Robust Resource Allocation Using Edge Computing for Delay Sensitive Tasks in Vehicle to Infrastructure (V2I) Networks

Razin Farhan Hussain, Mohsen Amini Salehi, Anna Kovalenko
High Performance Cloud Computing (HPCC) Laboratory,
School of Computing and Informatics
University of Louisiana at Lafayette, LA, USA
{razinfarhan.hussain1, amini, aok8889}@louisiana.edu

Omid Semiari
Georgia Southern University
osemiari@georgiasouthern.edu

***Abstract*—~~Vehicle to Infrastructure (V2I) systems~~ require ~~quick~~ and reliable services ~~due to the hazards of the road environment~~. Services provided in these networks (*e.g.,* traffic information and weather updates) are generally delay intolerant. Hence, ~~V2I systems~~ must provide consistent real-time response to ~~be useful for their users (*e.g.,* drivers or autonomous vehicles)~~. Maintaining real-time response can be particularly difficult when there is a surge of request arrivals to ~~edge devices, known as Base Stations~~. Accordingly, our goal is to ~~make the V2I system robust and reliable for its users. That is, we~~ enable the system to maintain its performance even in the presence of uncertain task arrival. To achieve robustness, we propose a resource allocation model that can load balance (*i.e.,* dynamically utilize resources from neighboring base stations), when the system is oversubscribed. We propose a method to calculate the probability of completing an arriving task on time on different base stations. Then, the load balancer assigns the task to the base station that maximizes the probability of completing the task before its deadline. We evaluate our proposed model under different workload conditions and when different scheduling policies are deployed within ~~the~~ base stations. Simulation results demonstrate that our proposed allocation ~~model~~ decreases deadline miss rate by up to 40% when ~~compare to the~~ conventional V2I systems.**

***Index Terms*—Base Station, Vehicular Networks, End-to-End delay, Edge Computing, Vehicle to Infrastructure (V2I).**

I. Introduction

Recent advancements in communication and computation technologies ~~has~~ stimulated a rapid development of vehicular networks. Federal Communications Commission [1] has already reserved 5.850 to 5.925 GHz frequency band for ~~Vehicle to Everything~~ (~~aka~~ *V2X*) communications. ~~Vehicle to Infrastructure~~ (V2I) communication is one prominent form of V2X that draws the majority of work to itself. The United States Department of Transportation (DoT) considers V2I communications the future of Intelligent Transportation Systems [1].

~~Infrastructure, in V2I,~~ refers to all edge and core technologies that facilitate communication and computation of vehicular tasks. ~~In V2I systems, as~~ shown in Figure 1, ~~while vehicles are driven through the roads, requests for services are sent to the *Base Stations*~~. A Base Station is ~~a stationary edge device installed on the road side, communicates~~ with vehicles, and can process vehicular tasks [2]. Upon task completion on the Base Stations, its results are sent back to the requesting vehicle. Examples of such vehicular tasks can be wrong way driver warning [5], cooperative forward collision warning [9], and lane change warning [5]. This type of tasks can tolerate a short end-to-end delay [1]. End-to-end delay is composed of three contributing factors [10], namely up-link delay, processing delay, and down-link delay. In fact, there is no value in executing tasks after their tolerable delay. In this research, we model the tolerable end-to-end delay of each task as an individual hard deadline for the task.

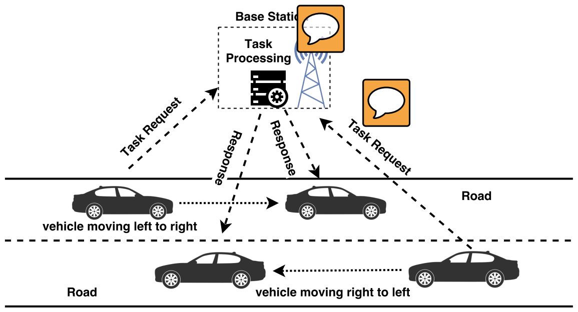

Fig. 1: A Vehicle to Infrastructure (V2I) scenario where vehicles send requests to a Base Station and receive the response. A Base Station is a roadside unit with communication and computation abilities.

For providing effective support to the vehicular services, it is commonly suggested by literature that V2I ~~system~~ should be enhanced by incorporating wireless network infrastructure with cloud computing [3]. For instance, the vehicular cloud computing (VCC) has been initiated in [4] as a subsection of mobile cloud computing. As both VCC and conventional cloud systems incur a high latency [13], which cannot be used for delay intolerant tasks of V2I systems. However, Base Stations' computational power can be harnessed as an edge computing system. ~~Being close to end users, the edge mitigates the delay produced by communication with the cloud~~. Various Base Stations in a V2I system can potentially be heterogeneous, both in terms of computational characteristics and communication medium to the core network (*e.g.*, wireless, optical fibre and wired [2]).

