Efficient Edge Server Placement under Latency and Load Balancing Constraints for Vehicular Networks

Sabri Khamari
Univ. Bordeaux, Bordeaux INP, CNRS,
LaBRI, UMR5800,
F-33400 -Talence, France
sabri.khamari@u-bordeaux.fr

Toufik Ahmed
Univ. Bordeaux, Bordeaux INP, CNRS,
LaBRI, UMR5800,
F-33400 - Talence, France
tad@labri.fr

Mohamed Mosbah
Univ. Bordeaux, Bordeaux INP, CNRS,
LaBRI, UMR5800,
F-33400 -Talence, France
mohamed.mosbah@u-bordeaux.fr

GLOBECOM 2022 - 2022 IEEE Global Communications Conference | 978-1-6654-3540-6/22/$31.00 ©2022 IEEE | DOI: 10.1109/GLOBECOM48099.2022.10000721

Abstract— **Vehicular applications in smart cities, such as assisted and autonomous driving, require sophisticated data processing, low latency, and high throughput data transmission. Edge Computing is a leading approach designed to meet those application requirements. By deploying Edge servers at the network's edge, close to the vehicles, such applications can be successfully delivered while adhering to low-latency and high-throughput requirements. However, optimal placement of Edge servers is challenging since it necessitates a trade-off between quality of service and deployment cost. Latency can be reduced by placing as many Edge servers as feasible close to the vehicles, however, this results in significant deployment costs. This work addresses the problem of optimal Edge server placement. It solves this problem using integer linear programming, considering the relation between delay and cost, as well as the capacity of Edge servers in realistic road traffic scenarios. The proposed generic methodology is designed to reduce the cost of deploying Edge servers by combining the achievement of the desired latency threshold with workload balancing between Edge servers. We evaluate the efficiency of the proposed solution mathematically and through simulations based on open data from real vehicles traffic on roadways of Bordeaux, France. The obtained results demonstrate that our solution outperforms existing Edge server placement approaches, especially on workload balancing.**

Keywords— Edge servers' placement, Vehicular networks, Edge Computing, Optimization.

I. INTRODUCTION

Vehicular networks are considered a significant component of future Intelligent Transportation Systems (ITS). Vehicular applications provide safer and more comfortable driving, better traffic efficiency, and a variety of value-added applications. Although vehicular applications are emerging and becoming computation-intense, they require complex processing of data and substantial storage space [1]. Usually, vehicles generate and transmit data to faraway cloud computing infrastructures with abundant resources for processing. Cloud computing enables the provision of resources on-demand, cost savings, scalability, and rapid application development [2]. However, the data transmission delays between users and the Cloud can be long and unpredictable, rendering them unsuitable for time-sensitive vehicular applications [3].

To overcome these limitations, the edge computing concept has been developed. Indeed, placing storage and compute resources at the network's boundary, as close as possible to end-users, results in lower latency and more responsive real-time operation [4]. The fundamental concept of Edge computing is to deploy a collection of Edge nodes or servers across a geographical region. As a result, processing and storage resources are made accessible to end-users, typically within a single hop.

Edge server deployment in urban vehicular networks is a major challenge. Increasing the number of Edge servers helps to reduce latency for the envisioned vehicle applications but increases cost. Edge server physical locations are crucial for ensuring high performance. Inefficient Edge server placement results in increased latency and drastically uneven workloads among Edge servers, i.e., some Edge servers are overwhelmed while others are underutilized [5]. This raises concerns regarding the optimal placement of Edge servers in metropolitan environments, where a massive volume of data is generated due to the high density of users, while considering deployment cost and network performance constraints.

This paper aims to establish an optimization model and a scientific methodology for efficiently placing Edge servers. More particularly, it proposes an integer linear programming approach for minimizing the cost of deploying Edge servers while simultaneously fulfilling the desired latency threshold and balancing workloads between Edge servers. We elaborate a methodology based on open data from real-world vehicles traffic from the city of Bordeaux to provide a realistic solution for Edge server placement [6]. This methodology can be extended to other sources of data traffic. The proposed solution is evaluated through extensive simulations and the obtained results show that our solution can effectively solve the Edge server placement problem in a real-world vehicular environment. It outperforms existing Edge server placement strategies, including Random, Top-K, K-means, and Wang et al.[7]. Our solution is based on multi criteria optimization, we satisfy the latency, the workload balancing and the cost.

