Virtual Machine Management for Dynamic Vehicular Clouds

by

Tarek Refaat

Thesis submitted to the

Faculty of Graduate and Postdoctoral Studies

in partial fulfillment of the requirements

for the Ph.D. degree in Electrical and Computer Engineering

School of Electrical Engineering and Computer Science

Faculty of Engineering

University of Ottawa

Dedication

To my Father, the selfless role model, whose pride in me drives me ever higher;

my Mother, the silent hero, whose endless faith in me keeps me believing;

my sister Noha, the very embodiment of altruism, excellence, and humility combined;

my brother Seif, truly my best friend, my confidant, and partner-in-crime;

my sister Hana, the very definition of a kind soul, a sensitive heart and a source of joy;

most affectionately, to my beloved wife Samar,
who stood by my side during my happiest and my darkest times,
believing in me when even I had lost hope;

my 'other brother' Karam, who has contributed to this thesis, and enriched my life…
in more ways than he knows;

Profs. Hassanein H. Amer & Ramez M. Daoud, without whom I wouldn't be here.

Finally, I must honor all my Skype and long-distance supporters: Adam Amin, Chris Scoufaridis, Mayada Serageldin, Omar Khaled, Yara Enany, Yomna ElFaramawy, Ziyad Omar, and the AUC Whatsapp group (you know who you are)!

Abstract

Vehicular clouds involve a dynamic environment where virtual machines are hosted on moving vehicles, leading to frequent changes in the data center network topology. These frequent topological changes require frequent virtual machine migrations in order to meet the service level agreements with cloud users. Such topology changes include fluctuations in connectivity, signal strength and quality. Few studies address vehicles as potential virtual machine hosts, while there is a significant opportunity in capitalizing on underutilized resources. Due to the rapidly changing environment of a vehicular cloud, hosts frequently change or leave coverage. As such, virtual machine management and migration schemes are necessary to ensure cloud subscribers have a satisfactory level of access to the resources. This thesis addresses the need for virtual machine management for the vehicular cloud. First, a mobility model is proposed and utilized to test a set of novel Vehicular Virtual Machine Migration (VVMM) schemes: VVMM-U (Uniform), VVMM-LW (Least Workload), VVMM-MA (Mobility Aware) and MDWLAM (Mobility and Destination Workload Aware Migration). Their performance is evaluated with respect to a set of metrics through simulations with varying levels of vehicular traffic congestion, virtual machine sizes and load restriction levels. The most advanced scheme (MDWLAM) takes into account the workload and mobility of the original host as well as those of the potential destinations. By doing so a valid destination will both have time to receive the workload and migrate the new load when necessary. The behavior of various algorithms is compared and the MDWLAM has been shown to demonstrate the best performance, exhibiting migration drop rates that are negligibly small. Finally, an integer linear program formulation based on a modified single source shortest path problem is presented, tested and successfully shown to be a benchmark that can be used in comparison to the proposed heuristics.

Acknowledgements

I wish to acknowledge all those that supported me along my path to this point:

Prof. Hussein T. Mouftah, for giving me the opportunity, for his faith in me, his endless support, patience and guidance, Prof. Burak Kantarci for his generous mentorship, constant guidance and immeasurable assistance, the examiners for their time and effort in evaluating my work, and last but not least, Dr. Mahmoud Gad for his assistance and moral support.

Table of Contents

Dedication ii

Abstract iii

Acknowledgements iv

List of Figures x

List of Tables xii

List of Acronyms xiii

List of Symbols xv

Chapter 1 Introduction 1

1.1 Background 1

1.2 Virtualization 3

1.3 Motivation 5

1.4 Objectives 6

1.5 Contributions 8

1.6 Thesis Outline 9

1.7 List of Publications 10

Chapter 2 Virtual Machine Migration and Vehicular Clouds: State-of-the-Art 11

2.1 Introduction 11

2.2 Virtual Machine Migration and Management: Overview 13

2.2.1 Virtual Machine Migration Requirements 13

2.2.2 Virtual Machine Migration Schemes 15

2.2.3 Traditional Virtual Machine Migration 16

2.2.4 Volume-based Migration Scheme 18

2.2.5 Migration of Co-located Virtual Machines 20

2.2.6 Virtual Machine Migration in Dynamic Clouds 22

2.3 Vehicular Cloud State-of-the-Art 24

2.3.1 VC Architecture and VM Migration Schemes 27

2.3.2 State-of-the-Art VC Studies 29

2.4 Discussion 31

2.5 Obstacles and Gaps in Existing Studies 32

2.6 Summary 33

Chapter 3 Vehicular Virtual Machine Migration 35

3.1 Introduction 35

3.2 Simplified Mobility Model 37

3.2.1 MATLAB and Revised Simplified Mobility Model 40

3.2.2 Virtual Machine Workload Initialization 45

3.3 Vehicular Virtual Machine Migration – General Scheme Description 46

3.4 Detailed Scheme Description 49

3.5 Summary 52

Chapter 4 Preliminary Analysis, Refined Mobility Model and Advanced Migration Schemes 54

4.1 Introduction 54

4.2 Metrics and Simulation Results 56

4.3 Addressing Previous Assumptions 61

4.3.1 Revised Grid Topology and Advanced Mobility Model 62

4.3.2 Advanced Workload Initialization and Distribution 65

4.4 Refined and Expanded Vehicular Virtual Machine Migration Schemes 65

4.5 Detailed Description of Revised Vehicular Virtual Machine Migration Schemes 69

4.6 Conclusion 72

Chapter 5 Analysis of Advanced Schemes Incorporating Non-Ideal Communication. 74

5.1 Introduction 74

5.2 Detailed Performance Evaluation 75

5.2.1 Simulation Settings and Metrics 75

5.3 Simulation Results 80

5.3.1 Effect of Varying Congestion Levels and VM Size on the Percentage of Dropped Migrations ($\boldsymbol{D}$) 81

5.3.2 Effect of Varying Congestion Levels and VM Size on the Percentage of V2I migrations ($\boldsymbol{V}$) 84

5.3.3 Effect of Varying Congestion Levels and VM Size on the Average Percentage Utilization of Collective Available Resources (***R***) 86

5.3.4 Effect of Varying Congestion Levels and VM Size on the Average Percentage Utilization of Available Bandwidth 88

5.4 Non-ideal Communication and Time Safety Factors 92

5.4.1 Simulation Results and Analysis 93

5.5 Conclusion 94

Chapter 6 Multiple-Instance Single Source Shortest Path Formulation 96

6.1 Introduction 96

6.2 Potential Optimization Problems 97

6.3 Shortest Path Problem 98

6.4 Multiple-Instance Single Source Shortest Path Problem Formulation 99

6.4.1 Objective Function and Constraints 104

6.4.2 ILP Metrics 110

6.4.3 ILP Preliminary Runs 111

6.5 Comparing to Heuristics 116

6.6 Summary 119

Chapter 7 Conclusions 120

7.1 Concluding Remarks 120

7.2 Future Research 123

References 125

Appendix A Confidence Analysis 142

Appendix B Elsevier Rights for Reuse 145

List of Figures

Figure 2.1: Migration Stages [CLA05] 17

Figure 2.2: De-duplication in Live gang migration [DES11] 21

Figure 2.3: Vehicular Cloud Overview [WHA14] 24

Figure 3.1: Modeled Grid Layout 38

Figure 3.2: Trajectory Setup Process 42

Figure 3.3:– VM Initialization Stages 46

Figure 3.4:– General VVMM Algorithm 49

Figure 4.1: Revised Architecture for a Vehicular Cloud. 56

Figure 4.2: Average Percentage of Dropped VMs 59

Figure 4.3: Percentage Improvement vs. VVMM-U 59

Figure 4.4: Average Utilization Fairness over 30 runs 60

Figure 4.5: Average Utilization Fairness for a single Iteration 61

Figure 4.6: The layout of the revised grid 63

Figure 4.7: Algorithm 2 - General VM Migration Pseudocode 68

Figure 5.1: Percentage of dropped migrations vs. Congestion – 10MB VM Size 82

Figure 5.2: Percentage of dropped migrations vs. Congestion – 25MB VM Size 83

Figure 5.3: Percentage of dropped migrations vs. Congestion – 50MB VM Size 83

Figure 5.4: Percentage of V2I migrations vs. Congestion – 10MB VM Size 84

Figure 5.5: Percentage of V2I migrations vs. Congestion – 25MB VM Size 85

Figure 5.6: Percentage of V2I migrations vs. Congestion – 50MB VM Size 85

Figure 5.7: Average percentage resource utilization vs. Congestion – 10MB VM Size 86

Figure 5.8: Average percentage resource utilization vs. Congestion – 25MB VM Size 87

Figure 5.9: Average percentage resource utilization vs. Congestion – 50MB VM Size 87

Figure 5.10: Average percentage utilization upload bandwidth vs. Congestion – 10MB VM Size..... 88

Figure 5.11: Average percentage utilization upload bandwidth vs. Congestion – 25MB VM Size..... 89

Figure 5.12: Average percentage upload bandwidth vs. Congestion – 50MB VM Size 89

Figure 5.13: Average percentage utilization download bandwidth vs. Congestion – 10MB VM Size..... 90

Figure 5.14: Average percentage utilization download bandwidth vs. Congestion – 25MB VM Size..... 90

Figure 5.15: Average percentage utilization download bandwidth vs. Congestion – 50MB VM Size..... 91

Figure 6.1: Example of Single Source Shortest Path Graph Formulation 101

Figure 6.2: Example of Multiple Instance Single Source Shortest Path Graph Formulation 103

Figure 6.3: Visual Representation of Reasoning behind Selecting Edge Cost Values 107

Figure 6.4: MISSSP Graph based on 1st Test and Solution 114

Figure 6.5: MISSSP Graph based on 2nd Test and Solution 115

Figure 6.6: Percentage of dropped migrations vs. Congestion – 50MB VM Size..... 118

Figure 6.7: Average percentage utilization upload bandwidth vs. Congestion – 50MB VM Size..... 118

Figure 6.8: Average percentage utilization download bandwidth vs. Congestion – 50MB VM Size..... 119

List of Tables

Table 2.1 – Comparison between Conventional Clouds and Vehicular Clouds 26

Table 2.2 – Vehicular Cloud Applications and VM Hosts 28

Table 2.3 – Suitability Analysis of Existing VM Management Schemes for the VC..... 31

Table 2.4 – Traditional Cloud VM Migration for the VC – Issues and Potential Solutions.... 33

Table 3.1 – Possible Vehicle Trajectories..... 39

Table 5.1 – Simulation Settings 79

Table 5.2 – Simulated Scenarios 80

Table 5.3 – Simulation Results for Non-Ideal Conditions with Safety Factor – Light Congestion, 25MB VM Size 94

Table 6.1 – Vehicle Status at each Time for the 1st Test 112

Table 6.2 – Vehicle Status at each Time for the 2nd Test 112

Table 6.3 – VM host at Time = 0 112

Table 6.4 – ILP Solutions for the 1st and 2nd Test 113

List of Acronyms

APSP	All Pair Shortest Path
CaaS	Cooperation-as-a-Service
CC	Cloud Computing
CompaaS	Computation-as-a-Service
DES	Discrete Event Simulation
ENaaS	Entertainment-as-a-Service
FTP	File Transfer Protocol
GVCC	Generalized Vehicular Cloud Controller
HDD	Hard Disk Drive
I2V	Infrastructure-to-Vehicle
IaaS	Infrastructure-as-a-Service
INaaS	Information-as-a-Service
IoT	Internet of Things
ILP	Integer Linear Program
IP	Internet Protocol
ITS	Intelligent Transportation System
LTE	Long Term Evolution
M2M	Machine-to-Machine
MDWLAM	Mobility and Destination Workload Aware Migration
MSSP	Multiple Source Shortest Path
MISSSP	Multiple Instance Single Source Shortest Path
NaaS	Network-as-a-Service
NP	Non-Polynomial
PaaS	Platform-as-a-Service
PHY layer	Physical Layer
RC	Roadside Cloud
RU	Roadside Unit
SaaS	Software-as-a-Service

SLA	Service-Level Agreement
SSSP	Single Source Shortest Path
STaaS	Storage-as-a-Service
SVCC	Specified Vehicular Cloud Controller
TCP/IP	Transmission Control Protocol/Internet Protocol
UDP	User Datagram Protocol
V2I	Vehicle-to-Infrastructure
V2V	Vehicle-to-Vehicle
VANET	Vehicular Ad-Hoc Network
VC	Vehicular Cloud
VCC	Vehicular Cloud Computing
VM	Virtual Machine
VMM	Virtual Machine Manager
VSR	Volume-to-Size Ratio
VVMM	Vehicular Virtual Machine Migration
VVMM-LW	VVMM-Least Workload
VVMM-MA	VVMM-Mobility Aware
VVMM-U	VVMM-Uniform

List of Symbols

A_{PD}	Total percentage dropped
A_{UF}	Average Utilization Fairness
B_{avg}	Average bandwidth available for an individual VM
B_{max}	Maximum bandwidth capacity of the grid
BW_{vijt}	Bandwidth value for the transition from vertex c_{it}, vehicle i at time t, to $c_{j(t+1)}$, vehicle j at time $t+1$ (the next column in the graph), for VM v
c_{it}	Vertex for vehicle i at time t
CQ_t	Normalized connection quality at time t
C_{src}	Source vehicle for VM v
D	Drop row in MISSSP grid
D_c	Cost of a drop (transitioning to the drop row D at time $t+1$)
D_i	i^{th} dropped migration value (VM units)
DL_{BW}	Percentage of downlink bandwidth utilization for MISSSP
dl_t	Total download bandwidth utilized, at time t, throughout the entire grid (MB/s)
dl_m	Maximum download bandwidth available in the grid (MB/s)
$Drops$	Drop percentage for MISSSP
d_t	Total dropped VM workload, at time t in the grid (MB)
e_{vijt}	Cost of transitioning from vertex c_{it}, vehicle i at time t, to $c_{j(t+1)}$, vehicle j at time $t+1$ (the next column in the graph), for VM v
I	Infrastructure row in MISSSP grid
I_c	Cost of migrating to the infrastructure, row I at time $t+1$
Int_t	Normalized interference due to concurrent migrations at time t
l_m	Maximum VM load capacity the grid can offer (MB/s)
L_{max}	Maximum load capacity of a vehicle
l_t	Total VM load hosted on all the vehicles currently the grid, at time t (MB)
L_{vijt}	Load from vertex c_{it}, vehicle i at time t, to $c_{j(t+1)}$, vehicle j at time $t+1$ (the next column in the graph), for VM v

M_c	Cost of migration from a vehicle to another vehicle
P_{max}	Maximum value for P_t (set at 0.5)
P_t	Probability, at time t, a packet is dropped due to interference or connection instability
r_e	Rate of entry (cars)
SD_m	Maximum value for SD_t, the diagonal distance from one corner of the grid to the other
SD_t	Distance between source and destination
s_t	Total successfully migrated VM load, at time t, throughout the entire grid (MB)
T	Simulated/observed time (s)
T_d	Time needed to migrate destination workload (s)
T_m	Total number of migrations
T_s	Time needed to migrate source workload (s)
U_D	Unifying Destination
UL_{BW}	Percentage of uplink bandwidth utilization for MISSSP
ul_m	Maximum upload bandwidth available in the grid (MB/s)
ul_t	Total upload bandwidth utilized, at time t, throughout the entire grid (MB/s)
$V2I$	Percentage of infrastructure migrations for MISSSP
v_t	Total VM load, successfully migrated to the infrastructure, at time t (MB)
WL_{ti}	i^{th} car's workload (VM units) at time t
x_{vijt}	ILP decision variable for edge from vertex c_{it}, vehicle i at time t, to $c_{j(t+1)}$, vehicle j at time $t+1$ (the next column in the graph), for VM v

Chapter 1

Introduction

1.1 Background

In the past decade, the field of telecommunications and specifically cloud computing have and continue to experience significant advancements, from the research and development stages to implementation. Handheld devices becoming more powerful, paired with increased affordability, has brought the Internet out of the offices and libraries and into every aspect of our surrounding environment. This has warranted the coining of terms such as Internet of Things (IoT), Machine-to-Machine (M2M), among others [HWA11].

The future of computer systems will rely heavily on Cloud Computing (CC). It is important to explain the concepts of CC and existing implementations to clarify what this thesis will address. Cloud computing is, in a broad sense, the utilization of resources that are not located locally. The resources are connected via a network (wired/wireless/both) and the user/client accesses these

resources based on certain parameters. The key advantage here is that users/clients pay for resources, only when they need them, as opposed to investing money in a similar resource (in a high-end computer for example) and not fully utilizing said resource, whether in terms of capabilities or in terms of running time. The average consumer (let alone companies and institutions) invests significant resources in hardware and software that is rarely fully utilized. The cloud concept facilitates sharing of resources by utilizing an on-demand philosophy. The user will only pay for the hardware/software they need, and only when they actually utilize it [VOO10]. The user experiences the illusion of unbounded resources, the pay-on-demand freedom as well as the control of how high or low the level of involvement is from his/her side. This is facilitated when selecting which type of cloud to use. In general, there are three levels of abstraction in cloud computing: Infrastructure-as-a-Service (IaaS), Platform-as-a-Service (PaaS), Software-as-a-Service (SaaS).

The levels of abstraction determine at which level the user/client interacts with the resources. For example, IaaS enables the client to access resources almost directly, with the cloud offering its resources with minimal software or intermediaries between the client and the hardware. The hardware (or resource) offered include storage, computing power among others. The higher level of abstraction offered in PaaS presents a framework on top of the resources for programming and application development. This level of abstraction is focused on developers rather than average customers. Finally, SaaS is the highest level of abstraction that offers the client everything from the hardware to the actual software to be run. The client merely utilizes a network interface and a browser to access the software [VOO10]. There are other levels of abstraction defined in various studies, but these shall be introduced later.

Cloud deployments can be Public, Private or Hybrid. Public clouds are resources that are accessible by clients which are not restricted in any way. Private clouds are resources that are managed by a certain organization/institution and access is limited to specific clients. A hybrid cloud is a pool of resources that come from both public and private clouds. Clients accessing the cloud operate require some guarantee that the resources they request will be available. This is achieved by the Service Level Agreement (SLA). The SLA contains parameters including the requested resource capabilities, the preferred activity time-frames, and the likelihood of resource availability during the preferred time-frame and outside of it [BOS10]. The cloud service providers are committed (legally bound) to meet the SLA parameters as the client has/will be paying for the service. If an SLA violation occurs (or may occur), the service provider must rectify the situation or compensate the client. The client is not concerned with where, which and how he/she accesses the resources, as long as they are available and meet the SLA parameters. This is achieved by virtualization, which is explained next.

