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

Razin Farhan Hussain
High Performance Cloud Computing (HPCC) Lab.
School of Computing and Informatics,
University of Louisiana at Lafayette
Louisiana, USA.
razinfarhan.hussain1@louisiana.edu

Mohsen Amini Salehi
High Performance Cloud Computing (HPCC) Lab.
School of Computing and Informatics,
University of Louisiana at Lafayette
Louisiana, USA.
amini@louisiana.edu

Anna Kovalenko
High Performance Cloud Computing (HPCC) Lab.
School of Computing and Informatics,
University of Louisiana at Lafayette
Louisiana, USA
aok8889@louisiana.edu

Omid Semiari
Department of Electrical and Computer Engineering,
Georgia Southern University
Georgia, USA.
osemiari@georgiasouthern.edu

ABSTRACT

Autonomous vehicles require fast and reliable services to manage road hazards. Services provided in these networks (*e.g.,* traffic information and natural disaster updates) are generally delay intolerant. Hence, Vehicle-to-Infrastructure (V2I) systems must provide consistent real-time response to autonomous vehicles. Maintaining real-time response can be particularly difficult, at a disaster time, when there is a surge of request arrivals to Base Stations with edge computing capabilities. Accordingly, our goal is to make the system robust (*i.e.,* able to maintain its performance) against 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 Base Stations. Simulation results demonstrate that our proposed allocation model decreases deadline miss rate by up to 90% when compared with conventional V2I systems.

ACM Reference Format:
Razin Farhan Hussain, Mohsen Amini Salehi, Anna Kovalenko, and Omid Semiari. 2018. Robust Resource Allocation Using Edge Computing for Delay Sensitive Tasks in Vehicle to Infrastructure (V2I) Networks. In *Proceedings of 47th International Conference on Parallel Processing (ICPP'18).* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

ICPP'18, August 2018, Eugene, Oregon USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM.
https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Recent advancements in communication and computation technologies have stimulated a rapid development of vehicular networks. Federal Communications Commission (FCC) [1] has already reserved 5.850 to 5.925 GHz frequency band for Vehicle-to-Everything (*V2X*) communications. Vehicle-to-Infrastructure (V2I) communication is one prominent form of V2X that draws the majority of work to itself. In V2I, infrastructure refers to all edge and core technologies that facilitate communications and computation of vehicular tasks.

As shown in Figure 1, autonomous vehicles send their service requests to Base Stations while travelling a road. A Base Station (BS) is capable of communicating with vehicles, and can process vehicular tasks [3]. 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 [9], cooperative forward collision warning [8], and lane change warning [9]. This type of tasks can tolerate a short end-to-end delay [1]. End-to-end delay is composed of three contributing factors [21], namely up-link delay, processing delay, and down-link delay. In end-to-end delay "uplink delay" and "down-link delay" can be defined as communication delay. Whereas processing delay can be defined as "computational delay". In case of delay sensitive tasks, 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.

For providing effective support to the vehicular services, it is suggested by the literature that V2I systems should be enhanced by incorporating wireless network infrastructure and cloud computing [25]. For instance, the vehicular cloud computing (VCC) has been initiated in [11] as a subsection of mobile cloud computing. However, both VCC and conventional cloud systems incur a high latency [17], thus, cannot be used for delay intolerant tasks of V2I systems. Nonetheless, Base Stations' computational power can be harnessed as an edge computing system. With edge computing, vehicular services can be managed directly at the BS with low latency, without requiring to communicate 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 [3]).

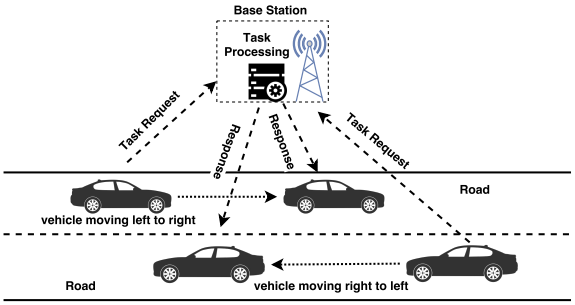

Figure 1: A Vehicle to Infrastructure (V2I) scenario where vehicles send requests to a Base Station and receive the response. A Base Station is a roadside unit with communication and computation abilities.