A problem arises during an emergency (*e.g.,* a natural disaster and road accidents) when a rapid increase in service requests to Base Stations significantly affects the tasks' response time. In fact, in this situation, ~~Base Station~~ resources become oversubscribed ~~in a way that it~~ cannot meet the deadline of all arriving tasks. Accordingly, our goal, in this research, is to design the V2I system to be robust against uncertain task arrival. *Robustness* in literature is defined as the degree to which a system can maintain a certain level of performance even with given uncertainties [6]–[8]. In this research, we measure robustness of the V2I system as the number of tasks that can meet their deadlines.

~~The specific research question we addresses in this paper is how to make the V2I system robust?~~ That is, how to allocate arriving tasks to Base Stations, so that the number of tasks meeting their deadline is maximized? To achieve robustness, any solution needs to overcome uncertainties of the system. In particular, uncertainty exists in the task arrival to the Base Station and in the communication delay.

Previous research works either discard these uncertainties (*e.g.,* [2]) or focus on the uncertainty introduced by communication (*e.g.,* [1]). Alternatively, to assure robustness of the V2I system, we propose a probabilistic resource allocation model that copes with uncertainties introduced by both communication and computation. Our proposed model is aware of the connectivity amongst Base Stations (*i.e.,* edge nodes) and their heterogeneity. In the face of oversubscription, we devise a load balancer at the Base Station level that can leverage the computational capabilities of other Base Stations to improve robustness of the V2I system.

Allocating vehicular tasks in a V2I system is proven to be an NP-complete problem [11] [12]. Therefore, a large body of research has been dedicated to develop resource allocation heuristics in V2I systems [13]–[15]. We leverage our proposed probabilistic model and develop a heuristic for the Base Station load balancer. To evaluate the performance of our proposed heuristic, we simulate the V2I environment and analyze its behavior under various workload conditions.

~~In summary, the~~ main contributions of this paper are as follows:

- Proposing a model that encompasses the uncertainties exist in communication and computation. The model is leveraged to determine the probability of completing an arriving task before its deadline on different Base Stations.
- Developing a load balancing heuristic that functions based on the proposed model and increases the robustness of the V2I system.
- We analyze the performance of our proposed heuristic under various workload conditions.

The rest of the paper is organized as follows. Section 2 introduces the scenario and assumptions made to describe the model. Section 3 includes the problem statement and the proposed system model with formulation. Section 4 discusses probabilistic allocation approach. Sections 5 and 6 present heuristics and simulation. Experimental stage and performance evaluations are particularly described in section 7. Finally, section 8 ~~wraps up the article with a conclusion~~.

II. Problem Statement and System Model

A. Problem Statement

In V2I communication ~~an ideal scenario of service happens as follows. While driving through the roads~~, vehicles request services from the Base Stations. As the request is generated, it travels from the vehicle to the Base Station. The result is sent back to the requesting vehicle when the processing is completed. The majority of requests are delay-sensitive which means that request response needs to reach the vehicle within a certain time frame. This time frame is defined in our research as an individual deadline for each specific task.

Therefore the problem can be defined as, *how to allocate arriving tasks to a base station in a V2I network so that the number of tasks missing their deadline is minimized?*

B. System Model

*1) **Formulation**:* In our scenario a set of tasks is generated by vehicles and sent to the Base Stationfor processing. Every task has its own deadline within which it has to be completed. ~~Our system allocates the tasks to Base Stations considering individual deadlines~~. Allocation algorithm aims to maximize the number of tasks meeting their deadline. According to the problem definition, the set of arriving tasks can be defined as "T", where T = $\{t_1, t_2, t_3, t_4 \ldots, t_n\}$ and the set of Base Stations "BS", where BS = $\{bs_1, bs_2, bs_3, bs_4 \ldots, bs_m\}$. The set of tasks that meet their deadline can be denoted as T_s, which is the subset of T ($T_s \subseteq$ T). It is assumed that a task t_i is allocated to the base station bs_j when the task t_i can meet its deadline δ_i in that specific base station bs_j. The problem can be formulated as:

$$\begin{aligned} \underset{t_i \in T_s}{\text{maximize}} \quad & \sum_{i=1,j=1}^{n,m} t_i bs_j, where\ t_i \in_i \& bs_j \in BS \\ \text{subject to} \quad & t_i \leq \delta_i \end{aligned}$$

*2) **Assumptions**:* We assume that V2I services are provided to the moving vehicles on the road by means of the Base Stations. In our system model Base Stations are stationary edge devices with memory storage, computational capacity, and short wireless range transmission system [15]. ~~It is also assumed that a receiving Base Stationis connected to the one hop distance Base Stations. Therefore, a receiving Base Stationcan transfer tasks to the neighboring Base Stations if needed.~~ ~~Vehicles in the vicinity of the Base Station (within its transmission range) generate service requests.~~ Assuming the variety of services is offered by a vehicular network. We categorize arriving tasks by a task type. Task types represent different requests a vehicle can submit (e.g., Hazards Around the Area, Nearby Gas Stations, Weather Forecast). ~~According to the task type, delay tolerant tasks are bigger in data size~~ (need more time to execute). Respectively, delay-sensitive tasks are smaller. Due to heterogeneity ~~across~~ ~~the~~ Base

Stations, ~~the tasks of different type h~~ave different expected completion times ~~in~~ different Base Stations.