The remainder of this paper is organized as follows. Section II presents related works in Edge servers' deployment. Then, we introduce the proposed model for the efficient placement of Edge servers in Section III. Section IV presents our methodology and simulation setup. Finally, we conduct performance analysis in Section V, and we provide some concluding remarks in Section VI.

II. RELATED WORK

Most existing studies on vehicular networks have addressed the cost-effective placement of roadside units principally from a

communication perspective [8]. A large share of works targeting Edge computing architecture focused on resource allocation and scheduling [9]. Even though there are a few solutions explicitly targeting Edge servers' placements in vehicular networks. The paper presented in [10] developed an optimization problem and a methodology that the infrastructure providers can use to deploy Edge computing devices in a smart city. The authors proposed a mixed-integer linear programming formulation that minimizes the deployment cost of Edge computing devices by jointly satisfying a target level of network coverage and computational demand. However, this work considers the deployment of Edge servers alongside roadside units, leading to the deployment challenges of the RSUs to ensure a good coverage ratio of the network. Even if this work aims to ensure a certain level of computational demand, it did not consider the latency requirements of vehicular applications. Wang et al. [7] studied the Edge server placement problem in mobile Edge computing environments for smart cities. They formulated the problem as a multi-objective constraint optimization problem that places Edge servers in some strategic locations to balance the workload between Edge servers and minimize the Edge server access delay. Then, they adopted mixed integer programming to find the optimal solution. Even if this work did not target the vehicular network specifically, it is the most related to the problem considered in this paper. We will consider it for comparison with our proposed solution. Laha et al. [11] considered the Edge server placement in 5G-enabled urban vehicular networks. They addressed the efficient deployment of a limited number of Edge servers in an urban scenario under a restricted budget. They considered the structural properties of the road network using complex-network-based centrality metrics and the vehicular traffic distribution to rank the candidate sites for the Edge servers. The problem is mapped to the 0-1 Knapsack problem to maximize the total profit based on the previously cited metrics. However, this work focuses on the coverage ratio, the ratio of the number of vehicles covered by the deployed Edge servers to the total number of vehicles in the network and did not consider the latency and workload balancing constraints.

Our work aims to provide an efficient Edge servers placement approach to minimize the deployment cost of Edge servers by jointly satisfying a target latency threshold and ensuring workload balancing between Edge servers.

III. Efficient Placement of Edge Servers

Before proceeding further, let us introduce the reference architecture illustrated in Figure 1. We consider vehicles moving in an urban area. These vehicles run AI-based applications and rely on the Edge servers for advanced learning capabilities depending on the collected data from the surrounding environment. Thus, vehicles communicate with roadside units (RSUs) wirelessly using ITS-G5/IEEE802.11p, while edge servers communicate with RSUs via IP broadband connection, typically via optical fiber. Multiple RSUs can be connected to a single Edge server. Edge servers process the data offloaded by vehicles. Additionally, these Edge servers are connected to the Internet, which provides extra storage and processing resources to enable the Cloud-Edge continuum. Edge servers are also connected to the road operator's platform that provides V2X services and security mechanisms (e.g. AAA and PKI).

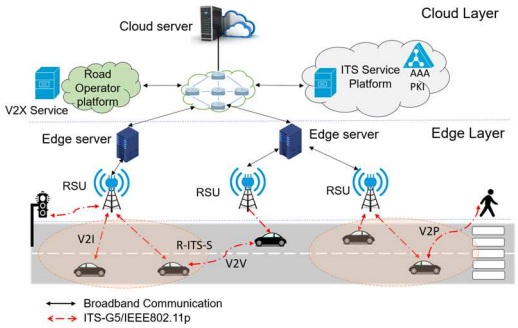

Figure 1. Vehicular Edge computing reference architecture

In the next section, we propose "OptPlacement", our solution for an efficient Edge server placement under real traffic and server's capacity constraints, followed by the scientific methodology used for evaluation.

