Game Theory-based Allocation Management in VCC Networks

Binal Tejani

Computer Science

Submitted in partial fulfilment
of the requirements for the degree of

Master of Science

Faculty of Mathematics and Science, Brock University
St. Catharines, Ontario

Abstract

Vehicular Ad-hoc Networks (VANETs) have contributed significantly towards improving road traffic management and safety. VANETs, integrated with Vehicular Clouds, enable underutilized vehicular resources for efficient resource management, fulfilling service requests. However, due to the frequently changing network topology of vehicular cloud networks, the vehicles frequently move out of the coverage area of roadside units (RSUs), disconnecting from the RSUs and interrupting the fulfillment of ongoing service requests. In addition, working with heterogeneous vehicles makes it difficult to match the service requests with the varying resources of individual vehicles. Therefore, to address these challenges, this work introduces the concept of clustering resources from nearby vehicles to form Combined Resource Units (CRUs). These units contribute to maximizing the rate of fulfillment of service requests. CRU composition is helpful, especially for the heterogeneity of vehicles, since it allows clustering the varying resources of vehicles into a single unit. The vehicle resources are clustered into CRUs based on three different sized pools, making the service matching process more time-efficient. Previous works have adopted stochastic models for resource clustering configurations. However, this work adopts distinct search algorithms for CRU composition, which are computationally less complex. Results showed that light-weight search algorithms, such as selective search algorithm (SSA), achieved close to 80% of resource availability without over-assembling CRUs in higher density scenarios. Following CRU composition, a game-theoretical approach is opted for allocating CRUs to service requests. Under this approach, the CRUs play a non-cooperative game to maximize their utility, contributing to factors such as fairness, efficiency, improved system performance and reduced system overhead. The utility value takes into account the RSS (Received Signal Strength) value of each CRU and the resources required in fulfilling a request. Results of the game model showed that the proposed approach of CRU composition obtained 90% success rate

towards matching and fulfilling service requests.

Acknowledgements

I would like to express my deepest gratitude and appreciation to my supervisor, Dr. Robson E. De Grande, who has been my constant support throughout my thesis. He has constantly motivated me and provided me guidance, supervision and timely and insightful feedback to improvise at every stage of my thesis. I would like to thank him for his immense patience in addressing and settling every doubt that I encountered during my thesis. Without his constant motivation, patience and guidance, the successful completion of this thesis was not possible.

I would like to express my sincere gratitude towards my fellow students, Abubakar Saad and Thiago Gomides D'Silva for their constant support and timely acknowledgement to several problems that I encountered in terms of coding. Their insightful ideas in several aspects of my thesis have contributed greatly in achieving the desired results for my thesis.

I would like to express my special appreciation to the Department of Computer Science at Brock University for providing me with an opportunity to conduct my thesis in an ethical work environment with all necessary facilities provided.

And lastly, I would like to thank my family for their constant support, encouragement and for always keeping my morale high in fulfilling my dream of successfully completing my thesis.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

The Intelligent Transportation System (ITS) has gained considerable attention in the modern world due to its applications in road safety and the transportation industry, fulfilling the QoS (Quality of Service) requirements of users and operational efficiency through the advanced techniques of information transmission. Due to the significant advancements in road and traffic safety, considerable research works have been focused on the advances of vehicular networks integrated with cloud computing technologies. These advancements have provided users on the road with practicability and convenience in terms of road safety and traffic management and has led to the development of Vehicular Cloud Computing (VCC) [26].

VCC networks provide users access to information, such as heavy traffic routes and accident scenarios. This information sharing can be enabled through vehicles in close vicinity to each other, forwarding and exchanging information through cloud networks. The vehicles and roadside infrastructure (RSUs) are essential elements that compose VCC networks. They contribute towards efficient data processing and acquisition, through devices such as smart sensors and actuators built into them. They also play an active role in resource sharing by collaborating their underutilized resources for resource allocation and management [22].

The vehicles and RSUs in the VCC network make use of an extensive range of communication modes, which include: V2V (vehicle-to-vehicle) and V2I (vehicle-to-infrastructure). The RSUs are an integral part of the V2I communication since these units allow vehicles to transmit and download data from the networking platform. A VCC network may consist of several RSUs that are usually hard wire connected so that they have an infinite lifetime in the network and can be used for providing

services to users if/when mobile vehicles in the network move outside of the transmission range of the requester vehicle. V2V enables vehicles in close vicinity to transmit and receive data from each other without the involvement of RSUs, thus contributing significantly towards reduced power consumption and transmission delay.

VCC allows for dynamic resource allocation, which contributes towards efficient scaling in large networks [9]. VCC networks are generally classified based on the services they offer, such as NaaS (Network as a Service), SaaS (Sensing as a Service) [11] and CaaS (Computation as a Service). This thesis primarily focuses on CaaS, as demonstrated in [15], which suggests that idle vehicles in a parking lot contribute to building a strong VCC network. CaaS works by each vehicle contributing a small number of their computing resources to the cloud system, which would help fulfill user service requests in the surrounding area.

Previous works have considered stochastic models such as SMDP and MDP for clustering resources within a VCC network, disregarding the heterogeneity of vehicles (vehicles that may have varying amounts of computing resources). However, recently, different manufacturers make vehicles with varying amounts of resources. Therefore, it is vital to consider the heterogeneity of vehicles and their underutilized resources for resource management in VCC networks. We address this by clustering vehicle resources into small, medium, and large sized pools.

These pools, called Combined Resource Units (CRUs), are composed based on a predefined maximum CPU, memory and storage capacity. Resources from vehicles in the surrounding area are used to assemble these units until their capacity is maximized. Unlike approaches that introduce complexity in the management and allocation process, we introduce a relaxed method where virtual resource units are assembled proactively for fulfilling requests. Our proposed method utilizes distinct search algorithms called ELSA (Exhaustive Linear Search Algorithm), RSA (Restrictive Search Algorithm), FASA (First Available Search Algorithm) and SSA (Selective Search Algorithm), which are based on search principles in matching predefined resource cluster sizes. A resource distribution heuristic is also implemented, which attempts to impose resource cluster distribution according to the available resources supported by the network.

Following the CRU composition process, a game theoretical model is implemented

and tested for successfully allocating CRUs to user service requests. This model ensures that CRUs are assigned to users in a fair and efficient manner, maximizing the service fulfillment success rate within a minimum time frame. The game model aims at maximizing the utility of each request. The utility value is derived from the *RSS (Receiving Signal Strength)* value and the *additional resources* offered by a CRU to a service request. Based on the utility value of each CRU, it is allocated to a service request for fulfillment using the *Alpha-Beta Pruning* algorithm, which is considered as an optimization technique for the traditional minimax algorithm.

1.1 Motivation

Vehicles are equipped with a significant amount of on-board resources that remain under-utilized. These resources can be utilized towards fulfilling service requests. Moreover, the vehicles in a VC network display high mobility, which results in a dynamically changing network topology, which affects the connectivity status of the vehicles. This leads to increased time delays in processing requests. It is also a time-consuming process for service requests to be matched and fulfilled with individual vehicles from a large pool of vehicles. Therefore, addressing these problems, this work proposes a more relaxed approach of clustering resources of nearby vehicles in a virtual unit of three different predefined sizes. Distinct search algorithms are implemented for assembling these units. The different pools of resource units utilize maximum underutilized vehicular resources while also maximizing the rate of fulfillment of services. The distribution heuristic estimates the number of units that can be safely assembled in the network, and the game theory model allows for fair and efficient allocation of resource units to service requests.

1.2 Objectives

This work aims at clustering vehicular resources into virtual units, maximizing the use of underutilized vehicular resources in a VCC network for fulfilling user service requests. The following important factors are to be considered:

1. Vehicular resources need to be clustered into Combined Resource Units where a minimum amount of resources is not unavailable.

2. Such units must be organized into different pools to provide maximum success rate of service request fulfillment.

3. A heuristic method must be able to estimate the different pools of CRUs that the network can safely assemble, to avoid overloading and congestion.

4. A resource allocation strategy must efficiently and fairly be assigned to service requests.

5. The CRU composition and allocation strategies need to scale to match urban computing environments.

6. The assignment of resources must guarantee service delivery and fairness.

7. The connectivity level of vehicles must be ensured through a propagation model to avoid interruption of service requests.

1.3 Contribution

This section organizes the contributions made through this work. The first section presents the contributions of CRU composition process and the second section presents the contributions of the CRU allocation process.

1.3.1 CRU Composition

The CRU composition process is conducted to assemble virtual units of resources, that contribute towards service allocation and resource provisioning.

1. RSU-based resource discovery: The RSU triggers the CRU composition process by communicating with vehicles in the vicinity to provide their resources for clustering into virtual units.

2. CRU assembly algorithms: Distinct search algorithms called *ELSA*, *RSA*, *FASA* and *SSA* are implemented for clustering the resources into different pools of CRUs, based on a predefined CRU capacity. These algorithms are based on the principle of a bin-packing algorithm.

3. CRU distribution heuristic: A distribution heuristic is implemented to estimate the different pools of CRUs that the network can safely assemble.

The results of the performance of CRUs in the VCC network have been accepted for publication in the IEEE ISCC-2021 conference.

1.3.2 CRU Allocation

After the completion of the CRU composition process, the CRU allocation process is conducted for fulfilling multiple service requests. The contributions of this section are organized as:

- Game theory-based models and algorithms: Two decision-making algorithms, *Exhaustive MnM* and *Efficient Pruning*, are implemented for finding an ideal CRU match for service requests. These algorithms are based on the principles of the Minimax and Alpha-Beta Pruning algorithms, respectively.
- The Efficient Pruning method presents a more coherent fair solution search since it considers a more realistic utility function for the CRUs.
- Two different utility functions, one which is distance-centered and the other which is RSSI-centered, are defined for *Exhaustive MnM* and *Efficient Pruning* algorithms, respectively.

1.4 Outline

The remainder of this thesis is organized as follows:

- Chapter 2 explains the background of Vehicular Cloud Computing, its emergence in the cloud computing world, state-of-the-art architecture of VCC networks, the taxonomy of vehicular cloud computing and resource allocation and management in VCC networks.
- Chapter 3 presents relevant related works to resource management in VCC networks. These works have been categorized based on efficiency, mobility and game theory approaches.
- Chapter 4 presents the problem statement, highlighting problems that motivated us to conduct this research.
- Chapter 5 discusses the proposed approach in detail, introducing the resource clustering method of CRU, the different search algorithms for clustering resources into CRUs, and a distribution heuristic. It also discusses a game model that is used to efficiently allocate the assembled CRUs to user service requests, maximizing the service fulfillment rate.

- Chapter 6 presents the experiments and simulations conducted for studying the practicality of our proposed approach in real-world scenarios and discusses the results.
- Chapter 7 concludes the thesis and describes some future work directions.

Chapter 2

Background

The integration of Vehicular Ad Hoc Networks (VANETs) and cloud computing led to the emergence of Vehicular Clouds. VANETs work as an underlying communication system where Vehicular Clouds are assembled through vehicles. This chapter briefly describes VANETs and their contributions in the emergence of vehicular cloud computing.

2.1 VANETs

VANETs emerged as a new technology with the recent developments in the vehicular and communication technologies. VANETs are known for the integration of new generation wireless network technologies to vehicles. They provide vehicle-to-infrastructure (V2I) wireless communication which can help ensure transportation safety and communication reliability for moving vehicles in urban road environments. They also provide vehicle-to-vehicle (V2V) communication, which allows vehicles to directly communicate with each other or vehicle communication with nearby road equipment, referred to as roadside units (RSUs), forming V2I communication. VANETs are known for their ability to provide safety applications such as road and traffic conditions, as well as non-safety applications such as video streaming, data download, advertisement diffusion, allowing to make travel more comfortable for users. However, despite the several advantages offered by VANETs, challenges such as high vehicle mobility, intermittent connections and limited bandwidth impacts the reliability of the services and applications offered by VANETs. [2]

2.2 Internet of Vehicles (IoV)

The Internet of Vehicles (IoV) is an emerging paradigm and an important application of the IoT technology in the field of ITS. The IoV is responsible for carrying out effective wireless communication between vehicles and vehicles, roads, pedestrians and the internet. It also contributes towards intelligent traffic management, intelligent dynamic information services and intelligent vehicle control network integration [27]. The IoV is known to provide some services such as collision warning, traffic congestion detection, route planning, infotainment, etc., through the support of cloud computing. These services contribute to making human traffic travel convenient and efficient. Cloud computing has served as the backbone of IoV due to capabilities of providing computing resources at any time and place through the internet. Cloud computing in IoV has also served as a significant medium for data storage, data processing and data analysis. In the communication architecture of VANETs, vehicles can obtain road information only through two modes of communication: vehicle-to-vehicle and vehicle-to-infrastructure. This leads to a small and limited network. IoV, on the other hand, consists of smart vehicles that have various in-built sensors to detect the status of other vehicles, and communication devices that enable them to build connections with other vehicles and/or the Internet in the network. IoV introduced the vehicle-to-everything (V2X) communication mode, allowing vehicles to connect to anything that is able to share information about the surroundings of a vehicle [14].

However, since there has been a significant increase in the number of vehicles and mobile devices, road safety management has become a serious issue. The increase in the number of vehicles burdens the centralized cloud data center in the cloud computing paradigm for efficient storage, offloading, processing and management. In addition, since the cloud data centers are located relatively far from end users, it poses a serious problem of high processing latency, especially for latency-sensitive applications of the IoV. For instance, real-time applications such as an ambulance requiring its surrounding traffic information to reach at the rescue location on time, or a moving vehicle requiring instantaneous information about road accidents, collisions, road surface conditions, etc..

2.3 Vehicular Cloud Computing (VCC)

Initial research reveals that only communication resources of VANETs and traditional Internet Cloud Computing had been widely explored. However, some statistics reveal that over a billion vehicles around the world spent many hours in an idle state in parking lots, garages, driveways, etc., and these vehicles are equipped with sufficient computing and storage resources that could be used for fulfilling services or sharing important data among nearby vehicles. These features have contributed to making vehicles as the perfect nodes in the cloud computing environment. Keeping in mind all of these points, [22] introduced the concept of vehicular cloud, which integrates the underutilized communication, computing and storage resources of vehicles to provide services to authorized users.

In VCC, vehicles are treated as smart devices that are equipped with multiple sensors and therefore are capable of gathering and providing useful road and traffic information to requesting users. The idle resources of vehicles may help city traffic authorities in delivering information to vehicles approaching a particular traffic congested or accident highway road. This would help drivers change their route and save time. Therefore, the integration of cloud computing with vehicular networks is said to provide a wide range of vehicle-based services to the drivers and passengers, such as improvised road safety, smart traffic control, entertainment services, and contribute towards minimizing high memory bandwidth consumption, processing and communication latencies, response delays to network requests, and improvise location-awareness.

2.3.1 VCC architecture

[16] defined the architecture of vehicular networks, keeping in mind the different types of cloud scenarios in VANET. It is the first architecture defined for VCN and is divided into 3 different sub-architectures:

- Vehicular Clouds (VC): This comprises a VANET infrastructure, gateways and brokers. The VC is further divided into static clouds and dynamic clouds.
- VANET using clouds (VuC): In this architecture, VANETs make use of cloud architectures while on the move. RSUs act as mediators for vehicles to access cloud services. These RSUs are wired-connected to the clouds and provide

high-speed communication to cloud services access. Infotainment and real-time traffic information are some services provided by this architecture.

- Hybrid Clouds (Inter-vehicle clouds): This architecture is a collaboration of VC and VuC. VC plays the dual role of a provider and consumer. Some services offered by this architecture include NaaS, P2P and IaaS.

[26] proposed the VCC architecture which comprises 3 tiers:

- Inside-vehicle: This tier keeps track of the health and mood conditions of the driver by collecting useful information inside the car such as pressure and body temperature. This information is detected using different types of vehicle in-built sensors such as body sensors, smartphone sensors, environmental sensors and driver behavior recognition sensors which can detect the reflexes and intentions of the driver. This information is then stored in the cloud or is used as an input for various software applications. The information for storage in the cloud is forwarded through the on-board units in vehicles that are built with wireless communication broadband, navigation system, a map, and location of the RSUs.

- Communication: This consists of 2 components: i) V2V- If a driver is detected with faulty behavior on the road such as abruptly changing directions, over speeding while driving or some mechanical failure in the vehicle, emergency warning messages (EWMs) are forwarded to cloud storage and nearby vehicles. These messages carry with them important information such as current location, speed and direction of the vehicle and ii) V2I- This allows for the transfer of information and data between vehicles, cloud and infrastructure over wireless networks such as 3G, internet, LTE, etc.. V2I has greatly minimized accidents, delays and road traffic congestion on highways, thus enhancing road safety levels.

- Cloud: This layer consists of 3 internal layers: application, infrastructure and platform. It is responsible for computing complex and large computations in minimal time. The application layer has several real-time services and applications, such as human health and activity recognition, environmental recognition, etc. The infrastructure layer consists of cloud storage and computation.

