Effect of RSU Placement on Autonomous Vehicle V2I Scenarios

B. KARA and A. ÖZGÖVDE

***Abstract*— Edge computing has become a prominent computing strategy as mobile devices and Internet of Things (IoT) became popular in the last decade where cloud computing proved partly insufficient meeting the computational requirements of these devices/applications. Unlike cloud, edge computing can provide low latency in communication, high quality of service, and support for high mobility. Connected and autonomous vehicles scenarios can be considered as an important application field for edge computing as these are the key requirements to implement a vehicular network. In this paper, we aim to present a remedy to one of the crucial problems in vehicular networks: efficient RSU placement by addressing network coverage and computational demand. We propose an RSU placement framework for generating placement models based on traffic characteristics of a target area. Our work differs from previous studies in that we focus on both communication coverage and the computational demand aspects simultaneously. The proposed framework in this study can be used by infrastructure providers for designing an efficient RSU placement while building a smart city. Moreover, our work includes extending capabilities of a simulation framework designed for edge computing scenarios. To demonstrate the effectiveness of our proposal we evaluated the performance of various placement models in realistic settings.**

Index Terms— **Road Side Unit (RSU), Edge Computing, V2I (Vehicle to Infrastructure)**

I. Introduction

WITH THE increasing popularity of mobile devices and Internet of Things (IoT) during the last decade, cloud computing had been leveraged to solve the problem of making complex computations with limited device resources by provisioning remote computing and storage resources. Edge computing, on the other hand, was suggested as a new computing paradigm when the limitations of the centralised data centres started to emerge. Satyanarayanan et al. describe these limitations as long WAN latencies and bandwidth-induced delays [1]. Because of these limitations, cloud computing is not a suitable computing strategy for scenarios which requires real-time data processing and relies on fast feedback. Edge computing is a good candidate to solve these problems by bringing computing resources to the edge of the network, usually one hop away from the user. The features of low latency in communication, high quality of service and support for high mobility makes edge computing an optimal solution for the computational requirements of a wide range of applications in different domains. Connected and autonomous vehicles scenarios are considered as a good application field for edge computing [2]. Fig. 1 shows the system components in a reference scenario.

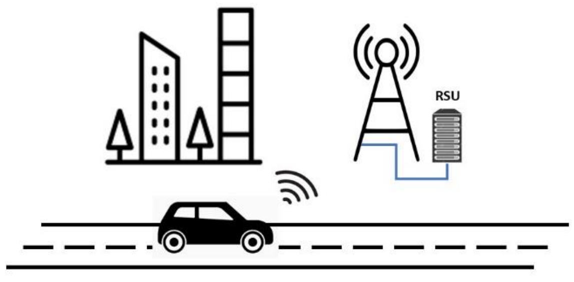

Fig.1. System components for the reference scenario

Using their advanced sensors, connected vehicles collect data from their environments. In current state of automotive technology, vehicles process this data to interpret their environment and enable assisted and autonomous driving for a safe navigation. For example, using their ultrasound, infrared, radar and video sensors, vehicles can detect other vehicles on the road, stop for pedestrians, and handle any unexpected circumstances [3]. On the other hand, the automotive industry is working to develop Vehicular Ad-hoc Networks (VANETs), to enable vehicles to share information with other vehicles and road side units (RSUs) through vehicle-to-vehicle (V2V) and vehicle-to-infrastructure (V2I) communication channels [4]. VANET is an essential part of Intelligent Transportation Systems (ITS) which are defined as the future of transportation.VANETs can be utilized for a broad range of safety and non-safety applications, allow for value added services such as vehicle safety, automated toll payment, traffic management, enhanced navigation, location-based services such as finding the closest fuel station, and infotainment applications such as providing access to the internet [5].

Dedicated short-range communication (DSRC), which is a candidate for use in a VANET, offers the potential to

BARIŞ KARA, is with Department of Computer Engineering Galatasaray University, Istanbul, Turkey,(e-mail: bariskara35@gmail.com).
https://orcid.org/ 0000-0002-2759-7447

ATAY ÖZGÖVDE, is with Department of Computer Engineering Galatasaray University, Istanbul, Turkey,(e-mail: aozgovde@gsu.edu.tr).
https://orcid.org/ 0000-0001-9688-766X

Manuscript received December 31, 2019; accepted June 11, 2020.
DOI: 10.17694/bajece.668615

effectively support V2V and V2I safety communications by providing high data transfer rates with minimum latency [6]. The primary motivation for deploying DSRC is to enable collision prevention applications. These applications depend on frequent data exchanges among vehicles, and between vehicles and roadside infrastructure.

Road Side Units (RSU) are the communication units in VANET which are fixed along the road side or in dedicated locations such as at the junctions or near parking spaces [7]. They are equipped with an antenna to enable wireless communication based on IEEE 802.11p radio technology, a processor, and a read/write memory [8]. Barskar et al. describe main functions and procedures associated with the RSUs as follows [7]:

- To extend the communication range of the ad hoc network for redistributing the information to other vehicles
- Running safety applications and acting as an information source
- Providing internet connectivity to the vehicles

The components of the V2I scenarios can be mapped to edge computing elements as follows:

- RSUs are the edge computing units because of their proximity to the vehicles, providing computational, storage resources and high bandwidth link, and transfer data with minimum latency
- Vehicles are the resource poor clients as they have limited computation and storage resources due to the requirements of small-size and low-cost hardware systems (Yu et al., 2013)
- Vehicular applications are edge applications as they demand complex computation and large storage

Applications collecting information from multiple vehicles have a great potential of increasing road safety and improving quality of traffic. Satyanarayanan proposes a scenario in which crowd sourcing and edge computing can be harnessed to create a shared real-time information system for situational awareness [9]. They claim that collected information can be used to detect critical situations such as accidents, icy road conditions, fallen rocks and advisory messages can be conveyed to the other drivers. Another study proposes an application for intelligent traffic management at intersections to minimize accidents, traffic congestion and environmental costs of road traffic using V2V and V2I communications [10]. Katsaros et al. also design an application that could improve fuel consumption and reduce traffic congestion in junctions using vehicular data through same communication channels [11]. Another study proposes a merging algorithm that optimizes the performance of connected fully automated vehicles through a freeway merging segment for a scenario relying on V2V and V2I communications [12].

All these applications deployed into RSUs receive data from vehicular applications such as trajectory, speed, destination coordinates, etc. in short intervals, aggregate and process it in real time and send response back to senders or to the relevant vehicles within the network range. Here again, low latency and high quality of service are the key factors to build this ecosystem.

RSUs placed in an area should meet two requirements. First, network coverage of the area should be maximised, so that vehicles can stay connected to the RSUs at any time during their journeys and edge applications can work without excluding any territories. Second, edge computing units have limited resource capacities compared to the cloud datacentres [13] and computational demand of the edge applications should be met by the RSUs. It is expected to observe different levels of traffic density in different parts of an area. Placing insufficient number of RSUs in a territory with high traffic volume creates computational demand more than RSUs can handle and this could result in system failure. On the other hand, placing more RSUs than required in a territory with low traffic volume could result in waste of resources and loss of money.

Deploying a specific number of RSUs into an area is a challenging work since satisfying two requirements at the same time brings us to a trade-off problem. RSUs should be placed in an area in a way that satisfies both network coverage for vehicles and computational demand for the edge applications at maximum level considering the traffic density on the road network.