1.2 Virtualization

Virtualization is a key aspect of Cloud Computing. Virtualization in essence is the treatment of resources as a black-box, offering access via a Virtual Machine (VM). A VM is typically an operating system hosted within a virtual environment, where the hardware is simulated by software. An example of such a case is running an instance of Linux within a Windows operating system, which is installed normally on a computer. The hardware hosting the Linux image is actually simulated virtually by the software in the Windows operating system, such as VMware [MIC14], [VM14]. The Windows XP VM is oblivious to it being run in a virtual environment and the resources allotted to it can be controlled by the main operating system (in this case

Windows 7). In cloud computing, there is a Virtual Machine Manager (VMM) or Hypervisor, which is a light operating system (or middle ware) which is installed on the hardware pool. This hypervisor is merely the controlling entity, managing the resources allotted to the VM (or VMs) running on the hardware. Each VM is completely oblivious of its neighbors (even though they are sharing the same resource) and only perceive the resources that they are allotted. If a VM is allotted 50% of the actual hardware capability, it is completely oblivious to the existence of any surplus hardware. The hypervisor is the software that is aware (and in control) of the full hardware pool [RIM09].

A VM goes through a certain life-cycle:

1. A request is made to the resource administrator.
2. The administrator checks resource availability, and loads, configures then starts the VM.
3. VM in operation.
4. Release of resources once (if) task is complete.

The provisioning process that occurs before running the VM involves selecting a server from the hardware pool. The server needs to have enough capacity. Appropriate software, drivers and image are loaded. The machine needs to be configured and then the VM can be started [ELR10]. The VM can be manually installed or a preconfigured VM, a clone VM or an imported VM can be used. The preconfigured or clone VMs are preferred, in order to minimize delays. For several reasons and in multiple techniques, VMs may need (and often are) to be migrated from one physical machine to another. This will be the main focus of this thesis and is discussed in further detail in what follows.

1.3 Motivation

Cloud Computing and Vehicular Ad-Hoc Networks (VANETs) are heavily researched fields. Researchers foresee CC as the gateway to the future paradigm, the IoT [HWA11]. The IoT is a vision where billions of uniquely identifiable objects are connected to the Internet. The more intelligence and networking capabilities a device has, the more functionality it can offer [GUB13], [HAM14]. Vehicles, due to the research in the field of VANETS and major automotive manufacturers, are rapidly becoming mobile computers [WAN14]. The average vehicle, at this time, is equipped with storage, computing and often networking capabilities, while less complex/powerful than a dedicated personal computer. Much of the time, these resources are underutilized [JAM12], [WHA14]. As such, these resources can be considered a potential pool of physical machines to serve a cloud computing deployment. The future of computing is gradually being migrated toward the cloud where connected vehicles in a VANET can be considered as a mobile data center with limited processing and storage capacities in comparison to the conventional cloud data centers. The field of VANETS is well studied in the literature [CHE14], [CHO11], [LAN08]. [MAR11], [SOM08], [TOR12]. The linking of cloud computing, VANETs and Intelligent Transportation Systems (ITS), can be found in the Vehicular Cloud (VC) [BIT12], [HUS12], [JAW11], [MER13A], [MER13B], [MER13C], [ZAB14]. The VC is a main convergence point for this thesis.

The VC with vehicles as physical hosts for the cloud is still a new concept, and as will be shown in the literature, few studies have addressed the management of virtual machines with the vehicular cloud. The resources available in a VC being utilized can greatly decrease the resource-hungry needs of the conventional cloud. Furthermore, the VC is possibly the more

demanding of dynamic clouds, and findings for the VC can be extended to generally dynamic clouds. This is the main focus of the thesis as it is a key determining factor in the feasibility of VC implementations.

There are various related cloud types such as fog computing or mobile edge computing. These forms of cloud computing are not necessarily alternatives but other potentially parallel approaches. The VC can be considered a sub-form of fog computing, whereby vehicles can handle a part of the cloud implementation, paired with the conventional cloud. The advantages of the VC versus fog computing using other mobile devices (phones/laptops) include that vehicles are far less utilized than mobile devices, thereby not draining processing/storage resources that such devices are hungry for, let alone the precious battery life, which is not an issue for vehicles.

Users can be incentivized to lease their resources in exchange for various forms of compensation, such as reductions in cellular data bills, credit for refueling/carwash services, credits for the cloud administrator's services/products, or possibly discounts for various in-vehicle infotainment services.

1.4 Objectives

An essential functionality in conventional cloud computing deployments is VM migration and management. However, VM management methods in VCs are relatively few. It is important to study conventional CC VM management methods to assess their feasibility for the VC. It is important to note three main differences between a conventional and a vehicular cloud: 1) VC consists of a data center network with mobile hosts rather than fixed servers, 2) VC network topology changes rapidly and frequently, in stark contrast to conventional clouds, 3) Hosts in the

VC have limited computation and storage capacity in comparison to the hosts in conventional cloud data centers. Due to these main differences, existing VM management methods must be analyzed in terms of suitability for the VC, to determine potential modifications needed to make said methods suitable. These modifications and possibly new VM migration techniques for VCs will be the goal of this thesis.

There are currently few studies which address vehicles as virtual machine hosts for the vehicular cloud. As such, this thesis aims to tackle a virtual machine management and migration scheme for vehicular clouds with vehicular VM hosts. Efficient resource management will also be a central aspect in the research. The objective of this thesis is to propose, test and refine a VM management system for the VC. The following list summarizes the breakdown of objectives for this thesis:

1. Develop a mobility model for VM migration scheme testing,
2. Propose and test VM migration schemes,
3. Evaluate scheme performance using various metrics,
4. Re-evaluate performance while incorporating limitations of existing communication protocols, testing non-ideal channel conditions and potentially revising algorithms,
5. Compare performance with optimized theoretical performance benchmark.

The scope of the thesis is limited to the above contributions, but it does not cover aspects including:

1. Processing and pre-migration communication delays,
2. Realistic modeling of the channel, medium contention, fading/shadowing,
3. Communication/network users outside the VC,

4. Communication overhead due to subscribers using the VC,
5. Security and privacy of the VC,

These limitations are not entirely disregarded, such as non-ideal communication conditions, network congestion and minimizing the bandwidth utilization with the goal of leaving as much bandwidth available for primary users of the cloud. It is also important to note that there are several assumptions, mentioned throughout the thesis, that can also be considered limitations of this study, and removing said assumptions is important in future research.

1.5 Contributions

This thesis has several main contributions:

1. Designing a coded mobility model to reflect an urban setting including: varying levels of congestion, vehicles moving at various speeds with a realistic mobility pattern, traffic lights, and temporarily parked vehicles.
2. Proposing and testing several novel virtual machine migration algorithms for the vehicular cloud using the mobility model including: a benchmark algorithm for performance comparison, consecutively adaptive and intelligent algorithms,
3. Incorporating and analyzing limitations of an existing wireless communication protocol Long-Term Evolution (LTE) to observe the impact of network congestion and communication imperfections,
4. Proposing and testing a novel way to utilize the Single Source Shortest Path (SSSP) problem for generating an Integer Linear Programming (ILP) formulation of the vehicular

virtual machine migration problem. This is then compared with the previously proposed heuristic algorithms and showing it as the new benchmark for performance.

The findings of this thesis can be further extended to general mobile clouds and has potential to be expanded upon in several ways, such as incorporating the use of a network modeler or exploring multi-destination migrations for resilience. The future research is outlined in the final chapter.

1.6 Thesis Outline

The remainder of this thesis is organized as follows: Chapter 2 presents a survey of the state-of-the-art of virtual machine migration schemes for the conventional and the vehicular cloud. Chapter 3 presents the preliminary contributions of this thesis: the simplified mobility model and the preliminary vehicular virtual machine migration schemes. The simulation metrics and results of the preliminary contributions are presented in Chapter 4, followed by the refined mobility model and the updated algorithms. Chapter 5 presents the simulation metrics used to analyze the advanced algorithms in the refined mobility model, as well as presenting and simulating the non-ideal communication scenario. Chapter 6 presents an Integer Linear Program formulation of the problem for comparison with the heuristics. Finally, Chapter 7 discusses future work and the conclusions of this thesis.

1.7 List of Publications

1. T. K. Refaat, B. Kantarci, H. T. Mouftah, "Optimal Virtual Machine Migrations in Vehicular Clouds," *Submitted for review in the IEEE Transactions on Emerging Topics in Computing/Special Issue on Next Generation Wireless Computing Systems*, March 2017.
2. T. K. Refaat, B. Kantarci and H. T. Mouftah, "Virtual Machine Migration and Management for a Vehicular Cloud," *Elsevier Journal on Vehicular Communications*, vol. 4, pp. 47-56, April 2016.
3. T. K. Refaat, B. Kantarci and H. T. Mouftah, "Dynamic Virtual Machine Migration in a Vehicular Cloud," *Proceedings of the IEEE Symposium on Computers and Communication (ISCC)*, MoCS3.2.1-MoCS3.2.6, Madeira, Portugal, June 2014.
4. T. K. Refaat and H. T. Mouftah, "Future Applications of the Internet of Things with Resilience," *Poster presented at the 5th Annual WiSense Workshop*, Ottawa, ON, 2013.

Chapter 2

Virtual Machine Migration and Vehicular Clouds: State-of-the-Art

2.1 Introduction

In the Cloud Computing context, the word ‘dynamic’ typically describes the cloud subscriber as opposed to the (static) cloud resources. An example of this includes users accessing cloud-based storage via his/her laptop or mobile phone/tablet. The concept of a dynamic resource remains quite novel. As presented earlier, typical cloud resources are hosted in static data centers. The paradigm of a dynamic resource for cloud services is appealing due to the ubiquitous nature of hardware with networking capabilities in the present day. Rarely are cloud-friendly devices available without a near-constant connection to the Internet. It is a logical step to capitalize on such abundance of hardware to further reduce the costs burdening cloud providers.

Utilizing mobile resources rather than static physical resources introduces a range of potential issues that would impede the effectiveness of such an implementation. These issues include:

- Unpredictable mobility patterns – devices moving (to some extent randomly) are difficult to manage,
- Resource availability – due to the mobility of the devices and the often battery-powered nature of said devices, network coverage and battery life can impact resource availability,
- Network congestion – dynamic resources would not necessarily be utilizing a dedicated network, and are susceptible to congestion in shared networks,
- Connection issues – wireless protocols would be the more appropriate choice for dynamic resources, potentially exposing the resources to communication channel instabilities,
- Device heterogeneity – unlike server farms, dynamic resources are quite varied, including tablets, mobile phones, vehicles, laptops.

Currently, resources housed in data centers offer solutions to the above issues, at the cost of maintaining, managing and funding the center. These static resources avoid mobility issues, communication and network issues, due to the center being hard-wired. Aside from failure of a resource, there are no reasons for availability fluctuations. However, the costs involved in a static data center, capable of hosting cloud services, are significant.

The focus of this thesis is virtual machine migration, specifically for the vehicular cloud. It is important to refer to existing virtual machine migration techniques in the traditional cloud, to assess their feasibility for dynamic/vehicular clouds. After which, further studies regarding vehicular clouds will be discussed. The next section presents an overview of virtual machine migration and management techniques.

2.2 Virtual Machine Migration and Management: Overview

In [MIS12], the authors present a comprehensive overview of the need for VM migration within general CC implementations. There are several main reasons to migrate a VM:

Consolidation of Servers: Servers may host multiple (possibly related/interconnected) VMs on separate machines. This is less preferable than (if possible) having these VMs on the same, if not on closer physical hosts. This also allows the machines which are now left unused to be put into standby or even off states, saving energy.

Balancing Processing Load: Physical hosts may have uneven loads and it is often preferable to migrate some VMs to evenly distribute loads. This gives physical hosts a more efficient level of utilization that can be advantageous in the event of increased demands.

Counteracting Hot/Cold Spots: In the pool of physical hosts, VMs are assigned to various machines. Multiple VMs may be hosted on the same physical machine (known as "co-location") while some machines may be unutilized/underutilized [DES11]. A hotspot occurs when the physical resources allotted to a VM are near the limits that may cause SLA violations. On the other hand, a 'cold spot' occurs when the resources are underutilized. Hotspots are undesirable due to the potential SLA violations while cold spots are undesirable because they are a direct result of inefficient resource utilization.

2.2.1 Virtual Machine Migration Requirements

In [MIS12], the authors present several heuristics for VM migration. These heuristics address: when to migrate, which VMs to migrate, and where to migrate the VMs.

When to migrate is a decision that the technique deployed must make based on ‘triggers’ or events that lead to migration [MIS12]. Examples may be a routine/scheduled migration, executed to preventively avoid the need to migrate. Another instance can be the existence of a hotspot. These entail detection methods that must detect the hotspot before it is ‘too hot’ (the VM’s resources are reaching storage/computation limits). Some detection techniques involve performance thresholds as metrics. These can also involve more intelligent and predictive methods to proactively migrate a VM before an inevitable hotspot. Migration of course can be avoided if the physical host has excess capacity to offer the VM in need, otherwise, migration is necessary. Further details of hotspot detection can be found in [WOO07]. Another reason for migration would be an uneven distribution of loads across a group of physical hosts, reducing the overall efficiency. Some physical hosts may have spare storage/processing capacity and this can be utilized to increase the overall efficiency of the data center. As discussed earlier, physical machines have hypervisors that can be used to monitor a host’s utilization. These values can be used to determine whether to migrate VMs for load balancing or for consolidation (to power down/off certain machines) with the goal of maximizing efficiency. Finally, new VMs introduced to a physical host and/or VMs being removed (due to completion or migration), can be considered triggers for migration assessment. If the removal or addition of a VM changes resource utilization enough to meet certain requirements, a migration may be initiated.

Selecting which VM(s) to migrate is an issue that is also discussed in [MIS12]. Such a decision can involve comparing potential migration candidates only. In some more complex decision methods, both the source and destination host’s states, prior to and as a consequence of the migration are considered. While the general goal is to select the VM requiring minimal effort to migrate, a potential goal could be to take into account the impact of the migration as well, in

order to decide whether the effort (minimal or otherwise) is worthwhile. The VM selected could be migrated due to it being the one with constrained resources. Several references present other more holistic approaches, taking into account various metrics concerning the physical resources available/needed [MIS11][SIN08][WOO07]. Other heuristics take into account ties between VMs (in terms of communication, shared memory), in order to avoid migrating a VM to a location where it would actually cost more to run resulting in network overhead [DES11][WOO09]. In [MIS12], the authors highlight the need for heuristics to address selecting the destination of a VM migration. Similarly to when selecting which VM to migrate, methods can be concerned with the available resources at the destination, the ties between VMs and the various overheads involved in migrating. Once when, which, from where and where to migrate the VM have all been decided, the migration itself takes place. This is discussed next.

2.2.2 Virtual Machine Migration Schemes

In [WOO07], the authors present a method for handling hotspots that includes several stages. As mentioned earlier, after hotspot detection, there is a need to estimate the necessary changes in resource allocations (the provisioning stage). The algorithm to deal with the hotspot involves selecting VMs and destinations for migration. The authors discuss that the problem of finding an optimal mapping of all VMs to physical machines, to avoid SLA violations, is in fact analogous to the “multidimensional bin packing problem.” This is an NP-hard problem. Determining whether a feasible solution exists is also NP-hard [WOO07]. As such, the authors resort to a heuristic approach. The goal for their approach (aside from which VM and where to migrate) is to minimize the amount of data that is transferred, also referred to as “migration overhead”. This will be referred to in this thesis as network (rather than migration) overhead. The method of

migration used is in essence hot/live migration. This is as opposed to cold migration, where the VM is inactive. A live migration is more complex as it requires multiple retransmissions due to changes in the VM state while operational during migration [ELR10].

2.2.3 Traditional Virtual Machine Migration

A typical migration involves several stages as shown in Figure 2.1. Note that skipping Stage 2 is what makes a migration cold as opposed to hot/live. Another way to consider live migration, in an analogy to wireless network handover techniques, is as a make-before-break process. Only upon guaranteeing the destination is correctly set and ready to host the VM, does the source cease hosting. If the migration cannot be successfully completed for any reason, the source aborts and resumes operation (or the migration attempt repeats) [CLA05]. Also note that the user only experiences actual downtime during Stages 3 and 4.