Upon arrival of the task to a Base Station, it is assigned an individual deadline. Individual deadline includes task arrival time and the end-to- end delay the task can tolerate. We assume in our case that communication delay (uplink and downlink delay) can be significant. Thus, we also consider a communication delay for a deadline calculation. For arriving task t_i, deadline δ_i can be defined as: $\delta_{\mathrm{i}} = arr_{\mathrm{i}} + E_{\mathrm{i}} + \varepsilon + \beta$, where arr_{i} is the arrival time of the task, E_{i} is the average task completion time, ε is a constant value defined by the Base Station(slack time) and β is the communication delay.

In our system model, we assume that tasks arrive at the Base Station randomly and arrival rate is not known in advance. Receiving Base Station is considered oversubscribed, which means that it receives the number of tasks beyond its capacity to execute within a deadline. Therefore, some tasks are projected to miss their deadline. If such tasks are delay-sensitive, then they are dropped. Dropping tasks is a usual practice in oversubscribed real-time systems [18]–[20]. In such systems, the execution of a task that is going to miss its deadline has no value. For instance, a vehicle requests an update about the road conditions of a specific street. If there is no response by the time the vehicle has reached the street, there is no more value to the requested update. ~~In this research,~~ we define the task robustness as the probability of task to meet its deadline in a particular Base Station. For example, if an arriving task has a greater probability to meet its deadline in the Base Station "A" among a cluster of Base Stations, then Base Station"A" provides a greater robustness for this task. As we aim to provide a robust system from the user perspective, it is assumed that a cloud server is connected to a cluster of Base Stations for a failover or computation-intensive tasks which can not be processed in any of the Base Stations.

*3) **Delay Estimation**:* In a V2I systems, three distinct factors contribute to the definition of the end-to-end delay (D_{V2I}). ~~They are d_U, d_{BS} and d_D~~ [10]. Therefore, V2I end-to-end delay (D_{V2I}) can be defined as:

$$D_{\mathrm{V2I}} = d_{\mathrm{U}} + d_{\mathrm{BS}} + d_{\mathrm{D}} \quad (1)$$

Where d_U = average uplink delay, d_{BS} = average delay in the Base Station and d_D = average downlink delay. From the equation 1, d_U and d_D can be defined as follows. For a task t_i requested by the vehicle "i" ~~to~~ the Base Station "m", the uplink delay ~~from "i" to "m"~~ is

$$d_{\mathrm{U}} = \frac{L_{\mathrm{i}}}{g(i,m)} \quad (2)$$

and ~~for t_i traveling back from "m" to "i",~~ the downlink delay is

$$d_{\mathrm{D}} = \frac{L_{\mathrm{i}}}{g(m,i)} \quad (3)$$

, where L_i is the task ~~data~~ size, g(i,m) and g(m,i) is the ~~effective~~ transmission data rate for the link from "i" to "m" (uplink bandwidth) and from "m" to "i" (downlink bandwidth) respectively.

4) System Model Scenario: Upon the arrival to the Base Station the task gets into the load balancer. The load balancer works in an immediate mode to allocate arriving tasks to the Base Stations. It can allocate the task to the receiving Base Station or to the one-hop distance neighboring Base Station. Therefore, an arriving task gets immediately allocated by the load balancer. When the task is allocated, it enters the batch queue of the Base Station for processing. From the arrival to the end of the task processing a delay is imposed. Such delay can be defined as a "computational delay" (d_c). Therefore d_{BS} can be defined as : $d_{\mathrm{BS}} = d_c$, where d_c = average computational delay.

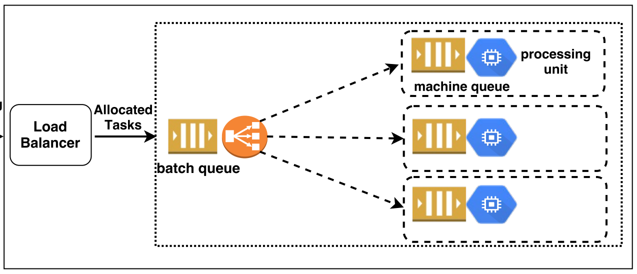

Fig. 2: ~~Base Station architecture.~~

III. Approach

A vehicle-generated task arrives at the Base Station through the uplink channel and enters the load balancer. The Load balancer is the component that allocates arriving tasks to appropriate Base Stations. It makes the decision of task allocation, considering that its robustness must be maximized. As reflection of this decision task is transferred to the Base Station that offers highest robustness. Every Base Station has two matrices. One is the Estimated Task Completion (ETC) time matrix [16]. Another one is the Estimated Task Transfer (ETT) time matrix. These two matrices help the load balancer to work efficiently.

The ETC Matrix contains estimated task completion time distributions ($X \sim \mathcal{N}(\mu, \sigma^2)$) for different task types in different Base Stations. Each ETC matrix cell contains two values, μ (mean) and σ (standard deviation). Together, this two values represent a normal distribution. This distribution is based on historical execution times of different task types on different Base Stations. In the ETC matrix, every column defines a Base Station and every row defines a task type. To capture the difference between the estimated completion times of task types we consider the worst-case scenario. Thus, we estimate completion time as the average of the previous completion times summed with its standard deviation.