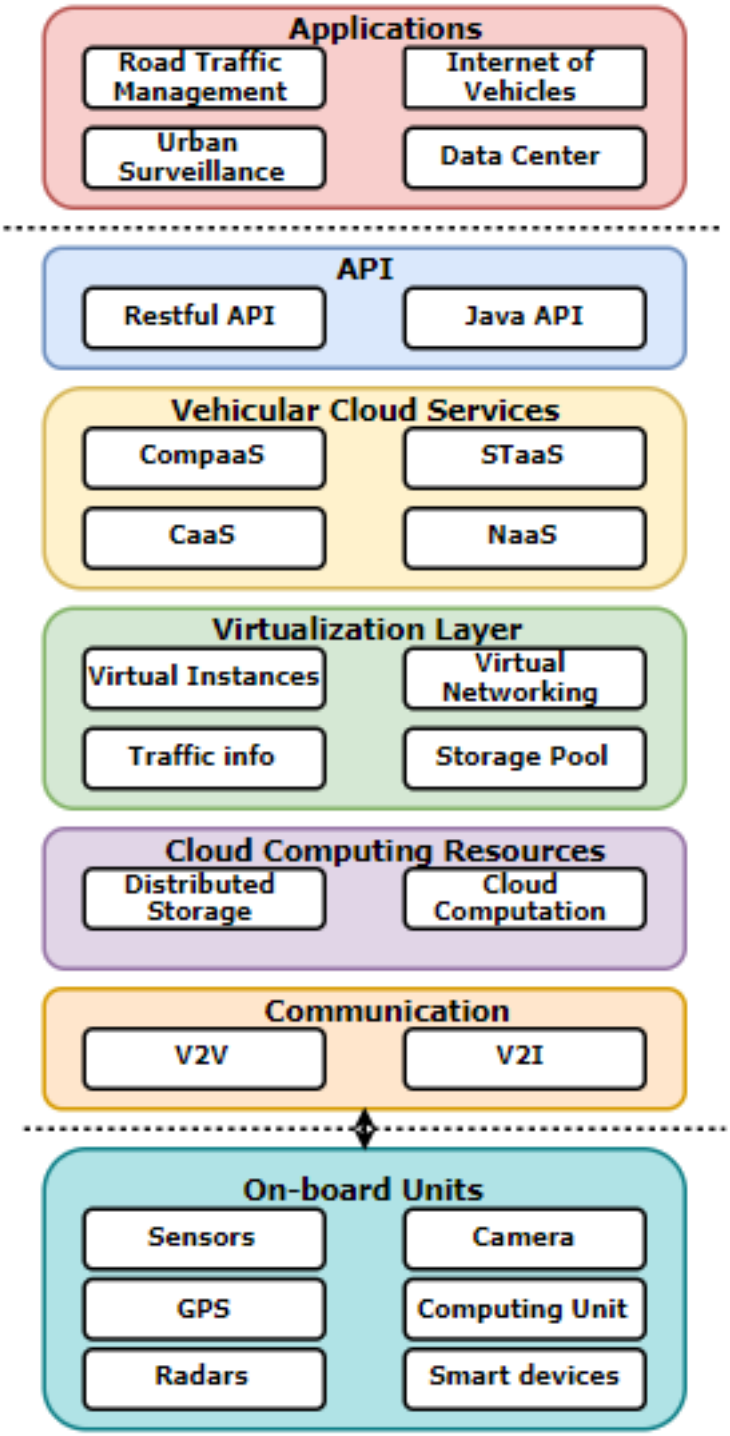

Figure 2.1: Vehicular Cloud Computing Architecture as proposed by [6]

2.3.2 Vehicular Cloud Taxonomy

The vehicular cloud taxonomy is explained based on the different services offered by the vehicular cloud: i) Network as a Service (NaaS), ii) Storage as a Service (StaaS), iii) Sensing as a Service (SaaS) and iv) Computation as a Service (CaaS).

1. Network as a Service (NaaS): This service exploits the communication capabilities of vehicles. Much research has been conducted to study the efficient utilization of communication resources in vehicles. Work such as [1] has researched on-time delivery issue of data by exploring the connectivity between WiFi enabled devices that use different routing policies. Some researchers also explored the static vehicles and their underutilized resources for utilization in the vehicular cloud environment. [15] proposed the approach of parked vehicle assistance which helps in establishing network connectivity in the vehicular cloud environment by using parked vehicles as static nodes in the system.

2. Storage as a Service (StaaS): Recent developments in storage technology has made it an inexpensive and contributing factor in the cloud computing environment. Little research has been conducted in exploring the underutilized resources to provide storage services in the vehicular cloud. The authors in [26] make use of static vehicular storage resources of parking lots to calculate communication cost, and propose a data-center system that consists of two tiers: tier 1, which has stable storage resources; and tier 2, which has unstable storage resources. These storage resources obtained are used to minimize the data access cost within the system.

3. Sensing as a Service (SaaS): Modern-day vehicles are equipped with a number of sensors that ensures safety while driving, as well as to identify the behaviour of the driver. Due to the constant mobility of vehicles, the sensing coverage is much expanded, which helps us in obtaining sensing date from geographically distributed vehicles. [11] proposes an approach in which parked vehicles in urban regions are utilized as relay nodes that will help in sensing vehicles on road that are not in the line-of-sight region, thus contributing to maximizing road and traffic safety. [32] introduces a system called POVA that helps in sensing traffic lights in large-scale urban scenarios. This system provides advantages such as low deployment cost and wide scale coverage. [28] introduces an incentive mechanism for crowd-sensing that allows smartphone users to be recruited for providing sensing services.

4. Computation as a Service (CaaS): Most modern-day vehicles are built with something called vehicle equipment (VE) which is like a small computer with a networking interface. These VEs enable vehicles to communicate with other vehicles, RSUs and sensors to exchange information and enhance road safety. These VEs also enable vehicles to be treated as computation resource providers of static vehicles that are parked. These resources contribute to building a date center with significant computation power.

The taxonomy also includes several applications, such as traffic and road safety management, urban surveillance, emergency management, IoV (Internet of Vehicles) and management of traffic lights.

2.3.3 Game Theory

Game theory was founded by John von Neuman as an optimal solution to many mathematical problems by providing possible outcomes which involves several parties, called players [13]. It involves a strategic interaction between players, following a certain set of rules, to yield possible outcomes. Game theory is different from other mathematical optimization models in ways that it allows players involved to have independent choices, with each of their choices having a possibility to influence the other. This leads to players developing similar interests in each others choices which can further be used to collaborate their assets or abuse each other's assets for personal benefits. Game theory is primarily used to achieve fairness and efficiency in certain situations.

Game Theory is defined as a tuple: $G = \{P, S, U\}$. P denotes a set of players or decision-makers that are actively involved in the game and contribute towards achieving an outcome. S denotes a set containing one or more strategies adopted by a player. Such strategies determine the action taken by a player at any time of the game. U denotes the utility function which is a reward that each player receives at the end of the game. The objective of the game theory tuple is to ensure that the end outcome of each player is favorable to all players in the game. It depicts how the decision of one player may affect the decisions of other players.

2.3.4 Resource allocation and management in VCC

Several studies have revealed that VCC networks enable dynamic allocation of resources which ensures guaranteed delivery of reliable services to users. Resource

management and provisioning in VCC networks are responsible for identifying and allocating sufficient resources to the user's requests, based on their QoS requirements. Resource management also deals with identifying the lack of availability of resources required to fulfill user service requirements. The work presented in [9] has mentioned the importance of resource management in terms of minimized execution time, cost and consumption of energy. However, the high mobility of vehicles and the frequently changing topology of the vehicular cloud environment introduces challenges such as instability of resources, which also complicates resource management and allocation in VCC [10]. Therefore, it is important to address the resource management challenges in the VCC network. A detailed view of resource allocation and management in VCC networks is presented in the next chapter, where we will discuss and compare the works of several authors.

Chapter 3

Related Works

Resource allocation and management in VCC networks have been significantly investigated in various works based on three factors: i) Efficiency, ii) Mobility and iii) Game Theory

3.1 Efficiency

Boukerche *et al.* [5] acknowledged the importance of resource management in VCC networks, considering various aspects such as: i) efficiency - which means that the resource allocation strategy must optimize the use of resources in a way that ensures resources have been consumed in their entirety, ii) QoS - all allocated resources must be sufficient to meet the QoS requirements of users, iii) fairness - all resources must be fairly allocated to user service requests.

Yu *et al.* [30] presented a 3-tier vehicular cloud architecture which is constituted by i) vehicular cloud, ii) roadside cloud and iii) central cloud. Their work focuses on collaborating the redundant physical resources in the intelligent transportation system network. Due to the resource-intensive characteristics of the architecture, resource management is considered a must. To address this, [4] proposes a game-theory model to allow the maximization of cloud resources allocation. Cloud and vehicle resources are represented as Virtual Machines (VMs). These VMs are present inside a cloudlet to obtain maximum resources. These VMs are then allocated to user requests based on the user request requirements. In order to achieve fair allocation of computing and storage resources, the authors set up 2 virtual resource counters, one for keeping track of the total computing resources and the other to keep track of storage resources. The threshold of the VRC marks the indication that the VM

can no further request for resources which means that all VMs in the network are allocated equal number of computing and storage resources.

Lin *et al.* [17] addressed resource allocation in VCC networks by proposing an SMDP model. The work constitutes integrating the computing resources of the vehicles and RSUs in the network to form resource units (RUs). It presents a 2-tier architecture which consists of a vehicular cloud composed of RUs and remote cloud which consists of powerful computing resources. User service requests are processed either through the VC or RC. Since this model considers heterogeneous vehicles, resource allocation and management in such networks in considered a complex model. Such a model contributed towards minimized consuming power and time for resource management in VCC.

Meneguette *et al.* [19] proposed a peer-to-peer protocol to address the resource management problem in the vehicular mobile cloud. The protocol is the first of its kind, that does not consider external infrastructure such as RSUs, and primarily depends on vehicles collaboration to provide resources for service execution. The objective of their model is to ensure maximum availability of resources as well to ensure that the resource utilization time by the vehicles is highest. To address this, they propose an efficient solution called SMARt (Search and Management Resource Protocol) that will help in search and management of resources in the vehicular cloud environment without depending on RSUs or any other external units for resources. To achieve this, it is important for vehicles to be able to collaborate with other vehicles to ensure management and sharing of resources. This protocol is based on the famous Gnutella peer-to-peer network that is composed of 5 basic concepts: i) ping message which contains information such as the location and request requirements of the requesting vehicle and the identification and location of controller and gateway; ii) pong message which contains the same fields in addition to the location and id of the node sending the ping message, status of the requested service, gateway id and the controller to which the sending node is connected; iii) query message which consists of a location that is similar to that of the requester, resources required and id and location of gateway and controller; and iv) query_hit message which presents the resource request confirmation as well as the location and id of the gateway and controller it is connected to. Through the peer-to-peer protocol, this approach implements multi-hop communication between vehicles to seek resources and fulfill services. Simulation results of this work showed improved results in terms of low search time

for resources at one hop and multi-hop from the requester. This approach also guarantees the availability of significant number of resources and low overhead, thereby increasing overall system performance.

Cordeshi *et al.* [8] designed a resource management controller that works well in a distributed VCC network. This controllers allows the exploitation of cognitive radio technology and the data fusion of soft input/soft output in the VCC environment. The primary goal of this model is to allow the vehicle smartphone devices that are computation and storage resource-constrained, to access V2I provided WiFi connections to perform traffic offloading. This offloading is done into multiple roadside units, also called cloudlets. This controller efficiently manages the traffic flows the VC, making the system more distributed and scalable. In order to support the cognitive radio based vehicular access to the VC, the authors proposed a unique protocol called intra-cluster access protocol. The authors divided this protocol into seven phases: channel estimation, channel propagation, channel sensing, data fusion, client scheduling, client upload, and acknowledgement phase.

Yu *et al.* [29] addressed the problem of resource management and sharing problem for bandwidth and computing resources to support mobile applications in VCC network. In their work, all service provider vehicles in the vehicular cloud environment collaborate with each other to configure coalitions. These coalitions will help the service provider vehicles to efficiently share their idle resources with each other. The authors propose a coalition two-sided game theory model to enable the collaboration of cloud service providers for the sharing of their idle resources. The authors divide this model into two phases: i) the cloud service provider explores the revenue and identifies if it is capable of working solo or must be integrated with the cloud environment; and ii) the service provider performs two functions of either leasing resources to other service providers in range or renting resources from them. This means that the service provider vehicles act as providers and requesters simultaneously. There are several options that the authors consider in this approach: i) a service provider can participate in a coalition to evaluate whether it could have a better utility; ii) a service provider at any time can change coalitions they participate in to improve their utility; iii) a service provider may work alone to improve their utility. To make these options available to the service provider, the authors introduced the Pareto optimality that would help the service providers in maximizing their utility or to ensure the avoidance of reduced optimality. The process of Pareto optimality is gradually

made stronger.

Another approach to resource management in VCC networks was introduced by Zheng *et al.* [31] who proposed an optimal computational resource allocation scheme which helps in significantly enhancing the total long-term expected rewards in the VCC system. This reward is calculated by taking two important factors into account: income and cost of the VCC system, as well as the variability of the available resources. This is one from a few works that have heterogeneous vehicles involved. When a user vehicle sends a service request message, it is the responsibility of the VCC network to make an immediate decision whether to process this request in locally in the VC or forward it to the RC. In order to deal with heterogeneous varying resources of vehicles, the authors slice the resources into virtual units such that each vehicle is accommodated with one such virtual unit. The reward that is calculated based on cost and income depends on power consumption and time of processing. The resource management and allocation problem is further solved as a semi-Markov decision process (SMDP). The SMDP model depends on 4 factors: state space, action space, reward model and transition probability distribution model of the VCC system. For obtaining optimal results, the authors use an iteration algorithm that contributes in efficiently enhancing the long-term expected total reward of the system.

3.2 Mobility

Arkian *et al.* [4] addressed the problem of resource management in VCC networks by proposing a vehicular cloud architecture that consists of vehicle clusters that will act as resource providers to service requests. The clustering technique is defined as the grouping of vehicular nodes to provide resources to service requests. The clustering technique has contributed greatly towards maximizing vehicular network performance. The clustering technique serves as an ideal solution to overcome challenges of high vehicle mobility and frequent topology changes in dynamic vehicular cloud networks. In the proposed technique, a cluster head (CH) is selected, which in fact acts as the cloud controller, and manages the creation, maintenance and destroying of a vehicular cloud. The CH assigns resources to vehicles in need and is also responsible for maintaining cloud resources. During a service request interval, if any vehicle that is serving the request moves out of range, it is the responsibility of the CH to assign a new vehicle to the service request. The cluster of vehicles is assembled based on some vehicle characteristics such as the direction, speed and current location. The main characteristic for cluster formation is the distance of vehicles from

each other which means that vehicles in close vicinity of each other are grouped under one cluster. Since the fuzzy logic does not involve complex mathematical models, it is opted as an ideal technique for the process of selection of CH. The fuzzy logic is used to calculate the *FitFactor* which is used in the selection process of the CH. Three metrics are considered for the selection of a CH: vehicle average speed, degree of neighborhood and quality of RSU link, in order respectively. In order for the CH to select the most ideal helper vehicle (the vehicle with the most underutilized resources), in a dynamic environment, an MDP-based reinforcement learning technique is applied. The proposed architecture for resource management is called a COHORT. This model proved to be efficient in terms of increased service completion rate. However, it also faces some challenges such as the CH selection process is time consuming and thereby minimizes network performance and increases system overhead.

Mustafa *et al.* [21] addressed the challenge of high mobility of vehicles which has an impact on resource allocation and management in the VCC environment. For the first time, their work introduced the concept of mobility prediction of the vehicles as a novel solution to minimize the impacts of resource mobility on the performance of the VC environment. Unlike several approaches, this approach also deals with VM migration for efficient resource management. Groups of vehicles present within the RSU coverage form cloudlets. For continuity of service, the VM assigned to a particular user makes continuous shifts between different RSUs in order to reach the destined cloudlet. The authors in [21] propose a mobility prediction model that is used to ensure that the resource management process is conducted in a reliable manner, without the overuse of resources which could be led by redundancy as well as to minimize the performance overhead due to service executions being interrupted by VM migrations. According to the proposed model, the lifetime of all available resources, which is defined as the time through which a resource is available for service execution, is predicted, which implies that a resource with maximum predicted lifetime is chosen. The model also sets up a predefined VM migration is scheduled to ensure accomplishment of service execution. The simulation environment set up by [21] is based on Nagel Shreckenberg CA model. Despite its simplicity, this model efficiently captures the real-time traffic details and, as a result, provides a two-dimensional matrix which defines the traffic characteristics through 2 metrics: definite road length and simulation time. The lifetime of available resources is predicted using a hybrid of linear regression (LR) supported by the artificial neural network (ANN) model. Simulation results of this model revealed successful performance of the vehicular cloud,

without the overuse of vehicular cloud resources as well as decreased overhead due to the minimal VM migrations in the network.

The approach in [7] is slightly different and one of its kind from existing resource management techniques in VCC. Keeping in mind the high mobility of vehicles, Brik *et al.* [7] introduced the concept of public bus harnesses which provides 2 important advantages: stability - in terms of time (since buses have fixed schedules of operating) and space - which means having a fixed line in an urban area. These 2 factors enable buses to act as cloud depositories, allowing provider vehicles to register their services at these directories. The authors in [9] introduce a new proactive protocol called DCCS-VC (Discovering and Consuming Cloud Services in Vehicular Clouds). In this protocol, vehicles present in the MVC network rent out their various resources to nearby vehicles. The architecture of this protocol comprises of 3 main entities: i) provider vehicles which identify the most appropriate cloud directory in its vicinity and register their services with it; ii) public buses to employ as cloud directories. The public buses are employed to allow provider vehicles to store their services and allow consumers to select from a wide range of services; and iii) consumer vehicles that discover services before making a request. This protocol also enables consumer vehicles to select from a wide range of services, in the case where they have discovered several possible service providers for their requested service. The bus selection process for cloud directories is based on Simple Additive Weighing (SAW) technique, based on quality criteria. The provider vehicles are localized so that consumer vehicles can consume requested services directly from the providers. To enable this, the authors propose a localization technique called the Grid-based Tracking Cell technique (GTC). To select the most appropriate provider vehicle, the consumers propose a fuzzy approach, accommodating 2 important factors: consumer preferences and service constraints (QoS). This has enabled in achieving efficiency in the ranking of provider vehicles, in terms of delay. Simulation results revealed that the proposed protocol significantly enhances service directory and consumption delays.

Sibai *et al.* [12] proposed a connectivity-aware service provision approach for addressing the challenge of resource allocation and management in VCC networks. In this approach, a vehicle is requested as a provider for providing the requested service. This service provider vehicle is selected based parameters such as mobility of vehicles and availability of the requested service. The authors propose a 3-tier architecture in which: i) level 1 defines the type of available services (data storage, sensor data, com-

puting, etc.); ii) defines the communication involved (V2V and V2I); and iii) vehicle sensors for data aggregation such as GPS, camera, smartphones, etc. The author also defines 2 types of vehicles: i) requester vehicle which is responsible for requesting for one or more available services in the cloud environment; ii) service provider vehicle that is responsible for assigning services to the requester vehicle. Since the provider vehicle initiates the servicing of the VC, it is also called the leader vehicle. The leader vehicle has various responsibilities like: i) exploring vehicles that are willing to provide services as ideal candidates; ii) initiating services of the cloud; iii) maintaining the cloud during an ongoing service; and iv) destruction of cloud after service fulfillment. The service provider vehicle may be a mobile or stationary vehicle in the cloud environment. A spatiotemporal algorithm is implemented to calculate communication duration and delimited communication interval. These values are then used in the process of searching and allocation of resources in the VC environment. This process is achieved by the requester sending a request message and implementing a mechanism of re-transmission that aims at controlling the broadcast storm, thus minimizing system overhead.