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' service time. In fact, in this situation, BS resources become oversubscribed and the BS 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 the literature is defined as the degree to which a system can maintain a certain level of performance even with given uncertainties [2, 4, 23]. In this research, we measure robustness of the V2I system as the number of tasks that can meet their deadlines. Therefore, the problem in this case is how to allocate arriving tasks to Base Stations, so that the system is robust (*i.e.,* the number of tasks meeting their deadlines 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[3] or focus on the uncertainty introduced by communication[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 [24] [15]. Therefore, a large body of research has been dedicated to develop resource allocation heuristics in V2I systems [2, 13, 15, 17, 19, 22, 25]. 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.

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 concludes the paper.

2 PROBLEM STATEMENT AND SYSTEM MODEL

2.1 Problem Statement

In V2I communication, 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?*

2.2 System Model

2.2.1 ***Formulation.*** In our scenario a set of tasks is generated by vehicles and sent to the Base Station for processing. Every task has its own deadline within which it has to be completed. 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 $\mathrm{T} = \{t_1, t_2, t_3, t_4 \ldots, t_n\}$ and the set of Base Stations "S", where $\mathrm{S} = \{s_1, s_2, s_3, s_4 \ldots, s_m\}$. The set of tasks that meet their deadline can be denoted as T_s, which is the subset of T ($T_s \subseteq \mathrm{T}$). It is assumed that a task t_i is allocated to the base station s_j when the task t_i can meet its deadline δ_i in that specific base station. The problem can be formulated as:

$$\begin{aligned} \underset{t}{\text{maximize}} \quad & \sum_{i=1}^{n}\sum_{j=1}^{m} t_{ij}, where\ t_{ij} \in \mathrm{T}_s \\ \text{subject to} \quad & t_i \leq \delta_i\ , where\ t_{ij} \in \{0,1\} \end{aligned}$$

In the above equation t_{ij} is an indicator random variable that indicates whether or not a task is allocated to a Base Station. Successful allocation represent 1 and otherwise 0. Therefore above problem is Binary Integer Linear Program(BILP).

2.2.2 ***Assumptions.*** 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 a certain wireless range transmission system [19]. Meanwhile, BSs are connected to one another via X2 backhaul links, enabling them to exchange tasks with their neighboring BSs. Variety of services are offered by a vehicular network.

We categorize arriving tasks by a task type. Task types represent different requests a vehicle can submit. For instance electronic toll collection, autonomous driving, intersection collision warning, traffic information, blind-spot warning, slow or stopped vehicle warning, curve speed warning, entertainment data or download services. Typically, delay-tolerant tasks require large size data packets(need more time to execute). Respectively, delay-sensitive tasks are smaller. Due to heterogeneity of Base Stations, different task types may have different expected completion times when processed at 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. In real-time scenario vehicles can request tasks that require high data rate (*e.g.,* HD-maps, entertainment data). On the other hand number of vehicles can be large and each vehicle may request several tasks simultaneously which may strain the capacity of wireless network. For above mentioned reasons communication delay (uplink and downlink delay) can have significant impact in vehicular services. Thus, we also consider a communication delay for a deadline calculation. For arriving task t_i, deadline δ_i can be defined as: $\delta_i = arr_i + E_i + \varepsilon + \beta$, 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 [5, 6, 13]. 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.

Here, we define the task robustness as the probability of task to meet its deadline in a particular Base Station. For an arriving task of type "i" in a Base Station j, the robustness can be defined as : $P_j(t_i)$. Therefore if an arriving task has a greater probability to meet its deadline in the Base Station "A" ($P_A(t_i)$) among a cluster of Base Stations, then Base Station"AâĂİ 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.

2.2.3 ***Delay Estimation.*** In a V2I systems, three distinct factors contribute to the definition of the end-to-end delay (D_{V2I})[21]. Therefore, V2I end-to-end delay (D_{V2I}) can be defined as:

$$D_{V2I} = d_U + d_{BS} + d_D \tag{1}$$

Where d_U denotes average uplink delay, d_{BS} denotes average delay in the Base Station and d_D denotes average downlink delay. From the (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 is

$$d_U = \frac{L_i}{R_i^m} \tag{2}$$

and the downlink delay is

$$d_D = \frac{L_i}{R_m^i} \tag{3}$$

, where L_i is the task packet size, R_i^m and R_m^i is the transmission data rate for the link from "i" to "m" (uplink bandwidth) and from "m" to "i" (downlink bandwidth) respectively.

