Preliminary Evaluation and Optimization of Task Offloading and Latency in Vehicular Edge Computing

Rim Sayegh, Abdulhalim Dandoush, Hela Marouane, Sahar Hoteit

Preliminary Evaluation and Optimization of Task Offloading and Latency in Vehicular Edge Computing

Rim Sayegh*†, Abdulhalim Dandoush ‡, Hela Marouane *, Sahar Hoteit†§
* ESME Research Lab, Ivry sur Seine, France
†Université Paris-Saclay, CNRS, CentraleSupélec, Laboratoire des Signaux et Systèmes (L2S), Gif-sur-Yvette, France
‡University of Doha for Science and Technology (UDST), Doha, Qatar
§Institut Universitaire de France (IUF), France
rim.sayegh@centralesupelec.fr, abdulhalim.dandoush@udst.edu.qa, hela.marouane@esme.fr, sahar.hoteit@centralesupelec.fr

***Abstract*—In vehicular edge computing (VEC), multi-access edge computing (MEC) plays a crucial role in enabling vehicles to offload computationally intensive tasks, thereby enhancing data processing efficiency. However, MEC encounters challenges related to limited resources and the complexity of optimal task offloading due to the dynamicity of the environment.**

This short paper provides a preliminary analysis of decision-making algorithms responsible for assigning vehicle tasks to MECs, with a focus on the latency metric. The study examines two types of vehicle applications—safety and infotainment—using the open source Simu5G simulator to assess how environmental complexity and different policies influence the delay. The results underscore the impact of these factors on the end-to-end (E2E) latency and highlight potential enhancements. Future work will explore strategies to further reduce E2E latency, improving the overall responsiveness and efficiency of VEC systems.

***Index Terms*—Multi-Access Edge Computing; Vehicular edge computing; Automated Vehicles; E2E Delay; 5G Simulation**

I. Introduction

Intelligent Transport System (ITS) represents a complex system including infrastructure (e.g., roads, traffic lights), with dynamic environmental conditions (e.g., weather and road conditions) and connected vehicles interacting with these elements. Different intelligent transport materials and software are deployed onboard or at a central cloud to enable a new generation of applications and services, such as collaborative auto-driving or safety applications. However, certain critical services require relatively large computing and storage resources, which are larger than the capability of vehicles. On the other hand, these services are sensitive to delay, and forwarding the information to a remote central cloud for processing may be inappropriate. To solve this issue, Multi-Access Edge Computing (MEC) [2] provides cloud-computing capabilities and an Intelligent Transport (IT) service environment. MEC significantly reduces latency by processing data at the edge, as demonstrated in studies where MEC-assisted applications showed improved response times compared to traditional cloud-based solutions [9]. The recent development of autonomous driving technologies has increased the demand for services and resources at the network edge. Inadequate servicing of autonomous vehicles at the network edge, due to limitations in resource allocation and scheduling, can lead to erroneous decisions and potentially serious accidents [8]. Therefore, evaluating MEC performance in 5GB networks is crucial. It is essential to optimize MEC resource allocation and communication segments for various services to ensure safety and efficiency. This paper aims to evaluate the performance of a decision-making algorithm (policy) used to allocate vehicle tasks to MEC nodes in a VEC environment. By studying the latency components associated with different application types, we seek to identify the limitations of the current approach. Through simulations conducted using OMNeT++ [3] and Simu5G [2], we analyze the effect of environmental complexities on task offloading delays and propose potential solutions to reduce End-to-End (E2E) latency, paving the way for future improvements in MEC performance. The remainder of the paper is structured as follows. Sections II and III introduce our system model and provide details of the simulation scenarios, respectively. Section IV covers the discussion of the current results. Lastly, Section V concludes the paper, highlighting limitations and suggesting future work.