As to be mentioned in Section II, majority of the existing works address RSU placement problem from communication aspect without considering resource consumption of the edge applications. On their survey addressing Mobile Edge Computing, Mach et al. describe the issue of finding an optimal way where to physically place the computation depending on expected user demands as an open research challenge [14].

The objective of this study is to implement an RSU placement framework for generating RSU placement models based on traffic characteristics of an area. We aim to provide a flexible tool that can be configured for designing a placement model in favour of network coverage or computational demand. Additionally, our work includes extending capabilities of an open source simulation framework, EdgeCloudSim[1], proposed to evaluate the performance of edge computing scenarios. By adding new modules to support simulations for V2I scenarios and designing realistic traffic scenarios for a target area in London city centre, we evaluate the performance of the generated placement models and validate their functionality [15].

Simulation results show that generated models satisfy network coverage and resource demand in different levels, and can be used to find the optimal placement of the RSUs in the target area. Therefore, our framework can serve as a reliable tool to be used as part of RSU deployment process by infrastructure providers.

The rest of the paper is organised as follows: Section II explains previous RSU studies addressing RSU placement and Edge Computing in Vehicular Networks. In Section III, we describe preliminary work including processing target area map and generating traffic dataset, then, our reference scenario, proposed placement framework and simulation environment is explained. Section II discusses the simulation

[1] https://github.com/CagataySonmez/EdgeCloudSim

results and validity of the proposed framework. Finally, we conclude the paper and outline the future work in section III.

II. LITERATURE REVIEW

A. RSU placement

Trullols et al. suggest a maximum coverage approach to the problem of information dissemination in intelligent transportation systems in their study, which can be considered as one of the earliest works addressing this topic as most of the research efforts had focused on the development of protocols and applications suitable for VANET until that period of time [16]. In their study, they propose a heuristic algorithm to solve the problem of maximizing the number of vehicles that get in contact with the Dissemination Points (DPs). Their results also show that, their suggested heuristics can be successfully employed to plan a deployment capable of informing more than 95% of vehicles with a few DPs.

Aslam et al. present two different solutions to the RSUs placement problem with objective of maximizing the information flow from vehicles to RSUs in an urban environment: Binary Integer Programming (BIP) method and a novel Balloon Expansion Heuristic (BEH) method [17]. BIP method utilizes branch and bound method to find optimal solution, whereas, BEH method uses balloon expansion analogy to find optimal solution. Both optimization methods were used to solve the optimization problem of minimizing the average reporting time. They have shown that the novel BEH method is more versatile and can be used to solve the optimization problem.

Balouchzahi et al. also propose an optimization method addressing RSU placement by formulating the problem using BIP [18]. In their work, highway and urban scenarios are separately formulated to improve the model scalability. Their simulation results show that the proposed model reduces the receiving time of traffic information and can reach to a satisfactory level of coverage using less RSUs.

Wu et al. tackle the same problem by presenting a placement strategy referred as Capacity Maximization Placement (CMP) based on Integer Linear Programming (ILP). Apart from direct communication of RSUs and vehicles, their study also covers multi-hop relaying, which takes place when the vehicle is out of RSU's transmission range [19]. To validate their findings, they compare the results of CMP with two other models: uniformly distributed placement and hot spot placement. The simulation result shows that the proposed model leads to the best performance among all mentioned models.

Our study differs from aforementioned works in a way that they only address the problem from communication and network coverage aspects without taking resource consumption and computational demand of the RSUs into account. Although a placement model can be optimized enough for a cost efficient RSU deployment in an area and provide a quality of communication at a certain level, it is not guaranteed that it can handle computational demand of the edge applications.

B. Edge Computing in Vehicular Networks

Yu et al. propose a hierarchical cloud architecture for vehicular networks [20]. Their architecture consists of central clouds, roadside cloud and vehicular cloud. Central clouds have sufficient cloud resources but large end-to-end communications delay. On the contrary, roadside and vehicular clouds have limited cloud resources but satisfy communications quality. In their study, they focus on efficient resource management in the proposed architecture and they formulate and solve resource competition among virtual machines in a game-theoretical framework

In their study, Datta et al. seek an alternative of cloud platform to support real time connected vehicular scenarios [21]. They design an IoT framework that includes an edge computing system for the connected vehicles to offer consumer centric services. Their framework primarily utilizes an edge computing platform to support network switching, resource discovery, provisioning, local processing for data fusion and storage of the high-level intelligence for vehicular scenarios. Salahuddin et al. present RSU Clouds as a novel way to offer non-safety application with QoS for VANETs [22]. RSU Clouds consist of traditional RSUs and micro datacentres. Their system can be reconfigured, at a cost, to meet the fluctuating service demands like cloud datacentres. They also focus on concepts such as resource management, minimizing VM migrations, control plane overhead, number of service hosts and infrastructure delay for their proposed architecture.

Although the research described in this section address edge computing in vehicular networks, the researchers mostly suggest new frameworks and architectures in which cloud and edge processing units, and mobile devices/vehicles are integrated into a new ecosystem. Then, they suggest solutions for computational challenges such as resource allocation, scheduling, VM migration, etc. for the computational resources. Our work can be considered as a complementary study in which we focus on provisioning a V2I infrastructure built on top of an existing architecture. Therefore, we assume that low-level computation and communication problems are resolved and we can propose solutions for higher level challenges such as efficient RSU placement

II. MATERIALS AND METHODS

A. Reference Scenario

In our reference scenario, we consider a smart city equipped with RSUs and support V2I communication. All the vehicles are smart or connected with the ability of running vehicular applications that connect to edge applications deployed into RSUs. Vehicular applications send one task to the nearest RSU per second in case the vehicle is in the network coverage of any RSU. When the task is successfully processed, RSU sends a response back to the vehicular application. There are 4 cases a task can fail:

• Coverage: Vehicle is not in range of any RSU's network

• Capacity: RSU is out of capacity and cannot process incoming task
• Bandwidth: Task cannot be sent through network due to congestion
• Mobility: Vehicle leaves the RSU network coverage after sending the task

We assume all RSUs have same hardware resources and the tasks sent by the applications are identical. In our scenario, each RSU has 1 Mbps bandwidth. Average task payload size is 1024 bytes for both upload and download operations. We also assume that each RSU has an equipped server with 600Mhz CPU and 500MB RAM, and average task length is 300 MI. Table 1 shows the parameters and their values for RSU and task configurations.

TABLE I
RSU AND TASK PARAMETERS AND VALUES

Parameter	Value
RSU Network Range	300m
RSU Bandwidth	1 Mbps
CPU	600 Mhz
Memory	500 MB
Average Task Payload Size	1024 byte
Average Task Length	300 MI
Task arrival rate	1 Hz

The simulations we run are based on these assumptions and parameters.

B. Preliminary Work

1) Target Area

For our scenario, we chose London city centre as the target area for deploying RSUs which covers an area of 3 by 3 kilometres. To be able to run traffic simulations and calculate RSU locations, we needed to extract the road network of the target area. To obtain the road network, we outlined the target area on OpenStreetMap which is a free collaborative map application, then we exported it in xml format. Since the map data includes a variety of information such as buildings, parks, restaurants, etc., we processed the file to only include road network elements such as motorways, intersections and traffic lights. Fig. 2 shows the map of the target area.