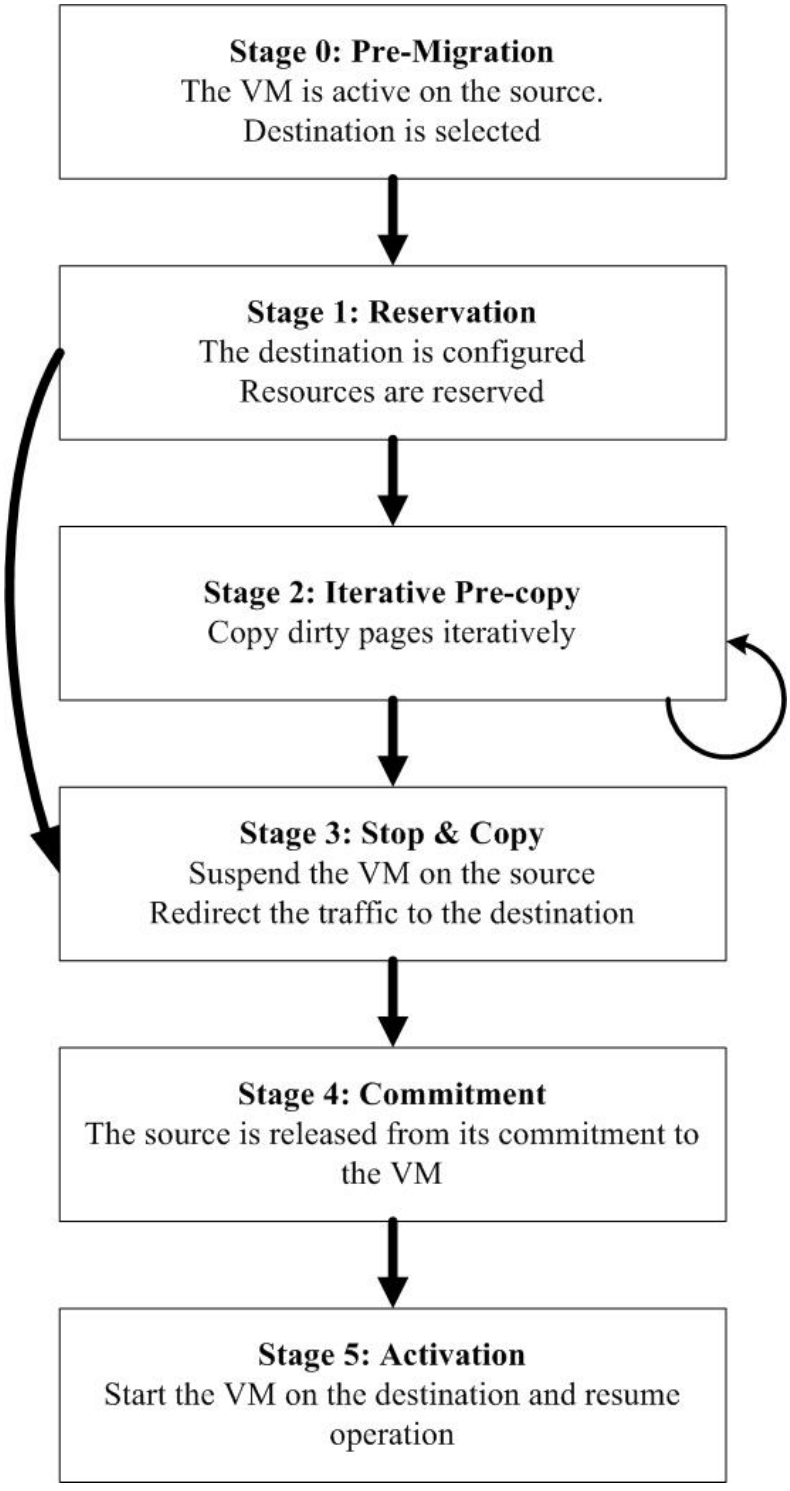

Figure 2.1: Migration Stages [CLA05]

2.2.4 Volume-based Migration Scheme

The heuristic presented in [WOO07] utilizes a similar live migration process as described but with some specific steps. Firstly, after the hotspot is detected, to avoid the NP-hard problem of deciding on a new mapping for all VMs, a single VM (or small subset) is chosen for migration. Using a simple greedy algorithm, the authors aim to migrate (if possible) the VMs from the machine that is experiencing the most severe strain, to, logically, those experiencing the least. This algorithm also takes into account the amount of data to be copied during the migration. A sorting technique is utilized.

The *volume* of a physical machine is calculated as:

$$Vol = \left(\frac{1}{1-cpu}\right)\left(\frac{1}{1-net}\right)\left(\frac{1}{1-mem}\right) \quad (2.1)$$

where:

cpu total utilization of the physical machine's CPU,

net total utilization of the physical machine's network interface,

mem total utilization of the physical machine's memory.

As the utilization of each resource increases, the volume will increase. These variables are assigned a value from 0 to $1 - \epsilon$.

The VMs are assigned a "volume-to-size ratio" (VSR), which is the volume of their corresponding physical machine divided by their size (memory utilized). It is important to note that these are values proposed by the authors in [WOO07] and can be modified depending on the

administrator's priorities. For example, more (or less) parameters of utilization can be considered when calculating volume, and size (in VSR) does not necessarily have to be just the memory utilized by the VM, it can include other parameters that define the VM size. Such modifications to the equations and parameters used can be found in [CHE12], where the authors take into account network performance when calculating volume. However, the following explains the unmodified algorithm:

1. Sort physical machines according to volume.
2. Sort VMs on each machine according to VSR.
3. Address highest VSR VM on highest volume physical machine.
4. Check if physical machine with lowest volume has resources that can sufficiently support that VM.
5. If so then it is 'assigned' to it, until migration time. If not, the next lowest volume physical machine is considered.
6. Repeat steps 4 and 5 until either a viable candidate is found or none are found. If none are found, repeat step 3, with the next highest VSR on the same physical machine.
7. Once a viable candidate is found, the process is repeated until the physical machine would no longer be a hotspot if all the VMs are in fact migrated to their assigned destinations.
8. Repeat the process again for the next hotspot until there would be no hotspots remaining if migration occurs.
9. Migrations are initiated using output of algorithm: A list of VMs and their assigned destinations.

It is important to reiterate for clarity, the fact that the algorithm performs no migrations until it completes stages 1 to 8. This means that VMs are assessed for potential migration destinations

based on volume and VSR. This is repeated for as many VMs and physical machines experiencing hotspots as needed and predicted effects of migration are used. Only after all assessments are made and VMs are assigned destinations (if any), does the algorithm begin VM migration. A potential study could consider conducting both the migration and the algorithm concurrently. This could prove some form of improvement.

There is a possibility that no physical machines have enough spare capacity to handle any of the VMs that need to be migrated to alleviate the hotspot. The authors counter this with a "swap" algorithm within the heuristic [WOO07]. As the term swap implies, the algorithm attempts to alleviate hotspots by swapping a VM (with a high VSR) from the hotspot, with several VMs (with a low VSR) from a low volume physical machine. The collective volume freed by the low VSR VMs needs to be enough to support the high VSR VM from the hotspot. This may not alleviate the hotspot initially, but will give more liberty for the regular algorithm to perform some migrations with the low VSR VMs (after they have been migrated). It is important to note that the heuristic only addresses one VM (on a hotspot) for potential migration. This is not the case presented in [DES11].

2.2.5 Migration of Co-located Virtual Machines

In [DES11], the authors present a different way to tackle migration. Firstly, when migrating a VM, it is not considered independently. The authors address multiple VMs, hosted on the same physical machine, that interact with one another. The study argues that it is better to migrate co-located VMs, and to migrate them simultaneously. Furthermore, the process of migration utilizes "de-duplication," resulting in what the authors called "live gang migration" [DES11]. This can be seen illustrated in Figure 2.2 and is described in what follows.

De-duplication is the main advantage of this method; whereby the pages (memory pages) moved during migration are minimized, because co-located VMs may have similar/identical pages, there is no need to transmit the same information multiple times, overloading the network. This of course will increase the computation overhead at the sake of network overhead, where the migration process needs to analyze the pages prior to migrating for similarities. This is the preparation phase, which is then followed by the migration. Pages are scanned periodically to find identical pages. Every iteration, only pages modified since the last scan are checked (to minimize computation). The identical pages are stored in a hash table. Finally, the migration takes place. All the co-located VMs are migrated.

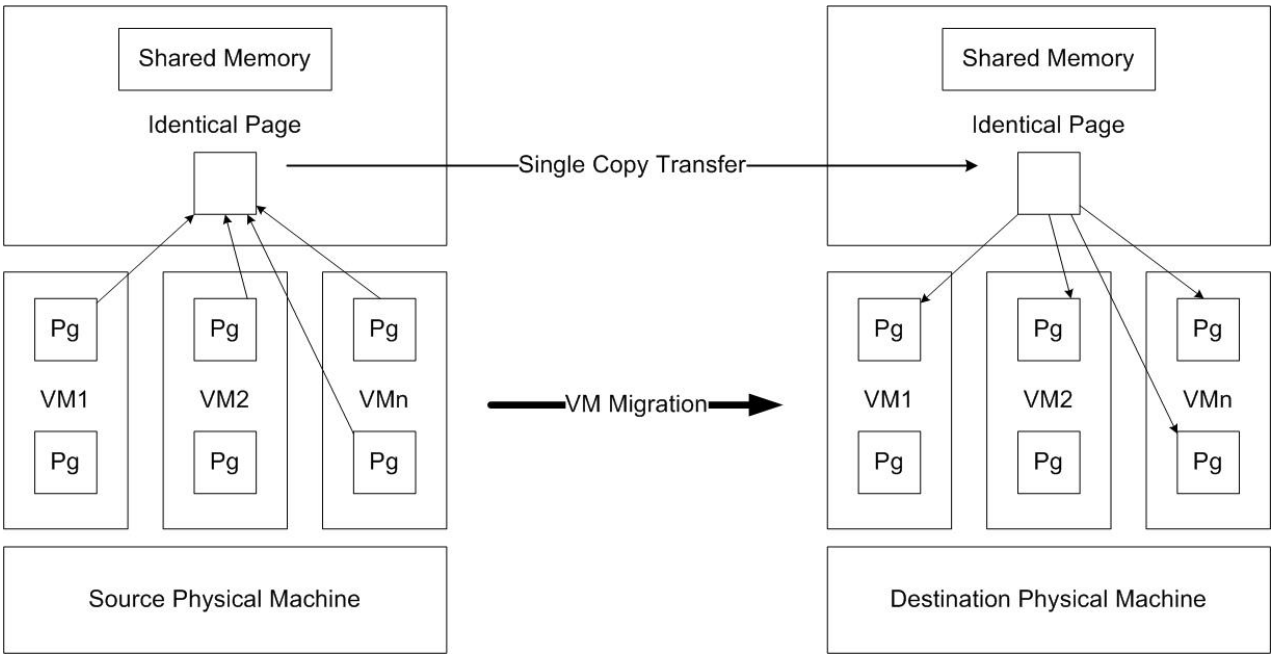

Figure 2.2: De-duplication in Live gang migration [DES11]

When migrating a VM, if a page contained is registered in the hash table, it is sent in full (the first time) and subsequently only an identifier is sent with the VM (thereby reducing the amount of data copied). The destination can then use the identifier to just copy the page locally. The

authors go into further detail regarding the actual mechanisms used to identify identical pages. Also, it is important to note that de-duplication can be done at a subpage level. The methods are all done while the VMs are inactive, which essential means it is a cold migration technique. The authors claim a live (or "online de-duplication") is not considered for further evaluation due to low performance [DES11]. The performance of the proposed method in comparison to the typical migration, compressing pages prior to migration online and in the offline case is shown to be an improvement. The total migration time is considerably less. Logically as the number of VMs increases, the true contribution of the method appears. Exact values can be found in [DES11].

2.2.6 Virtual Machine Migration in Dynamic Clouds

In [LIU16], the authors present a more recent study of virtual machine migration techniques, with a focus on dynamic clouds. The authors refer to the dynamic cloud in this context to mean a rapidly changing cloud, not a cloud with mobile resources.

The authors propose a migration strategy and several algorithms to suit clouds of a dynamic nature. They point out several potential causes for migration: a) Existing strategies often over-allocate resources to VMs, which assume that subscribers overestimate their needs, and as such resources can often go unused. Administrators assign VMs to the same physical resource, with the cumulative load 'promised' to the VMs is greater than the resource available, in hopes of minimizing unused resources. However, this could potentially cause a hotspot rapidly appearing, should a VM indeed request its full load, violating SLAs, b) alternative strategies avoid the previous issue by only allocating VMs their needed resources, at the cost of having unused resources reserved until the VM is released.

The authors propose a migration mechanism to solve this issue that involves periodic checking of and a setting of certain thresholds to determine which VMs will need to be migrated and when. This may involve several violations of the SLA but it is only temporary and is merely a trigger for migration.

The authors then address the selection methodology for the destination of migration. This involves calculating correlation coefficients for all VMs and servers using several equations. VMs that have the highest correlation with their host are those needing priority in migration. The correlation between the migrating VM and potential servers is calculated and the minimum is selected as the destination. Their simulations show the algorithm outperforms many existing methods.

In the literature, there are several other approaches to VM migration. These include prioritizing VMs and even the applications they are running to give lower downtimes to higher priority VMs and vice versa [JIA13]. Other studies address the issues involved with IP configuration when migrating over wide-area networks [BRA07][TRA06]. Further studies focused on VM migration can be found in [AHM15A], [AKO10], [BIL15], [BOU13], [DES16], [KAN16], [MAI16], [MAN16], [MIS11], [PAR09], [WAN13], [WU11], [XU10], [YE11], and [ZHA12]. As all the previously mentioned studies focused on traditional cloud computing scenarios, it is important to discuss Dynamic Clouds, in terms of mobility and not stability. The next section shall introduce the current state of research regarding Dynamic Clouds relative to the traditional cloud.

2.3 Vehicular Cloud State-of-the-Art

Dynamic or mobile clouds can refer to several specific implementations or contexts, but this thesis focuses mainly on vehicular clouds. The notion of the Vehicular Cloud has been studied from various perspectives. In essence, a VC (or vehicular cloud computing) is a cloud environment that incorporates resources offered by vehicles (among other more traditional sources) [ALM12][GER14][HE14][LEE14][MAN12][VIG14].

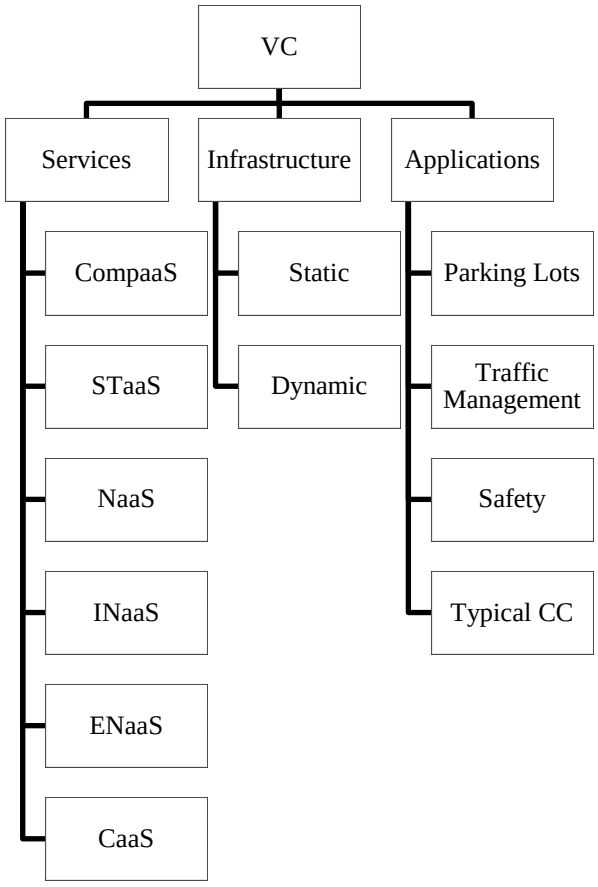

Figure 2.3: Vehicular Cloud Overview [WHA14]

Naturally, this involves several obstacles, or better put, areas of research, which arise from the field of vehicular networks overlapping with cloud computing. Among these obstacles are

energy efficiency, mobility of resources, wireless communication constraints, latency issues, scarcity and limitation of resources, safety issues, security and privacy issues, standardization issues and networking complexities. A high-level survey of vehicular cloud computing is illustrated in Figure 2.3 [WHA14]. Beyond the three levels of abstraction discussed earlier, there are studies which explain the other types of service that can be offered with various VC deployments [ARI12][DIN13][ELT10][GER12][GU13][MOU11]. VCs can offer Computing-as-a-Service (CompaaS) [WHA14], Storage-as-a-Service (STaaS) [ARI12], Network-as-a-Service (NaaS) [ARI12], Information-as-a-Service (INaaS) [DIN13], Entertainment-as-a-Service (ENaaS) [DIN13], Cooperation-as-a-Service (CaaS) [MOU11], and Traffic-Information-as-a-Service (TIaaS) [HUS13] among others. These various levels of service utilize the hardware on the vehicle. Vehicular electronic capabilities are constantly evolving (and improving) as a result of research in the field of Intelligent Transportation Systems [SAN12]. As such, there are several resources that are often underutilized by vehicle owners and operators.

In [WHA14], the authors discuss several ways to employ a VC, specifically in terms of deployment/set up. This is due to the main difference between VC and normal cloud deployments: the hardware being located on a group of vehicles, as opposed to a data center. A brief comparison is presented in Table 2.1.

In order to establish a vehicular cloud, there are three ways [WHA14].

1. Forming a cloud between a group of static/parked vehicles, which is the closest analogy to the conventional cloud, and is the easiest to manage [ELT10].

2. Forming a cloud and linking it to an infrastructure or backbone (be it cellular towers or a wired link to the Internet). This backbone would be very helpful in managing and orchestrating the cloud [OLA13].
3. Forming a cloud between a group of moving vehicles (with or without the assistance of the infrastructure). This is the most complex yet most common scenario that can be used to establish a VC [OLA11].

Table 2.1 – Comparison between Conventional Clouds and Vehicular Clouds

Comparison Aspect	Conventional Cloud	Vehicular Cloud
Communication Interfaces	Wired/Wireless	Wireless (unless in specific cases)
Power Sources	Mains/Grid	Local/Battery/Alternator
Hardware Ownership	Organization/Data Center/Governmental	Vehicle owner
Hardware Location	Data Centers/Organizations/Ad Hoc volunteers	Completely Ad Hoc volunteers (with compensation)
Hardware Physical State	Static	Mobile (unpredictable behavior)
Resource Capabilities	Varied (High, Medium, Restricted)	Mostly restricted
Vulnerability	Typical data center security threats	Wireless vulnerability/Resources are not at hand to monitor
Hardware Uniformity	Technology consistent within data centers	Varies from vehicle to vehicle
Hardware Resilience/Fault-Tolerance	Incorporated by data center administrator	May or may not be incorporated
Cost of Equipment	Significant overhead to invest in data center	Hardware already in existence

There is a greater focus in the literature on static clouds and/or heavy reliance on an infrastructure. This is logical as the obstacles faced are far less. Further studies on the VC can be found in [ALS14], [BAB12], [GER13], [WAN11], [YAN12A], and [YAN12B].

2.3.1 VC Architecture and VM Migration Schemes

In [ABI11], [QIN12], and [YU13], the authors envision the VC architecture at multiple levels. First, the central or conventional cloud is linked to the vehicular cloud via the roadside cloud (RC). The roadside cloud is a group of servers and wireless communication towers which are located in close proximity to the roads, unlike the central cloud which is housed in remote data centers. The roadside serves only the vehicular cloud with regards to cloud management, communication management and routing. The interface between the RC and the VC is a "roadside unit" (RU). In essence, the communication between the RC and the VC is analogous to (if not is exactly the same as) Infrastructure-to-Vehicle/Vehicle-to-Infrastructure (I2V/V2I communication) [CON08][NAR11][SAN08]. Communication within the VC is, in a similar fashion, Vehicle-to-Vehicle (V2V) communication.

The VC level requires dynamic scheduling between the vehicles for efficiently managed communication. The authors propose two possible ways to handle this: Generalized VC Controller (GVCC) and Specified VC Controller (SVCC). GVCC implies a centralized management approach. This also means that the hardware at the heart of the management requires more computation capability. SVCC is a decentralized (ad-hoc) management approach.