A. *System model*

The system is composed of a set J = {S1, S2, …, Sm} of potential locations for Edge servers and a set I ={R1, R2, …, Rn} of roadside units. Vehicles transmit data requests or messages to a roadside unit, which offloads processing to the Edge server. The total number of requests received by an RSU will be utilized to calculate its demands. RSU demand represents the amount of processing, memory, and storage resources that should be available on the edge server to handle all requests received by this RSU. We suppose that vehicles send their requests with the same frequency.

Given the notation in Table 1, our objective is to minimize the cost of deploying Edge servers while ensuring a target latency requirement and a load balancing to prevent Edge servers from being overloaded. K Edge servers are to be placed among m potential locations. The following constraints are considered in our formulation:

- Each roadside unit is connected to only one Edge server.
- Each Edge server handles vehicles' requests from the roadside units.
- Each RSU has a demand which represents the sum of requests of vehicles communicating with this RSU.
- Each Edge server has limited computing and storage capacity (considered equal in this work).

Table 1. Notations

Symbol	Description
I	Set of Roadside units (RSU)
J	Set of the potential location of Edge servers
n	Number of roadside units
m	Number of potential locations for Edge servers
K	The number of deployed Edge servers
c_j	Cost of deployment of Edge server at location j
L_{max}	Maximum latency (proportional to the distance)
L_{ij}	The latency between RSUi and Edge server location j
Pr_i	Processing demands of RSU i
Mr_i	Memory demands of RSU i
Sr_i	Storage demands of RSU i
Ps_j	Processing capacity of server j

Ms_j	Memory capacity of server j
Ss_j	Storage capacity of server j
x_j	Binary variable for Edge server placed at location j
y_{ij}	Binary variable for RSUi linked to Edge server at location j
Z	The total cost of Edge server's deployment

Accordingly, we formulate an integer linear programming model that captures the features of the model under consideration. We begin by defining the decision variables of the model:

$$x_j = \begin{cases} 1, if\ server\ is\ deployed\ at\ location\ j \\ 0,\ else \end{cases} \quad (1)$$

$$y_{ij} = \begin{cases} 1, if\ RSU\ i\ is\ linked\ to\ server\ j \\ 0,\ \ else \end{cases} \quad (2)$$

$$Z = \sum_{j \epsilon J} c_j x_j \ : \text{ total cost} \quad (3)$$

The objective function is defined as:

$$Min\ Z \quad ,i.e. \quad Min \sum_{j \epsilon J} c_j x_j \quad (4)$$

s.t

$$\sum_{j \epsilon J} y_{ij} x_j = 1 , \qquad \forall i \epsilon I \quad (5)$$

$$\sum_{j \epsilon J} L_{ij} y_{ij} x_j \leq L_{max} , \qquad \forall i \epsilon I \quad (6)$$

$$\sum_{i \epsilon I} Pr_i \ y_{ij} \leq Ps_j x_j , \qquad \forall j \epsilon J \quad (7)$$

$$\sum_{i \epsilon I} Mr_i \ y_{ij} \leq Ms_j x_j \qquad \forall j \epsilon J \quad (8)$$

$$\sum_{i \epsilon I} Sr_i \ y_{ij} \leq Ss_j x_j \qquad \forall j \epsilon J \quad (9)$$

$$x_j \epsilon \{0,1\} \qquad \forall j \epsilon J \quad (10)$$

$$y_{ij} \epsilon \{0,1\} \qquad \forall j \epsilon J, \forall i \epsilon I \quad (11)$$

We explain each expression in our model, starting from the decision variables in (1) and (2), x_j denotes a boolean variable indicating whether a candidate position is selected to deploy the Edge server. y_{ij} is a boolean variable indicating which Edge server each roadside unit is connected to. For example, $y_{34} = 1$ indicates that RSU "3" is connected to the Edge server deployed in position "4". The total cost function which depends on the number of edge servers and their deployment costs is presented in equation (3). The objective function in (4) aims to minimize the deployment cost by decreasing the number of Edge servers to be deployed where c_j denotes the cost of deploying an Edge server at location j from J. Next, we describe the constraints. Equation (5) ensures that each roadside unit is connected to one of the deployed servers. Therefore, no RSU is left unconnected or connected to more than one server. Constraint (6) guarantees that each RSU is connected to an Edge server without exceeding the maximum tolerated latency L_{max} which is defined by distance. Therefore, this constraint ensures that each RSU is within the deployed Edge servers' coverage range. Constraints (7–9) ensure that memory, storage, and processor requirements are met. They ensure that the total number of requests submitted to the Edge server by roadside units does not exceed its capabilities. Finally, constraints (10-11) ensure the integrality requirements of the decision variables.