II. System Model

Consider a VEC environment comprising a collection of n vehicles denoted as $V = \{v_1, v_2, \ldots, v_n\}$, and a set of m MEC hosts denoted as $H = \{h_1, h_2, \ldots, h_m\}$. The workflow of our scenario is illustrated in Fig. 1. In our scenario, each vehicle requests two types of applications. We implement the first application called "Safety app", where the vehicles request sensing task offloading from a centralized controller known as the "Orchestrator" [1]. Once the best MEC host is selected, it offloads and processes the vehicle's data and subsequently sends the necessary responses, such as navigation instructions, back to the vehicle. The second application involves video streaming [7], where vehicles transmit real-time video feeds to the MEC. The MEC processes these video streams to gather information about the vehicle's surrounding environment, which can be utilized for other applications to enhance decision-making and response efficiency. Each MEC ($h_j \in H$) is characterized by a maximum CPU, storage, and memory capacity, denoted by C_j, S_j, and R_j, respectively. Similarly, $C(v_i)$, $S(v_i)$, and $R(v_i)$ represent the CPU, memory, and storage requirements for a given vehicle task v_i. Each vehicle v_i offloads its tasks to a single MEC, where the assigning function gives $x_{ij} = 1$ if v_i offloads its task to MEC host h_j. We define the E2E delay as the sum of the *"initDelay"* and the

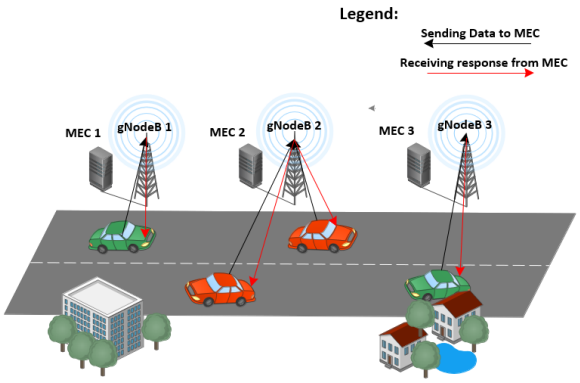


Fig. 1: Architecture of the autonomous driving scenario.

Total Response Time (TRT), as shown in Eq.(1). We define the *"initDelay"* as the sum of 3 sub latencies as follows: (i) the time spent to deliver the allocation request from the vehicle to the orchestrator. (ii) The time to allocate the request to the best MEC host according to the scheduling policy. And (iii) the time spent receiving the acknowledgment with the address of the best allocated MEC host to the vehicle. The "initDelay" corresponds to the decision-making time. We define in Eq. (2) the "Total Response Time" (TRT) as the time between the transmission of sensed data as a data packet from the vehicle to MEC and the reception of response data at the vehicle after data execution and processing by the MEC.

$$E2EDelay = initDelay + TRT \tag{1}$$

$$\begin{aligned} TRT =& ULPacketDelay + QueueingDelay \\ &+ ProcessingDelay + DLPacketDelay \end{aligned} \tag{2}$$

TABLE I: Simulation Parameters

Parameter Name	***Value***
Carrier Frequency	2 GHz
Numerology index	2
Simulation Time	200 s
Uplink rate (msg/s)	Pois(20)
Uplink payload (Bytes)	Exp(40000)
Downlink payload (Bytes)	313 [10]
Downlink Bandwidth (MBit/s)	0.05
Instruction Per Request (IPR)	Exp(500)
Min CPU Required (MIPS)	165130
Processing Speed (MIPS)	2356230
Road Mape Length	3 Km
gNodeBs inter distance	400 m
downlink rate (msg/s)	20
Session Duration (s)	uniform(3, 12)
Period between session (s)	uniform(3, 9)
Low-speed (km/h)	[10, 21]
Middle-speed (km/h)	[50, 80]
High-speed (km/h)	]80, 130]

III. Simulation Framework

Our objective is to assess the performance of task offloading policies and the ability of MEC hosts to support the most critical application, which is focused on ensuring safety and effective traffic management. To achieve this, we develop the "Safety App." Additionally, we evaluate the MEC hosts and policies under different environmental conditions and within a more complex scenario that includes an extra application, such as video streaming [7], implemented using Simu5G. We use the Simu5G as an open source based on the OMNet++ [3].

A. Vehicular application

- The "Safety App" represents a remote sensing scenario in which vehicles communicate with the infrastructure to ensure safety and optimize efficiency. Each vehicle operates the *"UESafetyApp"* function, while the MEC host processes the tasks from the vehicles using the *"MECSafetyApp"* function. Vehicles generate data packets according to a Poisson process, with inter-arrival times following an exponential distribution, as described in [6]. These packets travel through the uplink (UL) to the MEC host, where resources for each vehicle's request are allocated in a fair-sharing manner [4]. Each task requires a specific number of instructions for processing, referred to as Instructions Per Request (IPR) [5], [10]. The MEC host then processes the tasks, generates responses, and sends them back to the vehicles through the downlink (DL) path.
- The "VideoStreaming App" [7] is an application already implemented in the Simu5G. In this application, the MEC receives real-time video streams from the vehicle to accurately assess its surroundings and ensure safe steering.