The first notion of VM migration with respect to VC is introduced when describing potential applications presented in this study are shown in [YU13]:

Table 2.2 – Vehicular Cloud Applications and VM Hosts

Comparison Aspect	VM Host
In-vehicle infotainment (IVI)	-
Social networking between vehicles	-
Location-based services	-
Traffic data mining for real-time navigation	Central and Roadside Cloud
Distributed storage for video surveillance	Roadside Cloud
Cooperative download of a large file	Vehicles

For real-time navigation VMs are hosted on RCs, and are migrated along with motion path of the vehicle. This is needed to maintain that the VM is in the RC that has the vehicle within its coverage, to update the vehicle with real-time navigation. There are also VMs run on the central cloud, which handle the data-mining for the congestion information. For video surveillance, rather than store video on a bus Hard Disk Drive (HDD) and analyze it offline, again, utilize the VM on the RC. As the bus moves between RCs, it stores the next segment on the nearest cloudlet. The VM is migrated accordingly. Once an accident occurs, the videos are accessed on-demand (albeit in a fragmented fashion). Only in cooperative download of a large file do the authors suggest a method requiring vehicles to host the VM. When downloading a large file, the initiating vehicle selects other vehicles to help with the download. Each assisting vehicle hosts a VM, downloading parts and sending them to the initiator when complete (without any commitment to the infrastructure anymore). The initiator then assembles the file.

The next section presents the state-of-the-art with respect to vehicular clouds.

2.3.2 State-of-the-Art VC Studies

Vehicular cloud and Mobile cloud research is a broad field and topics range from issues relating to fault-tolerance [GHA15], security and privacy issues [AHM15B][YAN12A] and [YAN12B], to designing virtual and physical testbeds [LU16][PAS16], among other focuses [ADH16][BAR16][BIT15][CO15][EZI15][GAI16][HUS15][TAO15][KIM16][KUM15] [SHO16].

In [AHM15B], [ALO15], and [HUS15] the authors present a survey on the recent studies of vehicular clouds. The authors in [AHM15B] describe the vehicular cloud architecture in 3 tiers: vehicles, infrastructure, and traditional online cloud.

The grouping of all three tiers creates what the authors describe as a general architecture for vehicular cloud networking. The authors let the "cloud leader" define what specific type of cloud is implemented, be it a vehicle or the RU. The authors also specify several potential applications for such a scenario: video surveillance, bandwidth management of VC services, real-time navigation, remote traffic management.

In all cases, the authors focus generally on applications that would specifically benefit the vehicular cloud or benefit from the uniqueness of the vehicular cloud, not as an alternative or complementary asset to the traditional cloud model. As such the authors emphasize that security is of the utmost importance, on the cybersecurity level, on the wireless communication level and at the infrastructure level. The authors then outline the various threats and their criticality. In [ALO15], the authors address the importance of Quality of Experience, in terms of service provisioning.

The most recent and relevant studies to this thesis are [BAR16] and [LIU16]. While [LIU16] addresses dynamic clouds in terms of stability and not mobility, [BAR16] addresses the pairing of virtual machine migration and vehicular clouds. Sharing the same journal as our main publication, and citing our preliminary publication, the authors address the feasibility of VM migration for VCs.

In [BAR16], the authors also discuss the core focus of this thesis: virtualizing the resources of a vehicle for the cloud. Analogous to the traditional data center, the vehicles would serve as the physical machines. The users are forecasted to be mobile users, rather than the typical cloud user (business desktops etc.). The authors also target direct vehicle to vehicle migrations at predestined points in the topology. Unlike the proposed methods of this thesis, the authors use a history-based approach rather than an adaptive and intelligent decision making process.

The authors utilize an existing trace map of bus routes, and consequently use buses as their hosts, which while effective, does not address all the remaining (more importantly unused vehicles) in the grid. Based on the existing trace map, points of highest contact between buses are highlighted via a heat map and are designated as ‘hotspots’ where migration can occur, directly from bus to bus. This reduces the need for on-the-fly computation but potentially leaves room for cases when a migration cannot occur due to various reasons, such as lack of space, time etc.

The proposed method and the feasibility analysis is sound, however, the methodology proposed in this thesis will address many shortcomings of that study as well as prove to be potentially simpler and more effective.

2.4 Discussion

Aside from the lack of literature addressing VM migration schemes within a VC context, this section is a brief summary and commentary on the presented points regarding the pre-migration and migration phases in the various works discussed earlier. Table 2.3 presents the various issues presented earlier and their suitability in relation to the VC.

Table 2.3 – Suitability Analysis of Existing VM Management Schemes for the VC

Issue	VC Suitability
Deciding when to migrate [MIS12]	Hotspot detection algorithms no longer apply as triggers. The main trigger for migration would be vehicles leaving the coverage area. This requires estimation of mobility patterns [TOR12]
Deciding which VM to migrate [MIS12]	Existing algorithms may be too slow or complex with the limited resources
Deciding where to migrate [MIS12]	This requires either a shared knowledge between vehicles or a controlling node regarding utilization levels (or 'volume')
Cold migration [CLA05]	May not be suitable as it is not necessarily feasible to have nonoperational VMs stored on a moving vehicle
Live migration [CLA05]	Without modifications, the 5-stage process may require too much of a network overhead and a connectivity commitment between source and destination
Volume/VSR method [WOO07]	Again, requires frequent interaction vehicles with active VMs. However, it would only be suitable for determining where to migrate, because hotspot detection is not necessarily valid
Network performance aware method [CHE12]	More suitable than the unmodified VSR method as it considers more parameters (specifically network overhead) which is pertinent to VC
De-duplication/gang migration [DES11]	Only feasible if a vehicle can support multiple VMs
Priority-based migration [JIA13]	Only feasible if a vehicle can support multiple VMs unless priority comparisons are conducted across multiple vehicles

The lack of suitability can be considered an obstacle, and hence a potential module to address in the thesis. The potential gaps in research will be discussed in the next section.

2.5 Obstacles and Gaps in Existing Studies

Very few studies, at this time, address vehicles as virtual machine hosts for the vehicular cloud. Such a scenario necessitates an effective virtual machine management and migration scheme. The startup capital or overhead required for a completely new (or upgraded) infrastructure to prepare roadside cloudlets to host virtual machines is much greater than utilizing the already existing and underutilized hardware within vehicles currently in operation. Furthermore, the running costs of powerful servers and roadside units to support the vehicular cloud would be significantly higher if they are to host the majority of the computation/storage loads, as opposed to relying more heavily on the vehicles as hosts. The reason for focusing on CompaaS and STaaS specifically is that these are the most penetrative services for the average cloud user: Cloud-based computations and cloud-based storage.

As such, the goal of this thesis is to design, optimize and simulate a resource-aware scheme for virtual machine management and migration in the vehicular cloud. The specific services to be offered by the vehicular cloud in focus are Computing-as-a-Service and Storage-as-a-Service.

The issues discussed earlier with regards to suitability for VC are analyzed in terms obstacles and potential solutions are presented in Table 2.4.

Table 2.4 – Traditional Cloud VM Migration for the VC – Issues and Potential Solutions

Issue	Solutions
Deciding when to migrate [MIS12]	Require a method for vehicles (or RC) to detect appropriate migration time. This may rely on a method similar to cellular handover techniques, using received signal strength to predict location within coverage and distance from end of coverage footprint. This would be modified to include a predictive mechanism using speed and acceleration to estimate time within coverage remaining. Once near threshold of maximum migration time, initiate migration
Deciding which VM and where to migrate to [MIS12]	This is a job for the RC, as overseer or manager of the VC in order to decide who and where to migrate, based on analysis of the topology and geographic layout of the various vehicles in coverage. If this is a local decision, it mandates frequent exchange of status packets between vehicles with the dynamically changing environment. The factors taken into account when selecting which VM could vary depending on the computing resources available for the algorithm
Live migration [CLA05]	Assuming the migration decisions have been made, when, which and to where, migration needs a duration of constant connectivity (because the user is still connected while the migration occurs) between source vehicle and destination vehicle, unless a pausing mechanism is incorporated.
Volume/VSR method [WOO07], network performance aware method [CHE12]	Such a mechanism requires either frequent interactions (network overhead) and significant computation overhead between vehicles, or involvement of a roadside unit with some minimal intelligence (as opposed to a full data center roadside)
De-duplication/gang migration [DES11]	Possibly applicable if there are multiple vehicles that the RC can tell to migrate simultaneously due to their soon-to-be departure, to vehicles that are still in coverage
Priority-based migration [JIA13]	Again, would require heavy involvement from the RC and frequent interactions between vehicles and the RC

2.6 Summary

This chapter presented a survey of the state-of-the-art of the Vehicular Cloud as well as discussing virtual machine migration schemes in the traditional cloud. The aim of this chapter

was to present a background of the current state of research and to make way for the next chapter, introducing the contributions of the thesis.

Chapter 3

Vehicular Virtual Machine Migration

3.1 Introduction

One of the main obstacles associated with a vehicular cloud, and dynamic clouds in general, is the issue of VM migration. To avoid SLA violations, the cloud provider must ensure an acceptable degree of VM availability. This is a more critical concern in a dynamic cloud than a static cloud, as the resources can both frequently and unpredictably move in and out of network coverage. A resource outside network coverage, logically, is inaccessible to a cloud subscriber, and as such this could directly result in an SLA violation. While cloud service providers would attempt to minimize such violations by storing multiple redundant copies of VMs, this may not be enough to maximize availability.

Vehicles acting as VM hosts are analogous to physical machines in data centers. The more pressing reason for VM migration is not a hotspot, but a vehicle leaving network coverage. A

vehicle leaving network coverage will not be accessible by the cloud subscriber, let alone the administrator. However, if a vehicle can successfully migrate its hosted VM to another vehicle before it leaves network coverage, this would avoid the SLA violation. Unlike in static data centers, migration of a VM is not as straightforward as selecting an alternative physical location that has enough space. This thesis proposes, evaluates and compares several migration schemes that will address issues unique to a dynamic (specifically vehicular) cloud.

The analysis of the various proposed algorithms requires an environment within which a vehicular cloud can be deployed for evaluation. A physical testbed to accurately imitate a vehicular environment is very expensive, difficult to manage, and it is more practical to simulate the concept within a software environment [CON08][LAN08][MAR11][SOM08][TOR12].

There are several existing proprietary and open-source traffic/mobility model simulators available, each with its own inherent strengths and limitations. Examples include Simulation of Urban Mobility (SUMO) (open-source) and Vissim (proprietary) [KOT09][PAS16][PTV17]. Both tools can offer highly realistic and accurate traffic models, with varying degrees of scalability. Both microscopic and macroscopic simulations can be set up to model individual vehicles or aggregate flows. Such tools, especially open-source tools could prove very useful for more complex analyses in future research. The authors of [KOT09] present a more detailed analysis of various traffic simulators, analyzing features (such as a graphical user interface or 3D modeling), processing and memory performance, and scalability.

There are several motivations to develop a simple mobility model rather than to rely on the available tools. The model must be microscopic, scalable, light on processing, and most importantly, guarantee compatibility/interoperability with any potential tools to be used

throughout the thesis. The model should merely test the basic performance of the proposed algorithms. The model is initially quite simple, to focus on testing the proposed schemes, but it becomes more complex in later sections of the thesis. However, it is still simple enough to minimize computation requirements. The simplicity will also make tracing and debugging the migration algorithms easier.

3.2 Simplified Mobility Model

Rural or suburban settings are less appealing for a VC implementation than an urban/downtown setting due to the abundance of vehicles in higher spatial density, as well as an existing and high-capacity network coverage. A typical city center setting can be simplified to a grid-like region. For the sake of simplicity at this stage of the study, a grid comprised of three large horizontal roads intersecting with three large vertical roads, as shown in Figure 3.1, should suffice. The resulting layout can be considered a graph with 9 vertices, oriented in a 3×3 layout as shown in Figure 3.1, with the transitions from each vertex symbolizing a one-way section of the street from an intersection (or vertex) to the next. Each transition is assumed to be 500m long. These are generic values assumed to represent a typical downtown setting.

Based on this layout, there are a finite and known number of possible entry and exit points to the grid. The entry points are $S \in \{1, 2, 6, 7, 9\}$ and the exit points are $D \in \{1, 3, 4, 8, 9\}$. A vehicle is assumed to enter and leave the grid, and hence has a trajectory that can be defined as a sequence of intersections or transitions. For each vehicle i the trajectory is:

- $T_N = \{N_1, N_2, \ldots N_k\}$, where k is the length of the path, as a number of intersections (N_i).
- $T_L = \{L_1, L_2, \ldots L_{k-1}\}$, where L_i is i^{th} leg of the trajectory traversed.

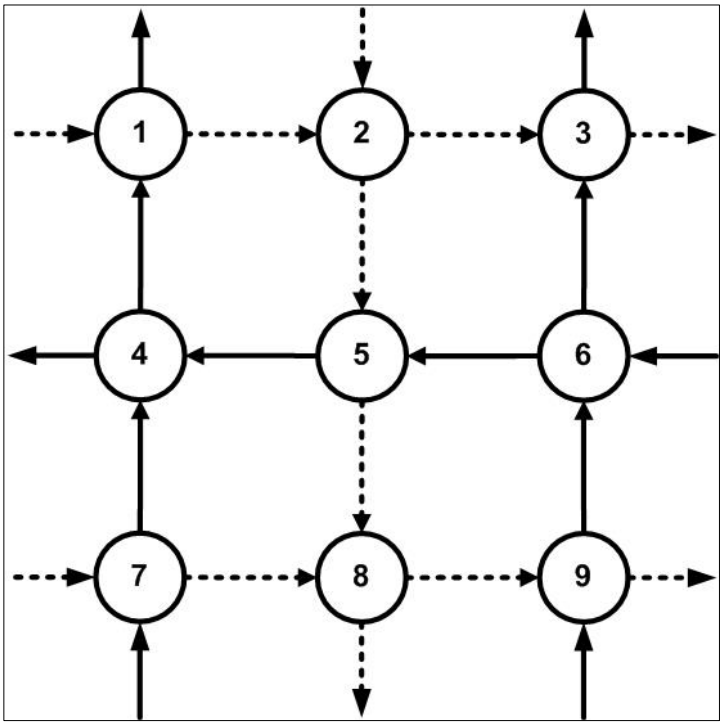

Figure 3.1: Modeled Grid Layout

It is also assumed that for all vehicles in the model:

1. $N_1 = S_i,\ N_k = D_i,\ S_i \neq D_i$. Vehicles must exit at a point other than the point of entry.
2. $L_i \neq L_j\ \forall\ i, j, i \neq j$. Vehicles are allowed to revisit intersections provided that they never use the same leg twice.

These assumptions are selected to minimize the complexity of the model to maintain focus on the contributions of the thesis, the VM migration schemes (to be presented in later sections). These assumptions mean there are a total of 39 possible trajectories (exhaustively), presented in Table 3.1. There are cases when multiple paths exist for one entry/exit pair.

Table 3.1 – Possible Vehicle Trajectories

Si	Di	N1	N2	N3	N4	N5	N6	N7	N8	N9
1	3	1	2	3						
1	3	1	2	5	8	9	6	3		
1	4	1	2	5	4					
1	4	1	2	5	8	9	6	5	4	
1	8	1	2	5	8					
1	9	1	2	5	8	9				
2	1	2	5	4	1					
2	1	2	5	8	9	6	5	4	1	
2	3	2	3							
2	3	2	5	8	9	6	3			
2	4	2	5	4						
2	4	2	5	8	9	6	5	4		
2	8	2	5	8						
2	9	2	5	8	9					
6	1	6	5	4	1					
6	3	6	3							
6	3	6	5	4	1	2	3			
6	4	6	5	4						
6	8	6	5	8						
6	8	6	5	4	1	2	5	8		
6	9	6	5	8	9					
6	9	6	5	4	1	2	5	8	9	
7	1	7	4	1						
7	1	7	8	9	6	5	4	1		
7	3	7	8	9	6	3				
7	3	7	4	1	2	3				
7	3	7	4	1	2	5	8	9	6	3
7	4	7	4							
7	4	7	8	9	6	5	4			
7	8	7	8							
7	8	7	4	1	2	5	8			
7	9	7	8	9						
7	9	7	4	1	2	5	8	9		
9	1	9	6	5	4	1				
9	3	9	6	3						
9	3	9	6	5	4	1	2	3		
9	4	9	6	5	4					
9	8	9	6	5	8					
9	8	9	6	5	4	1	2	5	8	

It is important to realistically consider the effect of congestion (car traffic) on a vehicle's top speed. Logically, as more vehicles flow through the grid, the vehicles will not be travelling as fast as they would be able to when the congestion is lower. As such, depending on the congestion level, a ceiling is imposed on the top speed permitted. With three congestion levels studied, High, Medium and Low, the speeds modeled are:

1. High congestion: 10, 20, 30Km/h.
2. Medium congestion: 10, 20, 30, 40Km/h.
3. Low congestion: 10, 20, 30, 40, 50Km/h.

A tool was necessary that could achieve the following:

1. Autonomously and rapidly create a mobility model, with various congestions levels,
2. Seamless incorporate this mobility model as an input to run the novel schemes,
3. Facilitate the rapid execution of the migration schemes,
4. Analyze the performance of each scheme,
5. Repeat all steps rapidly and autonomously collect results for multiple seeds.

Any involvement of multiple tools to achieve the above goals, would require inter-software compatibility, without user involvement. Such a scenario could be potentially non-existent, without significant modification. MATLAB was selected to design and build an engine to achieve the above 5 goals [MAT16]. This will be further explained in the following section, which is also summarized in the preliminary publication related to this thesis [REF14].

3.2.1 MATLAB and Revised Simplified Mobility Model

MATLAB provides a flexible and comprehensive coding environment for this thesis. As it was back to the drawing board, Algorithm 1 needed some refinements. The features of the

mobility model proposed earlier had a drawback that needed to be revised; the flow of vehicles through the grid was not realistic. Vehicles typically do not travel from intersection to intersection with a constant speed. As proposed earlier, cars instantaneously accelerate from 0Km/h to their designated speed, maintain it until the next intersection, and then decelerate from their speed to 0Km/h instantaneously. Also, with a certain degree of consistency, cars should continuously be entering, as well as leaving the grid. The arrival rate of vehicles is what would determine the congestion level. The process for creating the model in MATLAB is described next.

A total of 500 vehicles are modeled for every iteration. This will not cause a problem for congestion modeling as that will be handled based on the arrival rate and will later be addressed. Each vehicle follows the process shown in Figure 3.3 for trajectory assignment. As shown, selection of the various parameters determining which trajectory a vehicle is assigned is done so based on a uniform random distribution. The selection of the start time (‘based on window’), the selection of the speed (at various points along the path) and as a consequence the migration initiation time, require a more complex process.