The Estimated Task Transfer (ETT) time matrix also stores normal distributions of task transfer times. This distributions represent historic task transfer times for different task types from a receiving Base Station to the neighboring Base Stations. When a task is transferred from one Base Station to another, its transfer time is used to generate a normal distribution ($Y \sim \mathcal{N}(\mu, \sigma^2)$) with respect to its task type. The distributions are saved as an average (μ) and it's standard deviation (σ) in ETT matrix. Due to a real-time situation, the two matrices will be updated periodically. Thus, the cloud server sends a periodic pulse which updates the ETC and ETT matrices according to the current status of the system.

When the load balancer gets the task t_i of task type "i", it calculates the probability (P_i^j) of this task to meet its deadline δ_i across the Base Stations. For the receiving Base Station "j", the probability can be defined as $P_i^j(\gamma_i^j < \delta_i) = P_i^j(Z < z)$ where "z" is $(\delta_i - \mu_i^j) / \sigma_i^j$. We standardize the distribution with $\mu_i = 0$ and $\sigma_i = 1$. After calculating the "z" value, the probability can be found from the z-score table. For all of the neighboring Base Stations, before calculating the probability, we convolve the ETC matrix normal distribution with respective ETT matrix normal distribution. This convolution is important for the correct estimation of the probability of the task to meet its deadline in the neighboring Base Stations. This means we need to account for the transfer time as well as for the actual completion time. The convolved distribution is also a normal distribution which can be defined as: $W \sim \mathcal{N}(\mu, \sigma^2) = X \sim \mathcal{N}(\mu, \sigma^2) \circledast Y \sim \mathcal{N}(\mu, \sigma^2)$

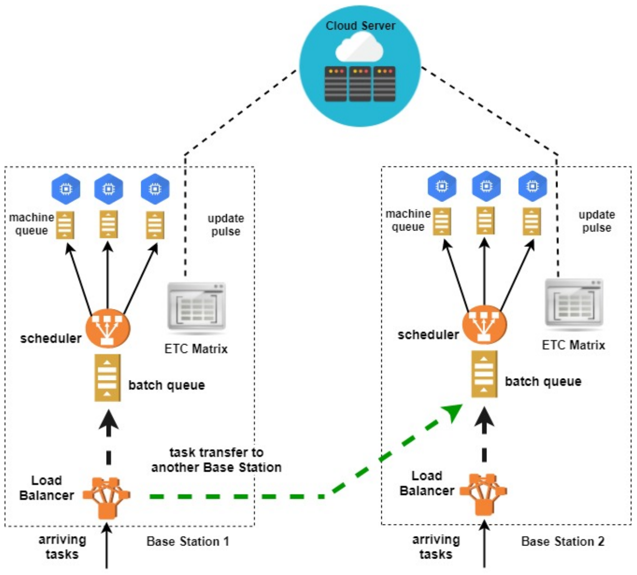

Fig. 3: A proposed model where the load balancer efficiently allocates arriving tasks

The resulting distribution ($W \sim \mathcal{N}(\mu, \sigma^2)$) is used to calculate the probability of the task in a specific Base Station. If a neighboring Base Station is "k" and task type is "i" then the probability can be defined as "P_i^k" where $z = (\delta_i - \mu_i^k) / \sigma_i^k$. When the probability of the received task in all of the Base Stations (receiving and neighboring) is calculated, it is allocated to the Base Station that offers the highest probability for the task to be completed within the deadline. When the task's probability to meet its deadline is zero (0), the task is dropped. Therefore this task will not be allocated to any of the Base Stations, nor it will enter the batch queue of any Base Station and increase the historic mean of completion times. Thus task dropping procedure during the over subscription situation will implicitly increase the probability of the other tasks to meet their deadlines.

IV. Heuristics

A. Load Balancer Heuristics

The load balancer is responsible for making an efficient task allocation decision so that the task can be processed within its individual deadline. The load balancer works in immediate mode. Therefore, upon the arrival, every task gets immediately allocated. A buffer is present in every load balancer in case many tasks arrived at the same time. Thus, the load balancer can store some tasks to serve them in the future. There is no re-allocation considered after the task has been allocated by the load balancer due to the overhead produced by additional task transfer.

1) Maximum Robustness(MR): According to the Maximum Robustness heuristic, load balancer allocates the arriving task considering its robustness across the Base Stations. The robustness of a task is defined as the probability of that specific task to meet its deadline in a particular Base Station. For instance, if the probabilities of the task to meet its deadline among 3 base stations "A", "B" and "C" are 0.8, 0.6, 0.5 respectively, then the task has the highest robustness in the Base Station "A".

At first, the load balancer calculates the probability of the arriving task to meet its deadline in receiving Base Station . After that, the deadline meeting probabilities in all the neighboring Base Stations are calculated. These probabilities are the set of values which represent robustness of the arriving task across the local Base Stations. From this robustness set, the heuristic chooses the Base Station that provides the maximum value (the highest probability) and allocates the task to that Base Station. Though, the highest probability needs to be greater than zero in order to be allocated. If there is no probability higher than zero within the set than the task is dropped.