3.3 Game Theory

Tao *et al.* [24] proposed a non-cooperative resource allocation game whose working is based on Gauss-Seidel iteration method. The primary aim of this method is the reduction of Nash Equilibrium Point (NEP) calculation time. The authors proposed the concept of roadside cloud which is formed by merging VANETs and cloud computing, for efficiently minimizing the communication time between the RSU and vehicles. The roadside cloud enables users and vehicles for data access and downloading services. This access to the roadside cloud is provided to the users through the RSU or dedicated servers contained inside the RSU. The roadside cloud also consists of roadside cloudlets that offers services to bypassing vehicles. It is assumed that vehicles participating in the data access competition are selfish in terms of trying to achieve the best desired network performance. In urban scenarios, vehicles access data through RSUs. For this reason, the authors propose a game theory approach to study the transmission of vehicle nodes. This game theory helps vehicles in minimizing their cost and maximizing their utility. The game theory results helps in achieving efficient cloud resource allocation results among vehicle nodes through the G-S iteration method. The G-S iteration method helps in addressing the problem of

resource allocation convergence in VCN, by calculating the optimal utility and flow rates of vehicular nodes.

Mohanty *et al.* [20] proposed a hierarchical interconnected vehicular cloud architecture for mobile vehicles. This cloud architecture collaborates the physical resources of mobile vehicles present in the ITS infrastructure to form a ubiquitous cloud environment for vehicular networks. This cloud environment combines the interconnected data center, infrastructure units and the physical resources of mobile vehicles that results in a significant and powerful resource cloud for vehicles. The 3-tier architecture which comprises of a vehicular cloud, roadside cloud and central cloud helps in organizing together, the abundant available resources of these clouds. Even though the central cloud is comprised of large resources, it also produces a large end-to-end communication delay. While the resources contained in the roadside and vehicular cloud are limited, but they provide good communication speed and quality. Due to these reasons, this architecture enables vehicles to select their particular cloud and its services with ease, thus making it flexible, application-friendly and compatible with heterogeneous WSN and CR technologies. The authors in [20] also propose a game theoretical approach as an effective resource allocation algorithm to accommodate the resource-demanding nature of the vehicular and roadside cloud. Resources present in the roadside and vehicular cloud are represented in the form of VMs. VM resource allocation in the roadside cloud must meet with 3 important factors: efficiency, QoS and fairness.

The authors in [3] proposed a new architecture called Smart Vehicle as a Service (SVaaS) to provide services to vehicles prior to their arrival in smart city environment. This architecture was proposed to expand the provisioning of vehicular services sharing and storing digital data, monitoring and sensing surrounding environment such as for traffic and accident information and on-demand mobile services, and to address the challenges of varying ownerships, costs, high demand levels, service requester and different rewards. This approach relies on the mechanism of predicting a vehicle's future location. This would help in achieving a service selection mechanism that is based on Quality of Experience (QoE), which in turn would help in selecting services that are needed by the vehicle before the actual arrival of the vehicle. The authors collaborate QoE with a TTP cloud entity to establish a game model that acts as a mediator between vehicle nodes and service providers. TTPs are established organizations and provide services to vehicle users. They provide an abstraction layer between

Table 3.1: Summary of works in resource management in VCC.

Work	Connectivity	Mobility	V2I	Probabilistic Model used
[7]		✓	✓	Own Method
[30]	✓	✓		Game Theory and Nash Equilibrium
[17]			✓	Semi Markov Decision Process
[19]		✓	✓	Peer-to-Peer protocol (Gnutella model)
[8]		✓	✓	Own Method (Intracluster Access Protocol)
[29]		✓	✓	Coalition Game Model and Pareto Optimality
[31]		✓	✓	Infinite Horizon SMDP model
[4]	✓			Fuzzy Logic and Q-Learning
[21]		✓		Artificial Neural Network
[12]	✓			Spatio-Temporal Similarity
[24]	✓	✓		Gauss-Seidel Iteration Method (GT)
[20]		✓	✓	Game Theory and VM Migration
[3]		✓		QoE Game Model
Proposed	✓	✓	✓	Game Theory and CRU

service users and providers that contributes significantly in the resource management and allocation in a smart city environment. In this approach, vehicles play the role of service requesters and providers. The location prediction mechanism is based on Dempster-Shafer theory and relies on several factors such as: user schedule, tasks, service interests and location history of the vehicle to predict future locations. The TTP cloud entity is composed of a service mediator module that incorporates the location prediction mechanism and is responsible for service discovery and selection. This module relies on game theoretical model which calculates a rating measure for each service provider based on an overall satisfaction of service from the participant. Game theory is used to ensure fair service distribution among providers or requester based on some factors such as resources, services available, requesters that are currently operating, service charges, game participants, game events and reputation values of QoE for each service.

Table 3.1 summarizes and presents a comparison of the resource allocation and management problem by different authors, based on factors such as connectivity, mobility and communication. It also displays how our work differentiates from all previous works. These related works deal with resource management. However, their performance and feasibility is impacted by several issues. The heterogeneous nature of vehicles and service requests is not addressed by most works. VM migration is not considered an ideal option when working with heterogeneous vehicles since it is difficult to deal with continuously changing data. The service drop rates would be higher in such a scenario when a VM migrates from vehicle A to B due to the varying

resources of A and B. Therefore, the challenges presented by these works motivated our work to propose the CRU composition approach, for conducting efficient resource management in VCC networks.

Chapter 4

Problem Statement

Our primary challenge to address in this work is to maximize the use of underutilized resources of vehicles towards fulfilling service requests in a urban dynamic VCC environment. The vehicles in the VCC environment can communicate with each other through V2V (vehicle-to-vehicle) or through V2I (vehicle-to-infrastructure), where RSUs provide coverage. When dealing with heterogeneous vehicles, the challenge lies in their varying resources, often resulting in under-utilization/wastage of resources. This adversely affects the success rate of service request fulfillment.

Consider an urban dynamic VCC network environment(Figure 4.1), consisting of a finite number of heterogeneous vehicles, defined as:

$$V = \{V_1, V_2, V_3, ..., V_n\}.$$

This VCC scenario resembles a real-world scenario to better understand the practicality of our approach in real-world. The vehicles in our scenario possess high mobility and are able to connect and communicate with each other to exchange information, data and resources. Due to the high mobility of vehicles, their connectivity in the network is abruptly affected when they move out of range of each other. This also affects an ongoing service that is being processed. To support connectivity and avoid disruption of services, the network also consists of a set of finite number of RSUs, represented as:

$$RSU = \{rsu_1, rsu_2, rsu_3, ..., rsu_k\}$$

Let us consider R to be our set of resource requests. This set will consist of p number of resource requests and it can be written as:

$$R = \{R_1, R_2, ...R_p\}$$

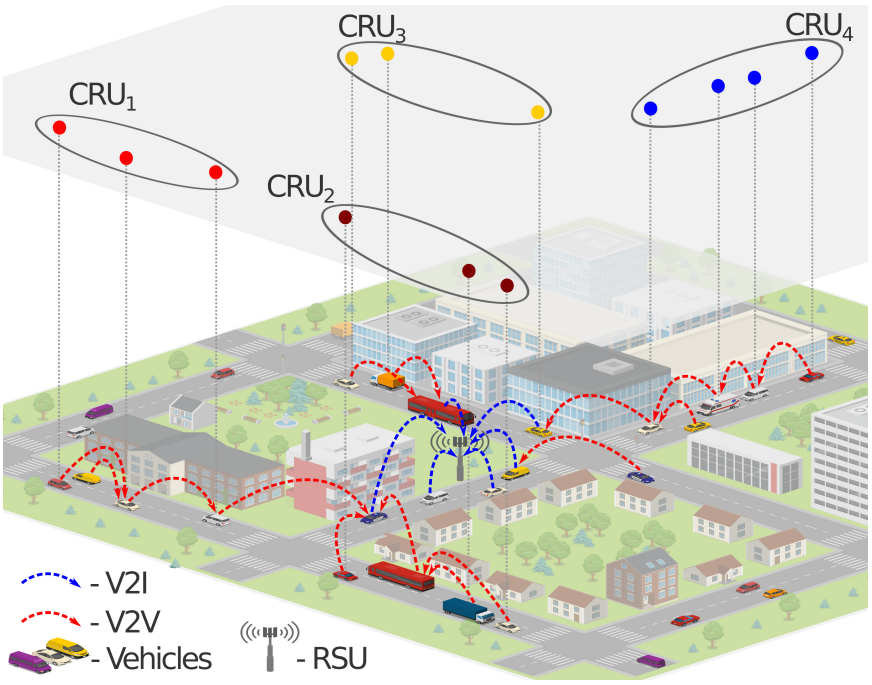

Figure 4.1: CRU composition in a VCC Scenario.

These vehicles are efficiently in-built with a set of resources that can be represented as::

$$r = \{r_1, r_2, r_3, ..., r_n\}$$

Each vehicle is equipped with sufficient CPU (MHz), Memory (MB) and Storage (MB) resources. These resources can be classified as:

$$r_n = \{CPU_i, Mem_i, Stor_i\}$$

There are several mobile smart users that are accessing the internet for download and upload of data, data sharing, etc. through mobile devices. However, several problems may be encountered:

1. The heterogeneous nature of vehicles may result in the under-utilization of vehicular resources during the resource-service matching process, thus resulting in minimal service fulfillment rate.

2. Mobile devices are resource constraint devices. This means that they are not sufficiently equipped with resources to always fulfill large user service requests such as downloading large media files, sharing data with other users, etc. Using mobile devices' resources for fulfilling service requests is considered a difficult and time-consuming process.

3. Since the connectivity between vehicles remains a rarely researched topic, the currently researched resource management algorithms do not produce efficient results due to the short communication time between vehicles. In such cases, it is difficult to meet with the QoS requirements of the users.

4. The high mobility of vehicles in Vehicular Clouds results in a dynamically changing network topology, which constantly affects the connectivity status of the vehicles. If a vehicle loses its connection with the centralized RSU, it is no longer considered valid for participating in resource sharing, which would lead to increased time delays in processing user requests.

5. Most works in the past have not considered heterogeneity of user requests.

6. It is time-consuming for user service requests to be matched and fulfilled with an appropriate vehicle from a pool of vehicles in the network.

Addressing the above stated problems, our approach takes advantage of the heterogeneous vehicles present in the network and utilize their available underutilized resources to form Combined Resource Units (CRUs) that are used to fulfill multiple user service requests simultaneously. Unlike previous works that have considered stochastic models such as SMDP, MDP, etc. for resource clustering techniques, our approach uses distinguishing search algorithms for conducting the CRU composition process. Following the CRU composition process, we employ the CRUs for fulfilling user service requests. A game theoretic model is used for conducting a fair and efficient allocation of the assembled CRUs to multiple user requests, simultaneously. The fairness and efficiency of the game theoretic model is achieved through the maximization of the utility value/payoff of each CRU.

Chapter 5

Combined Resource Unit

This work proposes the concept of collaborating the physical resources of vehicles in a virtual environment, forming a Combined Resource Unit (CRU). CRUs, integrated with the concept of a game theory model will contribute to the resource management problem in VCC networks, ensuring: i) fair allocation of CRUs to all service requests; ii) maintaining the QoS requirements of users service requests in a mobile vehicular environment; iii) efficient consumption of CRU resources allocated to the service requests; and iv) efficient utilization of the underutilized vehicular resources.

CRU is defined as the amalgamation and virtual combination of vehicles' resources into a single unit that can be used for efficient resource allocation in the VC environment. Unlike all past approaches related to resource management in VCC using virtual machine migration (VMM), the CRU approach is unique for several reasons:

- In previous works, each vehicle in the VC has its own virtual machine (VM) of resources which are used in the management and allocation process. However, this work clusters the resources of several vehicles into a single virtual unit and after resource allocation, these vehicles are assigned to the request for fulfillment.
- The composition of CRUs make the request allocation process easy and efficient by designing and assembling three different sizes of these units. This helps in reducing the search time for requests to be assigned and also helps in preserving more resources for future requests.
- The CRUs are assembled prior to service requests. This means that if there are multiple requests made by multiple users, the rate of fulfillment is much higher.

- The three different sizes of the CRUs ensures that all requests are allocated their desired number of resources for fulfillment while also ensuring that resources of these units are not wasted. This means that services can be matched with the CRUs during allocation. The overhead in the system is minimized since there are always enough CRUs available to be assigned to service requests and therefore, the load on the system is also reduced, making the system more scalable and economic.

Table 5.1: Notations

Notation	**Meaning**
VCN	Vehicular Cloud Network
V	All vehicles in the VCC network
n	number of vehicles in the VCC network
CRU	Combined Resource Unit
RM	Resource Manager (pertaining to characteristics of RSU
R	set of resource requests
RSU	Roadside Unit
V2I	Vehicle-to-Infrastructure communication
V2V	Vehicle-to-vehicle communication

5.1 VC Scenario

Consider an urban dynamic vehicular cloud network consisting of a finite number of heterogeneous vehicles. These vehicles possess high speed and are constantly moving around in the network. Two RSUs, which are static in nature, are deployed in the VC network, and are responsible for providing internet coverage to vehicles wirelessly through V2I (Vehicle-to-Infrastructure). They also help in maintaining connectivity in the network and exchanging information with the vehicles. From the deployed set of RSUs, rsu_m is considered as a resource manager since it aggregates all data and information in the region, besides connecting to neighbouring (in range) vehicles. Thus, rsu_m maintains information of vehicle's resources in our problem scenario. Also, rsu_m is considered as the primary RSU in the network since it is responsible for conducting resource allocation to incoming user requests.

There may be a few problems that can be encountered in the process of CRU composition and a few assumptions that must be made:

1. **Problem**: All vehicles in the VCC network have the same resources and features. This means that the three different types of CRUs that will be composed will have the same capacity and this would make the allocation process complex, leading to resource wastage.
   Assumption: We assume that, our VCC scenario consists of heterogeneous vehicles which have varying resources and features.

2. **Problem**: Failure in network connectivity can lead to disruption of ongoing services in the network.
   Assumption: The RSUs in the network have a lifetime connectivity and are hard wired connected to each other. This would lead to minimal service disruption.

3. **Problem**: There can be multiple requests requesting for resources at the same time. This would make it difficult in identifying which request must be processed first.
   Assumption: Requests can be queued based on their arrival time and processed accordingly.

5.2 Composition of Combined Resource Unit

The CRU composition approach is divided into two objectives: i) designing an algorithm suitable for CRU assembly and; ii) developing a CRU distribution heuristic to calculate the number of different CRU types that our network is able to assemble.

5.2.1 CRU Composition Algorithm

The different CRU composition algorithms are based on the principle of bin-packing problem. This optimization problem fits our CRU composition problem by allowing resources of vehicles to be packed/clustered into a finite number of predefined-capacity-CRUs. The goal of using bin-packing problem is to ensure efficient number of CRUs are composed in the network.

The bin-packing problem is classified under the non-deterministic polynomial time (NP) problems. These problems are solvable and can be verified in polynomial time. The bin-packing problem is further classified as a NP-hard problem, which is a subset of the NP problem. NP-hard problems are decision problems that cannot be verified in polynomial time. In instances such as ours, where a large number of vehicles are

involved, it becomes difficult to cluster significant number of resources into limited number of CRUs. However, the bin-packing problem allows the efficient clustering of resources in a dynamic scenario like ours, despite its worse-case NP-hard property, through the use of several approximation algorithms.

The various approximation algorithms that exist are divided based on: online heuristics and offline heuristics. Online heuristics include algorithms such as next-fit, first-fit, best-fit, worst-fit, almost worst-fit refined harmonic, and refined first-fit algorithms. Offline heuristics include algorithms such as first-fit decreasing, next-fit decreasing and modified first-fit decreasing. However, in this work, we only consider the online heuristic algorithms since the time complexity of the offline heuristic algorithms is significantly higher than the online heuristic algorithms.

It is important to note that the approximation ratio of these algorithms must not be less than 1.5, which is the approximation ratio of the bin-packing problem. In this work, we consider the next-fit, best-fit and first-fit approximation algorithms. The first-fit and best-fit algorithms are termed as any-fit algorithms that have an approximation ratio of 1.7. The next-fit algorithm presents an approximation ratio of 2, while the almost worst-fit algorithm presents a ratio of 1.7, which is better. However, in terms of time complexity, the next-fit algorithm performs better than the almost worst-fit algorithm. Additionally, in the almost worst-fit algorithm, vehicles are made to cluster into the second most empty CRU. It is difficult and time-consuming to perform such a complex search when large number of vehicles are involved.

The goal is to select an algorithm that makes use of maximum resources of vehicles, leaving behind little to no underutilized resources. This particular algorithm must also assemble CRUs using least number of vehicles while ensuring that a CRU's maximum capacity has been met. For all algorithms, we assume a scenario in which the RSU periodically probes vehicles that are within its communication range. The RSU maintains an updated table of the resource information of all vehicles, that can be used for the CRU composition process. The RSU is also assumed to maintain a CRU list, that is used for storing CRUs that are composed using different algorithms. When CRU composition begins, each algorithm calculates the total number of different-sized CRUs that it is required to assemble, by calling the *Distribution Heuristic* method. Once the array of CRU types have been established, the different CRU composition algorithms can be used.

- ***Exhaustive Linear Search Algorithm***: The Exhaustive Linear Search Algorithm, also called *ELSA*, is based on a greedy approach for composing CRUs. This method works by clustering vehicles in a CRU based on a *first-come-first-*

served approach, as described in Algorithm 1. This means that when a service request arrives at the RSU, it is assigned the first available CRU for service processing. This method is expected to generate flawed results since this method composes a CRU until it exceeds its predefined capacity. It does this so that it does not result in composing under-assembled CRUs. This could potentially serve as an advantage and disadvantage to the VCC system in terms of having minimal under-assembled CRUs and maximum over-assembled CRUs, respectively.

ALGORITHM 1: CRU Composition - Exhaustive Linear Search

```
   Data: V|v_{i_res} = {CPU, memory, storage}
   Result: CRU = {cru_1, cru_2, ..., cru_n}
 1 CRU = ∅;
 2 CRU_dist = distribution_heuristic(V, α, β, γ);
 3 foreach v ∈ V do
 4     foreach cru ∈ CRU do
 5         if v_i ∉ cru_j then
 6             cru = new(CRU, CRU_dist);
 7             if v_{i_res} ≤ cru_{j_Cap} then
 8                 cru_j ∪ v_i;
 9                 break;
10             end
11         end
12     end
13 end
```

- ***Restrictive Search Algorithm***: The Restrictive Search Algorithm, also called *RSA* is based on the principle of the next-fit bin-packing method. According to this algorithm, vehicle resources are tried to be clustered in the current CRU. If the vehicle fits, it is placed in the CRU. Otherwise, a new CRU is created to accommodate the current vehicle. The *RSA* is expected to be flawed due to several reasons: This algorithm does not allow new vehicles to be clustered into CRUs that previous vehicles did not fit in, thus leaving significant number of unfinished CRUs and under-utilized resources in the system.