2) Traffic Dataset

Due to the lack of publicly available vehicle trajectory dataset for the target area, we used Simulation of Urban Mobility (SUMO) framework to generate realistic traffic dataset. SUMO is an open source, microscopic and continuous road traffic simulation framework designed to handle large road networks [23].

Apart from its simulation capabilities, SUMO includes several scripts for traffic and road network operations. We converted the map data into a network file, which is the SUMO input format that defines the road network. Then, we used randomTrips tool, which is a python script that takes place in SUMO library, to generate the vehicle routes randomly on the road network. The output route file, along with the network file should be provided to SUMO to run the traffic simulation.

We defined two important parameters during route generation: simulation time and vehicle arrival rate. The simulation time we chose as 1 hour, aligns with the time of V2I simulation we conducted in the following steps. Vehicle arrival rate, on the other hand, defines the number of vehicles in the simulation and SUMO generates one specific route for each vehicle. This parameter is set to 1 by default. In our study, traffic density plays an important role on RSU placement process as the load on the RSUs depends on number of vehicles in the system. Thus, to cover scenarios with different traffic volumes, we run the script using different arrival rates. As a result, we generated 8 route files which include 500, 1000, 1500, 2000, 2500, 3000, 3500, and 4000 vehicles routes.

After that, by running SUMO traffic simulation for each route file for a simulation time of 1 hour, we produced 8 traffic output files which comprise our traffic dataset. Each file contains traffic data logged for each simulation second such as vehicle id, type, coordinates, speed, angle, lane, etc. As a result, 8 million logs were produced in total for the traffic dataset.

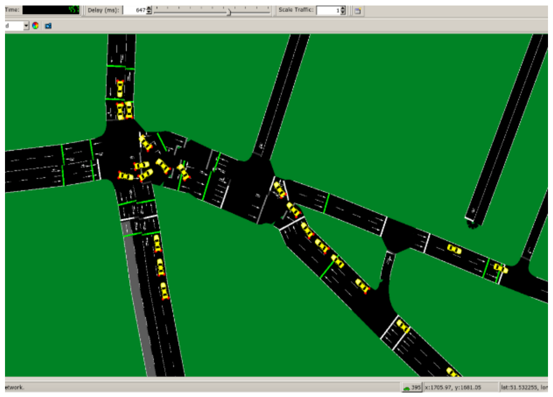
Fig. 2. SUMO traffic simulation

C. RSU Distribution Models

In this study, we propose 3 RSU distribution models: Uniform, Weighted, and Optimized. This section outlines the algorithms, implementations and results of the models.

1) Uniform RSU Distribution

This placement model only aims for full RSU network coverage by placing RSUs equidistant from each other without considering computational demand. Network range of the RSUs can reach up to 1000 meters ifthere are no obstructions, and 250-350 meters in cluttered urban areas [24]. For this model, we assumed that each RSU works best with a coverage of 150 meters due to the shadowing effect of the buildings, and we decided to place RSUs 300 meters far from each other. Therefore, to cover an area of 9 km² with RSUs working in their best performances, we needed to have 100 RSUs.

We developed a Java application as the implementation of the algorithm and referred it to *RSU Distributor*. In this application, we generated a grid on the area map by dividing it into cells each with the size of 300x300 meters. We referred to these cells as territories. Then, we placed one RSU into the centre of each territory, therefore 100 RSUs were evenly distributed to the area. Fig. 3. shows the RSU locations on the target area map based on the uniform distribution. It should be noted that, as its name suggests, an RSU should be placed on the road side to ensure the proximity to the vehicles. However, in an urban scenario in which a complex road structure exists, a territory includes multiple roads and we expect the placed RSU to serve to the vehicles across multiple roads within the coverage area.

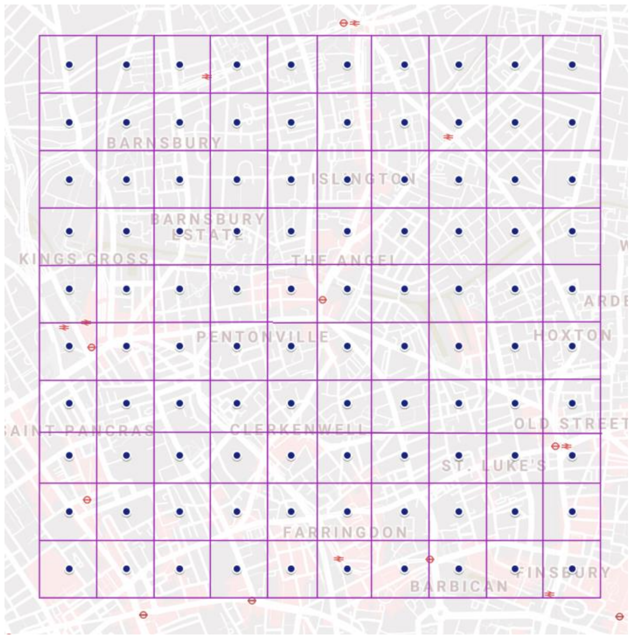

Fig. 3. RSU locations on uniform distribution model

2) Weighted RSU Distribution

We used Uniform Distribution model as base model to generate Weighted RSU distribution with a heuristic approach. This model addresses refining RSU locations set up for uniform distribution model by taking computational demand into account. In the uniform distribution model, despite of the full network coverage, we might have high task failure rates since RSUs might not meet the high computational demand using their limited resources. We may especially experience this problem in territories with higher traffic volumes i.e. traffic congestions. An external parameter, θ, is the relocation factor and determines number of the RSUs to be relocated. Relocation step addresses selecting θ% least utilized RSUs and move them to the territories where more computational resources are needed.

Thus, we aim to decrease resource originated task failures by bringing additional computational resources to meet the higher demand. On the other hand, relocated RSUs will cause coverage originated task failures as no RSUs will serve to vehicles at these territories. Value of θ should be assigned considering the difference of traffic volumes in different territories as this trade-off is only reasonable if total number of task failures decreases after the relocation.

a) Algorithm

The algorithm for this placement model consists of 4 steps:

- **RSU Selection:** This step addresses finding the RSUs placed at the territories with lower traffic volume, thus have low utilization rates. To detect these RSUs, we calculate task assignment rates for each RSUs in the uniform distribution. The RSUs with less task assignment rates are marked to be moved in the territories with higher resource demand. We select θ% least utilized RSUs in this step.
- **Territory Selection:** To detect the territories that need additional resources to meet the high computational demand, we analyse the performance of the RSUs in uniform distribution model under a heavy load. The territories containing the RSUs with higher task failure rates due to insufficient capacity are the candidates to support with additional RSUs.
- **RSU Distribution:** In this step, we first calculate a weight factor using task failure rates for each candidate territory. Then using the weight factor, we calculate number of RSUs to be assigned into each territory. Finally, we distribute the selected RSUs into these territories.
- **RSU Placement:** This step addresses placing selected RSUs into the candidate territories. The first RSU is placed in the middle of territory centre and neighbour territory centre with the highest computational demand among all neighbours. The second RSU is placed between the territory centre and neighbour territory centre with the second highest computational demand, and so on.