Table I summarizes the main simulation parameters. Notably, we simulate the worst-case scenario where all vehicles concurrently request MEC app execution.

B. Detailed Algorithms

We assess three distinct task offloading algorithms already integrated into Simu5G, which serve as the orchestrator's decision-making policies. These algorithms may serve as benchmarks in future studies when we introduce our own algorithm, designed to outperform them.

- *"MecHostBased"* is a pre-defined association algorithm that enables the orchestrator to select a MEC host based on an index provided before the simulation begins.
- *"MecServiceBased"* is a First-Fit algorithm that selects the first available MEC host meeting the application's resource and service requirements. This involves choosing the first host with the necessary capacity and services to support the MEC application.
- *"AvailableResourcesBased"* is a Best-Fit algorithm, evaluating all MEC hosts in the list to find the one with the necessary resources and the highest CPU speed.

The scenario is designed to evaluate how various parameters, such as the number of vehicles, their speed, and the number of MEC hosts, along with different MEC selection policies, affect the performance of the MEC host. This enables us to analyze the various factors contributing to E2E delay within safety services. In future work, we will propose a new algorithm to enhance these delay components further, building on the insights gained from this evaluation.

IV. Results and Discussion

This section analyzes how vehicle speed affects delays under various selection policies and evaluates their limitations in scenarios where vehicles request multiple applications.

1) Impact of Selection Policies and Vehicle Speeds: In

the first experiment, we simulate a safety application with 12 vehicles and 3 MEC hosts, evaluating different policies across low, middle, and high-speed ranges. The results, shown in Fig. 2, reveal that the "Best-Fit" policy performs the best by selecting the MEC host with the highest CPU speed, which effectively distributes vehicle tasks and balances the network load, reducing "QueuingDelay" and "ProcessingDelay" as shown in Fig. 2b and Fig. 2c. Vehicle speed impacts "ULPacketDelay" and "QueueingDelay," as shown in Fig. 2a and Fig. 2b. As vehicle speed decreases, "ULPacketDelay" rises because lower-moving vehicles lead to spending more time in certain areas, leading to a higher volume of messages at coverage boundaries and increased propagation delays. Likewise, "QueueingDelay" grows with reduced speed since vehicles connected to a base station and MEC server for extended periods produce more packets, causing a backlog and longer queues.

TABLE II: *" initDelay"* (ms) with Different Selection Policy

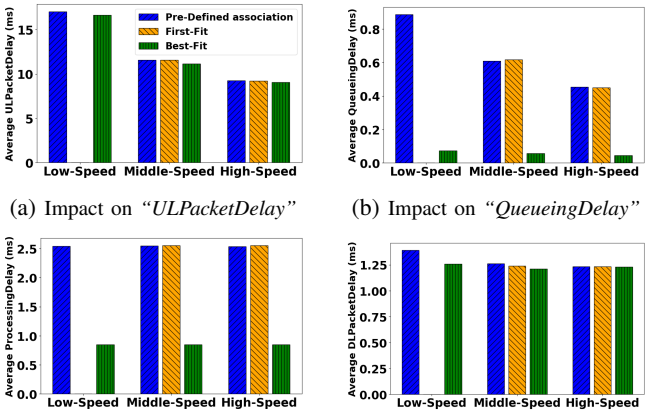
Selection Policy	***With low Speed***	***With middle Speed***	***With high Speed***
Pre-Defined association	54.34	53.63	52.71
First-Fit	53.56	52.87	52.37
Best-Fit	53.2	52.62	52.13

Table II shows that both the policy and speed have a minor impact on "initDelay," with the "AvailableResourcesBased" policy achieving the lowest "initDelay" due to effective load-balancing. Additionally, "initDelay" experiences a slight decrease with higher speeds. The experiment also highlights a limitation of the "First-Fit" policy at low speeds, where it allocates all vehicles to the most distant MEC host, resulting in packet reception issues at the base station. This means any packets received by the MEC and the associated delays are not calculated according to this policy, as demonstrated Fig. 2.