As the heuristic only selects one Base Station for the allocation, the situation when two or more Base Stations have the same highest probability needs to be considered. In such situation, the tie can be resolved by considering the sparsity of two concurrent distributions. The sparsity is found from the standard deviations of respective distributions.

2) No Redirection(NR): No Redirection heuristic does not consider the transfer of the task to the neighboring Base Stations. Therefore, whenever the arriving task enters the load balancer of a specific Base Station, it has to be allocated to that specific Base Station. Nevertheless, allocation decision is also based on the task's robustness. For the task to be allocated, receiving Base Station has to offer the probability greater than a certain value. If this condition is not satisfied then the task is dropped.

3) Minimum Expected Completion Time(MECT): MECT heuristic is one of the baseline approaches in the literature. The core concept of this heuristic is to minimize the expected completion time of each arriving task to make an allocation decision. At first, this heuristic uses the ETC matrix to calculate the average expected completion time (μ) for the arriving task in every local Base Station . After that, the heuristic selects the Base Station which offers the minimum expected completion time and allocates the task.

B. Scheduler Heuristics

After the load balancer task allocation is performed, the scheduler of every Base Station allocates the task to the VMs. In our research, we use conventional scheduling policies. The scheduling policies are used along with the load balancer heuristics and provided as follows:

1) First Come First Serve(FCFS): FCFS is one of the popular baselines for the task scheduling policy. According to this heuristic, the tasks that arrive earlier get scheduled first. In other words, the task which arrives first stays in the head of the queue and the task that arrive later stay in the tail. The scheduler allocates tasks from the head of the queue to VM's local queue for execution. Scheduling event occurs whenever a free spot appears in VM's local queue.

2) Shortest Job First(SJF): SJF leverages the tasks with the shortest execution time. At first, the heuristic arranges the tasks in the batch queue according to their execution time in ascending order. Therefore, the tasks with a shorter execution time stay in the head of the queue and are scheduled immediately whenever a free spot appears in VM's local queue.

V. Experimental Set up

To evaluate the performance of our system we use a CloudSim simulation [17]. Cloudsim is a discrete event simulator which provides Cloud and Edge Computing models. In our simulation Base Stations are the edge devices which only have limited computational resources [13]. Therefore, we implement Base Stations in a form of small datacenters, where each datacenter is a machine with 5 cores. Each core is utilized by one VM. All VMs in a Base Station are homogeneous, i.e. have the same computational power (MIPS). Nevertheless, Base Stations are heterogeneous i.e. have different computational powers (MIPS). We also implement the bandwidth for every Base Station to calculate the communication latency in our scenario. All these parameters are scalable. In this work, we implement a small scale simulation which can be scaled up in future work. In our simulation, we implement vehicular tasks as ”cloudlets” with additional parameters added to the CloudSim's default configuration. We create 3 Base Stations and a load balancer component which allocates the tasks according to our proposed heuristic. We also implement the ETC and ETT matrices for the load balancer to utilize.

A. Workloads

In our simulation, we randomly generate execution times using Gaussian distribution [13]. We use results of the Extreme Scale System Center (ESSC) at Oak Ridge National Laboratory (ORNL)[[19], [20]] to implement the arrival time for each task. Initially, every Base Station is assigned an individual workload to create an oversubscription situation and to obtain historic estimated task completion times for the ETC matrix. Together with an individual workload, our receiving Base Station is assigned the major workload. This major workload is the one we consider for our results evaluation, i.e. the three individual workloads do not affect the results explicitly. The workloads are seeded and changed from trial to trial. By manipulating the number of tasks in the initial workloads, we can control the system's oversubscription level.

VI. Experiments

To study the performance of our system thoroughly, we evaluated the system from two perspective. First one is the impact of increasing arrival tasks to Base Station when the system is oversubscribed with certain amount of initial tasks. The second experiment is on impact of deadline looseness on different heuristics to evaluate the performance of our proposed heuristic.

*a) **Impact of increasing arrival requests**:* In this experiment, we compare our load balancer (MR heuristic) with no load balancer(NR heuristic) by increasing the number of arrival requests to Base Station. We keep Base Station‘s environment at the same oversubscription level for every heuristic and only changing the number of tasks arriving to Base Station. We evaluate our proposed heuristic in terms of deadline missing requests.

*b) **Increasing over subscription level**:* In this experiment the impact of load balancer is tested with two different settings. In first setting we use 50 tasks to create over subscription in each Base Station. And in second setting over subscription is created with 100 tasks.

The simulation is run with starting workload of 100 tasks and incrementing 100 tasks in every trial. So simulation is run for 30 trial with different seeds using Load balancer and without Load Balancer. The average of every 30 trial is used for plotting barchart.

For this experiment the load balancer assigned the tasks to Base Station with highest probability. Here the probability condition of task dropping is zero which means that the highest probability value needs to be greater than zero. If there is no such Base Station then task is dropped or not allocated. This experiment is the implementation of MR heuristic which is mentioned in heuristic section. For this experiment two different scheduling heuristics are used. They are FCFS and SJF.The result plotting is given below :

From the above result it is visible that using load balancer leads to better performance. For smaller number of tasks difference between them is negligible. With the increased number of tasks load balancer works better than conventional scenario. From the two scheduling policy FCFS works better than SJF.