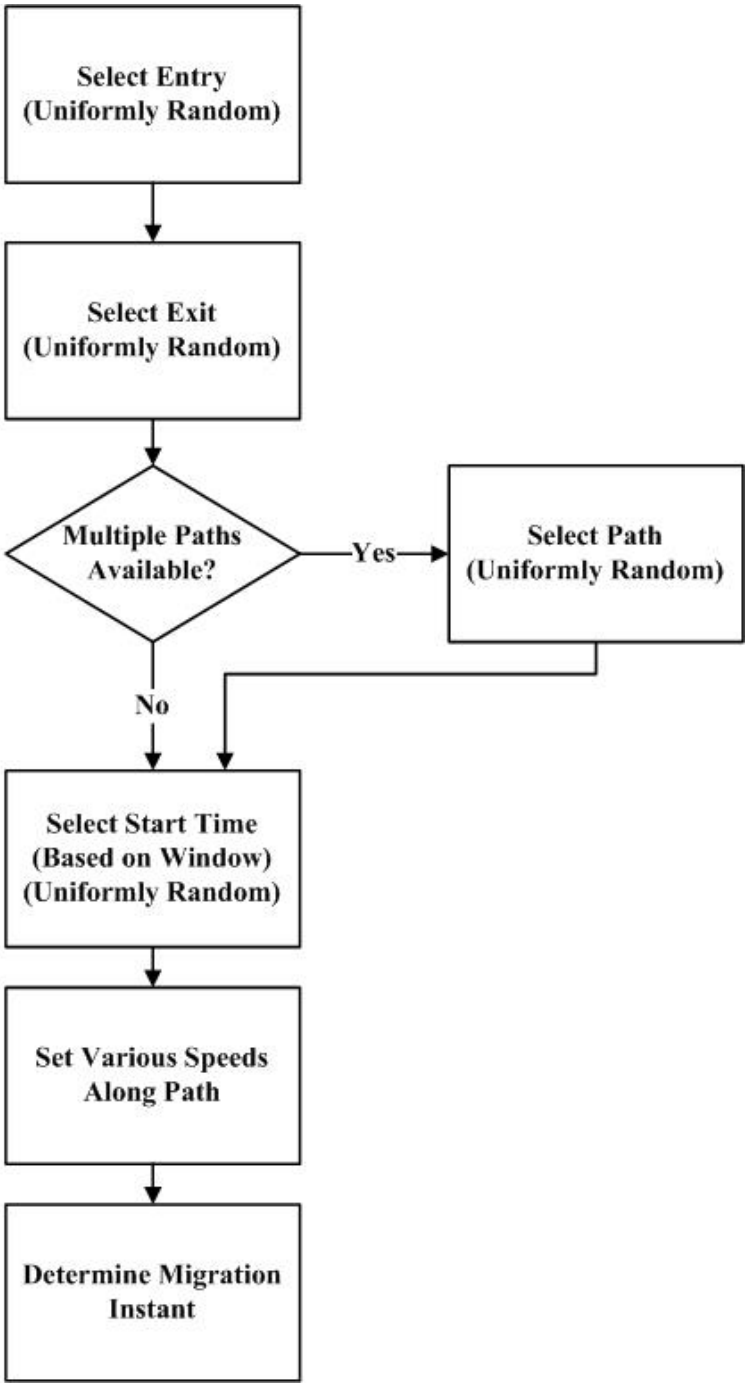

Figure 3.2: Trajectory Setup Process

3.2.1.1 Speed Initialization

Each vehicle, based on its trajectory, is assigned various speeds that vary randomly, but realistically along the path. As a rule of thumb, no abrupt speed changes occur. Speeds can only increase or decrease in steps of 10Km/h between every second of time passing. Furthermore, a car is forced to decelerate gradually from whatever speed it was moving at, near four seconds prior to reaching an intersection. This is such in consideration that at the worst-case speed of 50Km/h, the car will have time to decelerate by 10Km/h steps every second, so that by the time it reaches the intersection it will have stopped. During the intermediate points along the legs of the trajectory, the car selects a random speed (uniformly between the possible speeds mentioned above) and accelerates or decelerates to that speed in 10Km/h increments/decrements every second until it reaches the desired speed. Once that speed is reached, the random selection of the next speed is repeated and the vehicle may maintain its speed or change accordingly. This will achieve a more fluid mobility pattern and a more realistic simulation of vehicular motion.

3.2.1.2 Incorporating Congestion

As described earlier, the congestion level determines the top speeds a vehicle can be assigned. However, the actual implementation of the congestion levels is achieved by varying a 'five-car-time-window'. As the congestion level increases, the time-window decreases. Every time-window, five new cars enter the grid (at uniformly random points within this window). This translates to a higher number of cars entering the grid over a shorter time period. This can be measured using Equation (3.1).

$$r_e = \frac{cars}{window} \qquad (3.1)$$

Setting the time window to 60, 30 and 10 seconds for low, medium and high congestion respectively, achieves a rate of entry (r_e) of 5, 10, and 30 cars/min. The 500 modeled cars are assigned start times in groups of five. For example, for high congestion, the five-car-time-window is 10 seconds. The first five cars would have start times randomly distributed between 1 and 10 seconds, the second five cars between 11 and 20, and so on, until all 500 cars are assigned a start time.

At this point, the total duration for all cars to enter and complete their trajectories to the exit, is easily calculated in the code. However, there is a an undesirable consequence of this method: near the start of simulation, the grid will be empty, and cars will start to increase, eventually as time goes on, and the cars start to exit, at some point it will have a steady decrease of vehicles, until empty again. This is due to the finite number of vehicles modeled. In order to overcome this, a large number of vehicles is simulated (500) and an observation window, centered in the midpoint of the simulated time, will be used. The time needed from the first car entering the grid to the last car leaving the grid, is quite variable and significant. In order to make a fair observation of such a scenario, with continuous vehicles entering and leaving the grid, an observation window of 300 seconds, centered in the middle of the total time window is used. While the migration schemes will run over the entire simulated time, only the observation window will be analyzed to avoid the unrealistic behavior of filling/emptying the grid.

This sub-section concludes the initialization of the mobility model, in terms of vehicle mobility patterns. This has no bearing on the novel migration schemes except to serve as the testing environment. The following sub-section will present the first stage of configurations pertaining to the VC application.

3.2.2 Virtual Machine Workload Initialization

To simulate the cloud provider/administrator deploying VMs on the various vehicles, the workloads to be deployed must first be initialized. To minimize complexity while maintaining a degree of realism, three VM sizes are modeled: Light = 1 unit, Medium = 2 units, Heavy = 4 units.

Each vehicle is assumed to have a VM-hosting capacity of 8 units and is initialized to host only one type of VM, but can host a multiple number of the same type. After migrations occur, a vehicle will be able to host multiple types of VMs, depending on capacity. Figure 3.4 shows the selection process for each vehicle's initial VM workload. After selecting the type/size (or no VM), the number of VMs of that type to be hosted is selected. The figure however, shows the equivalent number of units resulting from that selection.

It is important to note that the VM sizes are left without specific units at this stage, as communication aspects are not assessed, except in terms of a time window τ seconds. This time window is designated as the maximum time that may be required for a successful migration to occur between two vehicles, from a vehicle determining it is in need to migrate, till the host successfully takes over the VM. This is arbitrarily set at a constant value of 25 seconds, for any migration. This is an assumption that will simplify the testing of the migration schemes, and will be revisited in later sections to study the communication/network effect on the scheme performance.

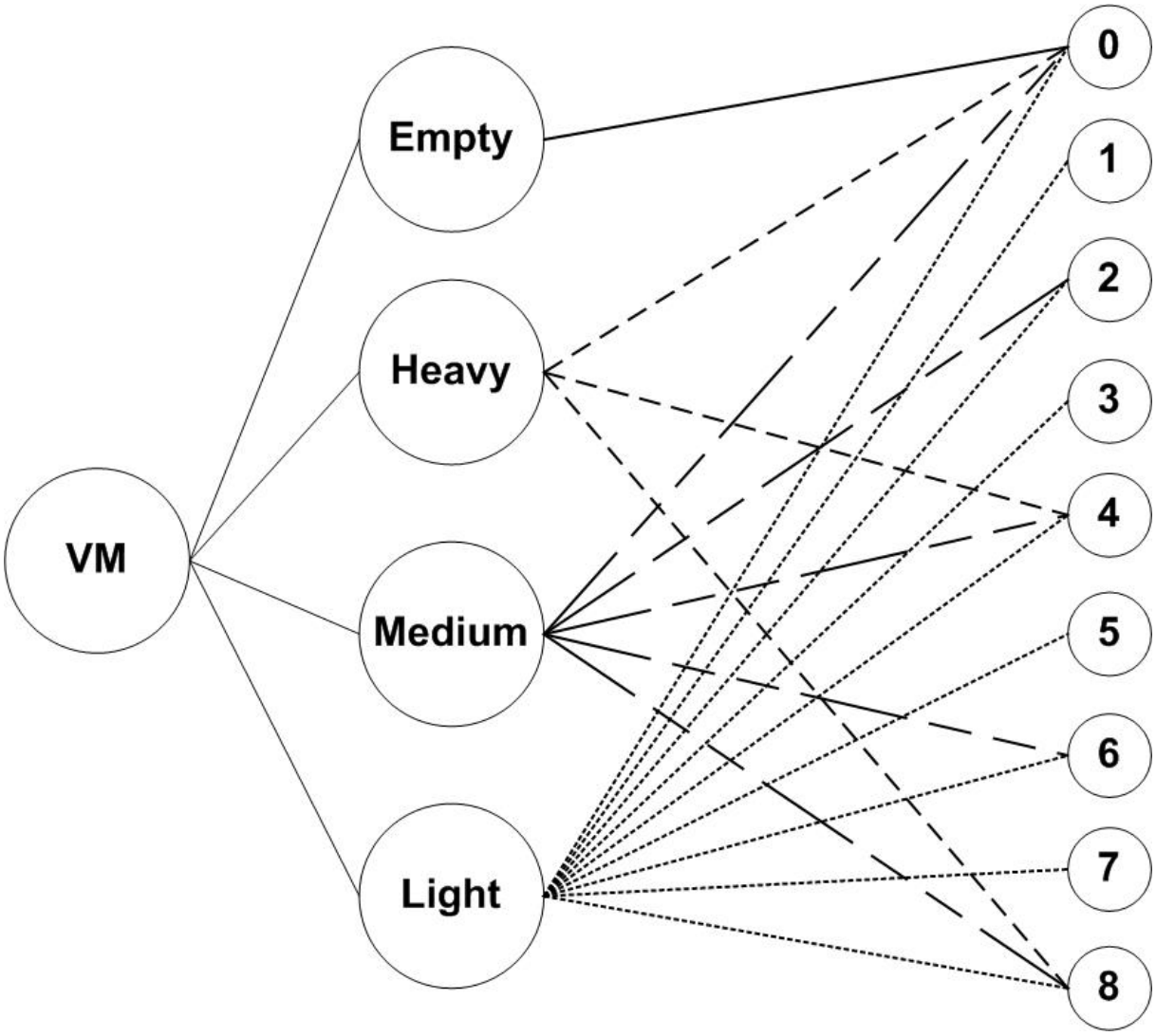

Figure 3.3:– VM Initialization Stages

3.3 Vehicular Virtual Machine Migration – General Scheme Description

This section presents the novel migration schemes to be evaluated, having sufficiently setup a basic mobility model and initialized all vehicles' workloads. The novel scheme Vehicular Virtual Machine Migration (VVMM) will address:

1. Frequent and unpredictable vehicular topology changes,
2. Host heterogeneity,
3. Minimizing RU intervention.

The metrics used to gauge the performance of the scheme will be introduced later.

The vehicular cloud under study involves multiple moving vehicles, hosting various VMs. As such, unlike a conventional cloud data center, the migration triggers are different. In essence, in a VC, and by extension, any Dynamic Cloud, a VM migration must occur if:

1. A vehicle/node switches from the communication range of one RU to that of another.
2. A vehicle/node is leaving the communication range of any RU to an area without coverage.

The above cases occur depending on the coverage map of the area under study, the number of vehicles/nodes, the terrain/topography, the communication protocol, the congestion level, the various vehicle/node speeds, and the vehicle/node mobility patterns.

In what follows, it must be noted that, the term unsuccessful migration is used to denote an undesirable outcome. Examples of such an outcome include the VM being migrated and stored in RU storage, rather than in a vehicle, or hibernated on the vehicular host until it returns to coverage. In both cases, the client may experience some degraded performance or downtime, but no loss of information is incurred.

In order to properly assess the scheme performance, three variations are tested and compared. All variations only address 1:1 migrations, where the destination of the migration can only be one vehicle. This is an initial assumption, which may be expanded in future refinements to the

algorithm if needed. The three variations are ‘uniform’, ‘least workload’ and ‘mobility-aware’, VVMM-U, VVMM-LW and VVMM-MA respectively.

VVMM-U assumes no intelligence in selecting migration destinations, and largely depends on proximity. VVMM-LW considers a further requirement, that being only candidates with an excess capacity large enough to handle the migration are considered and those with the least workload are chosen. Finally, VVMM-MA takes a further requirement beyond VVMM-LW, where a degree of mobility awareness is employed. Vehicles that qualify for candidacy must be predictably within range (and not intending to migrate) for the next τ seconds.

As shown in the flowchart in Figure 3.5 there are several stages that the vehicle follows in each scheme. The main *search criteria* are what changes depending on the scheme applied. The three schemes are elaborated further next.

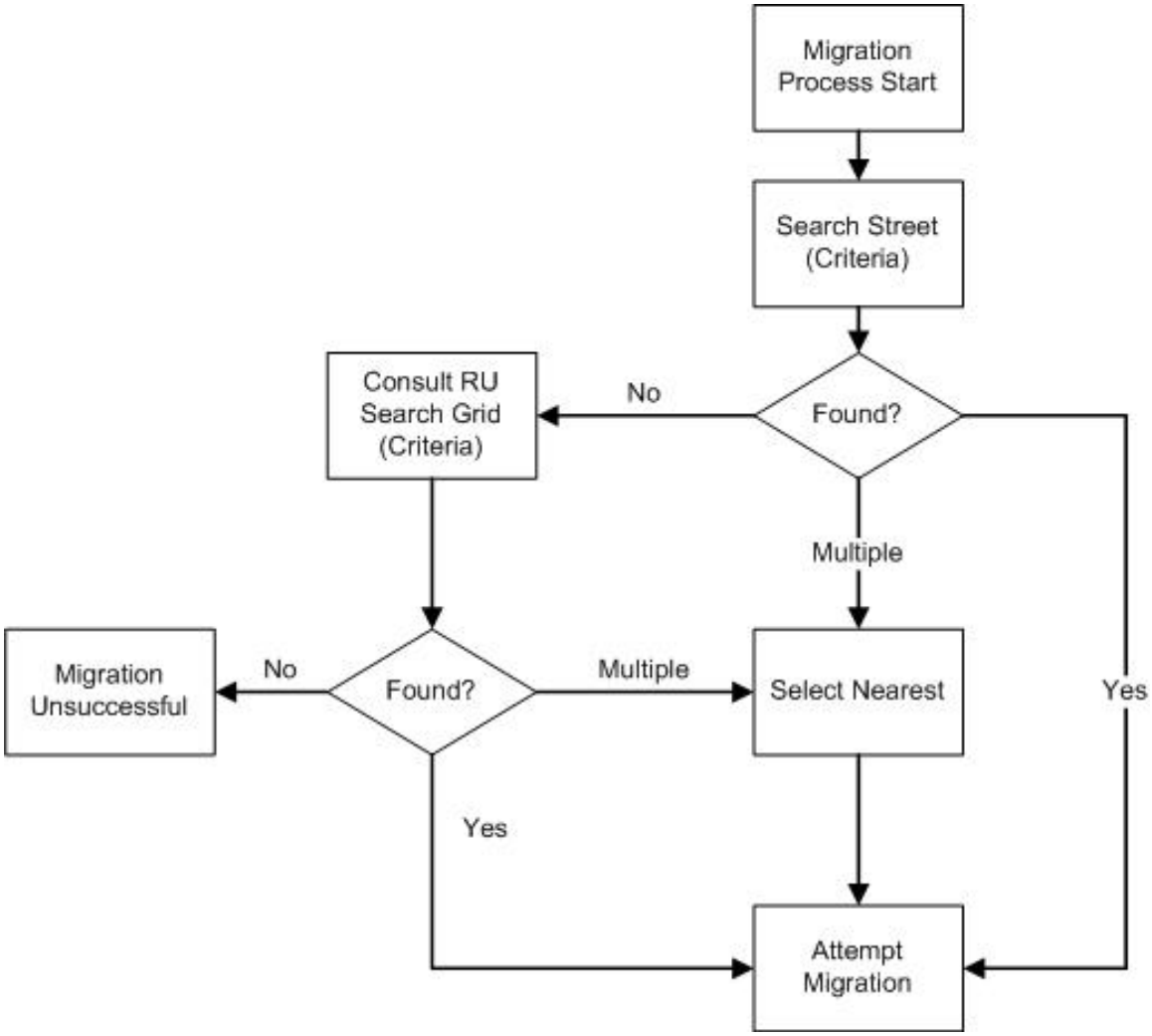

Figure 3.4:– General VVMM Algorithm

3.4 Detailed Scheme Description

VVMM-U: In the VVMM-U ('uniform') scheme, very little intelligence is involved in selecting migration destinations. A vehicle needing to migrate, need only select whatever vehicle is nearest it, on the same street, regardless of its excess capacity or future state. If multiple equidistant vehicles exist, it shall randomly select one. If it is the sole vehicle in the street, it shall refer to the RU, which shall utilize the same method, but instead of searching on the same

street, it will look at the entire grid, for the nearest vehicle, again disregarding its excess capacity or future state. This of course will result in an ‘unsuccessful migration’ in the case that the selected vehicle has less excess capacity than needed, or if the vehicle selected will itself be migrating (or is already in the process of doing so) within the next τ seconds.

