An RSU Placement Framework for V2I Scenarios

Baris Kara
Department of Computer Engineering
Galatasaray University
Istanbul, Turkey
e-mail: bariskara35@gmail.com

B. Atay Ozgovde
Department of Computer Engineering
Galatasaray University
Istanbul, Turkey
e-mail: aozgovde@gsu.edu.tr

***Abstract*— Edge computing has become a prominent computing strategy when mobile devices and Internet of Things (IoT) became popular in the last decade and cloud computing could not meet the computational requirements of some of these devices/applications. What edge computing can provide differently from cloud computing is 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 study, we aim to present a solution to one of the high level problems in vehicular networks: efficient Road Side Unit (RSU) placement by addressing network coverage and computational demand. We propose an RSU placement framework for generating RSU placement models based on traffic characteristics of a target area. Moreover, our work includes extending capabilities of a simulation framework designed for edge computing scenarios. Therefore, we can evaluate the performance of the generated models and validate their functionality by running simulations on this environment.**

Keywords-edge computing; V2I; connected vehicles; roadside units.

I. INTRODUCTION

With increasing popularity of mobile devices and Internet of Things (IoT) in 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 centralized data centers started to emerge. Satyanarayanan et al. [1] describe these limitations as long Wide Area Network (WAN) latencies and bandwidth-induced delays. Because of these limitations, cloud computing is not a suitable computing strategy for scenarios which require 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].

The components of the Vehicle-to-Infrastructure (V2I) scenarios can be mapped to edge computing elements as follows:

- Road Side Units (RSU) are the edge computing units in vehicular networks 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 [3].
- Vehicular applications are edge applications as they demand complex computation and large storage.

Applications deployed into RSUs receive data from vehicular applications such as trajectory, speed, destination coordinates, etc. in short intervals, aggregate and process them 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.

Deploying a limited number of RSUs into a smart city 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.

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, proposed by Sonmez et al. [4]. By adding new modules to support simulations for V2I scenarios and designing realistic traffic scenarios for a target area in London city centre, we validate the functionality of the proposed RSU placement framework and evaluate the performances of the generated RSU placement models

The rest of the paper is organised as follows: Section II reviews the related work. Section III introduces the simulation environment, V2ISIM. In Section IV, the

 ISBN: 978-1-61208-740-5

reference scenario is described, and Section V explains RSU placement models. In Section VI, we address the RSU placement results, in Section VII, we analyse the simulation results, and finally, Section VIII outlines the concluding remarks.

II. RELATED WORK

Previous research addressing edge computing in vehicular networks mostly suggest new frameworks and architectures in which cloud and edge processing units, and mobile devices/vehicles are integrated into a new ecosystem. Their main focus is to provide solutions for computational challenges, such as resource allocation and Virtual Machine (VM) migration [3] [5] [6]. Our work can be considered as a complementary study which is built on top of an existing architecture addressed by these studies.

On the other hand, RSU placement problem has been addressed by several studies in both highway and smart city scenarios. Highway scenarios have different traffic characteristics than smart cities (e.g. fast moving vehicles, sparse traffic, etc.), therefore, the communication infrastructure should be designed considering these requirements. Studies focusing on RSU deployment to the highways [7] [8] [9] differ from our study from this aspect.

There are also studies addressing RSU placement in smart cities. These studies mostly approach the problem from network coverage aspect without taking computational demand into account. As a result, the placement models presented in these works do not guarantee fulfilling computational requirements of the edge applications deployed to the RSUs. In their study, Liang et al. [10] formulate optimal RSU deployment problem as an integer linear program (ILP). In their model, V2I communication is extended with multi-hop Vehicle-to-Vehicle (V2V) communication. Also, RSUs in their model can have different configuration settings. In our scenario, we don't consider multi-hop communication in order to minimize the latency in V2I communication, and all the RSUs have same capabilities. Chi et al. [11] propose an RSU allocation algorithm with a concept of intersection priority. They aim to maximize the intersection coverage by deploying RSUs to the important intersections. Similarly, Gomi et al. [12] propose an RSU placement method by calculating placement priority for each intersection. In their work, they also consider road elements that affect radio wave spreading such as buildings and aim for a better communication performance. These methods consider the intersections as deployment points of RSUs, whereas in our model, we divide the target area into cells and build our deployment logic on top of these cells. Trullols et al. [13] propose a maximum coverage approach for modelling the problem of RSU deployment. Their model is based on deploying RSUs as Dissemination Points (DPs) and maximizing the number of vehicles that contact the DPs. In the study of Balouchzahi et al. [14] the problem of RSU placement formulated to binary integer programming. Unlike from the other studies, their work address highway and urban scenarios at the same time. Similarly, Premsankar et al., [15] use mixed linear integer programming formulation for the problem, but their formulation focuses on minimizing the deployment cost of edge computing devices by jointly satisfying a target level of network coverage and computational demand.