2.2.4 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 : $d_{BS} = d_c$, where d_c represents average computational delay.

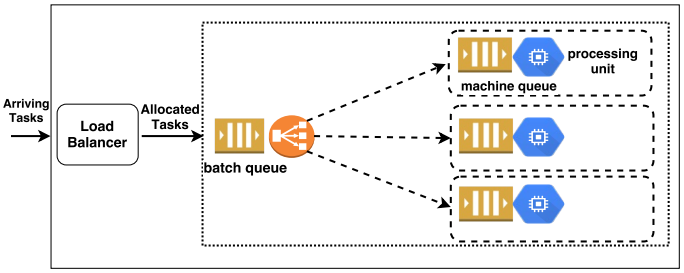

Figure 2: Architecture of the edge computing unit at each BS.

3 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 [10]. 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: $W \sim \mathcal{N}(\mu, \sigma^2) = X \sim \mathcal{N}(\mu, \sigma^2) \circledast Y \sim \mathcal{N}(\mu, \sigma^2)$

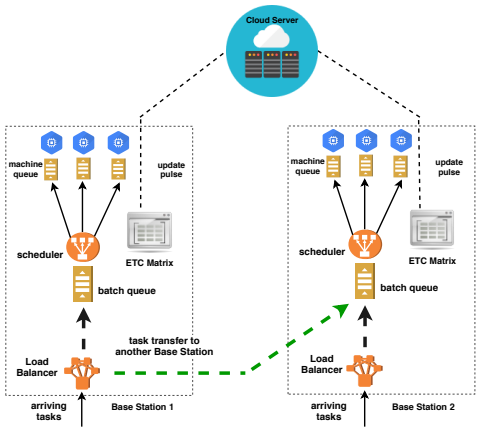

Figure 3: A proposed model where the load balancer efficiently allocates arriving tasks

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.

4 HEURISTICS

4.1 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.

4.1.1 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.

4.1.2 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.

4.1.3 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.

4.2 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:

4.2.1 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.

4.2.2 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.

5 EXPERIMENTAL SET UP

To evaluate the performance of our system we use a CloudSim simulation [14]. 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 [17]. 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.

5.1 Workloads

In our simulation, we randomly generate execution times using Gaussian distribution[17]. We use results of the Extreme Scale System Center (ESSC) at Oak Ridge National Laboratory (ORNL)[[6], [5]] 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.

6 EXPERIMENTS

To study the performance of our model thoroughly, we evaluate the system under various amounts of arriving tasks. We analyze the impact of the increasing workload on the oversubscribed system. Initially, each Base Station in the system has it's own workload and is proved to be oversubscribed (70 - 80% of arriving tasks miss the deadline). Additionally, we assign a testing workload to the Base Station which uses a load balancer to allocate them. The individual workload for each Base Stationis independent of the testing workload, as we assume the testing workload to be extra request bursts that can occur during the rush hours or the emergency situations. We consider the minimum testing workload to be a 100 tasks and the maximum to be a 1000 tasks. Each step we increase the testing workload by a 100 tasks and account for the number of tasks that missed their deadline. For each step, we perform 30 experiments and calculate the average number of tasks that miss the deadline. We compare the average number of tasks that missed the deadline while using MR and MECT heuristics. Figure 4 demonstrates the results of described experiments. The first image of figure 4 represents the results obtained using the FCFS scheduling algorithm. As we can see, the load balancer performs significantly better using the MR heuristic rather then MECT. About only 5% of all tasks of the testing workload miss their deadlines using the MR heuristic, where more than 90% of the testing tasks miss their deadline with the MECT heuristic. Apparently, the number of additionally arriving tasks does not significantly affect the performance of the algorithms, as the miss rate does not change much with the increase in the number of tasks. We assume that the performance of the MR heuristic depends on the level of the initial oversubscription of the system, but we leave finding the proof for our assumption for the future research. We can also notice that MR heuristic performs slightly better using the FCFS scheduling rather than SJF. The reason for that can be the common nature of all the arriving tasks (the burst times for different times are not much different). Overall, we can observe that MR heuristic performs noticeably better than our baseline heuristic, which can provide a better robustness for the whole system in the situations of oversubscription.