ALGORITHM 2: CRU Composition - Restrictive Search

Data: $V|v_{i_{res}} = \{\text{CPU, memory, storage}\}$
Result: CRU $= \{cru_1, cru_2, ..., cru_n\}$

```
1  CRU = ∅;
2  CRU_dist = distribution_heuristic(V, α, β, γ);
3  foreach v ∈ V do
4      foreach cru ∈ CRU do
5          if v_i ∉ cru_j then
6              cru = new(CRU, CRU_dist);
7              if v_ires ≤ cru_currentCap then
8                  cru_current ∪ v_i;
9                  break;
10             else
11                 cru = new(CRU, CRU_dist);
12                 cru_j ∪ v_i;
13             end
14         end
15     end
16 end
```

- ***First Available Search Algorithm***: The First-Available Search Algorithm, also called *FASA*, is based on the principle of first-fit bin-packing method. This algorithm works by iterating through a finite number of currently existing/previously composed CRUs until the first CRU in which current vehicle resources can be clustered is found. If no CRU that can accommodate a vehicle is found, a new CRU is created. Following the distribution heuristic, this algorithm begins the CRU composition process with composing small size CRUs first, followed by medium and large size CRU. The number of vehicles used in packing a CRU using this algorithm is intended to be minimal compared to the other methods. This algorithm also aims at evaluating the first previously created CRU for clustering resources, resulting in the composition of fewer CRUs and following the distribution heuristic accurately. However, *FASA* can fail to match efficient resource allocation requirements due to the fact that this algorithm does not consider alternative, more efficient assignment of resources in other CRUs, which could result in significant number of under-assembled CRUs.

ALGORITHM 3: CRU Composition - First-Available Search

Data: $V|v_{i_{res}} = \{\text{CPU, memory, storage}\}$
Result: CRU $= \{cru_1, cru_2, ..., cru_n\}$

```
1  CRU = ∅;
2  CRU_dist = distribution_heuristic(V, α, β, γ);
3  foreach v ∈ V do
4      foreach cru ∈ CRU do
5          if v_i ∉ cru_j then
6              cru = new(CRU, CRU_dist);
7              if v_ires ≤ cru_jcap then
8                  cru_j ∪ v_i;
9                  cru_jcap = cru_jcap − v_ires;
10                 break;
11             end
12         end
13     end
14     if v_ires ≠ CRU then
15         cru = new(CRU, CRU_dist);
16         cru_j ∪ v_i;
17     end
18 end
```

- ***Selective Search Algorithm***: The Selective Search Algorithm, also called, *SSA*, is based on the principle of best-fit bin-packing method. This algorithm works by iterating through the CRU list and accommodating a vehicle in the tightest-spot-available CRU. This means that a vehicle is assigned/clustered to a CRU until it is almost full so that there is the least wastage of CRU space. The *SSA* works by assembling large CRUs first, clustering vehicles with large resource capacity first. This serves as a benefit to the performance of the method since large CRUs can prevent accommodating smaller resource capacity vehicles, minimizing the total number of vehicles assigned to a CRU. *SSA* is expected to provide better results in terms of clustering CRUs to their maximum predefined capacity by clustering minimal number of vehicles, maximizing the rate of service request fulfillment. This would allow the algorithm to build a more stable VCC network by accurately matching the CRU distribution heuristic results. *RSA* and *FASA* oversimplify the search process by minimizing the attempts to match available resources with a set of CRUs under construction.

However, their search does not prime optimal assembly, leaving the algorithms to perform under-expected, in comparison to *SSA*.

ALGORITHM 4: CRU Composition - Selective Search

Data: $V|v_{i_{res}} = \{\text{CPU, memory, storage}\}$
Result: CRU $= \{cru_1, cru_2, ..., cru_n\}$

```
1  CRU = ∅;
2  CRU_dist = distribution_heuristic(V, α, β, γ);
3  while v∃V do
4      foreach v ∈ V do
5          find v_{i_res_max};
6      end
7      foreach cru ∈ CRU do
8          if v_i ∉ cru_j then
9              cru = new(CRU, CRU_dist);
10             if v_{i_res_max} ≤ cru_j then
11                 compare cru_j with CRU;
12                 cru_best ∪ cru_i;
13             end
14         end
15     end
16     if v_{i_res_max} ≠ CRU then
17         cru = new(CRU, CRU_dist);
18         cru_n ∪ v_i;
19     end
20 end
```

5.2.2 CRU Distribution Heuristic

The CRU distribution heuristic enables the calculation of the total number of different sized CRUs that the VCC network can assemble. An estimation technique is introduced that help us in defining our method. The distributed nature of the vehicular environment introduces complexity and difficulties in assessing the individual resources of each vehicle in the network. Therefore, this technique uses the sum of the CPU resources of all vehicles, represented as: $C = \{c_1, c_2, c_3, ..., c_n\}$, to estimate the total number of vehicles present in the network. The sum of the available vehicle CPU resources in the VCC network can be defined as:

Table 5.2: CRU size templates

CRU size	**CPU (MHz)**	**Memory (MB)**	**Storage (MB)**
Small	4	1024	5120
Medium	16	2048	10240
Large	32	4096	20480

$$C = \sum_{i=1}^{n} c_i \tag{5.1}$$

In a distributed, heterogeneous environment, service requests might match different CRU sizes. Consequently, we assume that our VCC network might comprise of a distribution of assembled CRUS. This distribution might follow the predefined CRU sizes, where we might have a population of CRUs that follow α small CRUs, β medium CRUs, and γ large CRUs. Let's denote the predefined resource capacities a for small, medium and large CRUs as $a = \{a_1, a_2, a_3\}$, respectively, as represented in Table 5.2. Using these values and our predefined CRU distribution parameters, we can define:

$$P = (a_1 * \alpha) + (a_2 * \beta) + (a_3 * \gamma) \tag{5.2}$$

Using 5.1 and 5.2, we can calculate the total vehicles in the VCC network as:

$$V = \frac{C}{P} \tag{5.3}$$

Let ρ represent the percentage of resources that can be accommodated in a CRU. We calculate the number of resources that can be allocated to a CRU as:

$$b_i = \frac{C * (\rho_i + 1)}{a_i} \tag{5.4}$$

where b_i represents the resource distribution and a_i represents the CPU capacities for CRU size $i = \{small, medium, large\}$.

Using Equation 5.4, we can also determine the number of CRUs N of each type that the network can assemble as:

$$N = \frac{b_i}{a_i} \tag{5.5}$$

Chapter 6

Game Theory-based CRU Allocation

The second objective of this thesis is resource provisioning and service request fulfillment in VCC networks. After the CRUs have been assembled, they are now ready to be assigned to service requests for fulfillment. The aim of the allocation model is to ensure that CRUs are assigned to service requests in an efficient manner, ensuring fairness among all users, resulting in maximum success rate of service fulfillment and minimal wastage of resources.

As a result, three different models have been explored in this work for conducting CRU allocation: *Naive FCFS*, *Exhaustive MnM* and *Efficient Pruning*. The *Exhaustive MnM* and *Efficient Pruning* models are decision making algorithms that follow a game theoretic-based decision-making approach. The goal of these models is to maximize the utility value of each CRU, based on which they are assigned to requests.

There are a few assumptions we consider for conducting the CRU allocation process:

- We assume that there is a static mobile device nearby the RSU that is responsible for collecting and forwarding the list of service requests from multiple requesters.
- The mobile device communicates with the RSU and other vehicles through V2I communication mode.

6.1 Naive FCFS Model

Allocation in vehicular environments utilizing CRUs as the fundamental resource element involves the already-known scheduling and management challenges of dis-

ALGORITHM 5: CRU Allocation- Naive-FCFS Approach

Data: $R = \{r_1, r_2, ...r_p\}$, CRU = $\{cru_1, cru_2, ..., cru_n\}$
Result: $\overline{CRU} = \{R, CRU\}$

```
foreach r ∈ R do
    foreach cru ∈ CRU do
        if r_icap ≤ cru_jcap then
            cru_j ∪ r_i;
            (r_i, cru_j) = completed;
            CRU ∪ (r_i, cru_j)
        end
    end
end
```

tributed and mobile computing systems. However, the particular scope of this work includes high mobility where the underlying network topology changes constantly and rapidly, aggravating the allocation challenges. At first, we can assume that a naive approach can fulfill service/resource requests as they arrive in the system, based on an first-come-first-served (FCFS) basis. Therefore, we name the naive approach as *Naive FCFS* model. The CRU allocation process using the FCFS approach is briefly explained in Algorithm 5. The algorithm takes the list of service requests and list of CRUs as input. It matches the first CRU on the CRU list with the first request on request list. If it matches, it is assigned to the request and the $(request, CRU)$ pair is marked as "completed". If not, the request is marked as "incomplete" and the next request is processed.

The FCFS algorithm is simple and covers all requests greedily. When trying to implement this approach of resource management using CRUs in a real-life scenario, it is not possible to get feasible results due to several reasons:

- The user request requirements may be more than the maximum capacity of the first available CRU in the list due to which the request cannot be accommodated and is dropped.
- The user request requirements may be minimal and it is allocated a large size CRU, which would lead to resource wastage.
- In the above approach, at a given time, only a single user request is processed and after the completion of the current user service allocation, the next request is taken into consideration. This approach is time consuming since it allows only a single request to be processed at a particular interval. In the modern-day scenario, there are long waiting times for users to get their services processed

which could serve as a drawback for the system. This approach could also lead to higher percentages of service drops which could affect the overall performance of the system.

6.2 Game Theory Model

In this section, we focus on developing a distributed system which allows efficient sharing of resources between RSUs and vehicles in the network. Keeping in mind a real-life scenario, at a particular interval, there are several service requesters requesting for resources to fulfill services, such as obtain road traffic information, download/upload data, and media files. CRU allocation strategy is designed in a way that meets the following requirements:

1. **Efficiency**. Each CRU is allocated in an efficient manner to ensure that all of its resources are utilized to its optimum level. The CRU distribution heuristic assembles three different sized pools of CRUs, which allows service requests to be assigned CRUs that precisely match their requirements. It minimizes the time in searching for individual vehicles that match the service requirements.

2. **QoS**. Each allocated CRU meets with the user's QoS requirements such that, all services assigned to the CRU are completed in a comfortable manner. The CRU allocation model is designed in a way that CRUs are assigned to multiple service requests simultaneously. The model ensures to deliver service request fulfillment in a timely manner.

3. **Fairness**. Each user is able to choose a CRU in a fair manner, without any biased behavior involved. Each service request is assigned a CRU that precisely matches its requirements, without being biased and offering a large or medium size CRU to a small request. CRUs are assigned to requests, leaving some room to accommodate extra requirements, if requested by the requester.

In our proposed approach, we present a game theoretical mechanism for resource provisioning and management in the VCC network. Game theory, when integrated with a VCC architecture that consists of a roadside cloud (RC) and vehicular cloud (VC), has proven to show efficient results for resource allocation and management [20]. The resources provided by RC and VC provide a good communication speed and quality, making the architecture flexible and application-friendly for heterogeneous vehicles and service requests. We introduce and compare two different models of

Game Theory: *exhaustive minimax (MnM)* and *efficient pruning* for conducting CRU allocation.

6.3 Objective of the Game

Here, we tackle the allocation problem through a game theory model, where it identifies the best set of CRUs for all existing user requests to maximize the rate of successfully assigned and fulfillment of requests. This assignment is based on two factors: connectivity level of the CRU and resource capacity. Each request must be assigned a CRU that optimally matches resource requirements and is at the closest distance from the RSU. This network distance decision serves as a policy to ensure that the connection is enabled at all times during service fulfillment. The model can consider multi-hop V2V and V2X connections, thus allowing vehicles to communicate directly with nearby vehicles or with anything that presents connectivity and is able to share information about resources. However, the complexity of the multi-hop connections would introduce high complexity in the allocation process. Therefore, in our current scenario, we only consider V2I communication mode for resource allocation and request fulfillment. At this design stage, we assume that only a single CRU is assigned to a request for task fulfillment.

Consider a finite multi-dimensional game scenario where n players are actively involved. The tuple for our game theory resource allocation model in VCC is defined as $G = \{P, S, U\}$, where P denotes a set of players that represent the incoming user requests that compete for resources (CRUs). These players/requests adopt a strategy S that help them in performing the action of resource allocation. The strategy S in terms of resource allocation is defined as the request meeting with the minimum resource requirements of CRU. The strategy set is defined as $S = \{CRU_j, noCRU\}$. CRU_j means a CRU is available from a pool of CRUs, and it matches the request requirements. $noCRU$ means there is no CRU available that matches request requirements, so no CRU is assigned. Every incoming request r_i adopts a strategy from the strategy set S after matching minimum resource requirements. U denotes the utility function that represents the reward that each request receives at the end of CRU allocation. Two factors contribute towards a request receiving maximum reward: resource allocation and distance. The utility function U is expressed according to Equation 5.

$$f(r_i, CRU_j) = y(r_i, CRU_j) \times d(rsu_m, CRU_j) \tag{5}$$

where

$$y(r_i, CRU_j) = (rel_{j_{cpu}} - r_{i_{cpu}}) \times \phi + (rel_{j_{mem}} - r_{i_{mem}}) \times \lambda + (rel_{j_{stor}} - r_{i_{stor}}) \times \theta \quad (6)$$

In Equation 6, $rel_{j_{cpu}}$, $rel_{j_{mem}}$, and $rel_{j_{stor}}$ represent the relative values of CPU, memory and storage respectively, offered by the j^{th} CRU. $r_{i_{cpu}}$, $r_{i_{mem}}$, and $r_{i_{stor}}$ represent the relative values of the CPU, memory, and storage respectively, requested by the i^{th} service. Consequently, we identify the difference between offered and requested resources. For instance, the value ($rel_{j_{cpu}}$ - $r_{i_{cpu}}$) indicates the additional offered relative CPU the CRU presents to the i^{th} request. The coefficients, ϕ, λ, and θ represent the predefined resource weights that indicate the importance of computation, storage, and data type resources in the workload of the j^{th} CRU.

Also, the last term of Equation 5, $d(rsu_m, CRU_j)$, corresponds to the geographical distance between the rsu_m and j^{th} CRU, which is calculated as a Euclidean distance, and is represented as:

$$rel_{dist_{rsu_m - CRU_j}} = |d_{rsu_m - CRU_j} - min(d_{rsu_m - CRU}|/max(d_{rsu_m - CRU}) - min(d_{rsu_m - CRU}) \quad (7)$$

where $d_{rsu_m - CRU}$ represents the set of all distances between rsu_m and all CRUs. Identified in the urban road-segment topology, rsu_m is the closest RSU to the j^{th} CRU.

There are several properties that make our resource management problem appropriate for an extensive-form game model:

- Each CRU payoff value is a function of its distance from the RSU. Since a CRU can be composed of a single or multiple nodes/vehicles, the distance of a CRU from RSU is calculated as an average of the cumulative sum of the distances of each vehicle that is clustered in a CRU, from the RSU. This distance is a function of the CRUs own distance from the RSU and the distances of other CRUs from the RSU.

- When two or more CRUs are located at the same distances from the RSU, the CRU that offers minimal additional resources (sufficient to fulfill a request and minimizing wastage of resources) is allocated to a service request.

- When two or more CRUs offer equal number of resources for service fulfillment, the CRU which is at the closest distance to the RSU is allocated.

- In an unlikely situation where two CRUs that are at equal distances from the

RSU and offer equal number of resources for service fulfillment, the CRU that is first on the list of CRUs is allocated.

6.3.1 Exhaustive MnM

After establishing the utility function (payoff value), we adopt an *Exhaustive MnM* algorithm, which is based on the principle of *minimax* algorithm and is used for solving the modeled game theory problem. This algorithm is used to find the optimal CRU for a service request, based on the utility value of the CRU. It considers the utility function defined in equation 5 and aims at maximizing it for each service request. Each service request is matched with CRUs that match minimum resource requirements and display a high utility value.

Our scenario represents an extensive form game in which our game is represented as a tree. In this tree, the nodes represent the different service requests and the edges represent the set of CRUs that are ideal for matching the service request requirements. The utility value of each CRU is calculated using Equation 5. The *Exhaustive MnM* algorithm is applied to the set of utility values of the matched ideal CRUs to find the optimal CRU for the service request.

Since we have inferred the connectivity status of a CRU as its distance from the RSU, we can establish that our utility value will be inversely proportional to the distance. This means that a CRU that is closest (at a shorter distance) to the RSU will acquire a higher value of payoff. Besides, the relative resources value, which defines the additional resources that a CRU can offer to a service request, will also be inversely proportional to the utility value. This means that a CRU that offers sufficient resources to fulfill a service request, without resulting in wastage of resources, is allocated to a request. Therefore, our utility function can be expressed mathematically as:

$$U = 1/(relative_{distance} \times relative_{resources}) \tag{8}$$

where $relative_{distance}$ is expressed as a Euclidean distance, as defined in equation 7, and $relative_{resources}$ is expressed as the additional resources, as defined in equation 6.

Algorithm 6 explains the working of the CRU allocation process and how *Exhaustive MnM* is applied for acquiring an ideal CRU match for a service request. The algorithm is implemented on a combinatorial search tree, with the nodes being the service requests and the edges being the selected CRUs. The algorithm begins by matching the service requirements with CRU resources and it calculates the utility

ALGORITHM 6: Exhaustive MnM - CRU Allocation

```
   Input: ur: node of a combinatorial search tree
   Result: V̄: set of pairs (ur, cru)
 1 V = ∅;
 2 if ur ≤ cru then
 3 |  V ∪ {ur, cru}
 4 end
 5 foreach {ur, cru} ∈ V do
 6 |  f(ur, cru) = y(ur, cru) × d(rsu, cru);
 7 |  Ū ∪ f(ur, cru)
 8 end
 9 Function Exhaustive MnM(n, depth, isMax) is
10 |  foreach f(ur, cru) ∈ Ū do
11 |  |  if depth = 0 then
12 |  |  |  V̄ ∪ f(ur, cru)
13 |  |  else
14 |  |  |  bestVal = −∞;
15 |  |  |  value = MnM(Ū, depth − 1, false);
16 |  |  |  bestVal = max(bestVal, value);
17 |  |  |  return bestVal;
18 |  |  |  V̄ ∪ bestVal
19 |  |  end
20 |  end
21 end
```

value of each CRU. After the utility values are calculated, its value is maximized with a threshold value $\alpha = -\infty$. This process is repeated for every child node/CRU of the tree, until the CRU with maximum utility has been found. At the end of the algorithm, the CRU with maximum utility is assigned to the service request for processing.