III. V2ISIM

We needed a simulation environment for our study in order to validate the functionality of the proposed RSU placement framework and compare the performances of the generated RSU placement models. For this purpose, we used EdgeCloudSim, which is an open source tool designed for simulating edge computing scenarios where it is possible to conduct experiments that consider both computational and networking resources [4]. We extended the capabilities of the framework by defining components and modules specific to V2I scenarios and referred to this extended simulation environment as V2ISim.

EdgeCloudSim is also extended from another simulation environment, CloudSim, which allows modelling of cloud computing infrastructures and application services [16]. While EdgeCloudSim implemented edge processing units and modelled edge computing network, we introduced RSUs as computing units for V2I scenarios and extended the network model for vehicular network. We also implemented a mobility module to integrate traffic scenarios into the environment.

The key components we implemented in V2ISim are as follows:

- **RSUManager:** This component is responsible for creating RSU instances in the system based on the configuration provided. The configuration should include RSU resource definition as well as Global Positioning System (GPS) coordinates in decimal degrees.
- **TrafficLoadGenerator:** A traffic input file, which includes vehicle trajectory data, should also be provided to the simulation environment. *TrafficLoadGenerator* is responsible for creating tasks using task characteristics received from task configuration file. When the simulation starts running, these tasks are scheduled for processing in due course.
- **TrafficTaskBroker:** This component is responsible for managing the lifecycle of a task. After the task is created, there are 3 stages it has to follow until it is completed: vehicular application submits task to the RSU, task is processed in the RSU and finally, the response is sent back to the vehicle. When the task reaches to one stage, it is rescheduled by *TrafficTaskBroker* for the next one.
- **RSUOrchestrator:** Its responsibility is to find the RSU that the task will be submitted. To achieve this, first, nearest RSU to the vehicle is detected. Then, if the vehicle is within the range of the RSU, task is submitted to it by *TrafficTaskBroker*.

To find the nearest RSU for a given vehicle position efficiently, all RSU coordinates are saved in a K-D tree (K-Dimensional Tree) when the application starts. A K-D Tree is a data structure for efficient search and nearest-neighbour(s) computation of points in K-dimensional space.

 ISBN: 978-1-61208-740-5

We used an open source K-D tree implementation in our application [17].

- **RSUMM1Queue:** We use M/M/1 queue model to simulate the network delay. This component is responsible for calculating task upload and download delays.
- **SimLogger:** Lastly, all the important task data, RSU data and as well as calculated system metrics are logged by *SimLogger* in different logging levels.

At the end of the simulation, 3 output files are generated for each traffic input file provided:

- **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. These values are used as metrics while comparing system performances from utilization aspect for different RSU placement models.
- **Task assignment logs**: It keeps the logs of number of assigned and failed tasks for each RSU.

IV. REFERENCE SCENARIO

This section outlines the reference scenario that we considered for V2I communication and the target area we used for the case study, and explains our approach on generating traffic dataset to run our experiments.

A. Scenario and Parameters

In our reference scenario, we consider a smart city equipped with V2I communication infrastructure. All vehicles are smart or connected with the ability of running vehicular applications. Vehicular applications send one task to the nearest RSU per second in case the vehicle is in the network coverage of any RSU and the data is processed by edge applications deployed to the RSUs. When the task is successfully processed, RSU sends a response back to the vehicle. 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 capacity 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 Central Processing Unit (CPU) and 500MB Random Access Memory (RAM), and average task length is 300 Machine Instructions (MI). Table I shows the parameters for RSU and task configurations. All the simulations run as part of this study are based on these values.