As explained above, to refine RSU locations using weighted distribution algorithm, we need to have two metrics from uniform distribution model: task assignment rates and task failure rates of the RSUs. To gather these results, we run V2I simulations on the target area using uniform distribution model and traffic dataset.

b) Simulation Framework

We used EdgeCloudSim as simulation framework which is an open source tool developed by Sonmez et al. [15] to conduct experiments for edge computing scenarios. We extended the capabilities of the framework by defining components and modules specific to V2I scenarios and we referred to this extended simulation environment as V2ISim.

The simulation environment served for two purposes in our study: first, by running simulations for uniform distribution model, we generated the inputs required for weighted distribution algorithm. Second, we needed a simulation environment to make experiments with generated distribution models, therefore we can evaluate the system performance and compare results in the following steps of the study.

TABLE II
SIMULATION PROPERTIES

Property	Value
Total number of traffic logs	8.147.468
Total number of RSUs	100
RSU placement model	Uniform
Simulation time	1 hour

c) Simulation for Uniform Distribution Model

The simulation environment requires two input files: vehicle trajectory data and RSU coordinates. As traffic input data, we provided the traffic dataset we generated using SUMO and as RSU coordinates, we used the coordinates we calculated for uniform distribution model.

We also configured RSU and task characteristics by providing parameters listed in Table 2. We set simulation time to 1 hour and 8.147.468 traffic logs were provided in the traffic dataset as total. Some important simulation properties can be seen in Table 2.

d) Simulation Results

It took 6 hours 10 minutes to run the simulation for uniform distribution model on a laptop with Intel Core i7-8850H CPU and 16GB RAM.

As a result of a simulation 3 output files are generated per traffic input file from the traffic dataset:

- **Generic logs:** this file includes most important simulation results such as number of successfully processed tasks, number of failed tasks, average service time, average network delay and average RSU utilization rate. The values logged in this file are used as metrics while comparing system performances for different RSU placement models.
- **RSU utilization logs:** this file keeps the utilization rates for each RSU logged for each simulation second. This values are used as metrics while comparing system performances from utilization aspect for different RSU placement models.
- **Task assignment logs:** this file is only generated for uniform distribution model simulation and it keeps the logs of number of assigned and failed tasks for each RSU. By processing these values, we can generate the 2 inputs required for weighted distribution algorithm: task assignment rates and task failure rates of the RSUs

e) Weighted Distribution Algorithm Implementation

We extended *RSU Distributor* Java application to implement weighted distribution algorithm. From 500 to 4000 vehicles, the simulation run once for each traffic input file and as a result, 8 task assignment log files which include more than 8 million task logs were produced in total. In the application, these logs were aggregated and processed to find the values of task assignment rates and task failure rates of the RSUs.

The number of RSUs we want to select and distribute into new cells are based on the value of θ, relocation factor. By providing 10, 20, and 30 for θ, we run the application and generated 3 different set of RSU placement models for weighted algorithm. For each value of θ, Table 3 shows the selected RSUs for relocation and Table 4 shows number of RSUs to be assigned to each territory.

TABLE III
RSU IDS SELECTED FOR RELOCATION

θ	RSU ids
10	3, 11, 39, 4, 9, 49, 5, 90, 88, 2
20	3, 11, 39, 4, 9, 49, 5, 90, 88, 2, 74, 89, 79, 69, 1, 91, 6, 70, 93, 12
30	3, 11, 39, 4, 9, 49, 5, 90, 88, 2, 74, 89, 79, 69, 1, 91, 6, 70, 93, 12, 84, 98, 92, 87, 8, 99, 14, 59, 80, 19

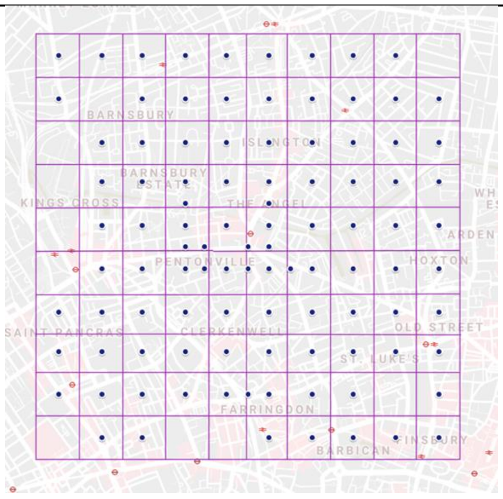

Fig. 4. RSU locations on weighted distribution model for θ = 10

TABLE IV

TERRITORY IDS AND NUMBER OF RSUS TO ASSIGN

θ	Territory ids and number of RSUs to assign
10	55(2), 54(1), 45(1), 35(1), 48(1), 33(1), 34(1), 65(1), 53(1)
20	55(3), 54(2), 45(2), 35(2), 48(2), 33(1), 34(1), 65(1), 53(1), 58(1), 47(1), 46(1), 75(1), 36(1)
30	55(4), 54(3), 45(3), 35(3), 48(3), 33(2), 34(1), 65(1), 53(1), 58(1), 47(1), 46(1), 75(1), 36(1), 25(1), 71(1), 38(1), 63(1)

Fig 3-5 shows the weighted distribution RSU placements for θ=10, 20, and 30, respectively.

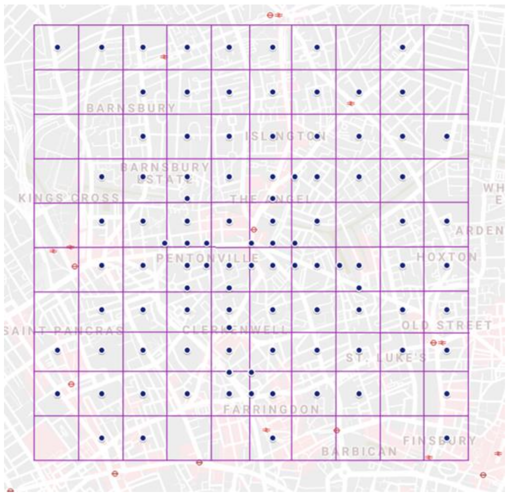

Fig. 5. RSU locations on weighted distribution model for θ = 20

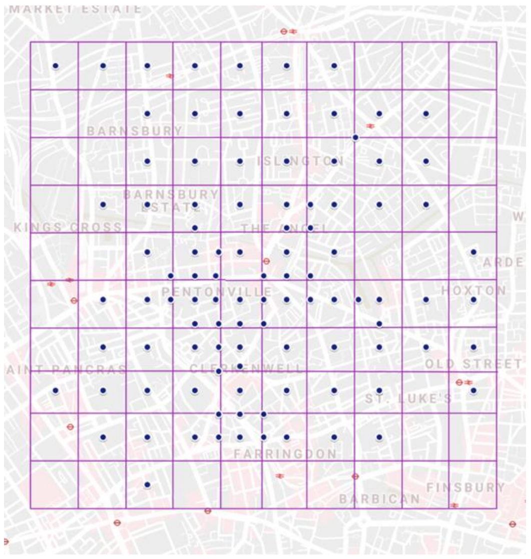

Fig. 6. RSU locations on weighted distribution model for θ = 30

3) *Optimized RSU Distribution*

As previously discussed, we have two criteria to fulfill while solving the RSU placement problem efficiently: network coverage and computational demand. Our approach for the optimized placement model is to use *Linear Programming* (LP) to address both of the requirements.

a) Algorithm

Similar to other models, optimized RSU distribution model also does its calculations on a grid generated on the target area map. For this purpose, we used the same grid as we generated for the uniform RSU distribution model to provide consistency. It should be noted that, working with smaller cell size would provide fine-grained results, however this results in an exponential growth on the number of formulations to define the mathematical model on LP.