VVMM-LW: In the VVMM-LW (‘least workload’) scheme, there is slightly more intelligence involved in the destination selection process. There is an added criteria that a) the destination of the migration must have enough excess capacity to host the load in question and b) the vehicle with the least workload (among those that qualify the first criterion) will be chosen. This process means that a vehicle will consider its neighbors in the same street, for excess capacity that satisfies its need. Upon finding one (or multiple) it will select the one(s) with the least workload. If multiple vehicles exist, with such criteria, it will select the nearest one. If no vehicles in the same street have enough spare capacity (or no vehicles share the street), the source vehicle will delegate the task to the RU, which will, using the same ‘LW’ criteria, search the entire grid for a valid candidate. In all stages of the scheme, the candidate vehicle’s future position/status is still not considered, so now there are less scenarios that lead to an ‘unsuccessful migration’. One case is where no vehicles are found whatsoever, with sufficient excess capacity. The other case is the scenario that, after selecting a destination with sufficient capacity, it turns out that the destination will itself be migrating (or is already in the process of doing so) within the next τ seconds.

VVMM-MA: The final scheme, utilizing ‘mobility aware’, is the scheme that introduces further intelligence to the process. Following the same process as VVMM-LW, the vehicle begins by searching its neighboring streets for vehicles with sufficient excess capacity, however, they must *also* satisfy the condition that none of them will be migrating (or are already in the

process of migrating) within the next τ seconds. There are several ways to predict mobility, which is important in deciding when to migrate (source) or predicting time remaining in the grid (destination) [TOR12]. One example of such a way would be to use signal strength to predict distance from the edge of the grid, and roughly calculate potential time remaining based on current speed.

Only if a vehicle satisfies the new condition, utilizing knowledge regarding its mobility pattern and future positions, will it be considered for the 'LW' stage. Among the vehicles that satisfy both the excess capacity and the mobility awareness requirements, the vehicle(s) with the least workload is/are selected. If multiple exist, the nearest is chosen. This results in a guaranteed 'successful migration'. In the event that no vehicles satisfy the above requirements, the vehicle again, refers to the RU, which searches the entire grid, with the same requirements. As described, this scheme now only results in an 'unsuccessful migration' if no vehicles satisfy the above requirements. This means that unlike the two previous schemes, if a vehicle is successfully selected, there is no reason for a migration attempt to fail.

The various schemes proposed both in this chapter and the next, are intended for local execution, on board of a vehicle intending to migrate its load. The schemes would necessitate some minor message exchanges prior to starting the migration. Messages would include information such as workload on-board, received signal strength or location information (to calculate time left in the grid). For messages exchanged in between vehicles in the same street, this is simple, but expanding to the grid could potentially involve some contribution from the infrastructure. The specifics can be further explored in future research. Computationally, the algorithms are not overly complex, mainly involving minor calculations and a simple sorting algorithm to select the best candidate. Scalability of the complexity involved in the sorting

algorithms could potentially be avoided by enforcing a maximum number of destinations to be considered by a migrating vehicle.

3.5 Summary

This chapter introduced the first significant contributions of this thesis. Dynamic Clouds, in specific Vehicular Clouds, were discussed from a unique perspective: mobile nodes acting as Virtual Machine hosts. The case of a vehicular cloud is chosen as the main case study for this thesis. One large obstacle of such cloud implementations is the lack of existence of a working virtual machine management scheme for the vehicular cloud. That is an essential obstacle that must be overcome, due to the dynamic nature of the vehicular cloud and computational capacity constraints of the vehicular servers. Unlike the conventional cloud, where virtual machines are hosted on servers located in data centers, the vehicular cloud involves virtual machines hosted on moving vehicles. This means that there are frequent migrations needed to maintain levels of service for the cloud users.

However, MATLAB was selected as the tool of choice due to its robustness. A grid-like layout in an urban environment was selected as the environment to test the main contribution of this section – a novel Vehicular Virtual Machine Migration (VVMM) scheme. Several variations of the scheme were proposed. They will be modeled and evaluated based on several metrics in the following chapter of the thesis. The proposed VVMM schemes will show a favorable performance, and the mobility model will be shown to serve its function well, however, there are several assumptions that should be and will be addressed in later chapters. For the Simplified Mobility Model: the point of entry and point of exit must be different, transitions cannot be

revisited, congestion levels are based on rate of entry of vehicles, and vehicles are constantly in motion (unrealistic). As for the Initialization assumptions: vehicles can only host one type of VM (at the initial stage), the grid is initially empty and will eventually empty again, VM sizes are without units, and are arbitrarily selected. Finally, general assumptions: all migrations occur seamlessly if a valid destination is found, taking arbitrarily 25 seconds, only Vehicle-to-Vehicle migrations are considered, communication and networking constraints are neglected.

It is important to address all these assumptions and further strengthen the proposed algorithms. The next section of the thesis will present a refined mobility and initialization model, followed by a more advanced analysis of the migration schemes.

Chapter 4

Preliminary Analysis, Refined Mobility Model and Advanced Migration Schemes

4.1 Introduction

Dynamic clouds and specifically vehicular clouds, like conventional clouds, need to migrate virtual machines from one physical resource to another. Migration occurs for various reasons in the conventional cloud, but more unique to vehicular clouds, a migration is a much more critical and frequent process. The mobility and dynamic nature of the physical resources means that there is a consistently unpredictable availability of the resource, meaning an unpredictable level of accessibility offered to the subscriber trying to access the resource.

The dynamic cloud and specifically the vehicular cloud, to truly be worth pursuing, must emulate, even on some level, a traditional data center. If it is agreed that the RU (housing the main focal point of the cloud) is a static node, it can be assumed that it has a physically bounded region

of coverage. This is limited to the number of network-providing antennae that report to this RU. This would be considered one cloudlet. Many cloudlets would exist over a large city. For now, it is important to focus on one cloudlet.

A group of streets in the same region, falling under the coverage of multiple collaborating network towers can be considered the boundaries of the data center. The physical machines within the center, rather than static servers, are the vehicles currently within the boundaries of coverage. As vehicles leave the grid and new vehicles enter the grid, the goal is to keep the VMs within the boundaries. This would be the *main trigger* for migration. As vehicles leave the region, they must successfully migrate their load to another vehicle that is still in the region, thereby keeping the VMs in the same physical region. Should such a migration be unachievable, a vehicle can attempt to migrate its load to the infrastructure, temporary storage elsewhere, be it in the RU or some other location online (managed by the same cloud provider). Figure 4.1 shows a visual representation of the revised VC approach.

In this chapter, the performance of the schemes proposed in the previous chapter will be analyzed, based on several metrics. Consequently, the revised approach to the VC will be explored and the migration schemes will be further refined. The next section presents the metrics and simulation results used for analysis of the previously proposed schemes.

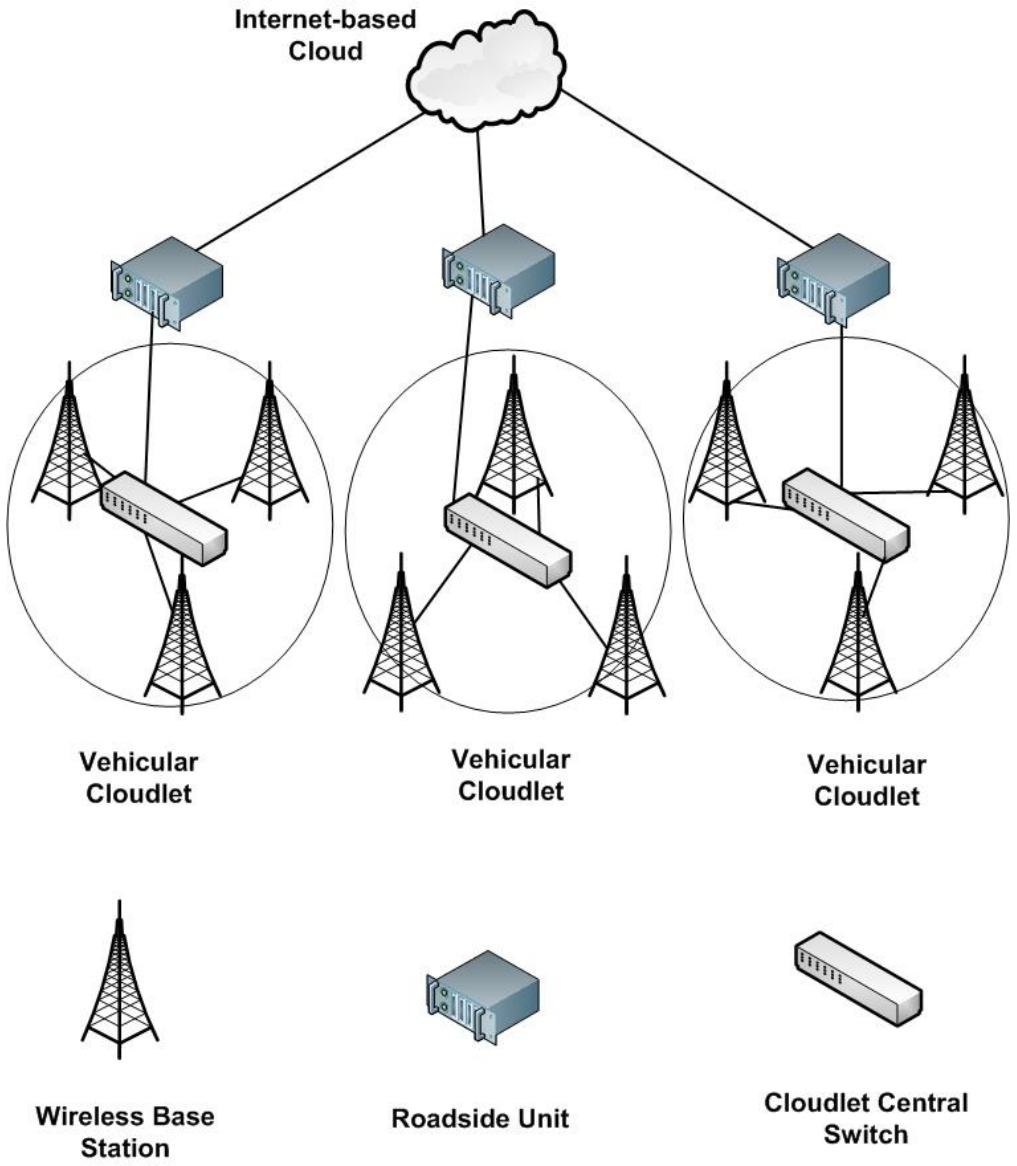

Figure 4.1: Revised Architecture for a Vehicular Cloud.

4.2 Metrics and Simulation Results

A MATLAB program was developed, using the previously discussed initializations and the three schemes to evaluate the performance of each scheme. Multiple runs were executed for each scheme. Each run, involves its own initialization process that starts from scratch, so every run

has 500 different cars, with different loads, speeds, start times, trajectories and migration points. Each congestion level was tested, with 30 iterations of each scheme being tested, with the 300 second observation window. Two main metrics were used to compare and analyze the scheme performance in the various scenarios.

A_{PD} = Average Percentage of Dropped VMs is formulated in Equation (4.1) as follows:

$$A_{PD} = \frac{\sum_{i=1}^{n} D_i}{\sum_{i=1}^{n} D_i + \sum_{i=1}^{m} S_i} \qquad (4.1)$$

where:

A_{PD} total percentage dropped,

D_i i^{th} dropped migration value (VM units),

S_i i^{th} successful migration value (VM units),

n total number of instances of a dropped migration,

m total number of instances of a successful migration.

The value of A_{PD} is computed and presented in Figure 4.2 for each congestion level and scheme. Also, the percentage improvement in A_{PD}, compared to VVMM-U, for VVMM-LW and VVMM-MA is shown in Figure 4.3.

The second metric used is A_{UF} = the Average Utilization Fairness (using Jain's Fairness Index) [JAI84]:

$$A_{UF} = \frac{1}{T}\sum_{t=1}^{300}\frac{(\sum_{i=1}^{n} WL_{ti})^2}{n \cdot \sum_{i=1}^{n} {WL_{ti}}^2} \qquad (4.2)$$

UF is the average fairness of the capacity utilization of the vehicles, over the 300 second observation window.

where:

WL_{ti} i^{th} car's workload (VM units) at time t,

n total number of cars in the grid at time t,

t time instant (seconds),

T total simulation duration.

The results show a notable performance improvement for VVMM-LW and VVMM-MA versus VVMM-U scheme. The greater improvement, logically, is found in VVMM-MA. There is also a clear performance improvement when in a high congestion, due to the availability of multiple migration destinations.

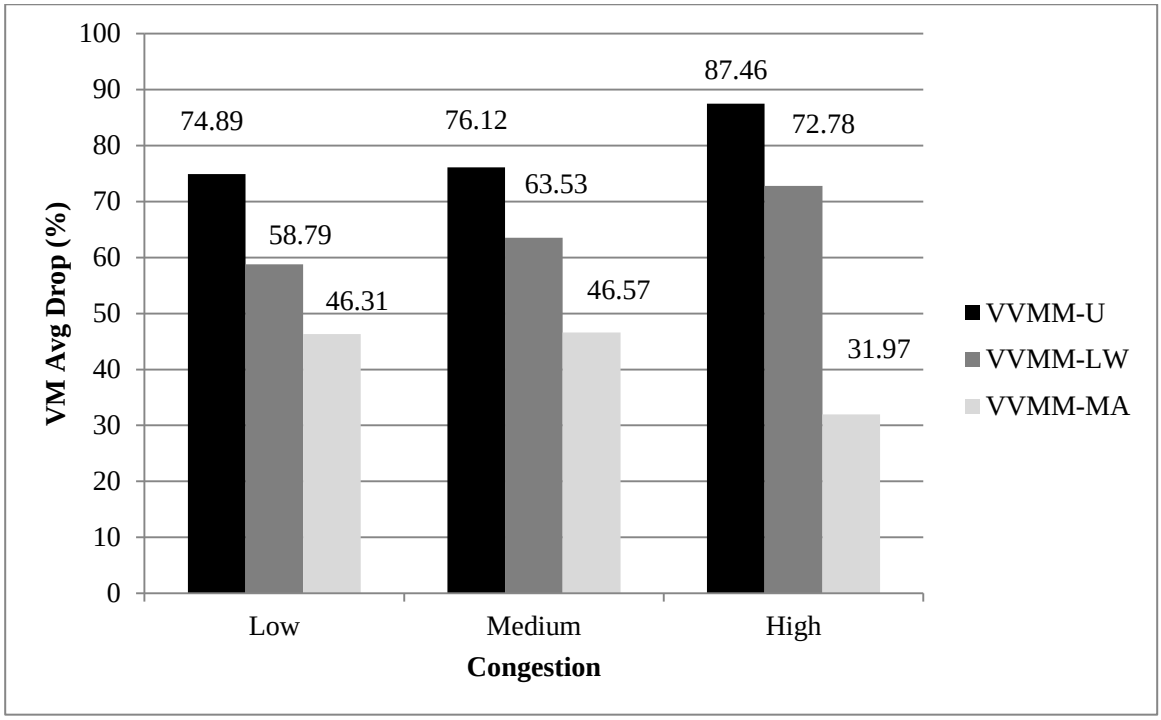

Figure 4.2: Average Percentage of Dropped VMs

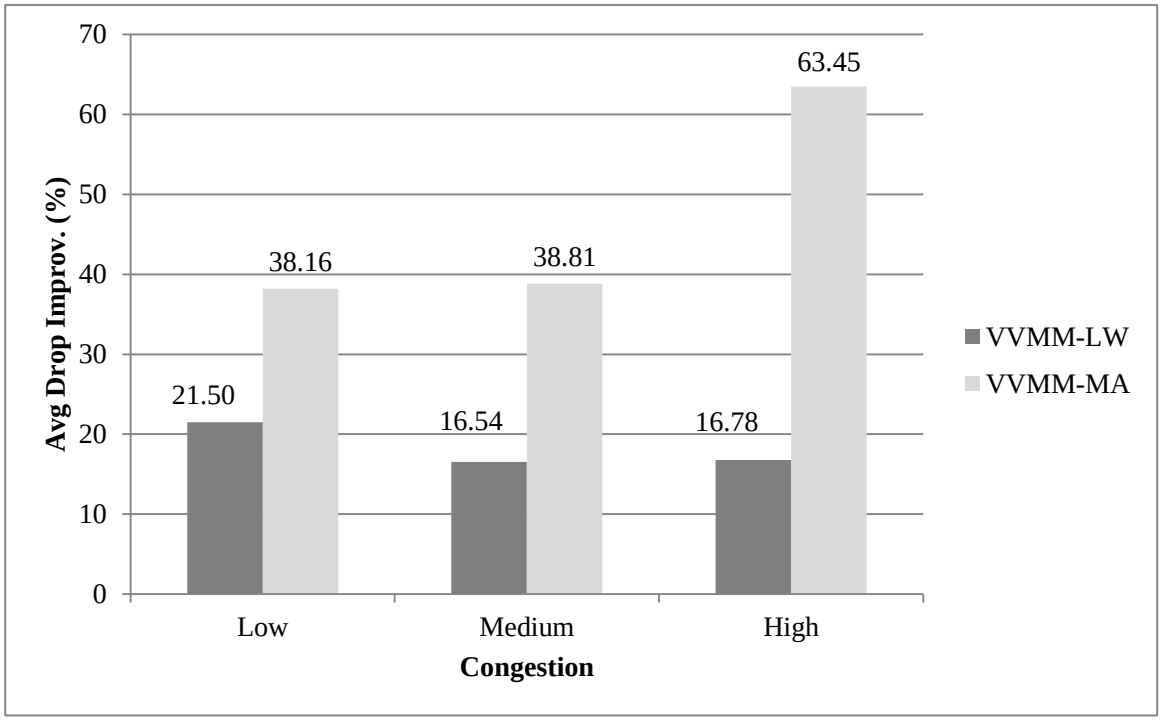

Figure 4.3: Percentage Improvement vs. VVMM-U

Using the second metric, fairness, it can also be seen in Figure 4.4 and Figure 4.5, that VVMM-MA also achieves a fairness value near 1, a clear improvement over the previous schemes. As shown in Figure 4.5, one iteration output, the fairness of the VVMM-MA scenario also experiences less fluctuation over the observation window. This being only one iteration, it is important to note that this is not accurately representative but more a sample of the fairness. It would be difficult to graphically show a similar figure for multiple seeds, as such Figure 4.4 is more representative figure.