However, the *Exhaustive MnM* model presents some drawbacks:

1. This algorithm is based on the principle of minimax algorithm which has a significant branching factor. The branching factor is defined as the number of moves/choices a player has to win an optimal outcome. This can be a time-consuming process, especially in complex, large-scale scenarios like ours when there are multiple requests and significant number of CRUs available for allocation. Therefore, we adopt an *efficient pruning* technique, that is an optimization for the *exhaustive MnM* algorithm. We further discuss about this in the next section.

2. The distance factor, as defined in Equation 7, is a linear distance. This means

that a CRU closer to the RSU gets a higher utility value than a CRU that is farther away from the RSU. However, in real-world scenarios, the communication reliability grows differently with distance. For instance, a CRU that is closest to the RSU, may have a poor connection. As it moves slightly away, it gains a strong connection as the signal strength improves. Therefore, it is important to consider the signal strength of each CRU, along with their distance from RSU and the additional resources it offers, for service allocation. We define a new utility function for the CRUs in the next section, considering the signal strength of the CRU.

6.3.2 Efficient Pruning

The *efficient pruning* algorithm is proposed as a performance enhancement of the *exhaustive MnM* algorithm to overcome its drawbacks. This algorithm is based on the principle of *alpha-beta pruning* which minimizes the number of moves of a node by half, yielding faster searches, resulting in minimal composition time.

The time complexity of *efficient pruning* and *Exhaustive MnM* can be defined as:

- *Exhaustive MnM*: $\mathcal{O}(b^d)$
- *Efficient Pruning*: $\mathcal{O}(\sqrt{b^d})$

where b represents the branching factor, which is defined as the number of children at each node of a tree data structure. The higher the branching factor of a search tree, the greater is the search time of the algorithm. d represents the depth of the search tree, which is defined as the number of edges between the root node and the leaf node in a search tree. A higher number of edges results in a greater depth of the tree. Therefore, we can conclude that *efficient pruning* algorithm is a better choice than *exhaustive MnM* since it avoids many unnecessary search moves across the tree.

This algorithm, like the *Exhaustive MnM*, also considers the utility value of CRUs to allocate them to service requests. However, we define the utility function of this model slightly different from the *Exhaustive MnM* model. As discussed earlier, it is important to consider the signal strength of a CRU in defining its utility function in terms of *Connectivity* and *Resource Provisioning* factors. Therefore, we define the connectivity factor of a CRU in terms of its *Receiving Signal Strength Indicator* (RSSI) value, which is defined as the estimated transmit power that a device is receiving from an access point or router. In other words, it measures how well the receiving device can hear a signal that is being transmitted from an access point or router, located

Table 6.1: RSSI parameter values

Parameter	**Value**
n	2 (for free-space path-loss model)
d	distance of a vehicle from RSU
d_0	1m
A	17.78dBm

anywhere in the network. This value determines if the receiving device has sufficient signal to establish a good wireless connection. The *RSSI* value is a negative value, expressed in dBm. The closer the value to 0, the stronger is the *RSSI*.

In a log-distance path loss model, the receiving signal strength of receiving nodes/vehicles can be expressed as:

$$RSSI = A - 10nlog_{10}(d/d_0) - X_\sigma \tag{6.1}$$

The *RSSI* value is determined from several factors of the path loss model of free space propagation such as:

- Path loss exponent, also called attenuation factor$n$. It measures the rate at which the value of RSSI decreases as the distance between transmitter and receiver increases. It depends on the specific propagation environment.
- Distance between transmitter and receiver d.
- Reference distance d_0.
- Received signal power A, in dBm.
- A Gaussian random variable that reflects attenuation caused by fading, X_σ

For the sake of simplicity, we can neglect the fade margin factor X_σ since we do not consider any shadowing or obstacles in our scenario. Therefore, considering the parameter values from table 6.1, our final *RSSI* equation can be defined as:

$$RSSI = A - 10nlog_{10}(d) \tag{6.2}$$

To define the utility function for the *Efficient Pruning* algorithm, we calculate the $relative_{RSSI}$ value for each CRU as:

$$relative_{RSSI} = |CRU_{j_{RSSI}} - min_{RSSI}|/max_{RSSI} - min_{RSSI} \tag{6.3}$$

ALGORITHM 7: Efficient Pruning - CRU Allocation

Input: ur: node of a combinatorial search tree
Result: $\overline{V}$: set of pairs (ur, cru)

$V = \varnothing$;
$\overline{U} = \varnothing$;
if $ur \leq cru$ **then**
  $V \cup \{ur, cru\}$
end
foreach $\{ur, cru\} \in V$ **do**
  $f(ur, cru) = y(ur, cru) \times d(rsu, cru)$;
  $\overline{U} \cup f(ur, cru)$
end
Function Efficient Pruning(n, alpha, beta, depth, isMax) **is**
  foreach $f(ur, cru) \in \overline{U}$ **do**
    if *depth = 0* **then**
      $\overline{V} \cup f(ur, cru)$
    else
      $result = -\infty$;
      $value = EfficientPruning(\overline{U}, -\infty, +\infty, depth - 1, false)$;
      $result = max(result, value)$;
      $alpha = max(alpha, result)$;
      if *beta > alpha* **then**
        $optimal = EfficientPruning(\overline{U}, -\infty, +\infty, depth - 1, false)$;
        $result = max(result, value)$;
        $alpha = max(alpha, result)$;
      end
      return $result$;
      $\overline{V} \cup result$
    end
  end
end

Using equations 6 and 6.3, we can define the utility function for Algorithm 7 as:

$$U = relative_{RSSI}/relative_{resources} \tag{6.4}$$

where $relative_{resources}$ is defined in 6. According to this equation of the utility function, it is directly proportional to the *RSSI* value of a CRU. Closer a CRU to the RSU, higher is its *RSSI* value which signifies strong signal strength and higher is the utility value. The $relative_{resources}$ factor remains inversely proportional to the utility value, similar to *Exhaustive MnM* algorithm.

Algorithm 7 explains the working of the CRU allocation process and how *Efficient Pruning* is applied for acquiring an ideal CRU match for a service request. The algo-

rithm is implemented on a combinatorial search tree, with the nodes being the service requests and the edges being the selected CRUs that match minimum service request requirements. There are two threshold parameters defined in this algorithm, $\alpha = -\infty$ and $\beta = +\infty$. The algorithm begins by matching the service requirements with CRU resources. Following the matching process, the utility value for each matched CRU is calculated and stored in a list. At this point, the *Efficient Pruning* algorithm is applied to the list of utilities The algorithm works by iterating over the nodes (CRUs) of the tree until the condition $\alpha \geq \beta$ becomes true. Once the condition becomes true, the algorithm prunes the remaining nodes and edges in the tree since it has found the ideal choice of CRU at the point where the condition becomes true.

The *Efficient Pruning* algorithm yields the same results as the *Exhsuative MnM* algorithm. However, since the latter individually loops through every CRU (edges of the tree) on the list to obtain an ideal match for a request, the process is inherently time-consuming. Additionally, the function considered in calculating the utility value for a CRU may not always be reliable. Therefore, *Efficient Pruning* is expected to overcome the drawbacks of the *Exhaustive MnM* model and display ideal choices of CRUs for service allocation. The CRUs selected as ideal choices for service requests represent more realistic selections since the algorithm considers details from the underlying communication media in calculating the utility value of each CRU. The optimal choices of CRU selected are considered more reliable than those obtained from the minimax algorithm.

Chapter 7

Performance Analysis

The proposed approach of CRU composition and allocation was analyzed and tested by conducting simulation experimental analysis. These simulations closely resemble real-time urban scenarios, with vehicles exhibiting high and varying mobility. This chapter will briefly discuss our VCC experimental scenario, the different parameters and performance metrics that are used to evaluate our proposed approach, and the results obtained by implementing the simulations.

7.1 Scenario

The simulation environment for our analyses is built using Veins, Omnet++, and SUMO. The simulator veins [23] was built for running simulations of vehicular networks, in support of omnet++ [25], which provides a C++ simulation library and framework for building network simulators. Sumo [18] provides a microscopic simulation traffic package that is built to handle large vehicular networks. Our simulation scenario is completely built on V2I (Vehicle-to-Infrastructure) communication mode. This means that vehicles in the network can connect to the internet and communicate with each other via RSUs.

7.2 Traffic Network Topology

To emulate a real-world traffic network topology, we employ the Manhattan grid network area as our urban scenario. Vehicles in our network present high and varying mobility displacement patterns. The Manhattan grid represents an urban area of $1000 \times 1000m^2$. The primary RSU, which is responsible for the composition and

Table 7.1: Simulation Parameter Settings.

Parameter	**Value Range**
Vehicle Density	100 - 1000
RSU Density	1
Vehicle Speed	13.90 m/s
Vehicle CPU	1 - 8 MHz
Vehicle Memory	64 - 1024 MB
Vehicle Storage	1280 - 5120 MB
Transmission Power	60mW
RSU comm. range	1000m
α, β, γ	0.5, 0.25, 0.25
Map	Manhattan Grid

allocation of CRUs is placed in the centre of the network. This would allow maximum vehicles in the network to connect and communicate with the RSU and share their resources for service processing. Two other RSUs, one at the top right corner and other at the bottom left corner, are placed in the network. These RSUs are responsible for connecting and communicating with vehicles that the primary RSU is unable to reach. A mobile device is placed at a close distance to the primary RSU. It is responsible for collecting all service requests from users and forwarding them to the RSU for processing.

Both set of experiments conducted in this thesis work employ the same traffic network topology. The first section comprises the performance analysis of the CRU composition process, where we discuss the different parameters, performance metrics and results of the proposed composition approach. The second section comprises the performance analysis of the CRU allocation and service fulfillment process. We define similar parameters for this approach as well, a new set of performance metrics, and discuss the results gathered from the simulations.

7.3 Performance Analysis of CRU Composition

The CRU composition requires a list of nearby vehicles as input in order to cluster resources efficiently. The clustering efficiency directly impacts resource allocation. Therefore, our vehicular network is well-defined with a combination of suitable parameters and performance metrics, that will contribute towards gathering desired experimental results.

7.3.1 Parameters

For evaluating our CRU approach, we have defined a combination of parameter settings in Table 5.1. In our simulation environment, the speed of the vehicles is set to 13.90 m/s, which resembles the speed of vehicles in an urban scenario with traffic conditions. This speed could result in vehicles frequently changing their position in the network, resulting in loss of their connectivity and highly dynamic nature of our VCC environment.

In our simulation scenario, the different densities of vehicles, ranging from 100 - 1000 vehicles, impact the composition of the CRUs, which is linked to the success rate of service fulfillment. The higher densities of vehicles result in the formation of large number of different pools of CRU, maximizing service fulfillment rate. For the sake of simplicity in our simulation analyses, we have considered CPU as the available resources of vehicles in the observed urban scenario, since it would be difficult to assess individual sets of resources of each vehicle in a distributed network.

A single RSU is deployed in the grid network and is responsible for initiating the simulation by connecting and communicating with vehicles in close vicinity, and composing different pools of CRU, based on the distribution heuristic data. The RSU is static and is responsible for initiating the CRU composition process by collecting resource information of vehicles and using them to assemble different pools of CRUs, by implementing the different search algorithms. The RSU also provides internet coverage to the vehicles through V2I.

For the sake of emulating real-world scenarios, our scenario depicts the vulnerability of the vehicles and RSU to collisions/accidents, attenuation and propagation. These factors contribute to strengthening the performance of our proposed CRU composition process to dynamic conditions such as changing network topology, loss of connectivity.

After the RSU has collected the resource information of vehicles, the *distribution heuristic* method is used to estimate the number of total different pools of CRUs that the network is able to safely assemble.

7.3.2 Performance Metrics

This work focuses on the composition of CRUs using maximum underutilized vehicle resources while maximizing the success rate of matching requests by aggregating different sized CRUs. We adopted several performance metrics to analyze and test our CRU approach.

Average number of vehicles per CRU size estimates the total number of vehicles that are utilized in composing the three different pools of CRUs, based on their maximum predefined capacity. The large CRU will be composed using more vehicles, as compared to the small and medium pools.

Resources available in composing a CRU is the second metric we adopted in conducting this experiment. This metric estimates the number of resources that are required to maximize the predefined capacity of a CRU.

The third metric we adopted in measuring the performance of our approach is *Ratio of Under-assembled/Total CRUs*. This metric measures how many CRUs assembled by each algorithm is composed under its predefined capacity. In other words, this metric calculates the total CRUs that are incompletely composed. This is an important metric since it determines how many resources remain under-utilized that could be used for maximizing the success rate of service fulfillment.

Ratio of Over-assembled/Total CRUs is our fourth performance metric. The results of this metric determine the number of CRUs that have been composed exceeding their predefined capacity. When CRUs are composed exceeding their predefined capacity, it results in reduced total number of CRUs in the network, adversely impacting the success rate of service fulfillment.

Accuracy of Distribution Heuristic is another metric observed for testing the performance of our approach. The results of this metric determine the ratio of composed CRUs to the expected CRUs as estimated by the *distribution heuristic* method. The *distribution heuristic* method precisely determines the total number of CRUs the network can safely assemble and support. Therefore, it is important that the algorithms are able to match the results of this method.

7.3.3 Results

The experiments conducted to analyze the CRU approach compared the performance of our distinctive search algorithms, based on the performance metrics defined above. These comparisons were made with different vehicle densities, and averages with 95% confidence intervals were calculated from 10 simulation runs for each vehicle density and different resource parameters defined in Table 5.1.

Figure 7.1 depicts the average number of vehicles that were used by the different CRU composition algorithms. CRU composition is concerned with using least number of vehicles in composing a CRU. Therefore, *SSA* begins the composition process by assembling large size CRUs first by using vehicles with maximum resources. As

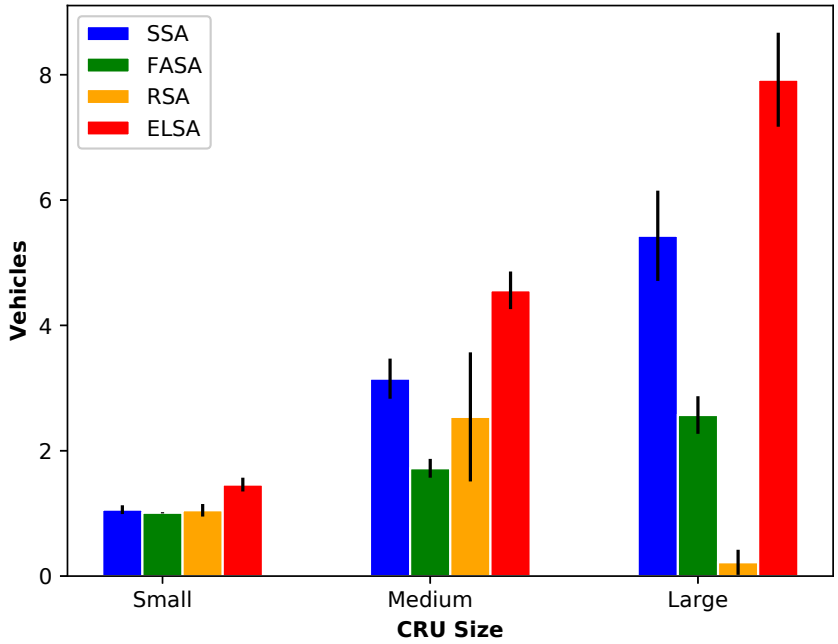

Figure 7.1: Average number of vehicles per CRU size.

depicted in Figure 7.1, *FASA* performs better in this area than the other algorithms. However, it also results in a significant number of under-assembled CRUs because of its inability to meet the maximum predefined capacity of each CRU type.

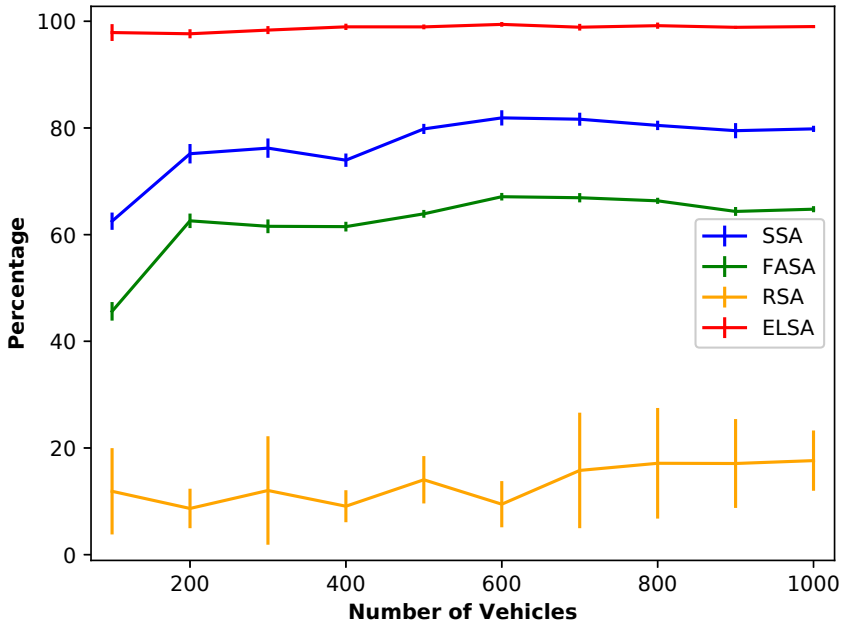

Figure 7.2: Percentage of resources assigned to CRU.

Figure 7.2 shows the relative amount of resources grouped by a CRU for different vehicle densities by the different algorithms. *ELSA* shows almost 100% resource utilization in the compositions. However, these results cannot be considered our best choice since the algorithm exceeds the predefined capacity of each CRU type. Hence, it results in clustering maximum resources in a single CRU, limiting the total number of CRUs in the VCC network and affecting service fulfillment rate. *SSA*, on the other hand, uses optimal number of resources in composing a CRU, resulting in composing minimal under-assembled and over-assembled CRUs. *FASA*, compared to *SSA*, assigns lesser resources to a CRU. However, it performs better than *RSA* that assigns the least resources for composing a CRU.