*c) **Impact of Deadline looseness**:* In this experiment we investigate the impact of deadline looseness on implemented heuristics. For this purpose initial over subscription is created with 3000 task workload in every base station. In every trial on top of initial over subscription, a batch of 1000 task with

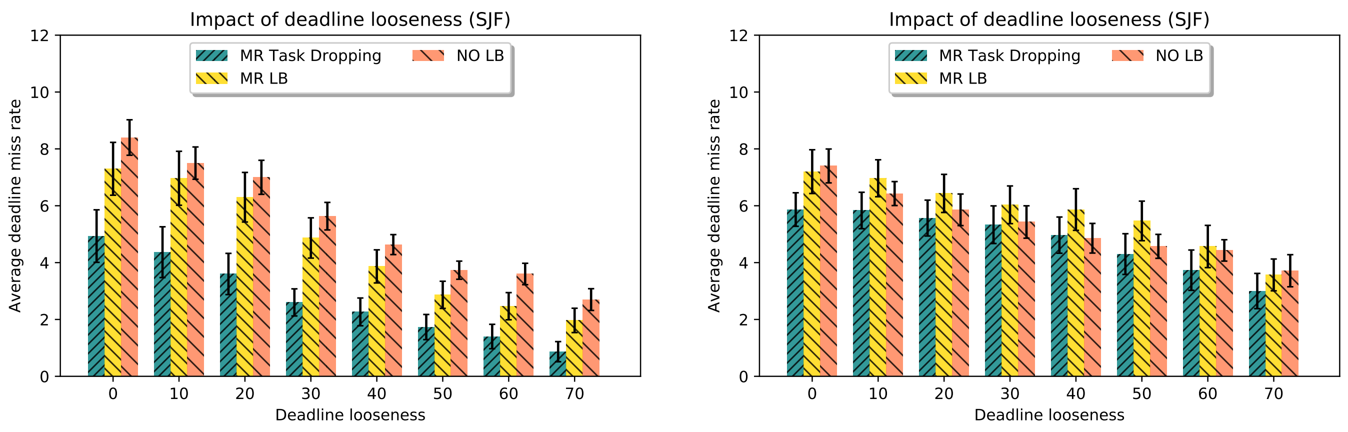

Fig. 4: Performance of heuristics on deadline looseness for FCFS and SJF

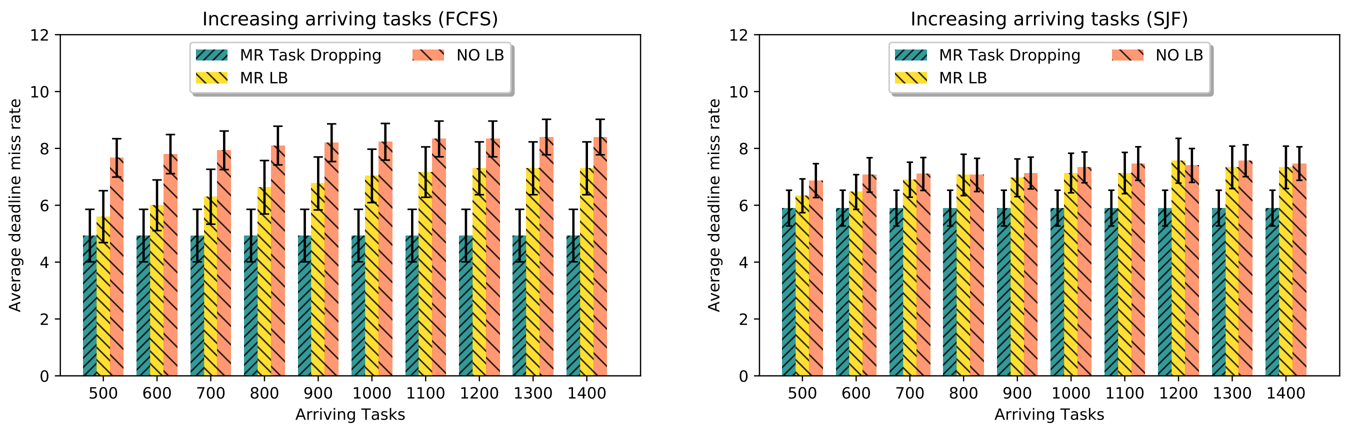

Fig. 5: Performance of heuristics on increased arriving tasks for FCFS and SJF

different arrival time is provided to the system. As slack is one of the parameter for calculation of deadline, it is used for loosen the deadline tightness. In every trial we run the simulation for 10 times using different seeds with MR load balancer, MECT load balancer and without load balancer. Then we take the average for plotting the result. We have increased the slack by adding integer value 10 in every trial starting from 0.12 to 90.12.The result is given below :

d) Task Dropping with given probability: In this experiment we have used our heuristic with certain threshold value as highest probability for task allocation decision. As such the heuristic allocates those tasks to specific Base Stations greater than the threshold probability.