TABLE I. RSU AND TASK PARAMETERS AND VALUES

Parameter	Value
RSU Network Range	250m
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

B. Target Area

We chose an area of 3 x 3 kilometres in London city centre as the target area for deploying RSUs. 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* [18], 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.

C. 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 [19]. Apart from its simulation capabilities, SUMO includes several scripts for traffic and road network operations. We used *randomTrips* script in SUMO library to generate random vehicle routes on the road network. The output route file, along with the network file should be provided to SUMO to run a traffic simulation.

In our study, traffic density plays an important role on RSU placement process as the computational demand depends on number of vehicles in the system. Thus, to cover scenarios with different traffic volumes, we generated 8 trajectory files for 500, 1000, 1500, 2000, 2500, 3000, 3500, and 4000 vehicles in the target area. Each file contains trajectory data logged for each simulation second, such as vehicle id, type, coordinates, speed, angle, lane, etc. As a result, more than 8 million logs were produced in total for the traffic dataset. Figure 1 shows heat maps of the generated vehicle trajectories for number of vehicles 500 and 4000. Traffic congestions can be observed in the centre of the map when higher number of vehicles used in the scenario.

 ISBN: 978-1-61208-740-5

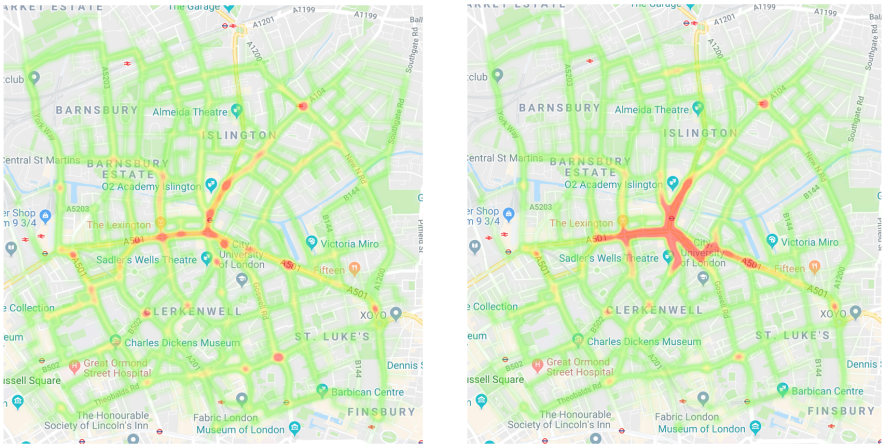

Figure 1. Heat maps of generated vehicle trajectories for the target area for number of vehicles (a) 500 and (b) 4000

V. RSU PLACEMENT MODELS

We generated 2 RSU distribution models addressing RSU placement problem in a smart city: Uniform RSU distribution and Weighted RSU distribution. This section outlines the algorithms we used for each distribution models.

A. Uniform RSU Distribution

Uniform RSU distribution serves for two purposes in our study: First, we used it as the base model which we made optimizations on the RSU locations in the next steps. Second, we used performance results of this model to compare with the results of generated placement models. The algorithm of this model is quite basic; after calculating the optimal distance between two RSUs, the number of RSUs required for full network coverage on the target area is calculated. Then, target area is divided into cells and RSUs are placed into these cells equidistant from each other. We referred to these cells as territories. Therefore, full network coverage is enabled in the target area, whereas computational demand is ignored.

B. Weighted RSU Distribution

Weighted RSU distribution model redistributes some RSUs in uniform distribution model by taking computational demand into account. In the uniform distribution model, despite of full network coverage, high task failure rates might be observed since RSUs might not meet high computational demand using their limited resources. It is especially expected to experience this problem in the territories with higher volumes of vehicle traffic, i.e., traffic congestions. An external parameter, θ, is the relocation factor, and it determines the 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.

Therefore, we aim to decrease capacity related task failures by bringing additional computational resources to meet the higher demand. On the other hand, relocated RSUs will result in network related 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.

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. RSUs with less task assignment rates are marked to be moved in
- the territories with higher resource demand. We select θ% of least utilized RSUs in this step.
- **Territory Selection:** To detect territories that need additional resources to meet 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 capacity related task failure rates 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.

VI. PLACEMENT RESULTS

We developed a Java application as the implementation of the suggested placement algorithms and referred to it as *RSU Distributor*.