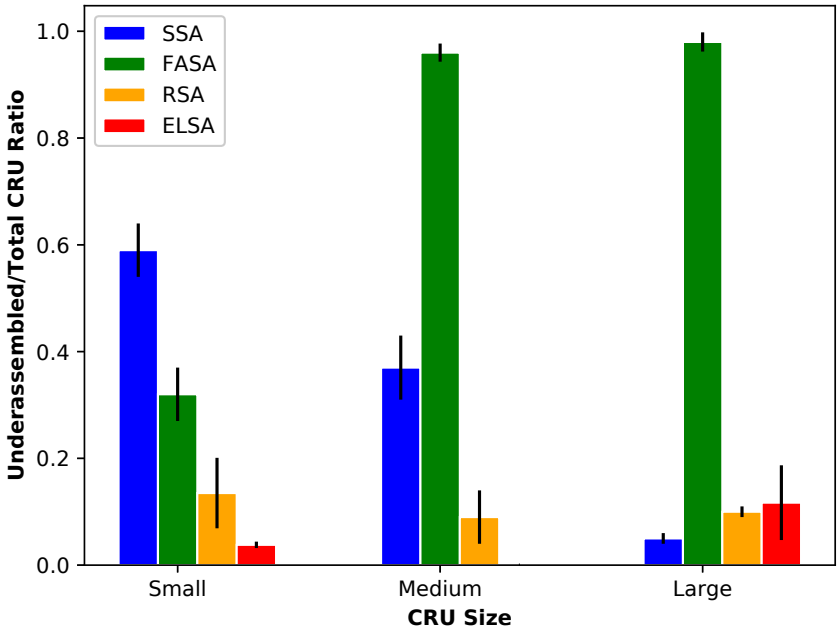

Figure 7.3: Ratio of under-assembled CRUs.

Figure 7.3 represents the comparison of the number of under-assembled CRUs between different approaches. As seen from the figure, since *SSA* starts the CRU composition process with composing larger CRUs first, it builds a minimum number of under-assembled CRUs. *FASA*, on the other hand, results in maximum number of under-assembled CRUs. This is because the algorithm checks for the already existing CRU for clustering vehicles, and if it fits, the vehicle is accommodated in that CRU. It does not check for other existing CRUs that could better accommodate a vehicle, resulting in increased number of under-assembled CRUs. The bar graph of *RSA* shows that it results in composing the least number of under-assembled CRUs. However, *RSA* does not match the distribution heuristic and it composes the least number of

total CRUs, most of which are under-assembled. Hence, these results are considered flawed rather an advantage for the VCC system. Lastly, *ELSA* follows a greedy approach, where it tries to assign maximum number of resources to a CRU, resulting in fewer number of under-assembled CRUs.

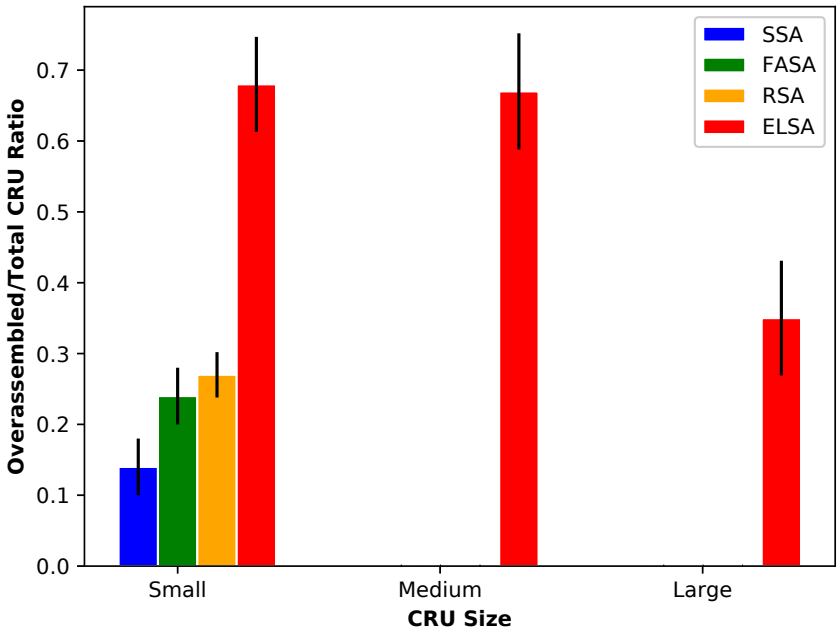

Figure 7.4: Ratio of over-assembled CRUs.

Figure 7.4 shows the comparison of the number of over-assembled CRUs between the different approaches. In the figure, *SSA* shows that it composes the least number of over-assembled CRUs and composes a minimum percentage of only small over-assembled CRUs. *FASA* also performs better in this area since it composes minimal number of only small over-assembled CRUs. As previously explained, the *RSA* minimally matches the distribution heuristic and results in composing significantly fewer CRUs in the VCC system, with a few small over-assembled CRUs. Hence, these results show a performance that does not benefit VCC resource management. Lastly, *ELSA* results in composing the maximum number of over-assembled CRUs since it follows a greedy approach, attempting to cluster maximum amount of resources in a CRU, exceeding the predefined CRU capacity.

Figure 7.5 compares the ratio of composed CRUs to the expected CRUs estimated by the distribution heuristic between the different approaches. The results of *FASA* and *SSA* demonstrate almost 100% accuracy. *RSA* matches the least with the distribution heuristic, since CRUs are closed off, before their maximum capacity has been reached, leaving no CRUs to cluster the remaining vehicle resources. *ELSA* begins

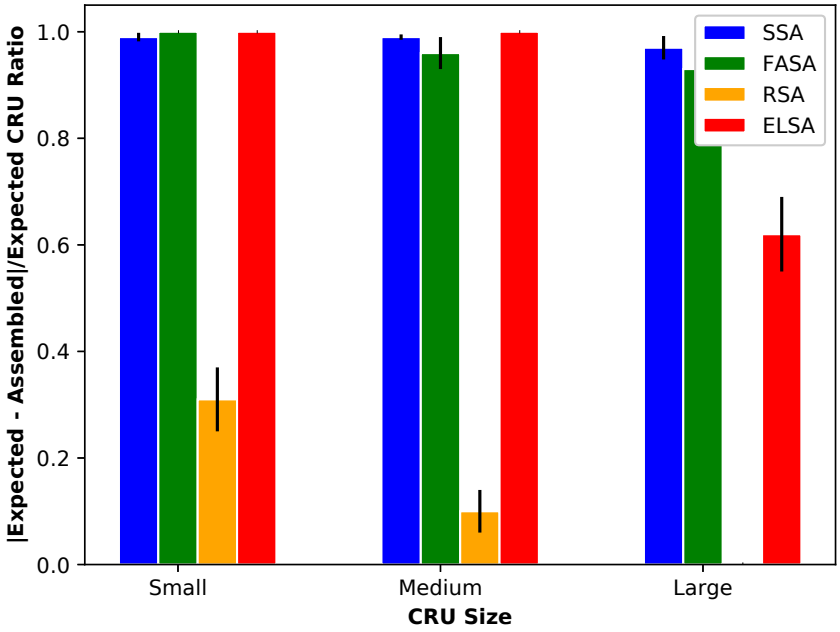

Figure 7.5: Distribution Heuristic Accuracy.

with assembling small sized CRUs, followed by medium and large. Therefore, it over-exceeds the capacity of small and medium sized CRUs by clustering more number of vehicles, thus leaving behind fewer vehicles to cluster the large size CRUs. As seen from Figure 7.3, *ELSA*'s bar graph shows more number of large under-assembled CRUs.

In terms of time complexity, both *ELSA* and *RSA* require only $\mathcal{O}(n)$ time to compose CRUs. The time complexity of *FASA* is $\mathcal{O}(n^2)$ and *SSA* is $\mathcal{O}(n \log n)$. Even though light-weight, *ELSA* results in composing CRUs exceeding their predefined capacity, and *RSA* leaves maximum CRUs incomplete.

After successfully implementing our search algorithms, we can conclude that the *SSA* is our ideal choice for conducting the CRU composition process. It results in utilizing maximum resources for composing CRUs, leaving behind minimal to no underutilized resources in the system. The algorithm composes least number of under-assembled and over-assembled CRUs, meeting accurately with the distribution heuristic results. The only drawback of *SSA* is it utilizes a higher number of vehicles in composing CRUs, compared to *FASA*. However, the several advantages of *SSA* overcome its only flaw, making it an ideal choice for conducting CRU composition in real-world scenarios. The worst performing algorithm is the *RSA*, since it results in significant number of under-assembled CRUs, and assigning minimum resources to CRUs for composition process. The algorithm composes the least number of total

CRUs, minimally meeting with the distribution heuristic results.

7.4 Performance Analysis of CRU Allocation

The CRU allocation takes place following the CRU composition process. After identifying the three different pools of CRUs that the network can safely assemble, the mobile device that is in vicinity of the RSU collects service requests from multiple requesters and forwards them to the RSU for CRU allocation and service fulfillment. We define a set of parameters and performance metrics that will contribute towards testing our allocation approach and gathering desired experimental results. The set of parameters remain the same, as defined in table 7.1, and the values of the coefficients, ϕ, λ, θ, defined in equation 6, are 0.6, 0.2 and 0.2, respectively. The network also has a mobile device that is always located in the vicinity of the RSU.

7.4.1 Experiments

Experiments have been conducted to analyze the CRU allocation process using the *Naive FCFS*, *Exhaustive MnM*, and *Efficient Pruning* models, based on the performance metrics that have been defined in Section 7.4.2. These comparisons were made with different vehicle densities and CRU composition algorithms, and the different resource parameters as defined in Section 7.4. We tested our approach with three different sets of service requests: 30, 50 and 100.

Our metric is based on measuring the number of requests that have been acknowledged and completely processed at the end of the allocation and service fulfillment process. We test the performance of this metric for a vehicle density ranging from 100-1000. We estimate the average number of requests that are completely processed by the three allocation models. These averages are calculated from 30 simulation runs for each vehicle density, with 95% confident intervals. The y-axis represents the ratio of complete/total number of requests, fulfilled by the 3 models and the x-axis represents the vehicle density.

7.4.2 Performance Metrics

Following the composition of different pools of the CRUs in the VCC network, this work focuses on allocating these CRUs to service requests for service fulfillment. We adopted several performance metrics to analyze and test the CRU allocation approach using game theory.

- **CRU-assigned requests**. It calculates the number of requests that have been assigned a CRU for service processing, since it matches minimum request requirements.

- **CRU-assigned and completely processed service requests**. It calculates the number of requests that have been assigned a CRU and have been completely processed. The vehicles that compose the assigned CRUs are in range of the RSU and provide their resources, resulting in 100% service fulfillment.

- **CRU-assigned and partially processed service requests**. It calculates the number of requests that have been assigned a CRU and have been partially processed. Of the vehicles that compose the assigned CRUs, some vehicles are not in range of the RSU to provide their resources, resulting in the request being partially processed or incompletely processed.

- **CRU-unassigned requests**. It calculates the number of requests that have not been assigned a CRU for service processing, since it does not match minimum request requirements.

7.4.3 Results

Simulations were conducted for evaluating the performance of all three allocation algorithms, with three sets of service requests; 30, 50 and 100 and vehicle density 100-1000.

Figures 7.6, 7.7 and 7.8 display the ratio of service requests that have been assigned a CRU by matching minimum service requirements. The line graphs of all three algorithms in 7.6 present a 100% success ratio of assignment. As the vehicle density increases, the network is able to assemble more number of CRUs which are able to match minimum service request requirements. However, at lower vehicle densities, we can observe that the ratio of success is 60% lower for *Naive FCFS* and *Exhaustive MnM* and 55% lower for *Efficient Pruning*. With low vehicle densities, the network assembles considerably fewer CRUs and since the requests are random in nature, there could possibly be more number of large size requests, requesting for large size CRUs. However, with low vehicle densities, the network assembles limited number of large size CRUs, resulting in unassigned service requests.

In Figure 7.7, *Naive FCFS* presents a 30% drop in success rate of assignment, as compared to *Exhaustive MnM* and *Efficient Pruning* algorithms. As the number of service requests increases, there are fewer CRUs that match minimum service

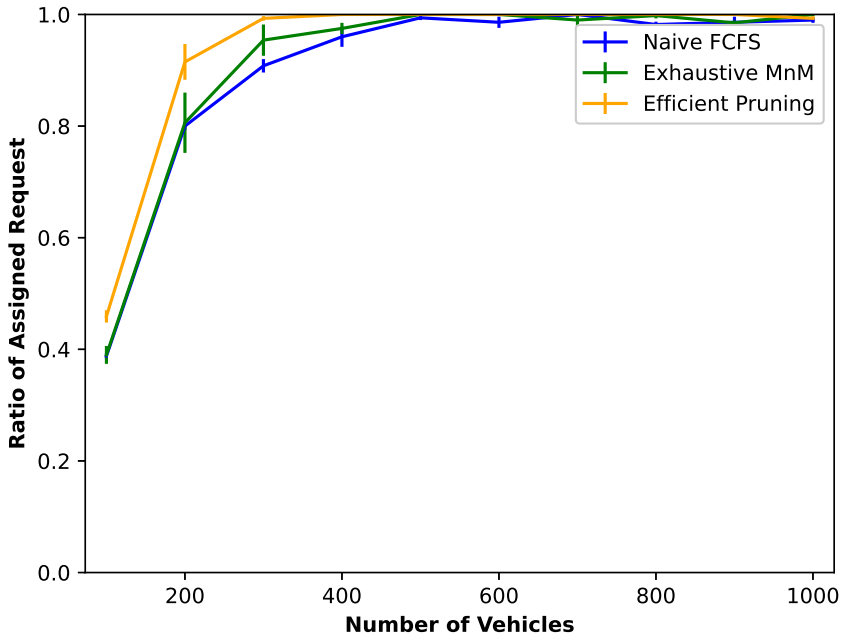

Figure 7.6: Ratio of Assigned / Total Number of Requests = 30

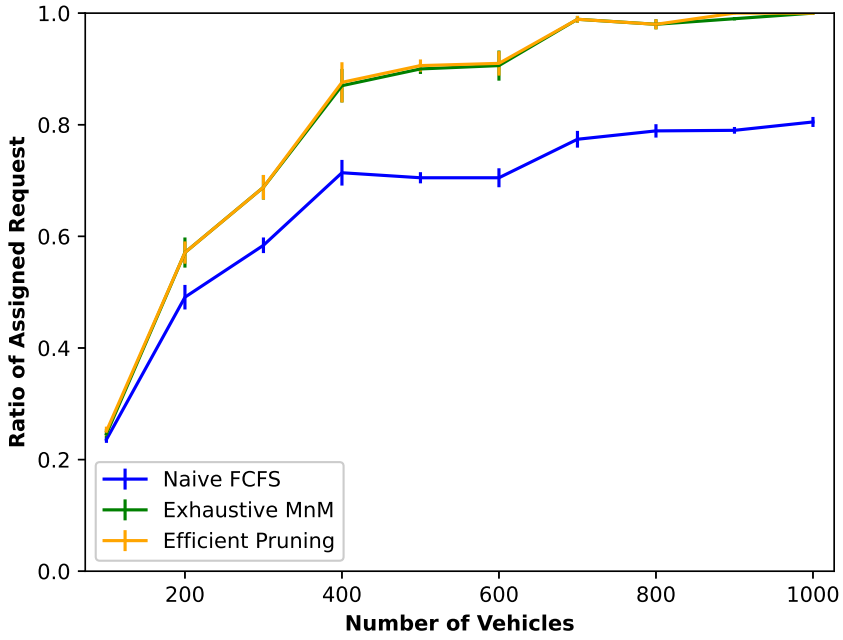

Figure 7.7: Ratio of Assigned / Total Number of Requests = 50

requirements based on a FCFS approach. However, *Exhaustive MnM* and *Efficient Pruning* present similar success ratio of assignment, with a drop of almost 20%, as compared to Figure 7.6. There may not be sufficient CRUs in the network that match with resource requirements of service requests. These are mostly CRUs belonging to

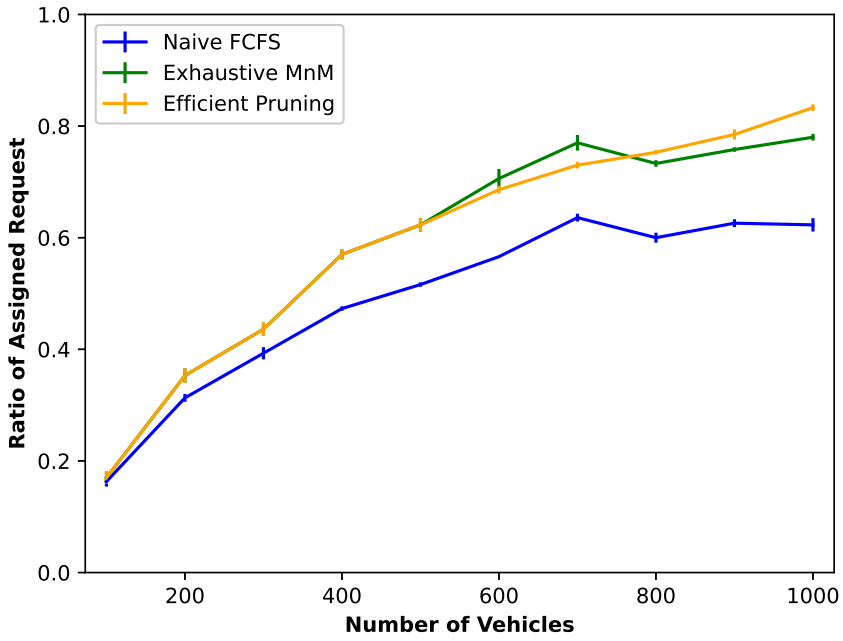

Figure 7.8: Ratio of Assigned / Total Number of Requests = 100

the large pool since the distribution heuristic method assembles minimal large size CRUs, to allow composing more CRUs in the network.

Figure 7.8 represents the ratio of assigned requests for all three algorithms when the number of requests is equal to 100. As the number of service requests increases, it affects the ratio of assignment of all three algorithms moderately. There is a drop in the ratios by 40%, 25% and 20% for *Naive FCFS*, *Exhaustive MnM* and *Efficient Pruning*, respectively. With increase in the number of requests, it becomes difficult for CRUs to match the resource requirements of all services, especially in a scenario like ours, where the algorithms assign only a single CRU to each request.