IV. Methodology and Simulation Setup

This section describes the preliminary processing necessary to generate the input data for our model. Then, the evaluation methodology and simulation setup are described.

A. *Preliminaries*

We consider the city of Bordeaux, France, as the target area to place Edge servers. A map of the Bordeaux center which bound an area of 15.10 km² is acquired from OpenStreetMap [12] (see Figure 2). To begin, we examine the placement of roadside units in this map using a uniform strategy. We used this approach because there is no available data on actual RSUs deployment. 27 roadside units were installed at road crossroads spaced by around 800 meters to achieve optimum network coverage. Next, we identify potential location for Edge servers using a straightforward technique. Between every three or four RSU, one location is assigned. This ensures that each RSU has a possible Edge server associated with it without exceeding the maximum acceptable latency L_{max} (which is defined by distance). A set of m=19 Edge server candidate locations is fed into an optimization model that determines the most strategic locations for Edge server deployment.

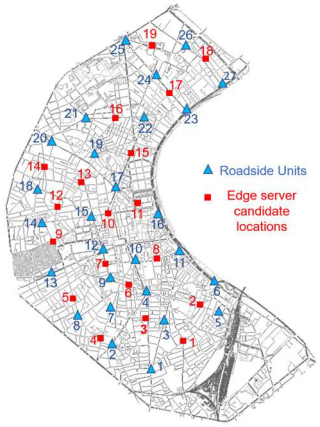

Figure 2 . Use case scenario: RSU locations and Edge server candidate locations on Bordeaux map

To perform simulations that are the most accurate representations of the real environment, we need to generate traffic traces that are identical to the actual traffic on Bordeaux's roads. We used the open dataset OpenDataBordeaux [6], which provides vehicles' numbers and speed at a geographical point. Based on this data, we identified three classes of roads: high traffic roads, moderate traffic roads, and low traffic roads. After that, we generate the road traffic trace using the RandomTrips tool provided by Simulation of Urban Mobility (SUMO) [13] along with the road classification as an input (by integrating the option Customized Weights of RandomTrips).

Once the traffic trace for our simulation is created, we determine the demands for each roadside unit. The RSU

demands are expressed by the number of messages or tasks transmitted to the RSU by the vehicles. Assuming that all vehicles transmit messages at the same rate, the RSU demands are calculated based on the number of vehicles in each RSU's covered region. Next, we perform a simulation to record the number of messages received by each RSU. Based on the results presented in Figure 3, the roadside units are grouped into three categories: high demand, moderate demand, and low demand. Table 2 presents the obtained results (Ri represents roadside unit at location i).

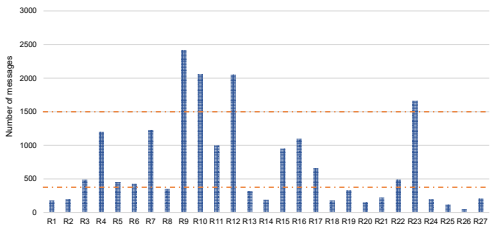

Figure 3. Roadside unit demands on Bordeaux's roads.