Binary Integer Programming (BIP), is a subtype of Linear Programming in which all decision variables are defined as binary. In our problem, each cell is a candidate for placing an RSU, meaning that we want to solve the problem that decides whether a cell has an RSU or not. Therefore, our decision variables refer to the condition of each cell having the value of 1 or 0, where 1 states that RSU should be placed, and 0 should not. As a result, we formulated the RSU placement problem using BIP.

b) Problem Formulation

For the grid consisting of 100 cells, the we define the decision variables as follows:

$$x_0, x_1, x_2, \dots, x_{99}$$

In the second step, we define the objective function using the decision variables. Our objective is fulfilling the constraints using minimum number of RSUs. Thus, the objective function is:

$$min \sum_{i=0}^{99} x_i$$

Finally, we define the constraints in which we model network coverage and resource demand as well as the criteria we want to set as their minimum values.

(1) Network Coverage

We start by formulating the total coverage (*R*) of the RSUs using the decision variables. As previously stated, the network range of an RSU is between 250-350 meters in the urban areas (Ligo et al., 2015). The cells have the size of 300x300 meters, whereas we consider the network range of an RSU as also 300 meters. Fig. 7 depicts the positioning of an RSU within a cell along with its network range. As it can be seen in the figure, the area of the RSU's network coverage exceeds the area of the cell. In this situation, when two RSUs are placed within the neighbour cells, there will be an overlap on the area coverage, and an optimized placement solution should minimize these overlaps. Total coverage is defined with this equation:

$$R = \sum_{i=0}^{99} x_i r_i - \sum_{i=0}^{99}\sum_{j=0}^{99} x_i x_j p_{i,j} + \sum_{i=0}^{99}\sum_{j=0}^{99}\sum_{k=0}^{99} x_i x_j x_k p_{i,j,k}$$

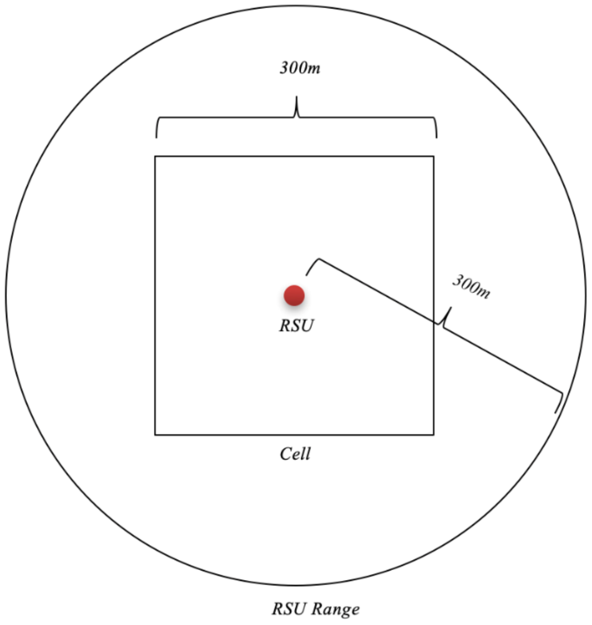

Fig. 7. The positioning of RSU within a cell

In the equation, first summation operation denotes addition of network coverage for all the placed RSUs (*i*). Then, as explained above, the overlapped areas as a result of neighbourhood should be subtracted from this sum, and the double summation operation indicates that (*ii*). According to this Fig. 8 which shows 4 example cells, the neighbourhood, which causes network overlapping, exists when RSUs are placed into these cells: A-B, A-C, A-D, B-C, B-D and C-D. Lastly, when there is a case of "L" shape neighbourhood, we need to add the overlapped area of these 3 cells to the equation as dual neighbourhoods takes out that amount of size from the sum as extra. For example, when RSUs are placed into A, B and D cells, the subtract operations defined at step *ii* will remove the overlapped areas for A-B, A-D and B-D neighbourhoods, and this will result in subtracting an extra overlapped area for A-B-D neighbourhood. The triple summation operation adds this area back to the equation (*iii*).

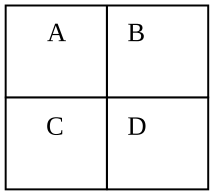

Fig. 8. The neighbour cells

(2) Resource Demand

We followed a similar strategy to network coverage while formulating the resource demand (*D*). The main difference is, we had to calculate the resource demand for each cell using the traffic dataset. Then, the same rules we used for the neighbourhood also apply here.

Therefore, total demand is defined with this equation:

$$D = \sum_{i=0}^{99} x_i d_i - \sum_{i=0}^{99}\sum_{j=0}^{99} x_i x_j s_{i,j} + \sum_{i=0}^{99}\sum_{j=0}^{99}\sum_{k=0}^{99} x_i x_j x_k s_{i,j,k}$$

(3) Constraints

Lastly, we define the constraints. The constraints below suggest that minimum network coverage and resource demand are user defined parameters and denoted by γ and λ. And all the decision variables are binary.

$$R \geq \gamma$$

$$D \geq \lambda$$

$$x_0, x_1, x_2, \dots, x_{99} = 1 \mid 0$$

Table 5 depicts the notations used in the mathematical formulations.

TABLE V
SUMMARY OF NOTATIONS IN THE MATHEMATICAL FORMULATIONS

Symbol	Description
i	Candidate grid cell for RSU placement
x_i	Binary variable for RSU placed at cell i
R	Total network coverage
D	Total resource demand
r_i	Network coverage for RSU placed at cell i
d_i	Satisfied resource demand for RSU placed at cell i
$p_{i,j}$	Overlapped network coverage for the RSUs at neighbour cells i and y
$s_{i,j}$	Overlapped supply for the RSUs at neighbour cells i and y
$p_{i,j,k}$	Overlapped network coverage for the RSUs at neighbour cells i , y and k
$s_{i,j,k}$	Overlapped supply for the RSUs at neighbour cells i , y and k

a) Implementation

We defined the formulations on an open source LP Solver. We solved the problem by assigning different values for γ and λ. When we targeted for for full network coverage (γ = 100) and full resource supply (λ = 100), it resulted that 79 RSUs needed to be placed. However, when we decreased both of the values to 99%, the outcome changed to 52 RSUs. For 90% coverage and supply, the problem was solved with 42 RSUs.

Since this tool is designed to serve as a framework to the infrastructure providers, the company will be free to use any values as parameters based on their financial and technical requirements. While they can set high values γ and λ, they can also aim for maximum coverage whereas they ignore the demand, or vice versa. For our simulation in which we compare the performances of the placement models, we use the values γ = 99 and λ = 99 since this combination result in a very efficient outcome..

To compare system performances and validate functionalities of the RSU placements we generated using RSU placement framework, we run a set of simulations on V2ISim. To achieve this, we processed the simulation output logs and plotted several graphs using *Python matplotlib* library.

Fig. 8 shows the final placement of the optimized distribution model.