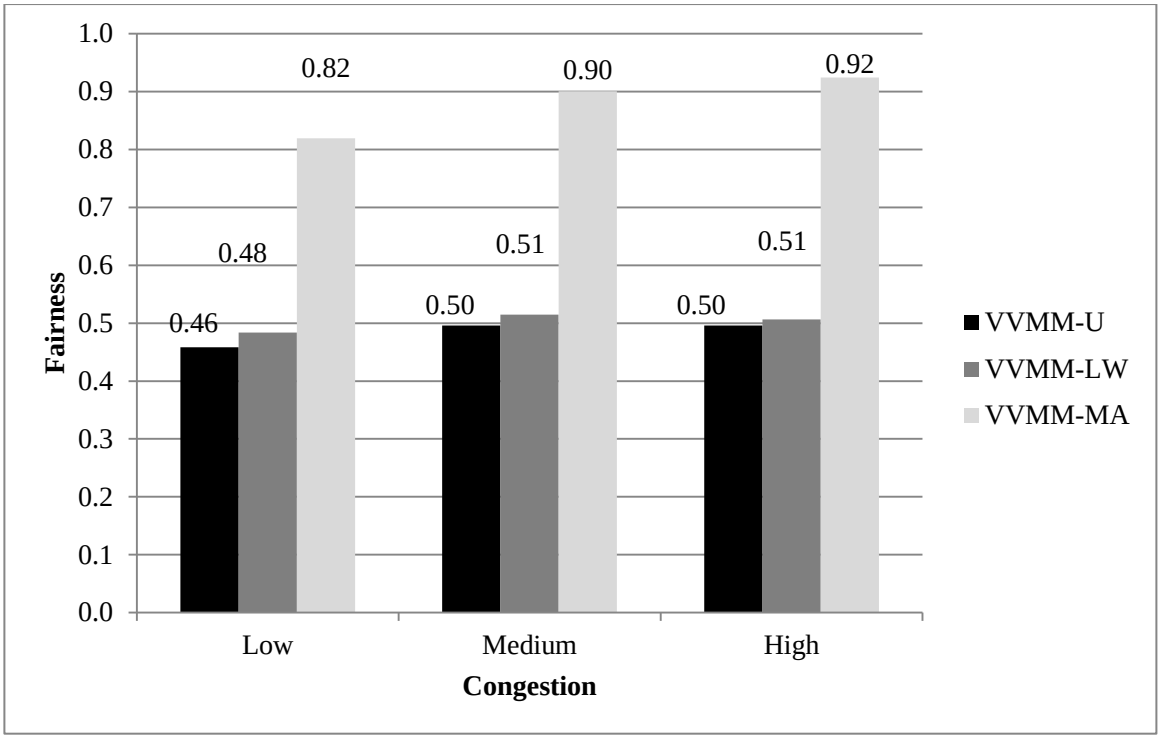

Figure 4.4: Average Utilization Fairness over 30 runs

In summary, the proposed migration schemes show a favorable performance based on preliminary analysis. They show the greatest performance in VVMM-MA with favorable conditions being high congestion. However, these preliminary results are neglecting many factors and require further refinement and analysis to solidify the contributions of this thesis.

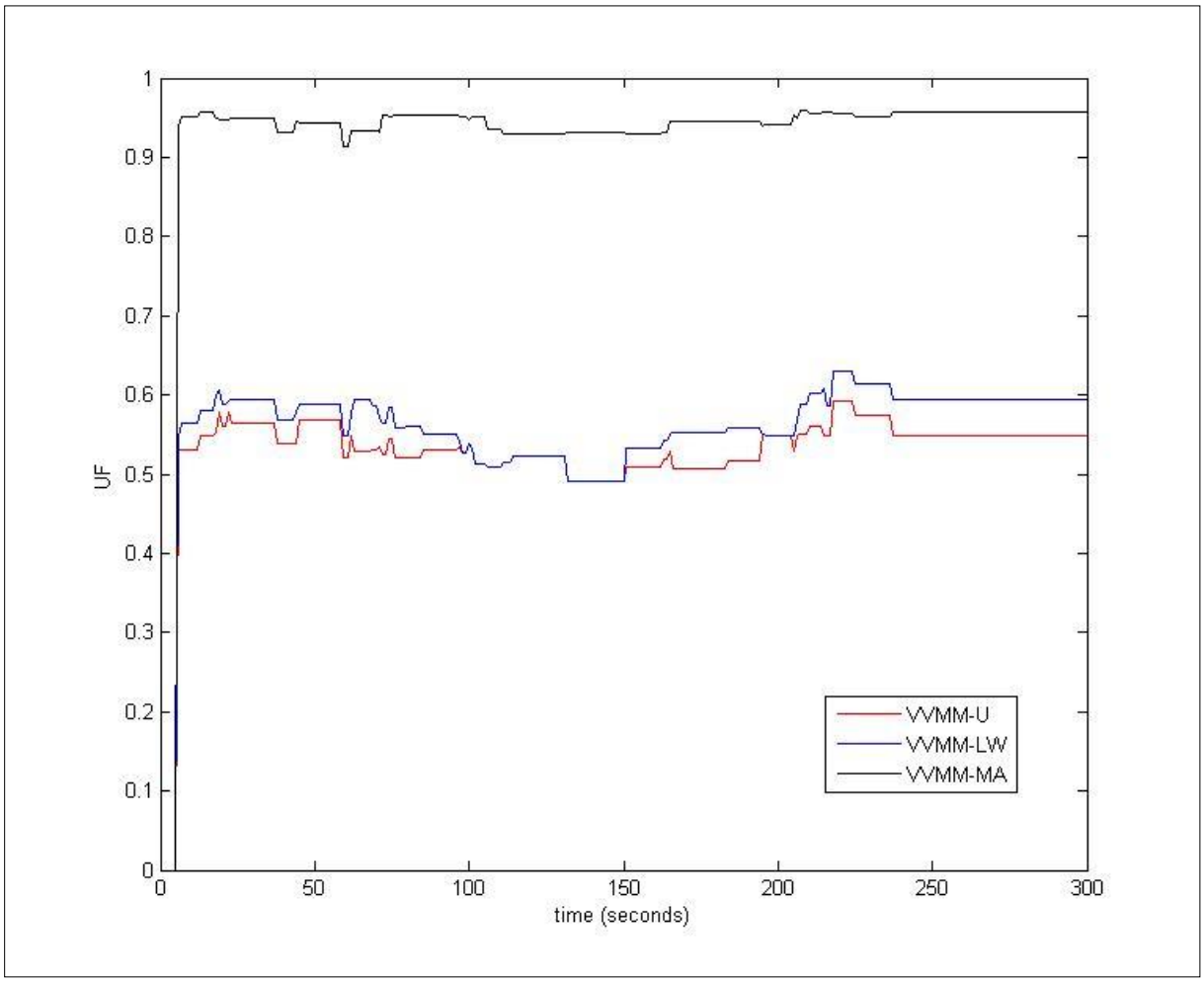

Figure 4.5: Average Utilization Fairness for a single Iteration

The next section addresses the assumptions made in the previous chapter as well as presents the refinements to the mobility model and the workload initialization.

4.3 Addressing Previous Assumptions

The previous chapter presented the preliminary mobility model, initialization process, and migration schemes [REF14]. This chapter then presented the metrics and simulation results of the schemes as proposed in the previous chapter [REF14].

In this chapter of the thesis several points are presented [REF16]:

1. A more specific approach to the VC architecture is envisioned,
2. The mobility model is revised,
3. The initialization process is refined,
4. Communication constraints and network congestion are taken into consideration,

The revised mobility model will be introduced next.

4.3.1 Revised Grid Topology and Advanced Mobility Model

The grid size is slightly expanded to 750m per transition. This now means the size of the grid is 1.5Km × 1.5Km. This is selected to represent a larger geographical area, covered by 3 wireless base stations. These base stations will depend on the protocol studied and will be discussed in a later section.

The mobility model is now more realistic, incorporating traffic lights at each of the 9 intersections. For each run, the grid is initialized for the duration of the simulated time with respect to the traffic lights. Each intersection is assigned a random sequence of red and green lights, of randomly selected durations of $t \in \{30, 45, 60\}$ seconds. This more accurately represents a real urban setting, unlike previous assumptions where vehicles would immediately leave an intersection after arrival.

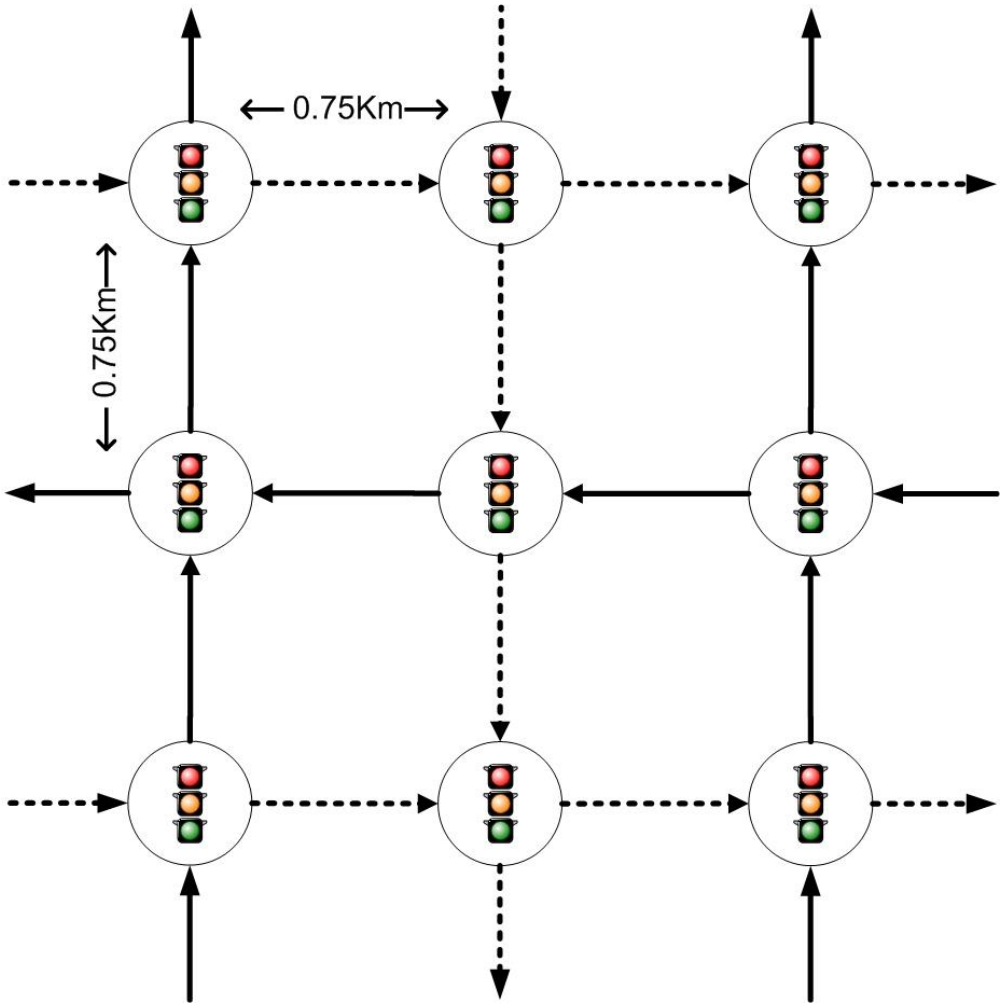

Figure 4.6: The layout of the revised grid

The mobility model is also redesigned to be more robust and random. Rather than utilizing the path initialization process proposed in the previous chapter, a different algorithm is used. Assumptions previously used are no longer applicable and the following limitations are addressed:

1. Points of entry and exit are no longer restricted from being the same,
2. Vehicles can start at any intersection,
3. Transitions can now be revisited,

4. Congestion levels are now more accurately modeled,
5. Vehicles can possibly park temporarily.

A vehicle enters the grid and begins to travel in a certain direction. As it moves along, its speed varies smoothly (but randomly) between 10 km/h and 50 km/h. As it reaches an intersection, it naturally decelerates, stops at a red light, and resumes its course once it turns green. If there are two possible directions to take, it randomly selects one of the two paths. However, unlike in the previous chapter, the vehicle is allowed to revisit a segment (between two intersections) up to 3 times. If a vehicle reaches a state when it has two options but both have been traversed 3 times, it selects the segment that is the nearest to an exit route for the grid.

Along its path, a vehicle can park at most once, for a duration of 10 minutes, or 1 hour, with a 20% or 5% probability respectively. As it is an urban/downtown setting, naturally, parking would more likely be a lower duration, hence the lower probability.

To model a realistic and dynamic flow of traffic, new vehicles are generated based on a probabilistic process considering the number of vehicles in the grid at each point in time. The number of vehicles in the grid is modeled so as to oscillate between two values ($0.8x$ and x) where $x \in \{25, 50, 100\}$ for $congestion \in \{low, medium, high\}$, respectively. The maximum speed is decreased as congestion increases, such that for $congestion = \{low, medium, high\}$, $max\ speed = \{50, 40, 30\}$, respectively. Leaving time of a vehicle is not known in advance as it is an output of the continuous trajectory generation. Thus, number of vehicles in the network varies every second. As the number gets closer to the lower limit $0.8x$, the probability a new vehicle arrival increases. On the other hand, as the number of vehicles gets closer to the upper limit x,

vehicle arrival rate decreases. Employing this methodology stabilizes the number of vehicles in the grid to reflect a certain congestion level.

4.3.2 Advanced Workload Initialization and Distribution

As in the previous chapter, the VMs are split into three sizes, light, medium and heavy, with size y, $2y$ and $4y$, respectively. However, to properly evaluate the performance of the proposed migration schemes, the VMs are not without units. The simulated test cases are $y \in \{10, 25, 50\}$ MB. Also, unlike the previous chapter, there are no restrictions on the initial workload a vehicle can host other than its maximum capacity. For the sake of simplicity, all vehicles are assumed to have the same capacity ($8y$). First the number of VM types hosted by the vehicle is randomly selected (from 1 to 3). Then, based on the number of VM types, any of the possible combinations is randomly selected. Finally, the exact number of each possible VM is randomly selected. These selections also include possibly initializing a vehicle to an idle load.

4.4 Refined and Expanded Vehicular Virtual Machine Migration Schemes

While previously, the communication protocols were unaddressed, this segment of the study assumes LTE will be used and evaluated for the refined model [3GP12][BAY11]. It is important to note that this is merely a current technology, and in the future, more advanced protocols would need to be studied. However, as the protocols increase in speed and robustness, the performance of the proposed algorithms will improve also.

Based on the LTE bandwidth, the available upload bandwidth serving the grid is now 3 × 50Mbps. The download bandwidth is higher than the upload and as such is not the main constraint. The bandwidth is a main factor that affects the probability of a successful migration. As more vehicles migrate concurrently, more bandwidth is used. The cumulative bandwidth utilized by the VC must not exceed the maximum capacity of the network covering the region. It is important to remember that as protocols advance, this capacity will increase and more vehicles can migrate concurrently. For the sake of this case study however, LTE bandwidth is used [3GP12].

Figure 4.7 shows Algorithm 2, the pseudocode that forms a basis for the proposed refined VM migration algorithms for the vehicular cloud. The goal is to select a migration destination for an *exiting* vehicle, in order to avoid any data loss. This refinement to the previous approach now facilitates a Vehicle-to-Infrastructure (V2I) migration, in the event that no viable vehicular host is found at migration time. While an unfavorable migration, it is more acceptable than a dropped VM, and as such, a dropped/failed migration will only define a vehicle that exits the grid having had no chance to unload its VM, be it to another vehicle or the infrastructure. Although the data should be retained in the grid, should no potential destination vehicle for migration can be found, the source vehicle will have to upload the VM to the infrastructure and this could entail storing the VM at a local RU, or it could mean uploading the VM to the Internet cloud. While the process outlined in Algorithm 2 is valid for all the algorithms proposed here, each has unique *search criteria* to be discussed next.

There are 3 refined versions of the previously proposed algorithms, benchmarked by the generic Algorithm – Vehicular Virtual Machine Migration with Uniform host selection (VVMM-

U). Each is explained below. They are named as they originally were but throughout the remainder of this section these abbreviated names refer to the revised algorithms.

VVMM-U: The benchmark algorithm used to gauge the performance of the proposed algorithms is VVMM-U. This follows the process described in Algorithm 2. The search criteria involve merely selecting a vehicle from the pool of candidates, initially searching at the street level, and at the grid level if there are no vehicles sharing the street. This selection process does not take into account any of the factors that could increase the chances of a successful migration. The migrating vehicle treats all potential candidates equally. The possibility that the selected migration destination may not have enough space, or enough time to receive the migration, is unpredictable and totally randomized. In theory, should there be absolutely no other vehicles to choose from, randomly, then the V2I migration would be resorted to, but this is only possible if there is only one vehicle in the grid, and such a case is not considered.

1: {*src* : migration source} // exiting vehicle
2: {*dst* : migration destination} // result of algorithm
3: Begin
4: *pool* : pool of potential candidates
5: let *pool* = vehicles in same street as *src*
6: if (|*pool*| > 0) then
7: let *pool* = vehicles in *pool* satisfying *search criteria*;
8: if (|*pool*| > 0) then
9: let *dst* = *V*, vehicle in *pool*, nearest to *src*
10: else
11: let *dst* = *I* // vehicle will attempt to migrate to infrastructure
12: end if
13: else

```
14:       let pool = vehicles in the same grid
15:       Go to line-6
16:   end if
17:   // destination selected, attempt migration
18:   if (dst = V ) then
19:       if (V  will migrate before src completes) then
20:       return Unsuccessful migration
21:       else
22:       if (V  does not have sufficient space) then
23:            return Unsuccessful migration
24:          else
25:            if (network has insufficient bandwidth) then
26:              return Unsuccessful migration
27:            else
28:            return Successful migration
29:            end if
30:       end if
31:       end if
32:   else
33:       if (dst = I ) then
34:            if (network has insufficient bandwidth) then
35:              return Unsuccessful migration
36:            else
37:             return Successful migration (to infrastructure)
38:            end if
39:       else
40:       return Unsuccessful migration
41:       end if
42:    end if
43:    End
```

Figure 4.7: Algorithm 2 - General VM Migration Pseudocode

4.5 Detailed Description of Revised Vehicular Virtual Machine Migration Schemes

VVMM-LW: The first modification to Algorithm 2 is Vehicular Virtual Machine Migration with Least Workload (VVMM-LW). Building upon the scheme described in the previous chapter (using the same title) but now referring to the revised approach. The algorithm involves search criteria that consider both the workload of the potential destinations and their distance from the migrating vehicle, to make a more informed decision. The soon-exiting vehicle, needing to migrate, will search for vehicles that have enough resource space to host its VM load. It will first search for candidates within its street. If there are multiple viable candidates, the source selects those with the least workload utilization. Should there be several potential candidates with equal resource space, it will select the one nearest in terms of distance. If there are no vehicles in the same street, it will search the overall cloudlet (grid). It will select the nearest among the viable candidates to migrate to. Should no viable candidates exist anywhere in the cloudlet, it will attempt to migrate to the infrastructure. This algorithm does not consider time needed for migration, and is likely to suffer from an unsuccessful migration due to the selected destination not having enough time remaining in the grid to receive the migration. However, VVMM-LW is expected to outperform the VVMM-U algorithm in terms of VM drop rate as it uses *a priori* knowledge on the workload profiles of potential destinations.