7 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 [17] [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

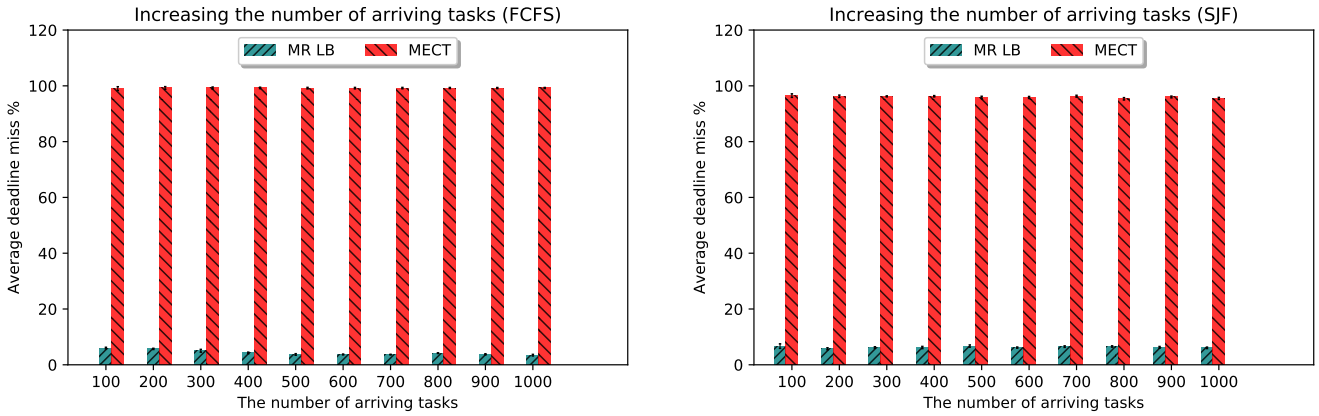

Figure 4: Performance of heuristics on deadline looseness for FCFS and SJF

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 [19] [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 [7] 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 [18] proposed an hybrid approach for vehicular content (*e.g.,* short video clips, sensor data) sharing among vehicles using V2V communication and V2I communication via cellular network. The main objective of this paper is to reduce high-cost V2I due to traffic volume in wireless network.

Although various extensive research works have been embraced in the field of vehicular systems [12, 16, 20], 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.

8 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 90%), when the system is seriously oversubscribed. As a result, the whole V2I system is more robust and reliable in emergency situation. From simulation results it is observed that for the various amounts of arriving task our proposed heuristic outperforms the MECT. About only 5% of all tasks miss their deadlines using the MR heuristic, where more than 90% of the testing tasks miss their deadline with the MECT heuristic. In future, we plan to extend our probabilistic model for cases that have heterogeneity within each Base Station. We also plan to study the impact of the initial oversubscription level and deadline looseness on the system. Another avenue of future research is to incorporate delay tolerant and intolerant tasks together in a Base Stationin a situation of oversubscription.

REFERENCES

[1] GG Md Nawaz Ali and Edward Chan. 2011. 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, 1–6.

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

[3] Kyoungsoo Bok, Seungwan Hong, Jongtae Lim, and Jaesoo Yoo. 2016. A multiple RSU scheduling for V2I-based data services. In *Big Data and Smart Computing (BigComp), 2016 International Conference on*. IEEE, 163–168.

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

[5] "Bhavesh Khemka, Ryan Friese, Sudeep Pasricha, Anthony A. Maciejewski, Howard Jay Siegel, and Gregory A. "2015". "Utility maximizing dynamic resource management in an oversubscribed energy-constrained heterogeneous computing system". *"Sustainable Computing: Informatics and Systems"* "5" ("2015"), "14 – 30".

[6] Bhavesh Khemka, Ryan Friese, Luis D Briceno, Howard Jay Siegel, Anthony A Maciejewski, Gregory A Koenig, Chris Groer, Gene Okonski, Marcia M Hilton, Rajendra Rambharos, et al. 2015. Utility functions and resource management in an oversubscribed heterogeneous computing environment. *IEEE Trans. Comput.*

64, 8 (2015), 2394–2407.