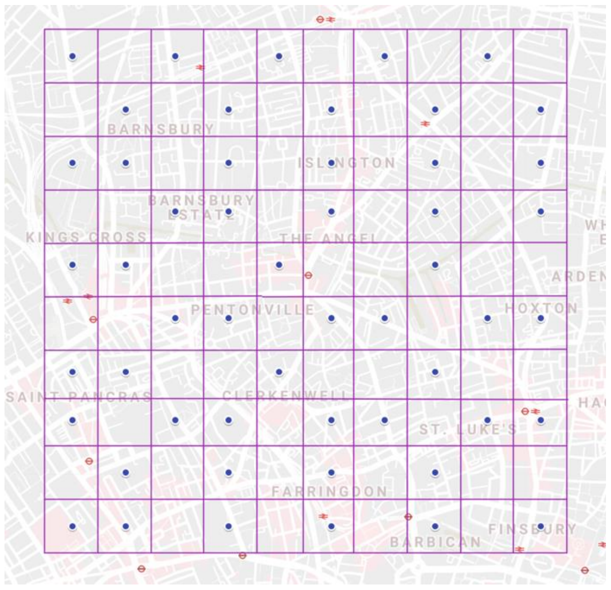

Fig. 8. RSU locations on optimized distribution model

We generated 3 distribution models: uniform, weighted, and optimized. The weighted model has 3 variations for the values of θ = 10, 20 and 30. Since we already had the results for uniform distribution model, we run the simulation for weighted and optimized placements. The simulation took 9 hours 6 minutes, 6 hours 36 minutes, and 6 hours 19 minutes for the weighted placement model respectively, and 7 hours 41 minutes for the optimized placement model. All the simulations were run on a laptop with Intel Core i7-8850H CPU and 16GB RAM.

We used same traffic dataset for all of the simulations. The dataset includes vehicle trajectory data files which represent different traffic densities. Therefore, we can evaluate system behavior under different loads. We classified the traffic densities into 3 categories:

- Number of vehicles below 1500 as low traffic volume
- Number of vehicles between 1500 and 3000 as medium traffic volume
- Number of vehicles more than 3000 as high traffic volume

The graph in Fig. 9 shows the comparison of task failure rates for uniform distribution, weighted distribution for θ = 10, 20, and 30, and optimized distribution. This can be considered as our most important metric while evaluating system performance. A system with low task failure rates is more reliable and functions better.

We can observe that the system functions best for the optimized distribution model under any traffic volumes, therefore we can suggest that optimized distribution model provides the best results among all models. The graph also shows that when the number of vehicles in the system increases, task failure rates also increase for all RSU distribution models consistently except for the optimized model. Considering the sharp increase between 3500 and 4000 vehicles for all models, we can claim that if the traffic density is over a threshold, RSUs will have difficulty handling the load and the system might even crash.

The graph shows us below 1000 vehicles, there is no significant gap between weighted distribution model for θ = 10 and uniform distribution model, however after this point we can observe an increase on this gap.

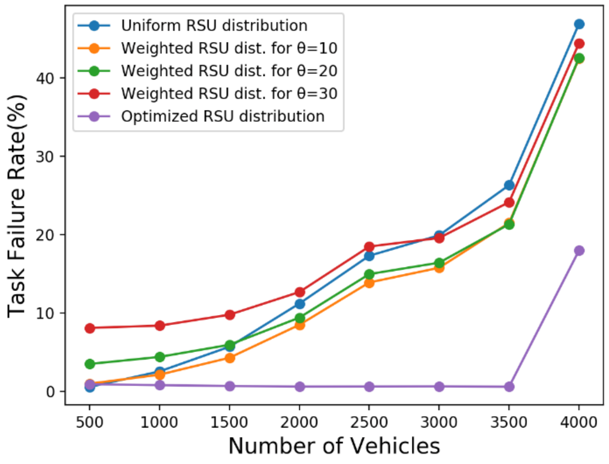

Fig. 9. Task failure rates

On the other hand, while uniform distribution model performs better than the weighted distribution models for θ = 20 and 30 under low traffic volume, weighted distribution model for θ = 20 outperforms it for medium traffic volume and weighted distribution model for θ = 30 outperforms it for high traffic volume. This is because while network coverage is a more important factor for the low traffic volume, resource capacity becomes more critical than the other factors when traffic density increases.

Lastly, the graph shows that relocating less utilized RSUs to the territories with higher load improves the system to a certain point. Weighted distribution model for θ = 10 outperforms uniform model for low, medium and high traffic volumes and it is the most optimal relocation factor among all the others. However, for θ = 20, weighted model only performs better for medium and high traffic volumes, and for θ = 30, it only functions better for high traffic volume. The reason for this is the trade-off between network coverage and resource capacity. When a less demanded RSU is relocated into a position to share the load in a busy area, capacity originated failure rates will decrease for the RSUs in the target territory, however coverage originated failure rates will increase for the original source territory.

As a result, by evaluating the results of Task Failure Rate graph, we can conclude that:

- optimized distribution model outperforms all others under any traffic load.
- uniform distribution model can be used for low traffic volume
- weighted model for θ = 20 can be used for medium and high traffic volumes
- weighted model for θ = 30 does not perform well under any traffic load

Fig. 10 shows the comparison of average service time of the RSUs in the unit of seconds. The service time is sum of download and upload delays and task processing time. As can be seen on the graph, increasing load is positively related to RSU service times for all distribution models except for the optimized model. Optimized distribution model performed better than the other models for all traffic volumes, and all weighted distribution models produced better results than the uniform model. The reason is, both download and upload delays and processing time depend on the demand on the RSU in that particular time. When an RSU needs to serve to higher number vehicles, they experience more delays on network and processing time. And as a result of sharing the high load with relocated RSUs, all weighted models provide better results in terms of service time.

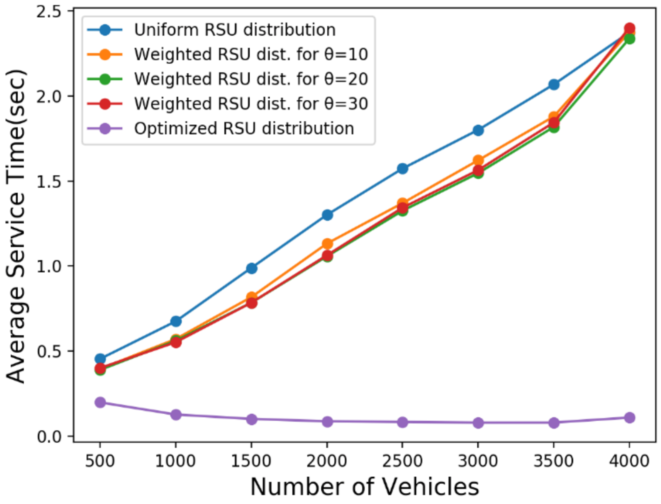

Fig. 10. Average service time

While measuring system performance, another important metric is the average utilizations of the RSUs. A system in which RSUs run with a low capacity is less efficient than another system with higher RSU utilization. On the other hand, a system with RSUs running in full capacity for a certain level of computational demand, is not able to sustain higher loads. Since the simulations we run with low and medium traffic volumes do not create significant load on majority of the RSUs, we compared utilization of the RSUs using only the results of the simulations run with 3500 vehicles. 3500 is the number which creates the highest traffic volume without breaking the system. Fig. 11 shows the histogram of average RSU utilization for uniform, weighted for θ=10, and optimized distribution models. The histogram shows that optimized model performs best in terms of RSU utilization because of two reasons: first, number of RSUs running in the lowest capacity (<10%) is lower than the other models, therefore RSU resources were used more efficiently. Second, number of RSUs running in high capacity (>%80) is also lower, therefore the load is distributed more evenly among the RSUs.