 ISBN: 978-1-61208-740-5

A. *Uniform RSU Distribution*

Network range of RSUs can reach up to 1000 meters if there are no obstructions, and 250-350 meters in cluttered urban areas [20]. In our scenario, 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 in total.

After assessing the number of RSUs to place, we processed the area map by dividing it into territories each with the size of 300 by 300 meters, and we assigned each territory an id sequentially. Then, we placed one RSU into the centre of each territory, therefore 100 RSUs were evenly distributed on the area. Figure 2 shows distribution of the RSUs into the territories based on the uniform distribution model.

B. *Weighted RSU Distribution*

To calculate the RSU coordinates for weighted RSU distribution model, we started by running V2ISIM for uniform distribution model, therefore, we generated the inputs required for weighted distribution algorithm: task assignment rates and task failure rates for each RSU. The simulation tool requires two input files: vehicle trajectory data and RSU coordinates. As traffic input data, we provided the traffic dataset we generated using SUMO and configured RSU and task characteristics by providing parameters listed in Table I. Some important simulation properties can be seen in Table II.

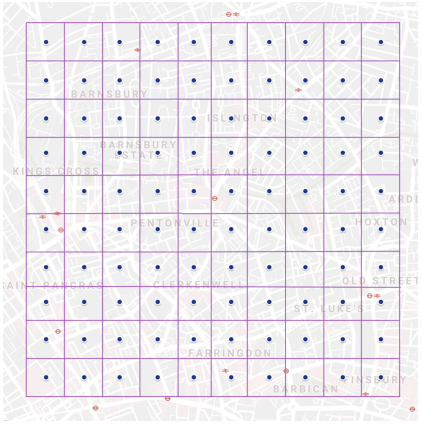

Figure 2. RSU Locations on Uniform Distribution Model

TABLE II. SIMULATION PROPERTIES

Parameter	Value
Total number of traffic logs	8 147 468
Total number of RSUs	100
RSU placement model	Uniform
Simulation time	1 hour

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

TABLE IV. TERRITORY IDS AND NUMBER OF RSUS TO ASSIGN

θ	RSU ids
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)

In *RSU Distributor*, simulation logs were aggregated and processed to calculate the values of task assignment rates and task failure rates of the RSUs. By assigning 10, 20, and 30 to θ, we run the application and generated 3 different RSU placement models. For each value of the θ, Table III shows the selected RSUs for relocation and Table IV shows the number of RSUs to be assigned to each territory.

After running RSU distributor with these inputs, 3 different distribution models were produced based on weighted distribution model algorithm. Figure 3 shows RSU placements for θ=10, 20, and 30 respectively.

VII. SIMULATION RESULTS

For generated RSU placement models, we run a set of simulations on V2ISim using a laptop with Intel Core i7-8850H CPU and 16GB RAM. Table V shows the time spent to run each simulation.

We classify traffic densities of the traffic input files we used for the simulations 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 Figure 4 shows comparison of task failure rates for uniform distribution and weighted distribution for θ=10, 20, and 30.

TABLE V. SIMULATION PROPERTIES

Simulation	Duration
Uniform RSU Distribution	6 hours 10 minutes
Weighted RSU Distribution θ=10	9 hours 6 minutes
Weighted RSU Distribution θ=20	6 hours 36 minutes
Weighted RSU Distribution θ=30	6 hours 19 minutes

