi, D. Tarchi, and S. A. Salinas Monroy, "Multi-Objective Computation Sharing in Energy and Delay Constrained Mobile Edge Computing Environments," *IEEE Transactions on Mobile Computing*, vol. 20, no. 10, pp. 2992–3005, 2021.

[19] V. D. Tuong, T. P. Truong, T.-V. Nguyen, W. Noh, and S. Cho, "Partial Computation Offloading in NOMA-Assisted Mobile-Edge Computing Systems Using Deep Reinforcement Learning," *IEEE Internet of Things Journal*, vol. 8, no. 17, pp. 13 196–13 208, 2021.

[20] X.-Q. Pham, T. Huynh-The, E.-N. Huh, and D.-S. Kim, "Partial Computation Offloading in Parked Vehicle-Assisted Multi-Access Edge Computing: A Game-Theoretic Approach," *IEEE Transactions on Vehicular Technology*, vol. 71, no. 9, pp. 10 220–10 225, 2022.

[21] U. Saleem, Y. Liu, S. Jangsher, X. Tao, and Y. Li, "Latency Minimization for D2D-Enabled Partial Computation Offloading in Mobile Edge Computing," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 4, pp. 4472–4486, 2020.

[22] R. Yadav, W. Zhang, O. Kaiwartya, H. Song, and S. Yu, "Energy-latency tradeoff for dynamic computation offloading in vehicular fog computing," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 12, pp. 14 198–14 211, 2020.

[23] G. Araniti, C. Campolo, M. Condoluci, A. Iera, and A. Molinaro, "Lte for vehicular networking: a survey," *IEEE Communications Magazine*, vol. 51, no. 5, pp. 148–157, 2013.

[24] J. G. Proakis, *Digital signal processing: principles, algorithms, and applications, 4/E*. Pearson Education India, 2007.

[25] J. Zhao, Y. Liu, K. K. Chai, Y. Chen, and M. Elkashlan, "Many-to-many matching with externalities for device-to-device communications," *IEEE wireless communications letters*, vol. 6, no. 1, pp. 138–141, 2016.

[26] A. Satpathy, M. N. Sahoo, L. Behera, and C. Swain, "ReMatch: An Efficient Virtual Data Center Re-Matching Strategy Based on Matching Theory," *IEEE Transactions on Services Computing*, vol. 16, no. 2, pp. 1373–1386, 2023.

[27] K. Bando, R. Kawasaki, and S. Muto, "Two-sided matching with externalities: A survey," *Journal of the Operations Research Society of Japan*, vol. 59, no. 1, pp. 35–71, 2016.

[28] F. Pantisano, M. Bennis, W. Saad, S. Valentin, and M. Debbah, "Matching with externalities for context-aware user-cell association in small cell networks," in *2013 IEEE Global Communications Conference (GLOBECOM)*. IEEE, 2013, pp. 4483–4488.

[29] C. Swain, M. N. Sahoo, A. Satpathy, K. Muhammad, S. Bakshi, and J. J. P. C. Rodrigues, "A-DAFTO: Artificial Cap Deferred Acceptance-Based Fair Task Offloading in Complex IoT-Fog Networks," *IEEE Transactions on Consumer Electronics*, vol. 69, no. 4, pp. 914–926, 2023.

[30] C. Cheng, L. Zhai, X. Zhu, Y. Jia, and Y. Li, "Dynamic task offloading and service caching based on game theory in vehicular edge computing networks," *Computer Communications*, 2024.

[31] C. Swain, M. N. Sahoo, A. Satpathy, K. Muhammad, S. Bakshi, J. J. P. C. Rodrigues, and V. H. C. de Albuquerque, "METO: Matching-Theory-Based Efficient Task Offloading in IoT-Fog Interconnection Networks," *IEEE Internet of Things Journal*, vol. 8, no. 16, pp. 12 705–12 715, 2021.

[32] C. Swain, M. N. Sahoo, and A. Satpathy, "LETO: An efficient load balanced strategy for task offloading in IoT-fog systems," in *2021 IEEE international conference on web services (ICWS)*. IEEE, 2021, pp. 459–464.

[33] C. Liu, K. Li, J. Liang, and K. Li, "COOPER-SCHED: A Cooperative Scheduling Framework for Mobile Edge Computing with Expected Deadline Guarantee," *IEEE Transactions on Parallel and Distributed Systems*, pp. 1–1, 2019.

[34] S. R. Sahoo, M. Patra, and A. Gupta, "MDLB: A Matching based Dynamic Load Balancing Algorithm for Road Side Units," in *2021 International Wireless Communications and Mobile Computing (IWCMC)*, 2021, pp. 291–296.

[35] P. A. Lopez, M. Behrisch, L. Bieker-Walz, J. Erdmann, Y.-P. Flötteröd, R. Hilbrich, L. Lücken, J. Rummel, P. Wagner, and E. Wiessner, "Microscopic Traffic Simulation using SUMO," in *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, 2018, pp. 2575–2582.

[36] J. Bock, R. Krajewski, T. Moers, S. Runde, L. Vater, and L. Eckstein, "The ind dataset: A drone dataset of naturalistic road user trajectories at german intersections," in *2020 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2020, pp. 1929–1934.

[37] D. Jiang and L. Delgrossi, "IEEE 802.11p: Towards an International Standard for Wireless Access in Vehicular Environments," in *VTC Spring 2008 - IEEE Vehicular Technology Conference*, 2008, pp. 2036–2040.

[38] F. Chiti, R. Fantacci, and B. Picano, "A matching theory framework for tasks offloading in fog computing for IoT systems," *IEEE Internet of Things Journal*, vol. 5, no. 6, pp. 5089–5096, 2018.