Figures 7.9, 7.10 and 7.11 displays the ratio of CRU-assigned and completely processed service requests. As observed from the figure, all three models perform significantly well, with a success rate of 70%. However, when we individually compare their performances, the *Naive FCFS* algorithm presented 20% and 28% lower ratios than *Exhaustive MnM* and *Efficient Pruning*, respectively, since each request is matched with the first available CRU and if the request does not match, it is termed as "Unassigned" and returned back to the mobile device. *Exhaustive MnM* and *Efficient Pruning* perform better than *Naive FCFS* since the two algorithms assign CRUs to service requests, considering the distance of a CRU from the RSU. However, *Exhaustive MnM* considers a linear distance that is not reliable in assigning a CRU to a service request. *Efficient Pruning* performs considerably well, with a drop in

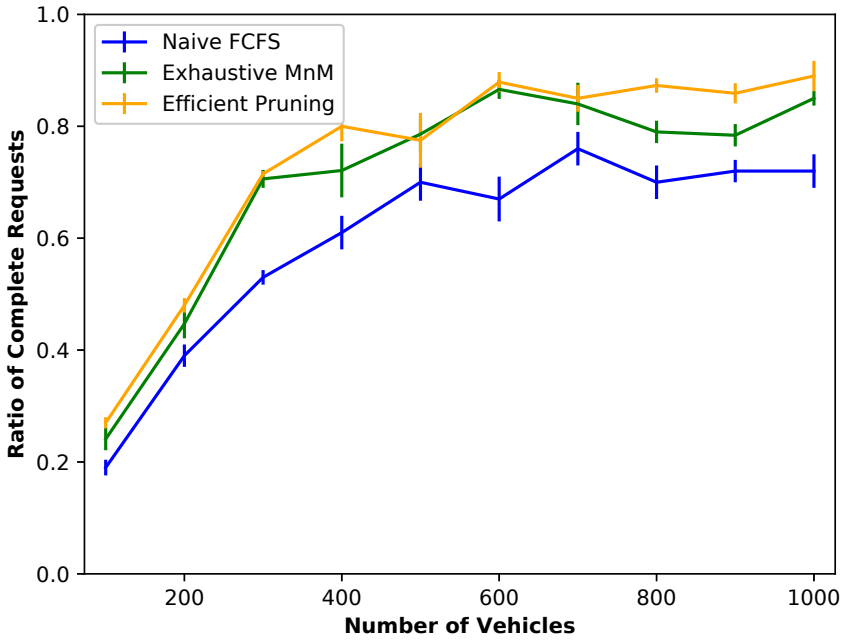

Figure 7.9: Ratio of Complete / Total Number of Requests = 30

success rate for a 500-vehicle density in a 30-requests scenario, displayed in 7.9. This happens due to the high mobility of vehicles, resulting in them quickly moving out of range of the RSU. The position of the RSU also affects the success rate for certain vehicle densities. The initial drop in success rate for vehicle densities 100 and 200 is due to the fact that when there are less vehicles in the network, there are fewer CRUs composed. It impacts the rate of service fulfillment since there would not be enough CRUs in the system to accommodate the requirements or the request would be too large and minimal large CRUs available for allocation. Additionally, when there are fewer service requests to process, the vehicles allocated for resource provisioning tend to remain in vicinity of the RSU, thus increasing the rate of service fulfillment.

In Figure 7.10, we see a moderate drop of about 20% in the success rate of complete requests for the *Naive FCFS* model, when the total number of service requests is equal to 50. It happens because, as we increase the number of service requests, there are either not enough CRUs available for allocation, especially in low density vehicles, or the CRU capacity does not match increased number of service requests since requests are allocated on a first-come-first-served basis. It results in smaller requests being assigned medium or large size CRUs, leaving behind no CRUs to fulfill medium and large size requests. It is a point to note that the mobile device does not forward service requests categorizing them by CRU sizes. However, *Exhaustive Naive* and *Efficient Pruning* present a steadily growing graph as the vehicle density

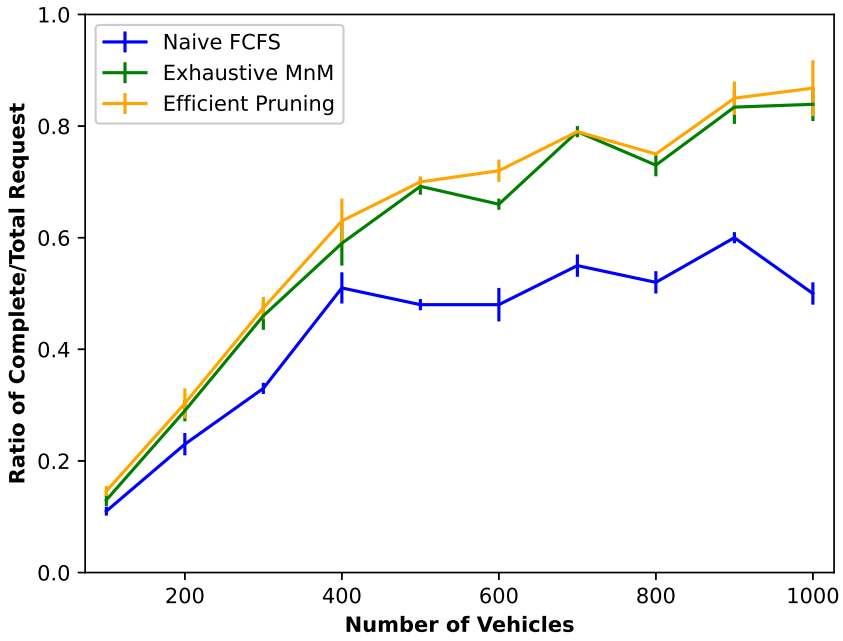

Figure 7.10: Ratio of Complete / Total Number of Requests = 50

increases. However, there is a slight decrease in the success rate for these two models, as compared to Figure 7.9, because as the number of requests increases and since these requests are random in nature (of varying resource requirements), there would not be sufficient CRUs to accommodate the needs of some requests.

In Figure 7.11, we notice a significant drop in the success rate of all three models, when the number of service requests is equal to 100. Since the CRU composition follows a distribution heuristic and assembles a balanced number of CRUs that the network can support. Therefore, this is a scenario where there are not sufficient CRUs available for allocation for lower vehicle density and a substantial group of these requests could possibly belong to the medium and large size pool, resulting in almost half the requests not being completely processed. Additionally, when the number of service requests is large, a larger number of CRUs (and therefore, larger number of vehicles) are assigned to fulfill these requests. However, these vehicles display high mobility and may move out of range of the RSU, resulting in a drop in the success rate of complete requests.

The *Efficient Pruning* model presents 20% and 10% higher ratios of success than *Naive FCFS* and *Exhaustive MnM*, respectively. The *Naive FCFS* model is based on the first-come-first-served approach, which is not considered reliable since majority of service requests would remain unfulfilled if the small size requests are allocated

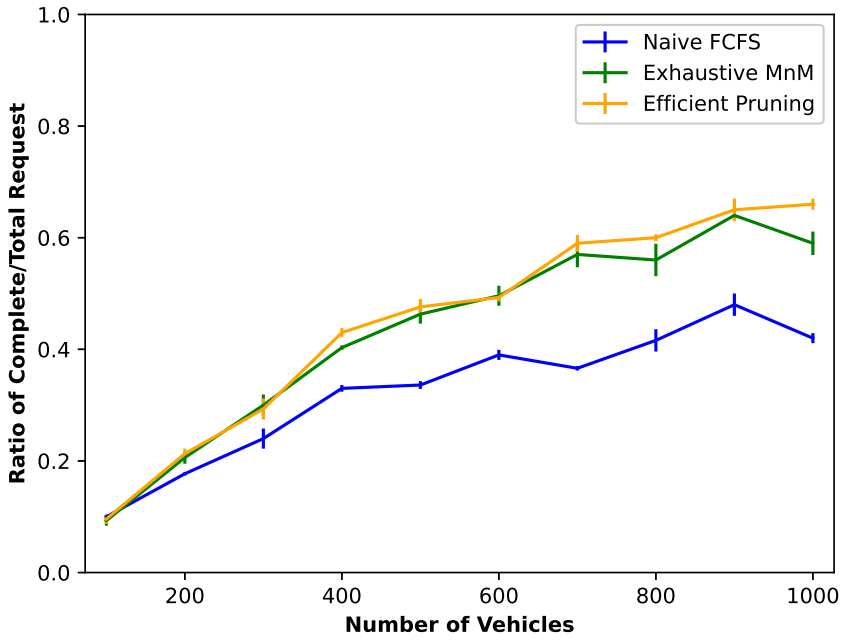

Figure 7.11: Ratio of Complete / Total Number of Requests = 100

medium and large sized CRUs. The distance factor used to calculate the utility value of a CRU in the *Exhaustive MnM* approach is a linear distance. This means that a CRU closer to the RSU is considered reliable for allocation than one that is farther away. However, in real-life scenarios, it could be possible that the CRU closest to the RSU displays a weak connection in comparison to the one farther away. Therefore, the *RSSI* value is reliable to determine the ideal CRU for service allocation, which makes *Efficient Pruning* our ideal choice for service allocation and fulfillment.

In Figure 7.12, we observe that *Naive FCFS* displays comparably larger number of partially-complete/incomplete service requests than *Exhaustive MnM* and *Efficient Pruning*. The algorithm does not consider distance of the CRU from the RSU and its resource capacity for service provisioning. It randomly allocates the first available CRU, which could be at a farther distance from the RSU. Therefore, it is unable to connect and communicate with the RSU to provide its resources for service fulfillment. The line graphs of figures 7.12, 7.13 and 7.14 for *Exhaustive MnM* and *Efficient Pruning* show a drop in the number of incomplete requests because as the vehicle density increases, there is some amount of congestion in the urban centre, reducing the mobility of the vehicles and therefore, allowing them to be in vicinity of the RSU to process the request. However, *Efficient Pruning* performs slightly better than *Exhaustive MnM* since it allocates CRUs based on their connectivity strength with the RSU, allowing majority vehicles to be in vicinity of the RSU to process requests.

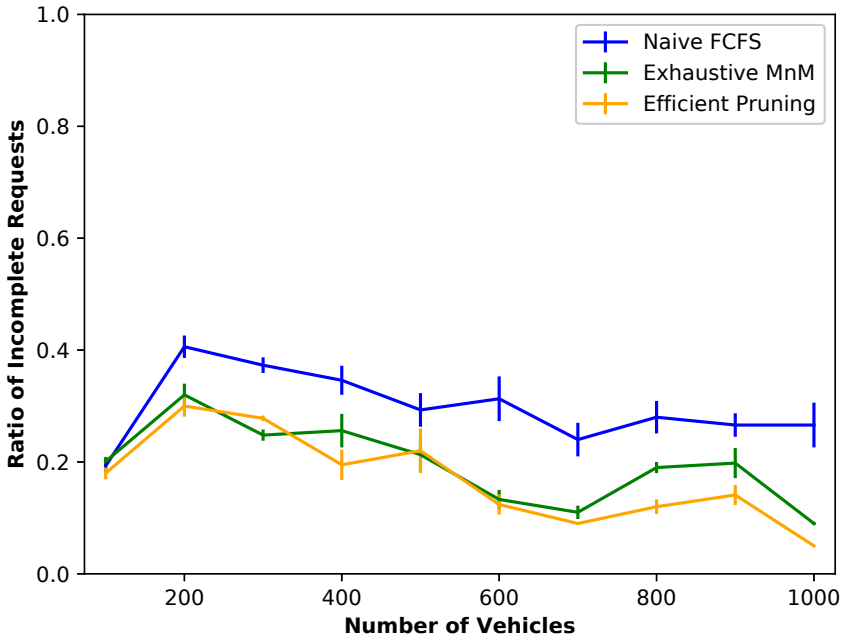

Figure 7.12: Ratio of Incomplete / Total Number of Requests = 30

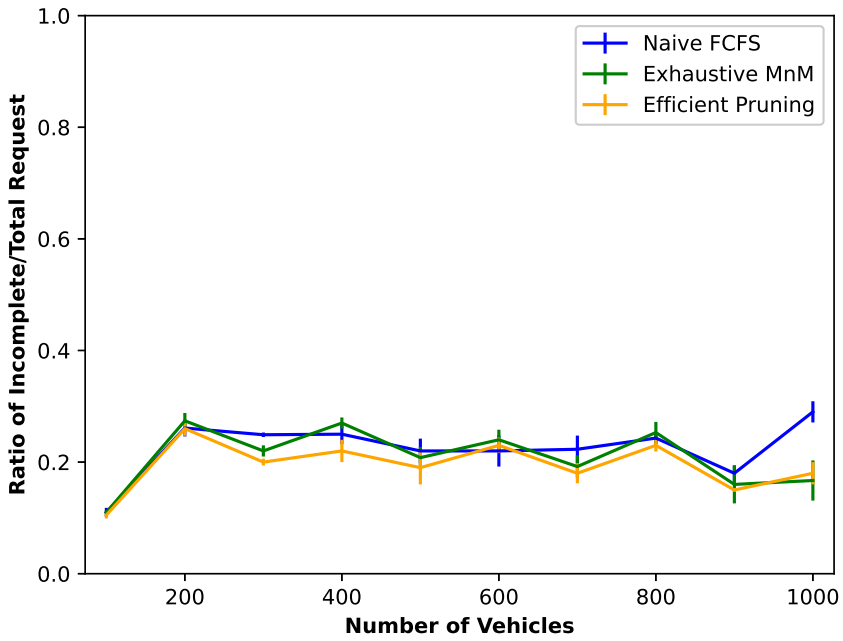

Figure 7.13: Ratio of Incomplete / Total Number of Requests = 50

At the lower vehicle density mark of 100 and 200, the ratio of incomplete requests remains unchanged for all three algorithms. For lower density vehicle scenarios, the congestion in the network is minimal, allowing the vehicles to freely move around. It results in most vehicles of assigned CRUs to quickly move out of range of the

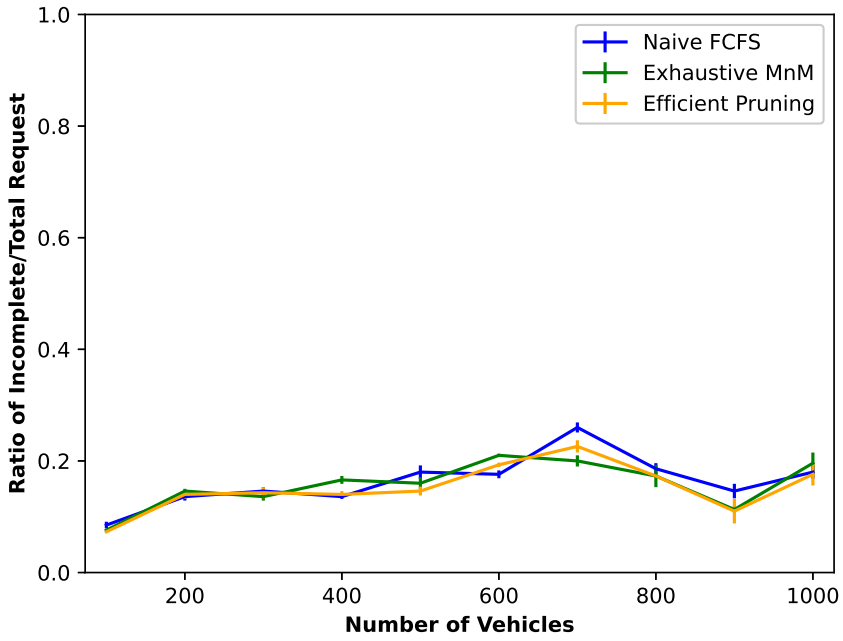

Figure 7.14: Ratio of Incomplete / Total Number of Requests = 100

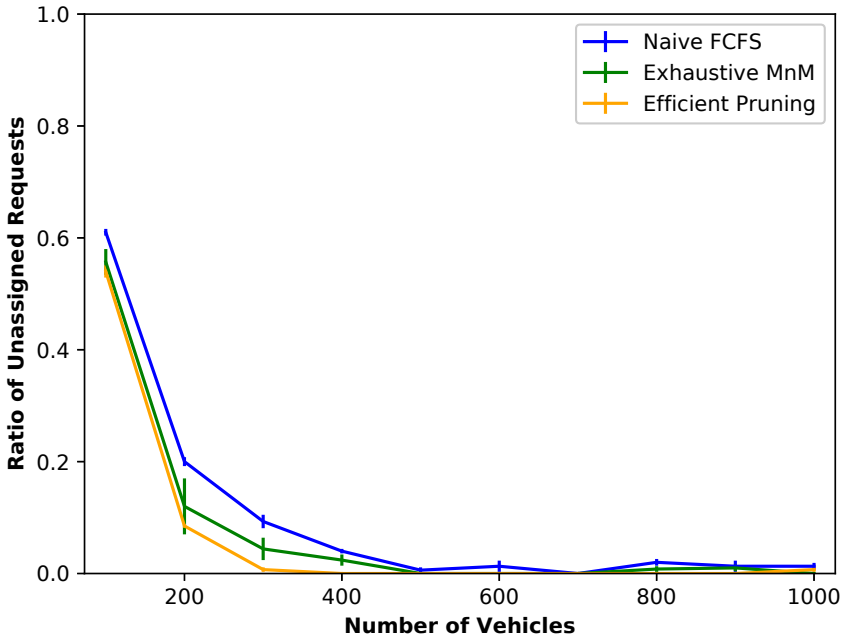

Figure 7.15: Ratio of Unassigned / Total Number of Requests = 30

RSU, leaving the request only partially processed. However, as the vehicle density increases, the vehicles of the assigned CRUs remain in range of the RSU, providing their resources for service fulfillment.