 ISBN: 978-1-61208-740-5

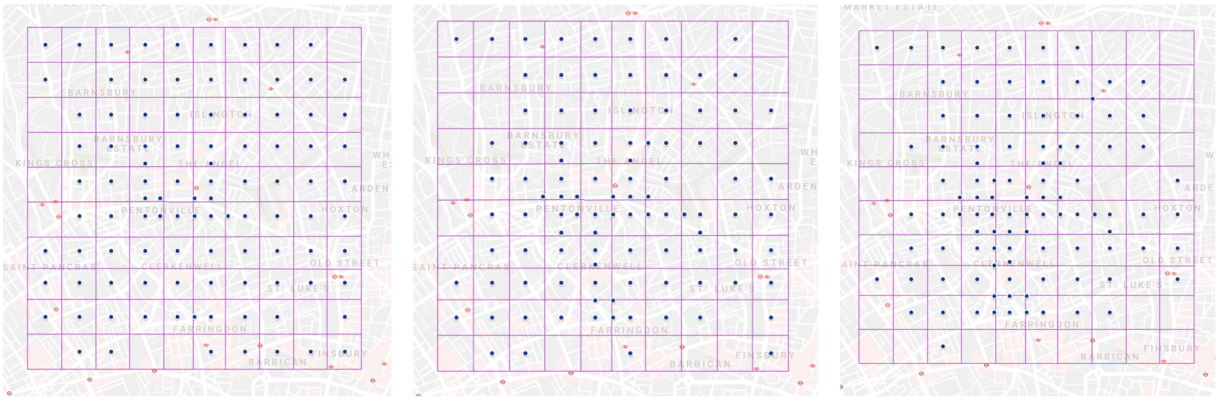

Figure 3. RSU Locations on Weighted Distribution Model for (a) θ=10, (b) θ=20, (c) θ=30

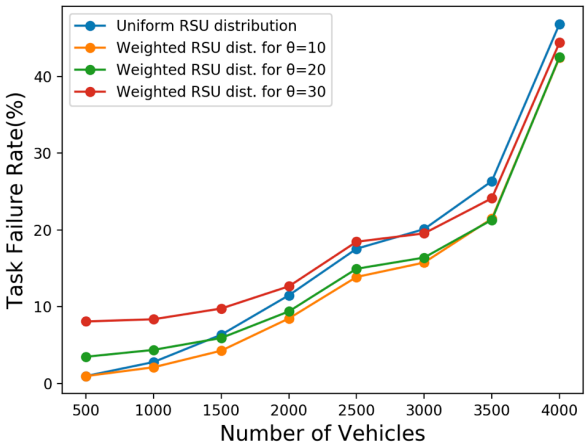

Figure 4. Task Failure Rates

Task failure rates 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 weighted distribution model for θ=10 under any traffic volumes. The graph also shows that when the number of vehicles in the system increases, task failure rates also increase for all RSU distribution models. 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 not handle the load and the system will crash. 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.

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:

- θ=10 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

Figure 5 shows the comparison of average service time of the RSUs in the unit of seconds. The service time is the sum of download and upload delays and task processing time. As can be seen on the graph, increasing load had a similar impact on RSU service times for all distribution models, and all weighted distribution models performed better than the uniform model for all traffic volumes. The reason is, both download and upload delays and processing time depend on the RSU demand 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.

While measuring system performance, another important metric is 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

 ISBN: 978-1-61208-740-5

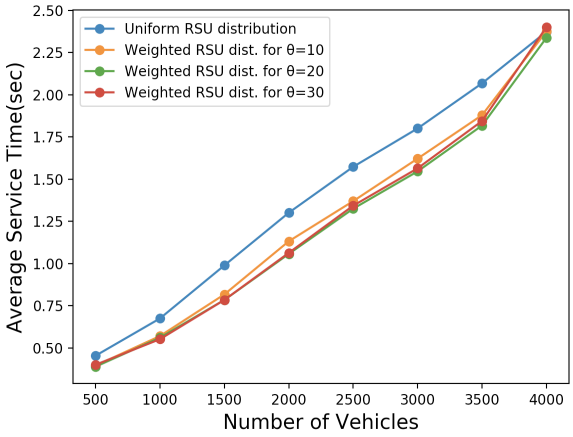

Figure 5. Average Service Time

majority of the RSUs, we compared utilization of 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.

Figure 6 shows histogram of average RSU utilization for uniform distribution model and weighted distribution models for θ=10. The histogram shows a significant improvement for weighted model in terms of RSU utilization because of two reasons: first, number of RSUs running in the lowest capacity (<10%) is lower than the uniform model, 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. This shows that relocating a less utilized RSU to share the high load in a territory provides good results in terms of utilization and can serve as a good optimization technique.

Figure 7(a) and 7(b) shows task failure reasons and breakdowns for uniform distribution model and weighted distribution model for θ=10 respectively. In uniform distribution, no task failure due to network coverage can be observed since it was specifically designed by addressing full network coverage. When the traffic volume is low, vehicle mobility is the reason for the majority of the task failures.