Table 2. RSU classification

Category	RSU
High demand (>=1500)	R9, R10, R12, R23
Moderate demand (400=<x<1500)	R3, R4, R5, R6, R7, R11, R15, R16, R17, R22
Low demand (<400)	R1, R2, R8, R13, R14, R18, R19, R20, R21, R24, R25, R26, R27

B. *Methodology and simulation setup*

Figure 4 provides a comprehensive overview of the methodology and simulation setup used to evaluate our placement strategy. The solution comprises two major parts. The first part is dedicated to solve the optimization problems utilizing AIMMS (Advanced Interactive Multidimensional Modeling System) [14] in conjunction with CPLEX12.10 solver. AIMMS is a perspective analytics platform that allows the modeling and development of optimization-based applications. An optimal solution is determined using the following inputs: the geographical coordinates of roadside units, the geographical coordinates of Edge server's candidate locations, the latency threshold (measured in distance), Edge servers' capacity, and RSUs demands (in terms of processing, memory, and storage). As a result, the solver provides the number of Edge servers to be deployed and servers' locations among the candidate locations. Additionally, the total cost of deployment is determined. Finally, one of the Edge servers is assigned to each roadside unit. The connection between the RSU and the server adheres to the latency constraint, and the aggregate of RSU demands linked to a single server must not exceed the capacity of the server. The computer used for this optimization problem has the following configuration: processor intel i7-10610U CPU @ 1.80GHz 2.30 GHz and RAM 16 GO.

The second part of this solution uses the obtained results from the first part as an input to conduct simulations using OMNeT++ [15] and the Artery framework [16]. In addition, the traffic trace generated by SUMO [13] is also required as input to the simulation. Finally, the simulation also integrates the radio model, which comprises the following characteristics: propagation model, obstacle loss model, path loss model, and the background noise model as presented in Table 3. The simulations were conducted on a computer with 16 GO RAM and the Intel i7-9700 CPU @ 3.00GHz x8 processor.

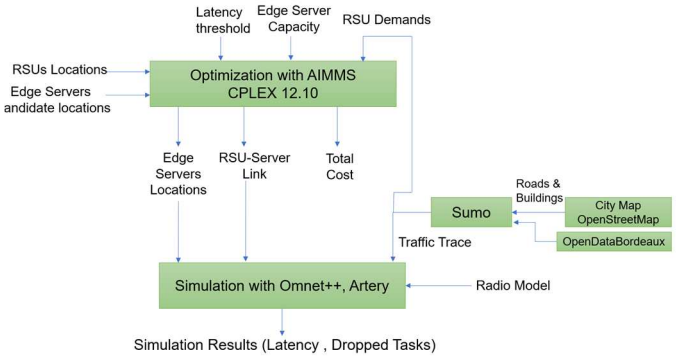

Figure 4. Overview of the used methodology to evaluate "OptPlacement".

Table 3 presents the different parameters used to find the optimal solution and to conduct simulations. The RSU's processing, memory, and storage demands depend on the roadside unit. RSU demands values are given by unit, which represents the number of received messages by second. The server capacity is expressed by unit, representing the capacity needed in terms of processing, memory, and storage to handle the received messages by second.

Table 3. Simulation parameters

Aimms parameters	
Edge server deployment cost at location i	1 unit
RSU demands (processing/Memory/Storage)	High Demand 30 unit Moderate Demand 20 unit Low Demand 10 unit
Edge server capacity (Processing/Memory/Storage)	150 unit
Latency threshold (in distance)	1500 m
OMNeT/Artery parameters	
Pathloss model	GEMv2 [17]
Propagation model	Constant Speed Propagation (speed of light)
Obstacle loss model	Dielectric Obstacle Loss
Background noise Model	IsotropicScalarBackgroundNoise (-110 dBm)
Message size (payload)	1000 Bytes
Message frequency	1 Hz

Figure 5 presents the vehicle density variation over simulation time according to the traffic trace generated from the OpenDataBordeaux dataset.

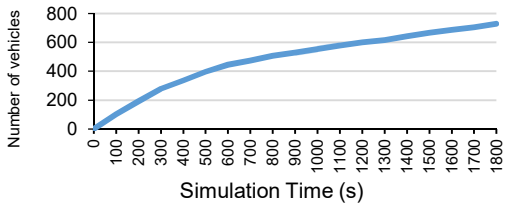

Figure 5. Vehicles' density over the simulation time.

Our methodology and solution are generic and may be used with any map or source of data traffic. According to the different inputs, the model provides the optimal solution for the efficient placement of edge servers.