VII. Related Work

Efficient resource allocation which can decrease the deadline missing rate is the major challenge for V2I systems. Oversubscription situations only make this challenge even more complex. The robustness and QoS can deteriorate due to the lack of efficient resource allocation in a V2I system. Thus, a proficient task allocation decision can decrease latency and significantly improve robustness.

There are several approaches proposed in the literature. In [13] [Jun Li et al. 2017] propose a local fog resource management model in Fog Enhanced Radio Access Network (FeRANs) based on V2X environment. The core concept of this paper is to improve the QoS at each individual fog node(e.g., Base Station) for real-time vehicular services. Authors propose two resource management schemes. Both schemes prioritize real-time vehicular services over other services. The model considers service migration from one fog node to another based on reserved resource availability.

[Ali et al. 2011] in [1] propose a multiple RSU scheduling to provide cooperative data access to multiple RSUs in vehicular ad-hoc networks. They categorize requests or tasks into two types (delay sensitive and delay tolerant) according to their data size. In oversubscription situation, authors propose to transfer delay tolerant requests to the neighboring RSUs with a lower workload.

In [15] [Liu and Lee 2010] suggest an RSU-based data dissemination framework to address challenges in vehicular networks. The system aims to efficiently utilize available bandwidth for both the safety-critical and the non-safety-critical services. An analytical model is proposed to investigate the system performance in terms of providing data services with delay constraints.

[Tong et al. 2016] in [21] offer a hierarchical architecture of the Edge to maximize the volume of mobile workload served in an oversubscription situation. Authors emphasize the advantage of hierarchical cloud architecture over a flat architecture. Nevertheless, hierarchical architecture also has some problems. For instance, it enables aggregation of the peak loads across different tiers to maximize the workload volume served, which is not efficient. As a solution authors propose a workload placement algorithm for mobile program/workload placement on edge cloud and provisioning computational capacity.

[Adachi et al. 2016] in [22] suggest a special model for local resource reservation and reallocation based on the priority of services.

Although various extensive research works have been embraced in the field of vehicular systems [23]–[25], they are limited to the issues of communication functionality and quality requirements from networking perspective. However, none of them considered V2I system in perspective of computational capacity and task processing in a Base Station (*e.g.,* edge device). Whereas our work can be distinguished as it contemplates a V2I system in both computational and communicational perspectives.

VIII. Conclusions

In this paper, we proposed a robust probabilistic resource allocation model that copes with the uncertainties exist both in communication and computation. We leveraged the probabilistic model to devise a load balancer that operates at the Base Station (edge) level and distributes arriving tasks to other Base Stations in the face of oversubscription. Experimental results express that our proposed model can significantly improve the robustness of the system (by up to 30%), WHEN??? . As a result, the whole V2I system is more robust and reliable in emergency situation. From simulation results it is observed that for deadline looseness our proposed heuristic outperforms other heuristics. MORE INTERESTING TAKE A WAY MESSAGES. In future, we plan to extend our probabilistic model for cases that have heterogeneity within each Base Station. Another avenue of future research is to incorporate delay tolerant and intolerant tasks together in a Base Station.

References

[1] G. M. N. Ali and E. Chan, "Co-operative data access in multiple road side units (rsus)-based vehicular ad hoc networks (vanets)," in *Telecommunication Networks and Applications Conference (ATNAC), 2011 Australasian*. IEEE, 2011, pp. 1–6.

[2] K. Bok, S. Hong, J. Lim, and J. Yoo, "A multiple rsu scheduling for v2i-based data services," in *Big Data and Smart Computing (BigComp), 2016 International Conference on*. IEEE, 2016, pp. 163–168.

[3] Rong Yu, Jiefei Ding, Xumin Huang, Ming-Tuo Zhou, Stein Gjessing, and Yan Zhang. Optimal resource sharing in 5g-enabled vehicular networks: A matrix game approach. *IEEE Transactions on Vehicular Technology*, 65(10):7844–7856, 2016.

[4] Mario Gerla. Vehicular cloud computing. In *Ad Hoc Networking Workshop (Med-Hoc-Net), 2012 The 11th Annual Mediterranean*, pages 152–155. IEEE, 2012.

[5] "Allied business intelligenc Inc." https://www.abiresearch.com/market-research/product/1016485-v2v-and-v2i-applications-and-use-cases/, accessed Dec 21, 2017.

[6] Shoukat Ali, Anthony A Maciejewski, Howard Jay Siegel, and Jong-Kook Kim. Measuring the robustness of a resource allocation. *IEEE Transactions on Parallel and Distributed Systems*, 15(7):630–641, 2004.

[7] Jay Smith, Vladimir Shestak, Howard Jay Siegel, Suzy Price, Larry Teklits, and Prasanna Sugavanam. Robust resource allocation in a cluster based imaging system. *Parallel Computing*, 35(7):389–400, 2009.

[8] Louis-Claude Canon and Emmanuel Jeannot. Evaluation and optimization of the robustness of dag schedules in heterogeneous environments. *IEEE Transactions on Parallel and Distributed Systems*, 21(4):532–546, 2010.