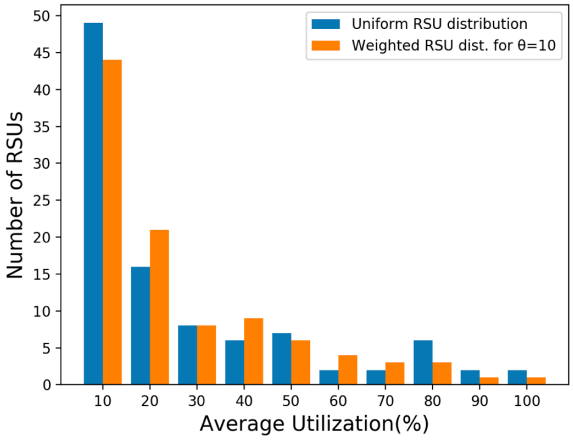

Figure 6. Average Utilization Histogram (3500 vehicles)

However, when traffic density increases, mobility failure rate decreases and RSU capacity failure becomes the main reason of the task failures. Also a high number of task failures can be observed due to exceeding bandwidth capacity when the number of vehicles is 4000 in the simulation.

On the other hand, when vehicle number is 500, network coverage is the main reason of the task failures for weighted distribution model for θ=10 since the vehicles within the range of relocated RSUs cannot connect to any RSU to assign their tasks. When traffic density increases, coverage and mobility failure rates decrease and RSU capacity failure becomes the main reason of the task failures.

VIII. Conclusions And Future Work

In this study, we proposed an RSU placement framework to be used for generating optimal RSU placement models based on traffic characteristics of a target area. Our solution addresses satisfying two criteria for RSU placement problem: network coverage and resource demand. Our work also includes extending capabilities of EdgeCloudSim, a simulation framework designed for edge scenarios, by introducing V2I components and modules. We referred to this extended simulation environment as V2I framework.

In order to validate the functionality of the proposed RSU placement framework, we generated a set of RSU distributions for a target area in London city centre based on uniform and weighted RSU distribution models. Then, we conducted experiments using V2I framework to compare their performances under different traffic loads. The experiments showed that weighted distribution model with replacement factor (θ) 10 performs best under any traffic load. Also we observed that weighted distribution models provided better results in terms of service time and resource utilization.

As future work, we plan further optimisations on weighted distribution model by eliminating input θ from the system and calculate optimal number of RSUs only based on given traffic input data. Also, 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 [21]. 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. Finally, in the next phases of the study, we might still have the requirement of working with simulation based traffic datasets as finding real traffic datasets is not always possible. In that case, in order to validate the results and prove the consistency, we will have an approach to generate multiple datasets using different traffic simulation environments.

 ISBN: 978-1-61208-740-5

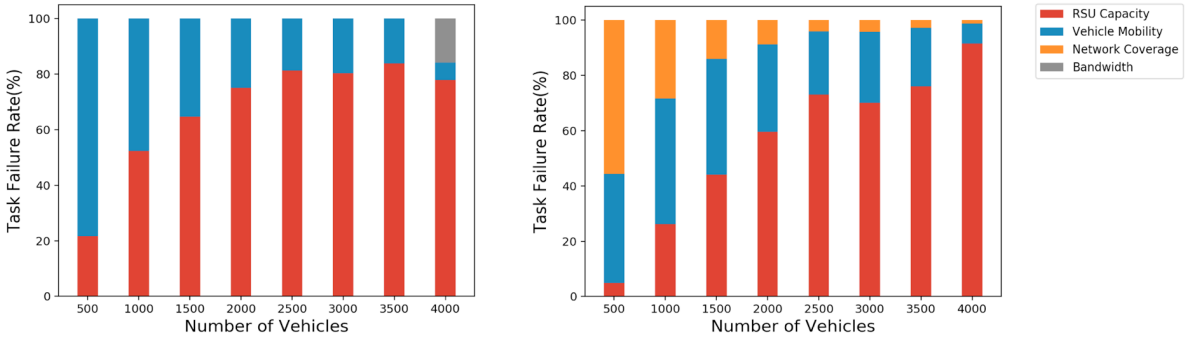

Figure 7. Task Failure Breakdown (a) Uniform Distribution Model (b) Weighted Distribution Model (θ=10)

ACKNOWLEDGEMENT

This work is supported by the Galatasaray University Research Foundation under the Grant No. 18.401.003.