V. PERFORMANCE EVALUATION

We compared the performance of our proposed approach OptPlacement with other placement strategies in terms of both latency and workload balancing. We select the following strategies: Random, top-K, and K-means strategies, as baseline comparisons because they are the most used strategies in such problems. Furthermore, Wang et al. approach [7] is also used for comparison since it is the most relevant work identified in the literature. We evaluated two variants of the random approach according to how the roadside units are connected to the Edge servers. The first one is Random-Random, where the RSUs are linked to the servers randomly. The second is Random-Nearest, where each RSU is linked to the nearest Edge server.

We measured for each strategy two critical metrics: the latency and the workload balancing. The latency metric represents the time needed to send a message from the vehicle to the Edge server. The workload balance is represented by the percentage of dropped tasks for each approach. For that, we assume that each Edge server has limited computing capacity and if the server is overloaded the arriving tasks are dropped. The workload balancing will depend principally on the number of roadside units connected to each server and their associated demands.

A. Placement results

According to our scenario, the results obtained by the AIMMS solver indicate that four (K=4) Edge servers should be placed in locations 1, 6, 12, and 19 (see Figure 2). This means that the total deployment cost of Edge servers is equal to 4 units (the optimal solution). The link between each roadside unit and the Edge server is shown in Table 4, as is the latency between the entities stated in terms of distance. For instance, Server 1, located at position 1, is used to connect R1 at a distance of 591 meters. Additionally, we show the Edge servers' workload, which is the aggregate of the demands of the roadside units connected to this server. It is 110 units for server 1.

Table 4. Optimal placement results

Edge server	Roadside ID (distance to the edge server)	Workload (units)
Server 1 (at location 1)	R1 (591m), R2 (1088m), R3 (458m), R5 (736m), R7 (1210m), R9 (1465m)	110
Server 2 (at location 6)	R4 (193m), R6 (1272m), R8 (975m), R11 (841m), R13 (1148m), R15 (1332m), R16 (1384m)	120
Server 3 (at location 12)	R10 (1441m), R12 (983m), R14 (293m), R17 (967m), R18 (439m), R19 (1036m), R20 (1120m)	120
Server 4 (at location 19)	R21 (1462m), R22 (1090m), R23 (1152m), R24 (563m), R25 (367m), R26 (550m), R27 (1275m)	100

This optimal solution is obtained in 42,11 seconds and with 129,1 Mb of used memory. The obtained results satisfy the latency and server's capacity constraints. All RSUs (R1 to R27) are connected to one of the Edge servers without exceeding the maximum latency (expressed in the distance) of 1500m which is used as an input to the model. Regarding the capacity constraint, for example, the sum of RSUs linked to server 1 is 110 units, which is less than server 1's capacity (150 units).

Table 5 below summarizes the obtained results using other placement approaches. For comparison purposes, in the Random, Top-K, and K-means approaches, we fixed K = 4, as the number of Edge servers to be deployed. This allows these approaches to be aligned with the solution obtained by our approach. Therefore, we consider that they have the exact deployment cost (K=4). In Wang et al. approach [7], two servers are deployed according to the optimal solution of the proposed model. In this approach, the maximum latency threshold is not considered, and therefore, two servers are deployed to reduce the cost.

Table 5. Placement results of the other approaches

Approach	Edge server (placement location)	Workload
Top-K	Server 1 (at location 7)	110
	Server 2 (at location 8)	160
	Server 3 (at location 10)	80
	Server 4 (at location 17)	100
K-means	Server 1 (at location 6)	150
	Server 2 (at location 8)	80
	Server 3 (at location 13)	90
	Server 4 (at location 17)	90
Wang et al. [7] approach	Server 1 (collocated with RSU 3)	250
	Server 2 (collocated with RSU 15)	200