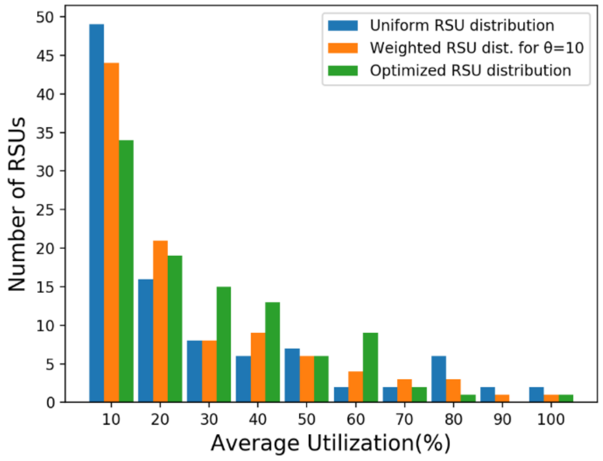

Fig. 11. Average utilization histogram

Fig. 12 (a), (b) and (c) show task failure reasons and breakdowns for uniform, weighted for θ=10, and optimized distribution models respectively. In uniform distribution no task failure due to network coverage can be observed since it was designed for the full network coverage. For uniform model when the traffic volume is low, vehicle mobility is the reason for the majority of the task failures. However, when traffic density increases, mobility failure rate decreases and RSU capacity failure becomes the main reason of the task failures. For weighted model, especially for the low traffic volume, network coverage failure is a significant failure reason as a result of RSU relocation. However, when traffic density increases, coverage and mobility failure rates decrease and RSU capacity failure becomes the main reason of the task failures. Lastly, for optimized model, network coverage is the main reason of the task failures for all traffic density levels, and we observe a spike on the bandwidth failures for 4000 vehicles.

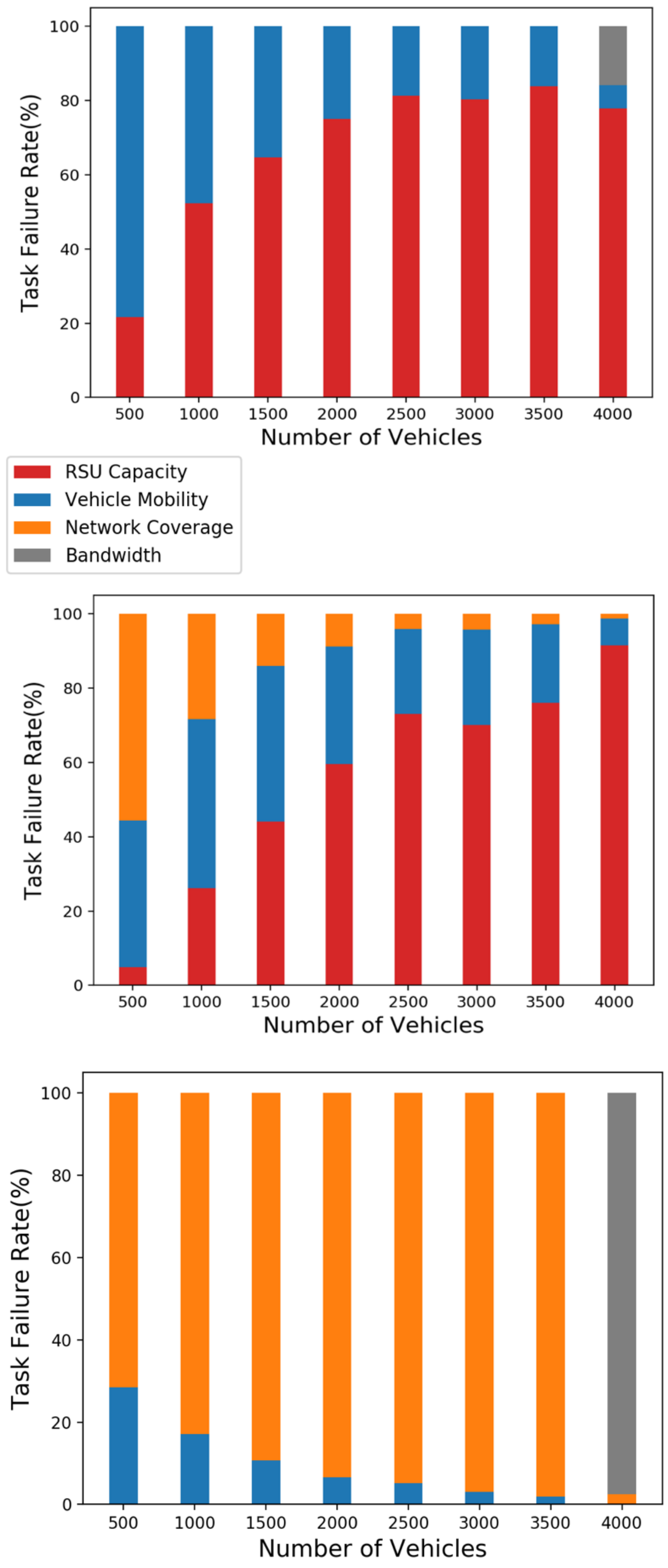

Fig. 12. Task failure breakdown (a) Uniform distribution model
(b) Weighted distribution model (θ = 10)
(c) Optimized distribution model

III. CONCLUSION

In this study, we propose an RSU placement framework to be used for generating optimal RSU placement models based on traffic characteristics of a target area. Two criteria should be satisfied for an RSU placement problem: network coverage and computational demand. The proposed framework includes 3 distribution models: uniform, weighted and optimized. Uniform distribution model addresses full network coverage and do not consider computational demand. This can serve as a suitable model for a road network in which sparse and evenly distributed traffic is observed on the road network. Weighted distribution is a heuristic model which uses uniform model as the base model. It addresses making improvements by considering the computational demand. The relocation factor (θ), which is an external parameter, is provided to this model to update RSU locations in favour of the computational demand. For a scenario with high traffic volume, it is expected to experience congestions on the road network and this might result in extra load on the RSUs serving in those territories. When the computational demand exceeds the capacity of an RSUs, they may become dysfunctional and this eventually would result a system crush. This scenario can be prevented by providing a meaningful value for θ. Thus, for an effective utilization of the framework, traffic characteristics of the target area should be carefully examined, and a suitable value should be assigned for θ. Lastly, optimized distribution model uses Linear Programming to generate an optimized RSU distribution. This solution guarantees a certain level of network coverage and resource supply using minimum number of RSUs. The constraints are defined with external parameters, γ and λ, and denotes coverage constraint level and and resource supply constraint level respectively. Thus, the company that uses this framework will be free to use any values as parameters based on their financial and technical requirements. While they can set high values γ and λ, they can also aim for maximum coverage whereas they ignore the demand, or vice versa.

We needed a simulation environment to test performance of the RSU placement models and validate their functionality. Since we could not find a simulation tool designed for V2I scenarios, we extended the capabilities of EdgeCloudSim, which is a simulation framework designed for edge scenarios. We introduced components and modules specific to V2I scenarios and referred to this extended simulation environment as V2ISim.