[7] Liang Tong, Yong Li, and Wei Gao. 2016. A hierarchical edge cloud architecture for mobile computing. In *INFOCOM 2016-The 35th Annual IEEE International Conference on Computer Communications, IEEE*. IEEE, 1–9.

[8] Tamer ElBatt, Siddhartha K. Goel, Gavin Holland, Hariharan Krishnan, and Jayendra Parikh. 2006. Cooperative Collision Warning Using Dedicated Short Range Wireless Communications. In *Proceedings of the 3rd International Workshop on Vehicular Ad Hoc Networks (VANET '06)*. ACM, New York, NY, USA, 1–9. https://doi.org/10.1145/1161064.1161066

[9] accessed Dec 21, 2017. Allied Business Intelligenc Inc. https://www.abiresearch.com/market-research/product/1016485-v2v-and-v2i-applications-and-use-cases/. (accessed Dec 21, 2017).

[10] Shoukat Ali, Howard Jay Siegel, Muthucumaru Maheswaran, Debra Hensgen, and Sahra Ali. 2000. Representing Task and Machine Heterogeneities for Heterogeneous Computing Systems. *Tamkang Journal of Science and Engineering* 3, 3 (2000), 195–207.

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

[12] Osamu Maeshima, Shengwei Cai, Teruhiko Honda, and Hirofumi Urayama. 2007. A roadside-to-vehicle communication system for vehicle safety using dual frequency channels. In *Intelligent Transportation Systems Conference, 2007. ITSC 2007. IEEE*. IEEE, 349–354.

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

[14] Rodrigo N. Calheiros, Rajiv Ranjan, Anton Beloglazov, César A. F. De Rose, and Rajkumar Buyya. 2011. CloudSim: A Toolkit for Modeling and Simulation of Cloud Computing Environments and Evaluation of Resource Provisioning Algorithms. *Softw. Pract. Exper.* 41, 1 (jan 2011), 23–50.

[15] J. K. Lenstra and A. H. G. Rinnooy Kan. 1981. Complexity of vehicle routing and scheduling problems. *Networks* 11, 2 (1981), 221–227. https://doi.org/10.1002/net.3230110211 arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/net.3230110211

[16] Gökhan Korkmaz, Eylem Ekici, and F Ozguner. 2006. Internet access protocol providing QoS in vehicular networks with infrastructure support. In *Intelligent Transportation Systems Conference, 2006. ITSC'06. IEEE*. IEEE, 1412–1417.

[17] Jun Li, Carlos Natalino, Dung Pham Van, Lena Wosinska, and Jiajia Chen. 2017. 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, 68–74.

[18] Yoshiaki Adachi, Hirozumi Yamaguchi, Teruo Higashino, and Takaaki Umedu. 2016. Cloud-assisted Dynamic Content Sharing among Vehicles. In *Computer and Information Technology (CIT), 2016 IEEE International Conference on*. IEEE, 516–523.

[19] Kai Liu and Victor CS Lee. 2010. RSU-based real-time data access in dynamic vehicular networks. In *Intelligent Transportation Systems (ITSC), 2010 13th International IEEE Conference on*. IEEE, 1051–1056.

[20] Tony K Mak, Kenneth P Laberteaux, and Raja Sengupta. 2005. A multi-channel VANET providing concurrent safety and commercial services. In *Proceedings of the 2nd ACM international workshop on Vehicular ad hoc networks*. ACM, 1–9.

[21] Ahmad Mostafa, Anna Maria Vegni, Rekha Singoria, Talmai Oliveira, Thomas DC Little, and Dharma P Agrawal. 2011. A V2X-based approach for reduction of delay propagation in vehicular Ad-Hoc networks. In *ITS Telecommunications (ITST), 2011 11th International Conference on*. IEEE, 756–761.

[22] Sung-Yeop Pyun, Woongsup Lee, and Dong-Ho Cho. 2016. Resource allocation for vehicle-to-infrastructure communication using directional transmission. *IEEE Transactions on Intelligent Transportation Systems* 17, 4 (2016), 1183–1188.

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

[24] J. D. Ullman. 1975. NP-complete Scheduling Problems. *J. Comput. Syst. Sci.* 10, 3 (June 1975), 384–393. https://doi.org/10.1016/S0022-0000(75)80008-0

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