B. *Performance results*

As previously stated, the considered metrics are the latency and the percentage of the dropped tasks to evaluate workload balance between servers. Figure 6 and Figure 7 depict the latency results obtained by the simulations for the aforementioned approaches. The findings show the average latency by second of messages received by the servers. Our approach OptPlacement achieves a latency that varies between 1,51ms in low density and 1,61ms in high density. The other approaches offer a latency in approximately the same interval. When we examine the findings more closely, we see that OptPlacement outperforms Random-Random and Wang et al. [7]. Additionally, Random-Nearest and OptPlacment have superposed curves and provide similar results. On the other hand, our solution presents a higher latency of 0,01ms (maximum) compared to Top-K and K-means approaches. This could be explained by the fact that those approaches use the nearest server criteria, i.e., each RSU is connected to the nearest server without considering other constraints like the server capacity and the workload balancing. For example, as depicted in Table 5, in the Top-K approach, server 2 with the capacity of 150 units, cannot handle all the tasks of the roadside units connected to it (160 units of total demands). Therefore, this minor increase of latency is acceptable for our solution to ensure that the server's capacity is not exceeded. On the other hand, the proposed model ensures that the latency threshold (L_{max}) constraint is always respected.

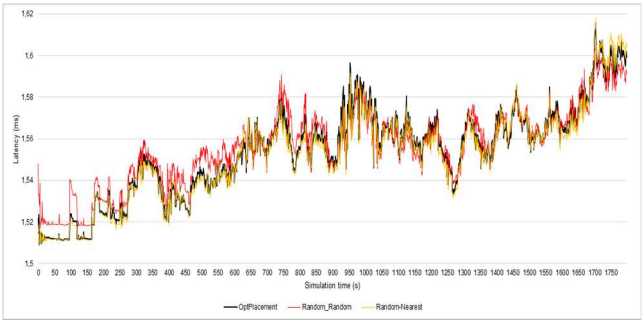

Figure 6. Latency simulation results: OptPlacement, Random-Random and Random-Nearest.

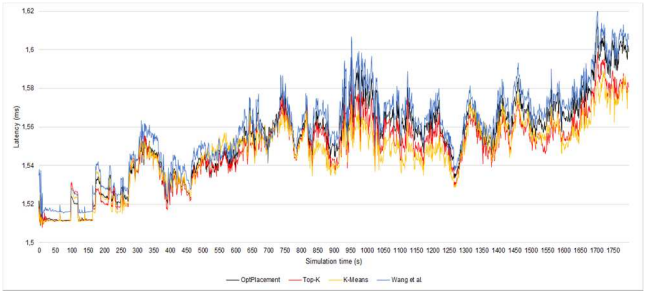

Figure 7. Latency simulation results: OptPlacement, Top-K, K-Means and Wang et al. [7].

To conduct a more thorough examination of the placement methods' performance, we investigated workload balancing amongst the Edge servers. If a server is overloaded the next arriving task is dropped. Figure 8 presents the percentage of dropped tasks for each approach. The results show that our approach can deliver the best performance as 2,90% of tasks are dropped. The K-Means and Top-K approaches have shown slightly better results in terms of latency. However, their performance in terms of workload balancing is not promising. The Top-K approach has 5,67% of dropped tasks, which is 95,5% higher than our approach, and the K-means strategy has 13,06% of dropped tasks. The percentage of dropped tasks in the Wang et al. approach [7] is increased by 394% compared to our approach. Those results validate that our proposed model can provide a better workload balancing compared to the other Edge server placement strategies while ensuring the latency constraint.

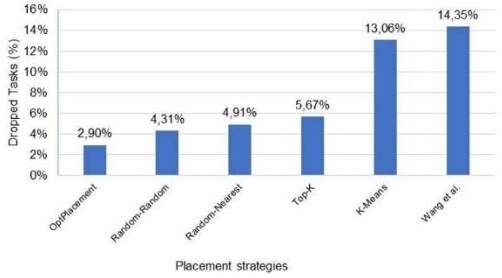

Figure 8. Dropped tasks results.

VI. CONCLUSION

Edge computing is an important emerging technology that can be used to improve the performance of a wide variety of applications, particularly vehicular applications. This paper provides an effective strategy for Edge servers' placement. To give a realistic Edge server location, we used a methodology based on the real-world traffic data set for Bordeaux. Our model is generic and applicable to other maps and different data set of traffic. Using AIMMS analytical platform, we obtained the optimal solution to the placement problem. Then, we conducted simulations to evaluate the solution's performance. The obtained results showed that our proposed solution is effective and outperforms existing approaches by ensuring load balancing among Edge servers and minimizing deployment costs. Therefore, our solution decreases the number of dropped tasks by an average of 56% compared to the other placement strategies. Furthermore, our solution satisfies the constraint of the maximum latency that could not be exceeded. Accordingly, our proposed strategy for Edge server placement achieves an optimal trade-off between deployment cost, workload balancing, and latency. As future work, we intend to investigate the service migration between Edge servers to ensure service continuity and decrease application delay.