In Figure 7.15, it is observed that there is a significant drop in the number of unas-

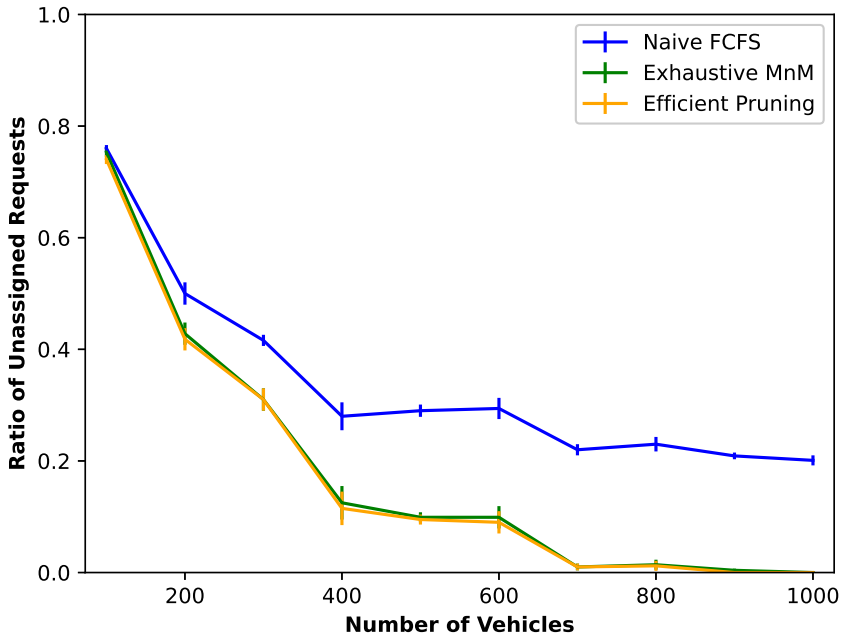

Figure 7.16: Ratio of Unassigned / Total Number of Requests = 50

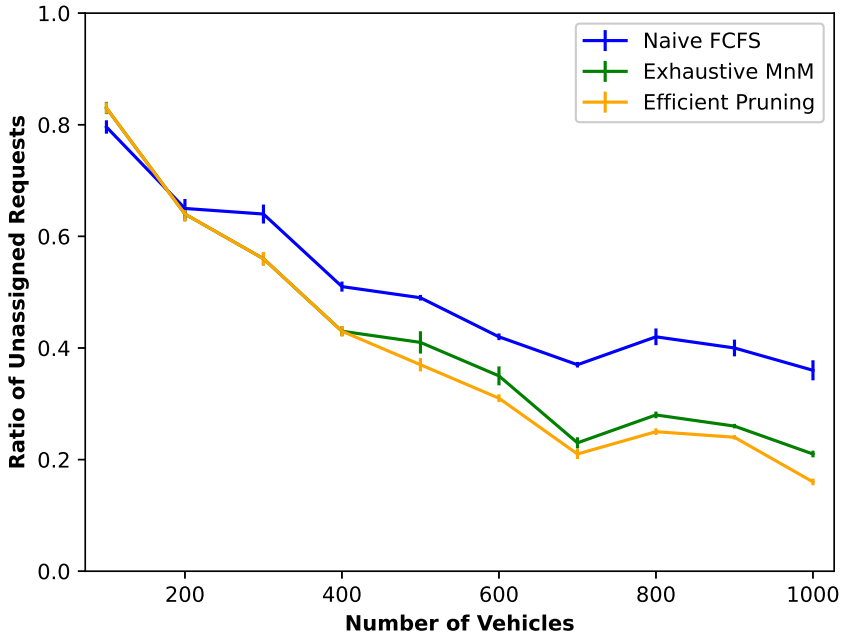

Figure 7.17: Ratio of Unassigned / Total Number of Requests = 100

signed requests until the vehicle density is 400 and then the graph becomes almost uniform for all three models. The CRU distribution heuristic allows the network to compose CRUs based on the number of vehicles in the network. Therefore, as the vehicle density increases, the total number of CRUs in the network also increases,

allowing sufficient service requests to be allocated. When closely observed, *Naive FCFS* presents 10% and 15% lower ratios than *Exhaustive MnM* and *Efficient Pruning*. The algorithm possibly allocates medium and large size CRUs to small size requests, leaving most medium and large size requests unassigned. Secondly, there are not sufficient small, medium or large size CRUs available to accommodate different capacity requests. On the other hand, *Exhaustive MnM* and *Efficient Pruning* display almost negligible number of unassigned service requests after the mark of 500-vehicle density.

As the number of service requests increase to 50 and 100 in Figures 7.16 and 7.17, respectively, there is a significant increase in the number of unassigned requests for the *Naive FCFS* model. As explained earlier, there are not sufficient CRUs available to accommodate requests on a first-come-first served basis. Additionally, there is a slight increase in the number of unacknowledged requests for low vehicle densities for the *Exhaustive MnM* and *Efficient Pruning* models, as compared to the 30-request scenario in Figure 7.15. As the number of service requests demand increases, there are fewer number of CRUs available for allocation. Such a case happens for requests that demand medium and large CRUs since the distribution heuristic estimates and composes fewer number of medium and large CRUs, to avoid congestion of the network.

The results of the three algorithms, as displayed in the graphs, confirmed the expected performance where *Efficient Pruning* can identify a combination of CRU allocations that matches a larger number of requests. The algorithm considers the connectivity of vehicles to the RSU and the number of resources a CRU offers to a request, resulting in minimal wastage of resources The *Exhaustive MnM* conducts a search that achieves an optimal result but with a longer time. Its lower performance compared with the *Efficient Pruning* approach occurs because pruning has a more realistic, communication-aware fitness function. This makes *Efficient Pruning* a connectivity-oriented algorithm. The only drawback of this model is that rarely, for some vehicle densities, this algorithm performs weaker than *Exhaustive MnM*. However, the several advantages of *Efficient Pruning* model overcome its only drawback, making it an ideal choice for conducting CRU allocation and service fulfillment in real-world scenarios. The worst performing model of the three is *Naive FCFS* since it does not consider the connectivity level and resource capacity of a CRU for allocation, resulting in vehicles being out of range of RSU and wastage of resources.

Chapter 8

Conclusion

This thesis has dealt with the resource allocation problem in VCC networks through a new relaxed approach, which introduces Combined Resource Units that aims at clustering resources of heterogeneous vehicles to minimize the number of underutilized resources, facilitating discoveries and assignments in a dynamic urban computing environment. Following CRU composition process, we have addressed the service allocation and fulfillment problem by allocating CRUs to service requests through a fair and efficient approach of game theory which implements the Efficient Pruning model. We conducted simulations to evaluate the proposed composition approach and allocation model.

8.1 Summary of Contributions

This work introduced the concept of clustering resources of heterogeneous vehicles into virtual units called Combined Resources Units. This approach helped us in identifying significant number of underutilized vehicles and their resources in the VCC network, and utilizing them for service provisioning. Through this approach, we configure our VCC network with three different pools of CRUs: small, medium and large, which helps us in minimizing the search time for matching CRU and request requirements and also maximizing the success rate of request fulfillment. By having different pools of CRUs, maximum number of service requests can be acknowledged and processed. We compared four different search algorithms for composing CRUs. We also established a distribution heuristic method which helped us identify the precise number of CRUs that a VCC network with certain vehicle density can safely compose, without congesting the network. The results of the distribution heuristic method displayed that *SSA*, *FASA* and *ELSA* performed adequately. Amongst the dif-

ferent search algorithms implemented, the *SSA* algorithm outperformed all the other algorithms. This algorithm resulted in composing least number of under-assembled and over-assembled CRUs, maximizing the use of available vehicle resources. This contribution has been accepted for publication at the IEEE ISCC-2021 conference, in the field of Vehicular Networks.

After conducting the CRU composition process, we allocate them for resource provisioning to service requests. We studied and compared the performance of three different models of allocation, by conducting experiments for different densities of vehicles and service requests. A Naive-FCFS model was defined as a baseline, fulfilling the minimum requirements of the CRU allocation process. However, this approach was inefficient since service requests are fulfilled on a first-come-first-served basis, resulting in wastage of resources. Next, we implemented an Exhaustive MnM model that is a recursive algorithm used in decision making scenarios. This approach calculates a payoff value for each CRU and assigns the one with highest payoff to a request. The payoff is calculated based on connectivity and resources provisioning factors. However, the connectivity of a CRU from RSU is calculated as a Euclidean distance, which is linear. Therefore, it is not considered reliable. Additionally, the Exhaustive MnM algorithm has a time complexity of $\mathcal{O}(b^d)$, making it inherently time-consuming since it loops through the list of all available CRUs, even though an ideal match has been identified.

To overcome the drawbacks of Exhaustive MnM algorithm, we adopted the Efficient Pruning algorithm for CRU allocation and service request fulfillment. The time complexity of this algorithm is $\mathcal{O}(\sqrt{b^d})$, minimizing the time consumed in searching for an ideal CRU for a service request. The utility value of this model is also calculated based on the factors of connectivity and resource provisioning. However, we define the connectivity level of a CRU in terms of its receiving signal strength indicator value. The CRU with a higher utility value is allocated to a service request for fulfillment.

There are some limitations of both the approaches proposed in this work:

- CRU Composition Approach. The current scenario pre-composes CRUs based on their predefined capacity. However, this could limit the fulfillment of excessively large sized service requests, maximizing the under-utilization of resources in the network for service fulfillment. Our current scenario has a significant number of incomplete/under-assembled CRUs, affecting the success rate of service fulfillment.

- CRU Allocation Approach. In our current scenario, a service request is allocated a single CRU for fulfillment. This impacts the success rate of fulfillment since the network may not have sufficient medium and large CRUs to fulfill large size requests.

In terms of our current simulation settings, the primary RSU is loaded with several responsibilities of triggering the start of CRU composition, assembling CRUs, implementing the distribution heuristic allocating CRUs to service requests and communicating with vehicles to provide their resources for service provisioning. This impacts the simulation time significantly, slowing the network and currently processing services.

8.2 Future Research Directions

Even though the proposed approaches are promising solutions to the resource management problem in VCC networks, it encounters several challenges, as defined above that can be addressed in future work.

The VCC network should be flexible to allow the composition of CRUs as and when it receives user requests. This flexibility should allow for maximum utilization of resources and minimal wastage of resources. Also, we could identify newer methods that minimize the number of incomplete CRUs to allow for a maximum success rate in terms of service fulfillment.

Instead of allowing only a single CRU to be allocated to a service request, we can make our approach flexible by allowing multiple CRUs to be allocated to a service request. This would also impact and significantly maximize the success rate of fulfillment, while minimizing the number of unassigned requests. The number of incomplete requests due to vehicles being out of range of CRU could be minimized by deploying multiple RSUs in the network that are connected to each other and can fulfill a request if the primary RSU is not in reach of vehicles.

We can introduce V2V communication in our work. This would allow vehicles to directly communicate with other vehicles, sharing information about their resource capacity and also helping fulfill service requests if CRU-assigned vehicles are not in range.

Since the RSU assembles CRUs following the distribution heuristic, we can prioritize service requests. Requests about road and traffic information can be prioritized over infotainment requests, ensuring sufficient CRUs are available to process priority requests first.

In our current scenario, at the start of simulation, we have a single RSU that broadcasts a message to all vehicles in range, requesting for their resources for composing CRUs. It is time-consuming since vehicles in range receive the message and respond back to the RSU, providing their resources. It would be interesting to have vehicles provide their resource information, to multiple RSUs, as soon as they arrive in the network. This would accelerate the CRU composition process, thus accelerating the processing of service requests.

Bibliography

[1] U. Acer, P. Giaccone, D. Hay, G. Neglia, and S. Tarapiah. Timely data delivery in a realistic bus network. In *2011 Proceedings IEEE INFOCOM*, pages 446–450, 2011.

[2] Saif Al-Sultan, Moath M Al-Doori, Ali H Al-Bayatti, and Hussien Zedan. A comprehensive survey on vehicular ad hoc network. *Journal of network and computer applications*, 37:380–392, 2014.

[3] Moayad Aloqaily, Ismaeel Al Ridhawi, Burak Kantraci, and Hussein T Mouftah. Vehicle as a resource for continuous service availability in smart cities. In *2017 IEEE 28th annual international symposium on personal, indoor, and mobile radio communications (PIMRC)*, pages 1–6. IEEE, 2017.

[4] Hamid Reza Arkian, Reza Ebrahimi Atani, Abolfazl Diyanat, and Atefe Pourkhalili. A cluster-based vehicular cloud architecture with learning-based resource management. *The Journal of Supercomputing*, 71(4):1401–1426, 2015.

[5] Azzedine Boukerche and Rodolfo I Meneguette. Vehicular cloud network: A new challenge for resource management based systems. In *2017 13th International Wireless Communications and Mobile Computing Conference (IWCMC)*, pages 159–164. IEEE, 2017.

[6] Azzedine Boukerche and E Robson. Vehicular cloud computing: Architectures, applications, and mobility. *Computer networks*, 135:171–189, 2018.

[7] Bouziane Brik, Nasreddine Lagraa, Nouredine Tamani, Abderrahmane Lakas, and Yacine Ghamri-Doudane. Renting out cloud services in mobile vehicular cloud. *IEEE Transactions on Vehicular Technology*, 67(10):9882–9895, 2018.

[8] Nicola Cordeschi, Danilo Amendola, and Enzo Baccarelli. Reliable adaptive resource management for cognitive cloud vehicular networks. *IEEE Transactions on Vehicular Technology*, 64(6):2528–2537, 2014.

[9] Nicola Cordeschi, Danilo Amendola, Mohammad Shojafar, and Enzo Baccarelli. Distributed and adaptive resource management in cloud-assisted cognitive radio vehicular networks with hard reliability guarantees. *Vehicular Communications*, 2(1):1–12, 2015.

[10] Wiseborn M Danquah and D Turgay Altilar. Vehicular cloud resource management, issues and challenges: A survey. *IEEE Access*, 8:180587–180607, 2020.

[11] D. Eckhoff, C. Sommer, R. German, and F. Dressler. Cooperative awareness at low vehicle densities: How parked cars can help see through buildings. In *2011 IEEE Global Telecommunications Conference - GLOBECOM 2011*, pages 1–6, 2011.

[12] Rayane El Sibaï, Talar Atéchian, Jacques Bou Abdo, Rami Tawil, and Jacques Demerjian. Connectivity-aware service provision in vehicular cloud. In *2015 International Conference on Cloud Technologies and Applications (CloudTech)*, pages 1–5. IEEE, 2015.

[13] Mark Felegyhazi and Jean-Pierre Hubaux. Game theory in wireless networks: A tutorial. 2006.

[14] Rim Gasmi and Makhlouf Aliouat. Vehicular ad hoc networks versus internet of vehicles-a comparative view. In *2019 International Conference on Networking and Advanced Systems (ICNAS)*, pages 1–6. IEEE, 2019.

[15] L. Gu, D. Zeng, and S. Guo. Vehicular cloud computing: A survey. In *2013 IEEE Globecom Workshops (GC Wkshps)*, pages 403–407. IEEE, 2013.

[16] Rasheed Hussain, Junggab Son, Hasoo Eun, Sangjin Kim, and Heekuck Oh. Rethinking vehicular communications: Merging vanet with cloud computing. In *4th IEEE International Conference on Cloud Computing Technology and Science Proceedings*, pages 606–609. IEEE, 2012.

[17] Chun-Cheng Lin, Der-Jiunn Deng, and Chia-Chi Yao. Resource allocation in vehicular cloud computing systems with heterogeneous vehicles and roadside units. *IEEE Internet of Things Journal*, 5(5):3692–3700, 2017.

[18] Pablo Alvarez Lopez, Michael Behrisch, Laura Bieker-Walz, Jakob Erdmann, Yun-Pang Flötteröd, Robert Hilbrich, Leonhard Lücken, Johannes Rummel, Peter Wagner, and Evamarie Wießner. Microscopic traffic simulation using sumo.

In *Proc. of the IEEE International Conference on Intelligent Transportation Systems*, pages 2575–2582, 2018.

[19] Rodolfo I Meneguette, Azzedine Boukerche, and Robson de Grande. Smart: an efficient resource search and management scheme for vehicular cloud-connected system. In *2016 IEEE Global Communications Conference (GLOBECOM)*, pages 1–6. IEEE, 2016.

[20] Prasant Mohanty, Lavitra Kumar, Madhuri Malakar, Suraj K Vishwakarma, and Motahar Reza. Dynamic resource allocation in vehicular cloud computing systems using game theoretic based algorithm. In *2018 Fifth International Conference on Parallel, Distributed and Grid Computing (PDGC)*, pages 476–481. IEEE, 2018.

[21] Ahmad M Mustafa, Omar M Abubakr, Omar Ahmadien, Ahmed Ahmedin, and Bassem Mokhtar. Mobility prediction for efficient resources management in vehicular cloud computing. In *2017 5th IEEE International Conference on Mobile Cloud Computing, Services, and Engineering (MobileCloud)*, pages 53–59. IEEE, 2017.

[22] Stephan Olariu, Ismail Khalil, and Mahmoud Abuelela. Taking vanet to the clouds. *International Journal of Pervasive Computing and Communications*, 2011.

[23] Christoph Sommer, Reinhard German, and Falko Dressler. Bidirectionally Coupled Network and Road Traffic Simulation for Improved IVC Analysis. *IEEE Trans. on Mobile Computing (TMC)*, 10(1):3–15, 2011.

[24] Jun Tao, Ziyi Zhang, Fuqin Feng, Jian He, and Yifan Xu. Non-cooperative resource allocation scheme for data access in vanet cloud environment. In *2015 Third International Conference on Advanced Cloud and Big Data*, pages 190–196. IEEE, 2015.

[25] András Varga and Rudolf Hornig. An overview of the omnet++ simulation environment. In *Proc. of the ICST International Conference on Simulation Tools and Techniques for Communications, Networks and Systems & Workshops*, 2008.

[26] Md Whaiduzzaman, Mehdi Sookhak, Abdullah Gani, and Rajkumar Buyya. A survey on vehicular cloud computing. *Journal of Network and Computer Applications*, 40:325–344, 2014.

[27] Kai Xiong, Supeng Leng, Jie Hu, Xiaosha Chen, and Kun Yang. Smart network slicing for vehicular fog-rans. *IEEE Transactions on Vehicular Technology*, 68(4):3075–3085, 2019.

[28] D. Yang, G. Xue, X. Fang, and J. Tang. Incentive mechanisms for crowdsensing: Crowdsourcing with smartphones. *IEEE/ACM Transactions on Networking*, 24(3):1732–1744, 2016.

[29] Rong Yu, Xumin Huang, Jiawen Kang, Jiefei Ding, Sabita Maharjan, Stein Gjessing, and Yan Zhang. Cooperative resource management in cloud-enabled vehicular networks. *IEEE Transactions on Industrial Electronics*, 62(12):7938–7951, 2015.

[30] Rong Yu, Yan Zhang, Stein Gjessing, Wenlong Xia, and Kun Yang. Toward cloud-based vehicular networks with efficient resource management. *IEEE Network*, 27(5):48–55, 2013.

[31] Kan Zheng, Hanlin Meng, Periklis Chatzimisios, Lei Lei, and Xuemin Shen. An smdp-based resource allocation in vehicular cloud computing systems. *IEEE Transactions on Industrial Electronics*, 62(12):7920–7928, 2015.

[32] Y. Zhu, X. Liu, M. Li, and Q. Zhang. Pova: Traffic light sensing with probe vehicles. *IEEE Transactions on Parallel and Distributed Systems*, 24(7):1390–1400, 2013.