REFERENCES

[1] M. Satyanarayanan, P. Bahl, R Caceres, and N. Davies, "The case for vm-based cloudlets in mobile computing," IEEE Pervasive Computing, vol. 8, no.4, pp. 14-23, Oct-Dec 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, vol. 5, no.4, pp. 73-74, Oct 2016.

[3] R. Yu, Y. Zhang, S. Gjessing, W. Xia, and K. Yang, "Toward cloud-based vehicular networks with efficient resource management," IEEE Network, vol. 27, no.5, pp. 48-55, Oct 2013.

[4] C. Sonmez, A. Ozgovde, and C. Ersoy, "EdgeCloudSim: An Environment for Performance Evaluation of Edge Computing Systems," International Conference on Fog and Mobile Edge Computing (FMEC), pp. 39-44, May 2017.

[5] S. K. Datta, R. D. F. Da Costa, J. Härri, and C. Bonnet, "Integrating connected vehicles in Internet of Things ecosystems: Challenges and solutions," IEEE 17th International Symposium on A World of Wireless, Mobile and Multimedia Networks (WoWMoM), pp. 1-6, Jul 2016.

[6] M. A. Salahuddin, A. Al-Fuqaha, M. Guizani, and S. Cherkaoui, "RSU cloud and its resource management in support of enhanced vehicular applications," IEEE Globecom Workshops (GC Wkshps), pp. 127-132, Dec 2014.

[7] T. J. Wu, W. Liao, and C. Chang, "A Cost-Effective Strategy for Road-Side Unit Placement in Vehicular Networks," IEEE Transactions on Communications, vol. 60, no.8, pp. 2295 – 2303, Jul 2012.

[8] X. Liya, H. Chuanhe, L. Peng, and Z. Junyu, "A randomized algorithm for roadside units placement in vehicular ad hoc network," IEEE 9th International Conference on Mobile Ad-hoc and Sensor Networks, pp. 193–197, Dec 2013.

[9] A. O'Driscoll and D. Pesch, "Hybrid geo-routing in urban vehicular networks," IEEE Vehicular Networking Conference, pp. 63–70, Dec 2013.

[10] Y. Liang, H. Liu, and D. Rajan, "Optimal placement and configuration of roadside units in vehicular networks," IEEE 75th Vehicular Technology Conference, pp. 1-6, May 2012.

[11] J. Chi, Y. Jo, H. Park, T. Hwang, and S. Park, "An Effective RSU Allocation Strategy for Maximizing Vehicular Network Connectivity," International Journal of Control and Automation, vol. 6, no. 4, pp. 259-270, Aug 2013.

[12] K. Gomi, Y. Okabe, and H. Shigeno, "RSU Placement Method Considering Road Elements for Information Dissemination," The Sixth International Conference on Advances in Vehicular Systems, Technologies and Applications, pp. 68-73, Jul 2017.

[13] 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, vol. 33, no. 4, pp. 432–442, Dec 2010.

[14] 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, vol. 9, no.9, pp. 851-861, Nov 2015.

[15] G. Premsankar, B. Ghaddar, M. Di Francesco, and R. Verago, "Efficient Placement of Edge Computing Devices for Vehicular Applications in Smart Cities," IEEE/IFIP Network Operations and Management Symposium, pp. 1-9, Apr 2018.

[16] R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. F. De Rose, and R. Buyya, "Cloudsim: A toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms," Software Practice and Experience, vol. 41, no.1, pp. 23-50, Jan 2011.

[17] S. D. Levy, "KD-Tree Implementation in Java and C#," https://simondlevy.academic.wlu.edu/software/kd, retrieved Aug 11, 2019

[18] https://www.openstreetmap.org, retrieved Aug 11, 2019

[19] P. A. Lopez et al., "Microscopic Traffic Simulation using SUMO," 21st International Conference on Intelligent Transportation Systems (ITSC), pp. 2575-2582, Nov 2018.

[20] 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," 43rd Research Conference on Communications, Information and Internet Policy, pp. 1-39, Nov 2015.

[21] R. Barskar and M. Chawla, "Vehicular Ad hoc Networks and its Applications in Diversified Fields," International Journal of Computer Applications, vol. 123, no.10, pp. 7-11, Aug 2015.

 ISBN: 978-1-61208-740-5