REFERENCES

[1] LIU, Lei, CHEN, Chen, PEI, Qingqi, et al. Vehicular edge computing and networking: A survey. Mobile Networks and Applications, 2021, vol. 26, no 3.

[2] APOSTU, Anca, PUICAN, Florina, ULARU, et al. Study on advantages and disadvantages of Cloud Computing–the advantages of Telemetry Applications in the Cloud. Recent advances in applied computer science and digital services, 2013.

[3] JAVADZADEH, Ghazaleh et RAHMANI, Amir Masoud. Fog computing applications in smart cities: A systematic survey. Wireless Networks, 2020, vol. 26.

[4] CAO, Keyan, LIU, Yefan, MENG, Gongjie, et al. An overview on edge computing research. IEEE access, 2020, vol. 8, p. 85714-85728.

[5] LI, Xingcun, ZENG, Feng, FANG, Guanyun, et al. Load balancing edge server placement method with QoS requirements in wireless metropolitan area networks. IET Communications, 2021, vol. 14, no 21, p. 3907-3916.

[6] OpenDataBordeaux, Bordeaux Métropole, https://opendata.bordeaux-metropole.fr/" [Accessed: 11 Juin 2021]

[7] WANG, Shangguang, ZHAO, Yali, XU, Jinlinag, et al. Edge server placement in mobile edge computing. Journal of Parallel and Distributed Computing, 2019.

[8] ZHANG, Rui, YAN, Feng, XIA, Weiwei, et al. An optimal roadside unit placement method for vanet localization. In : GLOBECOM 2017-2017 IEEE Global Communications Conference. IEEE, 2017. p. 1-6.

[9] SORKHOH, Ibrahim, EBRAHIMI, Dariush, ATALLAH, Ribal, et al. Workload scheduling in vehicular networks with edge cloud capabilities. IEEE Transactions on Vehicular Technology, 2019, vol. 68, no 9.

[10] PREMSANKAR, Gopika, GHADDAR, Bissan, DI FRANCESCO, Mario, et al. Efficient placement of edge computing devices for vehicular applications in smart cities. In : NOMS 2018-2018 IEEE/IFIP Network Operations and Management Symposium. IEEE, 2018. p. 1-9.

[11] LAHA, Moyukh, KAMBLE, Suraj, et DATTA, Raja. Edge nodes placement in 5g enabled urban vehicular networks: A centrality-based approach. In : 2020 National Conference on Communications (NCC). IEEE, 2020. p. 1-6.

[12] Open Street Map France, "https://www.openstreetmap.fr/" [Accessed: 11 Juin 2021]

[13] LOPEZ, Pablo Alvarez, BEHRISCH, Michael, BIEKER-WALZ, Laura, et al. Microscopic traffic simulation using sumo. In : 2018 21st International Conference on Intelligent Transportation Systems (ITSC). IEEE, 2018. p. 2575-2582.

[14] AIMMS, "Download AIMMS developer", 2020. [Online]. Available: https://www.aimms.com/english/developers/downloads/download-aimms/. [Accessed: 11 Juin 2021].

[15] VARGA, OMNeT++ Discrete Event Simulation System, Release 5.6, 2020.

[16] Riebl, Raphael, Obermaier, Christina, et Gunther, Hendrik-Jörn. Artery: Large Scale Simulation Environment for ITS Applications. In: Recent Advances in Network Simulation. Springer, Cham, 2019. p. 365-406

[17] BOBAN, Mate, BARROS, Joao, et TONGUZ, Ozan K. Geometry-based vehicle-to-vehicle channel modeling for large-scale simulation. IEEE Transactions on Vehicular Technology, 2014, vol. 63, no 9, p. 4146-4164.