[9] T. ElBatt, S. K. Goel, G. Holland, H. Krishnan, and J. Parikh, "Cooperative collision warning using dedicated short range wireless communications," in *Proceedings of the 3rd International Workshop on Vehicular Ad Hoc Networks*, ser. VANET '06. New York, NY, USA: ACM, 2006, pp. 1–9. [Online]. Available: http://doi.acm.org/10.1145/1161064.1161066

[10] A. Mostafa, A. M. Vegni, R. Singoria, T. Oliveira, T. D. Little, and D. P. Agrawal, "A v2x-based approach for reduction of delay propagation in vehicular ad-hoc networks," in *ITS Telecommunications (ITST), 2011 11th International Conference on*. IEEE, 2011, pp. 756–761.

[11] J. D. Ullman, "Np-complete scheduling problems," *J. Comput. Syst. Sci.*, vol. 10, no. 3, pp. 384–393, Jun. 1975. [Online]. Available: http://dx.doi.org/10.1016/S0022-0000(75)80008-0

[12] J. K. Lenstra and A. H. G. R. Kan, "Complexity of vehicle routing and scheduling problems," *Networks*, vol. 11, no. 2, pp. 221–227, 1981. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/net.3230110211

[13] J. Li, C. Natalino, D. P. Van, L. Wosinska, and J. Chen, "Resource management in fog-enhanced radio access network to support real-time vehicular services," in *Fog and Edge Computing (ICFEC), 2017 IEEE 1st International Conference on*. IEEE, 2017, pp. 68–74.

[14] S.-Y. Pyun, W. Lee, and D.-H. Cho, "Resource allocation for vehicle-to-infrastructure communication using directional transmission," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 4, pp. 1183–1188, 2016.

[15] K. Liu and V. C. Lee, "Rsu-based real-time data access in dynamic vehicular networks," in *Intelligent Transportation Systems (ITSC), 2010 13th International IEEE Conference on*. IEEE, 2010, pp. 1051–1056.

[16] S. Ali, H. J. Siegel, M. Maheswaran, D. Hensgen, and S. Ali, "Representing task and machine heterogeneities for heterogeneous computing systems," *Tamkang Journal of Science and Engineering*, vol. 3, no. 3, pp. 195–207, 2000.

[17] 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," *Softw. Pract. Exper.*, vol. 41, no. 1, pp. 23–50, jan 2011.

[18] Bhavesh Khemka, Ryan Friese, Sudeep Pasricha, Anthony A Maciejewski, Howard Jay Siegel, Gregory A Koenig, Sarah Powers, Marcia Hilton, Rajendra Rambharos, and Steve Poole. Utility driven dynamic resource management in an oversubscribed energy-constrained heterogeneous system. In *Parallel & Distributed Processing Symposium Workshops (IPDPSW), 2014 IEEE International*, pages 58–67. IEEE, 2014.

[19] B. Khemka, R. Friese, L. D. Briceno, H. J. Siegel, A. A. Maciejewski, G. A. Koenig, C. Groer, G. Okonski, M. M. Hilton, R. Rambharos *et al.*, "Utility functions and resource management in an oversubscribed heterogeneous computing environment," *IEEE Transactions on Computers*, vol. 64, no. 8, pp. 2394–2407, 2015.

[20] B. Khemka, R. Friese, S. Pasricha, A. A. Maciejewski, H. J. Siegel, G. A. Koenig, S. Powers, M. Hilton, R. Rambharos, and S. Poole, "Utility maximizing dynamic resource management in an oversubscribed energy-constrained heterogeneous computing system," *Sustainable Computing: Informatics and Systems*, vol. 5, pp. 14 – 30, 2015. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2210537914000420

[21] L. Tong, Y. Li, and W. Gao, "A hierarchical edge cloud architecture for mobile computing," in *INFOCOM 2016-The 35th Annual IEEE International Conference on Computer Communications, IEEE*. IEEE, 2016, pp. 1–9.

[22] Y. Adachi, H. Yamaguchi, T. Higashino, and T. Umedu, "Cloud-assisted dynamic content sharing among vehicles," in *Computer and Information*

Technology (CIT), 2016 IEEE International Conference on. IEEE, 2016, pp. 516–523.

[23] O. Maeshima, S. Cai, T. Honda, and H. Urayama, "A roadside-to-vehicle communication system for vehicle safety using dual frequency channels," in *Intelligent Transportation Systems Conference, 2007. ITSC 2007. IEEE*. IEEE, 2007, pp. 349–354.

[24] G. Korkmaz, E. Ekici, and F. Ozguner, "Internet access protocol providing qos in vehicular networks with infrastructure support," in *Intelligent Transportation Systems Conference, 2006. ITSC'06. IEEE*. IEEE, 2006, pp. 1412–1417.

[25] T. K. Mak, K. P. Laberteaux, and R. Sengupta, "A multi-channel vanet providing concurrent safety and commercial services," in *Proceedings of the 2nd ACM international workshop on Vehicular ad hoc networks*. ACM, 2005, pp. 1–9.