In our experiments, we used uniform, optimized, and weighted placement models. For the weighted model, we generated 3 variations for θ=10, 20 and 30. Also we generated a traffic dataset consisting of 8 vehicle trajectory files each representing a different traffic volume. Then, we run a simulation for each placement model using this dataset on V2ISim. The simulation results showed that optimized model outperforms all others under any traffic load. Also, we concluded that uniform distribution model can be used for low traffic volume, weighted model for θ=20 can be used for medium and high, and θ=30 can be used for high traffic volumes. These results align with our expectations and the experiments validate the functionality of the proposed RSU placement framework.

As future work, we plan to improve our communication model. In this study, we had our main focus on the communication between vehicle and RSU, however inter-RSU communication is an accepted form of communication in Vehicular ad-hoc network (VANET) in which RSUs can exchange data with each other [7]. By implementing this in V2ISim, task transfers between RSUs will be possible and task failures due to vehicle mobility will be prevented. Moreover, some technical factors that can impact the communication between vehicles and RSUs should be studied and findings should be reflected to the study. These can be determining the noise level for the RSUs in close proximity and shadowing effect of the buildings.

ACKNOWLEDGMENT

This work is supported by the Galatasaray University Research Fund under the grant number 18.401.003.

REFERENCES

[1] M. Satyanarayanan, P. Bahl, R. Cáceres and N. Davies, "The case for VM-based cloudlets in mobile computing," *IEEE Pervasive Computing*, 2009.

[2] P. Corcoran and S. K. Datta, "Mobile-Edge Computing and the Internet of Things for Consumers: Extending cloud computing and services to the edge of the network.," *IEEE Consumer Electronics Magazine*, 2016.

[3] E. Uhlemann, "Introducing connected vehicles [Connected vehicles]," *IEEE Vehicular Technology Magazine*, 2015.

[4] J. Santa, A. Moragón and A. F. Gómez-Skarmeta, "Experimental evaluation of a novel vehicular communication paradigm based on cellular networks," in *IEEE Intelligent Vehicles Symposium, Proceedings*, 2008.

[5] S. Zeadally, R. Hunt, Y.-S. Chen, A. Irwin and A. Hassan, "Vehicular ad hoc networks (VANETS): status, results, and challenges," *Telecommunication Systems*, vol. 50, no. 4, pp. 217-241, 2012.

[6] C. Y. Chang, "MAC protocols in vehicular ad hoc networks," in *Telematics Communication Technologies and Vehicular Networks: Wireless Architectures and Applications*, 2009.

[7] R. Barskar and M. Chawla, "Vehicular Ad hoc Networks and its Applications in Diversified Fields," *International Journal of Computer Applications*, 2015.

[8] M. Saini, A. Alelaiwi and A. El Saddik, "How close are we to realizing a pragmatic VANET solution? A meta-survey," *ACM Computing Surveys*, 2015.

[9] M. Satyanarayanan, "Edge computing for situational awareness," in *IEEE Workshop on Local and Metropolitan Area Networks*, 2017.

[10] L. C. Bento, R. Parafita and U. Nunes, "Intelligent traffic management at intersections supported by V2V and V2I communications," in *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, 2012.

[11] K. Katsaros, R. Kernchen, M. Dianati and D. Rieck, "Performance study of a Green Light Optimized Speed Advisory (GLOSA) application using an integrated cooperative ITS simulation platform," in *IWCMC 2011 - 7th International Wireless Communications and Mobile Computing Conference*, 2011.

[12] C. Letter and L. Elefteriadou, "Efficient control of fully automated connected vehicles at freeway merge segments," *Transportation Research Part C: Emerging Technologies*, vol. 80, pp. 190-205, 1 7 2017.

[13] C.-H. Hong and B. Varghese, "Resource Management in Fog/Edge Computing: A Survey on Architectures, Infrastructure, and Algorithms," *ACM Comput. Surv.*, vol. 52, no. 5, 9 2019.

[14] P. Mach and Z. Becvar, *Mobile Edge Computing: A Survey on Architecture and Computation Offloading*, 2017.

[15] C. Sonmez, A. Ozgovde and C. Ersoy, "EdgeCloudSim: An environment for performance evaluation of edge computing systems," *Transactions on Emerging Telecommunications Technologies*, 2018.

[16] O. Trullols, M. Fiore, C. Casetti, C. F. Chiasserini and J. M. Barcelo Ordinas, "Planning roadside infrastructure for information dissemination in intelligent transportation systems," *Computer Communications*, 2010.

[17] B. Aslam, F. Amjad and C. C. Zou, "Optimal roadside units placement in urban areas for vehicular networks," in *Proceedings - IEEE Symposium on Computers and Communications*, 2012.

[18] N. M. Balouchzahi, M. Fathy and A. Akbari, "Optimal road side units placement model based on binary integer programming for efficient traffic information advertisement and discovery in vehicular environment," *IET Intelligent Transport Systems*, 2015.

[19] T. J. Wu, W. Liao and C. J. Chang, "A cost-effective strategy for road-side unit placement in vehicular networks," *IEEE Transactions on Communications*, 2012.

[20] R. Yu, Y. Zhang, S. Gjessing, W. Xia and K. Yang, "Toward Cloud-based vehicular networks with efficient resource management," *IEEE Network*, 2013.

[21] S. K. Datta, R. P. F. Da Costa, J. Harri and C. Bonnet, "Integrating connected vehicles in Internet of Things ecosystems: Challenges and solutions," in *WoWMoM 2016 - 17th International Symposium on a World of Wireless, Mobile and Multimedia Networks*, 2016.

[22] M. A. Salahuddin, A. Al-Fuqaha, M. Guizani and S. Cherkaoui, "RSU cloud and its resource management in support of enhanced vehicular applications," in *2014 IEEE Globecom Workshops, GC Wkshps 2014*, 2014.

[23] P. A. Lopez, M. Behrisch, L. Bieker-Walz, J. Erdmann, Y. P. Flotterod, R. Hilbrich, L. Lucken, J. Rummel, P. Wagner and E. Wiebner, "Microscopic Traffic Simulation using SUMO," in *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*,

2018.

[24] A. K. Ligo, J. M. Peha, P. Ferreira and J. Barros, "Comparison between Benefits and Costs of Offload of Mobile Internet Traffic Via Vehicular Networks," in *43rd Telecommunications Policy Research Conference*, 2015.

BIOGRAPHIES

BARIŞ KARA received his B.S. degree in computer engineering from 9 Eylül University, İzmir,Turkey in 2010 and the M.S. degree in computer engineering from Galatasaray University, İstanbul, Turkey, in 2019.

Barış Kara is a senior software developer with 10 years of experience. He has been providing software development and IT consultancy services in the UK since 2016. He played major roles in design and development of many enterprise projects. Contributed to software projects of high-profile international and Turkish companies such as HSBC, O2, SKY, ATOS and Borsa Istanbul.

ATAY ÖZGÖVDE received BS, MS and PhD degrees from Bogazici University, Istanbul, in 1995, 1998 and 2009. He worked for Nortel Networks as an R&D engineer in various telecommunications projects between 1998 and 2001. In 2002, he started working as a research assistant in the Computer Engineering Department, Bogazici University. Currently, he is an assistant professor in the Computer Engineering Department, Galatasaray University. His research interests include wireless sensor networks, embedded systems, distributed systems, pervasive computing, SDN and Edge Computing. Atay Ozgovde is a senior member of IEEE.