2) Impact of Simultaneous Two applications on Task Offloading: In the second experiment, we evaluate how running both the "Safety App" and "VideoStreaming App" together affect task offloading. This was tested using three MEC hosts and 12 vehicles, with each vehicle requesting the initialization of both applications at the same time. The "Best-Fit" policy was applied. The results indicate that introducing the "VideoStreaming App" significantly increases the "initDelay" for both applications, exposing the algorithm's limitations, particularly in small-scale environments, as illustrated in Fig. 3. This suggests that even greater delays could be expected in large-scale environments when using these static algorithms.

V. Conclusion and Future Work

In this paper, we examined previous task offloading algorithms in VEC, particularly focusing on how speed impacts various aspects of E2E delay. The findings highlight a major limitation of existing algorithms, where the "initDelay", the time taken for decision-making, contributes most significantly to the overall delay, especially as complexity increases. Moreover, speed is critical in task offloading and must be factored in. Our ongoing work evaluates MEC performance under varying conditions, such as varying numbers of vehicles and MEC hosts. To address current limitations, we propose a dynamic algorithm that enables the migration of MEC applications during vehicle mobility, focusing on reducing E2E delay, particularly the initialization delay "initDelay". Unlike the static algorithm used in this study, which keeps vehicles allocated to the same MEC host throughout the simulation, the new algorithm will consider factors such as speed and task priority, alongside exploring alternative scheduling mechanisms.



(a) Impact on *"ULPacketDelay"* (b) Impact on *"QueueingDelay"*

(c) Impact on *"ProcessingDelay"* (d) Impact on *"DLPacketDelay"*

Fig. 2: The impact of the selection policy on the average Total Response Time (TRT) components with different Speed

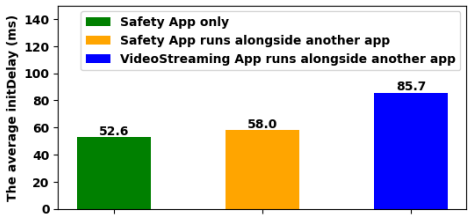


Fig. 3: The average "initDelay"

References

[1] ETSI MEC model in Simu5G. https://simu5g.org/users-guide/mec.

[2] Multi-access edge computing (mec). https://www.etsi.org/technologies/multi-access-edge-computing. [Online; accessed 21-May-2024].

[3] Omnetpp. https://github.com/omnetpp/omnetpp, 2024.

[4] G. Stea A. Virdis A. Noferi, G. Nardini. Rapid prototyping and performance evaluation of etsi mec-based applications. *Simulation Modelling Practice and Theory*, 123:102700, 2023.

[5] N. Auluck, A. Azim, and K. Fizza. Improving the schedulability of real-time tasks using fog computing. *IEEE Transactions on Services Computing*, 15(1):372–385, 2022.

[6] A. Bauer S. Kounev P. Heegaard F. Metzger, T. Hofeld. Modeling of aggregated iot traffic and its application to an iot cloud. *Proceedings of the IEEE*, 107:679–694, 2019.

[7] P. Ducange G. Stea G. Nardini, A. Noferi. Exploiting simu5g for generating datasets for training and testing ai models for 5g/6g network applications. *SoftwareX*, 21:101320, 2023.

[8] S. Hakak, T.R. Gadekallu, P. K. R. Maddikunta, S. Koppu, M. Parimala, C. d. Alwis, and M. Liyanage. Autonomous vehicles in 5g and beyond: a survey. *Vehicular Communications*, 39:100551, 2023.

[9] G. Nencioni M. A. Hathibelagal, R. G. Garroppo. Experimental comparison of migration strategies for mec-assisted 5g-v2x applications. *Computer Communications*, 197:1–11, 2023.

[10] O. Wendlasida, A. Andrea, J. Badii, C.T. Hind, and G. Rémy. Can edge computing fulfill the requirements of automated vehicular services using 5G network ? In *The 2024 IEEE 99th Vehicular Technology Conference (VTC2024-Spring)*, Singapore, Singapore, June 2024. IEEE.