VVMM-MA: In order to avoid unsuccessful migrations that occur due to the destination not having enough time remaining, Vehicular Virtual Machine Migration with Mobility-Awareness (VVMM-MA) refines the search criteria further. The migrating vehicle takes the amount of time

potential destinations have remaining in the grid into consideration. If a vehicle will not be in the grid long enough to receive the workload migration, it will not be considered as a viable candidate. Once a vehicle is forecasted to have enough time to receive the load, it is then checked for VVMM-LW criteria (resource space followed by proximity to the source). It is worthwhile noting that a viable candidate should have enough residual in-network time and memory/compute capacity to receive the workload migration. However, there are some cases when the migration will be marked as unsuccessful. Once again, same-street vehicles are considered first, then vehicles at the grid-level, finally the infrastructure is targeted failing any of the former options.

In order to decide the time of migration (specifically the instant a vehicle should start migrating), a vehicle considers its current resource workload and the expected total duration of migration based on an expected upload speed. If a vehicle is selected as a destination for migration, the destination workload will gradually increase as data is moved from source to destination. As the onboard workload increases, the time needed to migrate will increase accordingly. This will force the vehicle to set its expected migration time to an earlier point. A situation may occur where the updated migration time needed is greater than the residual in-network (or in-grid) time of the destination. In such a scenario, the destination vehicle will recalculate, and in its own interest, need and begin to have its new load migrate prior to completion of the ongoing migration. This will result in an unsuccessful migration for the source. This can be described as a '*first hop*' failure.

Another case could occur when the initial migration is successful and the destination now has more workload but its remaining in-network time is not enough to have its new workload migrate. This can be described as a '*second hop*' failure.

MDWLAM: To avoid the 'first hop' problem, the source vehicle must select a destination with remaining time in the grid, greater than or equal to the summation of the time necessary for the source to migrate its load, and for the destination to migrate its load. In order to avoid the 'second hop' problem, the migrating vehicle must further consider the time necessary for the destination vehicle to migrate both its current load, and the migrated load. As such, Mobility and Destination Workload Aware Migration (MDWLAM) utilizes Equation (4.3), to calculate the cutoff time for the search criteria.

$$Cutoff = 2(T_s) + T_d \qquad (4.3)$$

where:

T_s time needed to migrate source workload (s),

T_d time needed to migrate destination workload (s).

Only vehicles remaining in the grid longer than $Cutoff$ are considered as viable candidates. The source vehicle consequently selects the vehicle with the longest time remaining among the viable candidates. The idea behind this selection is to maximize data retention in the grid. In case there is a tie between multiple candidates for migration, the selection among them will involve the remaining criteria proposed by the VVMM-LW, i.e., resource space followed by proximity to the source. As all algorithms, same-street vehicles are considered first, then vehicles at the grid-level, finally the infrastructure is targeted failing any of the former options.

This algorithm involves the most well-informed decision in selecting a migration destination, and as such it is expected to outperform the previous algorithms.

4.6 Conclusion

The previous chapter presented the preliminary mobility model and initialization setup for the MATLAB environment. It also proposed the main migration schemes for the Virtual Machines in a Vehicular Cloud. This chapter presented the metrics and analysis of those previously proposed schemes in the described environment. The preliminary results showed a favorable performance for VVMM-LW and VVMM-MA, based on several metrics, with VVMM-MA proving to perform better than VVMM-LW.

However, the preliminary schemes involved several assumptions and simplifications. This chapter addressed those assumptions and revisited the entire process to propose and develop code for:

Advanced and more realistic mobility model:

The point of entry and point of exit are independent and can be the same. Intersections can be starting points, transitions can be revisited (with a limit on the number of revisits imposed), and congestion levels are based on a specific range. Flow of vehicles in and out is increased and decreased to maintain a steady congestion level within that range, vehicles can park temporarily, and intersections have red lights that can occur for various durations.

Advanced initialization and workload distribution:

Vehicles can host any number and type of VMs (at any stage) and the grid is more representative of a true vehicular environment, with vehicles entering and leaving continuously, while maintaining a specific level of congestion. VM sizes are now tangible values, where the smallest unit can be 10MB, 25MB or 50MB.

General refinements:

Bandwidth constraints are now taken into consideration, specifically using LTE speeds as an example, bandwidth constraints now impact capacity, number of concurrent migrations, as well as migration time, transmission delays based on workload size and data rate, but the impact of such constraints will still be studied in the next chapter, not only Vehicle-to-Vehicle migrations are considered, but now a last resort to avoid a dropped migration is possible, where a vehicle can migrate to the infrastructure, a Vehicle-to-Infrastructure (V2I) migration. Revisions to previously proposed migration schemes are made to incorporate the more advanced model, especially the network and bandwidth constraints. A final and further advanced scheme is proposed, MDWLAM, which will be analyzed in the next chapter, but should surpass the performance of the previous algorithms.

The next chapter will present more specific metrics and simulation cases to analyze the proposed advanced algorithms. More importantly it will also present a component that has been neglected thus far in the thesis: Non-ideal communication cases. Potential channel imperfections, due to communication congestion, and signal attenuation, due to source-destination distance, will be considered to further analyze the performance of the proposed schemes.

Chapter 5

Analysis of Advanced Schemes Incorporating Non-Ideal Communication

5.1 Introduction

The previous chapter presented the revised and more advanced mobility model, initialization setup and algorithms. It is important however, to analyze the performance of these revisions, using a new set of metrics. Furthermore, it is very important to incorporate the effects of non-ideal communication channels to avoid a simplified view of the algorithms proposed. The algorithms will be simulated with the revised mobility model and while incorporating a non-ideal setting, and will show a favorable performance.

5.2 Detailed Performance Evaluation

In order to evaluate the feasibility and performance of each of the proposed algorithms, a set of MATLAB simulations was conducted, with several metrics to be observed.

The previous chapter involved some preliminary simulations with several assumptions and simplifications: A simplified mobility model was used, restricting vehicles to select between a set of predetermined paths. VMs were of no tangible units and arbitrarily set to Light, Medium and Heavy. Vehicles were initialized to host only one type of VM. The system was observed for only a 5 minute window. Migration delays were all assumed to be fixed to 25 seconds, regardless of the VM size. The system was assumed to have infinite bandwidth, disregarding network congestion and capacity limitations of existing protocols. The MDWLAM algorithm was not tested. In this chapter, these restrictions and simplifications are addressed and the simulation settings have been revised as presented in the next subsections.

5.2.1 Simulation Settings and Metrics

The simulation model involves several main modules. First, the vehicle mobility model based on a certain topology is set. The vehicles are then assigned VM workloads. Finally, the VVMM algorithm is executed.

The mobility model, topology, VM initialization and subsequent application of the various proposed algorithms have been simulated with MATLAB. The simulations are used to evaluate the performance of each of the proposed algorithms, based on several metrics that are collected and analyzed. Based on the performance of each algorithm, restrictions can be heuristically proposed and tested, to further enhance system performance.

5.2.1.1 Metrics

The various metrics used to measure system performance include:

The percentage of dropped migrations (***D***) is calculated as shown in Equation (5.1).

$$D = \frac{\sum_{t=1}^{T} d_t}{\sum_{t=1}^{T} d_t + \sum_{t=1}^{T} s_t} \quad (5.1)$$

where:

T simulated time (s),

d_t total dropped VM workload, at time t in the grid (MB),

s_t total successfully migrated VM load, at time t, throughout the entire grid (MB).

The percentage of successful migrations that were in fact V2I migrations (***V***) is calculated as shown in Equation (5.2).

$$V = \frac{\sum_{t=1}^{T} v_t}{\sum_{t=1}^{T} s_t} \quad (5.2)$$

where:

T simulated time (s),

v_t total VM load, successfully migrated to the infrastructure, at time t, throughout the entire grid (MB),

s_t total successfully migrated VM load, at time t, throughout the entire grid (MB).

The average percentage utilization of the collective available resources (***R***) is calculated as shown in Equation (5.3).

$$R = \frac{\sum_{t=1}^{T} l_t}{T \cdot l_m} \qquad (5.3)$$

where:

T simulated time (s),

l_t total VM load hosted on all the vehicles currently the grid, at time t (MB),

l_m maximum VM load capacity the grid can offer, based on the expected average number of vehicles at the congestion level simulated (MB/s).

The average percentage utilization of the available upload bandwidth (***UL***) is calculated as shown in Equation (5.4).

$$UL = \frac{\sum_{t=1}^{T} ul_t}{T \cdot ul_m} \qquad (5.4)$$

where:

T simulated time (s)

ul_t total upload bandwidth utilized, at time t, by all vehicles in the grid (MB/s)

ul_m maximum upload bandwidth available in the grid (MB/s)

The average percentage utilization of the available download bandwidth (***DL***) is calculated as shown in Equation (5.5).

$$DL = \frac{\sum_{t=1}^{T} dl_t}{T \cdot dl_m} \qquad (5.5)$$

where:

T simulated time (s)

dl_t total download bandwidth utilized, at time t, by all vehicles in the grid (MB/s)

dl_m maximum download bandwidth available in the grid (MB/s)

All simulations are run for a constant $T = 3000$s. It is also assumed that the large area simulated, is covered by 3 LTE base stations, offering an upload bandwidth of 6.25MB/s (50Mb/s) and download bandwidth of 12.5MB/s (100Mb/s) each, resulting in a maximum available upload bandwidth $ul_m = 18.75$MB/s and download bandwidth $dl_m = 37.5$MB/s. The uplink being the higher constraint coupled with the guarantee that it will always be higher than any downlink communication created by the cloud, the focus will be on the uplink. Finally, as the number of vehicles is set to oscillate between two values, $0.8x$ and x (as presented in Section 4.3.1), if the computing resources of each vehicle is fully utilized, it would carry a load of size $8y$ (as presented in Section 4.3.2), resulting in the maximum VM load capacity that the grid can offer is bounded below by $l_m = (8y)(0.8x)$, and is bounded above by $l_m = (8y)(x)$ where x depends on the congestion level and y depends on the selected VM size for simulation. ***R*** is the calculated to be the average of the case using the lower and the upper limit of l_m. It is also assumed that a typical upload (and consequently download) speed is 1MB/s, for an individual vehicle attempting migration. These details are summarized in Table 5.1.

Table 5.1 – Simulation Settings

Variable	Value
Grid Size	1.5Km × 1.5Km
# of LTE Base stations	3
T	3000s
ul_m	18.75MB/s
dl_m	37.5MB/s
Avg UL/DL Speed	1MB/s
Low Congestion	20 ~ 25 vehicles
Medium Congestion	40 ~ 50 vehicles
High Congestion	80 ~ 100 vehicles

5.2.1.2 Simulation Stages

While measuring the above metrics, the model was simulated in several stages, for various degrees of congestion as well as different VM sizes. Initially, the performance of the various proposed algorithms is analyzed without any imposed restrictions on the available resources allocated to VM loads. As calculated using Equation (5.3), the average percentage utilization of the collective available resources (***R***) can be regarded as an indicator of the point of saturation, where the grid is unable to accommodate further loads. In a subsequent set of simulations, this value is set as an imposed, administrative limit, to the maximum load that can be allotted to the available resources in the grid. Such a restriction is analogous to a conventional cloud administrator managing the utilization of the available resources. Naturally the administrator is aware of the available resource 'space' and will only lease out a load that the resources can handle, to maximize performance. In this case, it is projected that such an imposed restriction, coupled with the utilization of one of the various proposed algorithms, should maximize performance.

5.3 Simulation Results

The various metrics and model settings introduced in the previous section, were used to simulate several scenarios. The simulations involved:

1. Vehicular Congestion: Low, Medium, High
2. VM Size: 10MB, 25MB, 50MB
3. Resource Restrictions: Unrestricted, Restricted

The breakdown of all the scenarios can be found in Table 5.2.

Table 5.2 – Simulated Scenarios

Congestion (# vehicles)	VM Size (MB)	Restriction (%)
Low (20 ~ 25)	10	UR
Low (20 ~ 25)	10	64
Low (20 ~ 25)	25	UR
Low (20 ~ 25)	25	50
Low (20 ~ 25)	50	UR
Low (20 ~ 25)	50	43
Medium (40 ~ 50)	10	UR
Medium (40 ~ 50)	10	66
Medium (40 ~ 50)	25	UR
Medium (40 ~ 50)	25	56
Medium (40 ~ 50)	50	UR
Medium (40 ~ 50)	50	46
High (80 ~ 100)	10	UR
High (80 ~ 100)	10	66
High (80 ~ 100)	25	UR
High (80 ~ 100)	25	56
High (80 ~ 100)	50	UR
High (80 ~ 100)	50	42

Each individual scenario was run for over 30 seeds and the metrics collected were averaged over all the seeds. “UR” refers to unrestricted in Table 5.2. The following sections present each metric to compare and analyze the performance of the proposed algorithms. For clarity in the figures, MDWLAM is shortened to MDW. Also note that a 95% confidence analysis was conducted and is shown in the figures (except if the value for Δ was less than 1%).

5.3.1 Effect of Varying Congestion Levels and VM Size on the Percentage of Dropped Migrations (D)

A VVMM algorithm is desired to have a low percentage of Dropped Migrations (***D***). Aside from the other metrics, an algorithm should most importantly have a low ***D***. While, for example, a low ***UL*** is also preferred, this should not come at the expense of a higher ***D***. This is due to ***D*** directly implying a loss of cloud data, potentially violating the SLA, while unfavorable values of the other metrics merely imply undesired grid conditions. In general, lower values of ***D***, ***V*** (to minimize RU involvement), ***UL***, ***DL***, are preferred whereas higher values for ***R*** are preferred.

There is a general trend of a decrease in unsuccessful migrations, as the algorithms become more advanced. This can be observed in Figures 5.1, 5.2, and 5.3, logically, showing that as the algorithm complexity increases, the ***D*** decreases. There is also a clear reduction as the restrictions to maximum data allowed are imposed. Figure 5.1 shows the ***D*** for 4 algorithms, for 3 congestions, both Unrestricted (labeled ‘UR’ in the figures) and Restricted (labeled ‘R’ in the figures), at a VM Size of 10MB. Figures 5.2 and 5.3 show the same, for 25MB and 50MB respectively.

It is important to elaborate what would appear to be an anomaly with VVMM-LW, performing ‘better’ in the unrestricted scenarios. However, this is valid behavior due to the following reason.

In the restricted case, there are more empty cars available. This means that the likelihood of finding a vehicle satisfying the workload requirements of the VVMM-LW scheme, are very high. This is reflected in lower V2I migrations in the restricted case, observable in the next subsection. This contributes to an overall higher percentage of drops, as V2I migrations in the VVMM-LW scheme are guaranteed success. Furthermore, in some aspects, an empty car may not necessarily be as appealing as one that is not empty, but has enough space to host the migration, in the VVMM-LW scheme. This is unique to the VVMM-LW scheme, as migrating vehicles only consider the workload of potential destinations, and not their time remaining in the grid. A vehicle that has a workload, and is not currently migrating, is more likely to have more time remaining in the grid than an empty vehicle. However, the difference in performance is quite minor, and the more advanced techniques rectify this anomalous behavior.

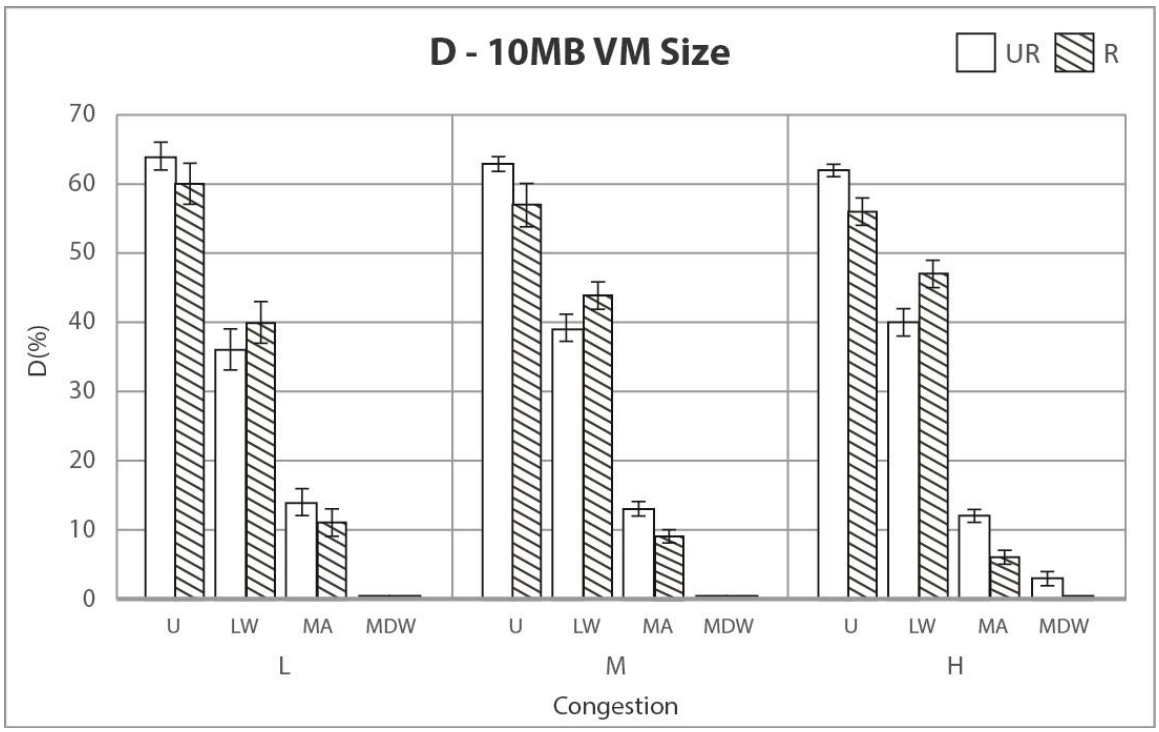

Figure 5.1: Percentage of dropped migrations vs. Congestion – 10MB VM Size

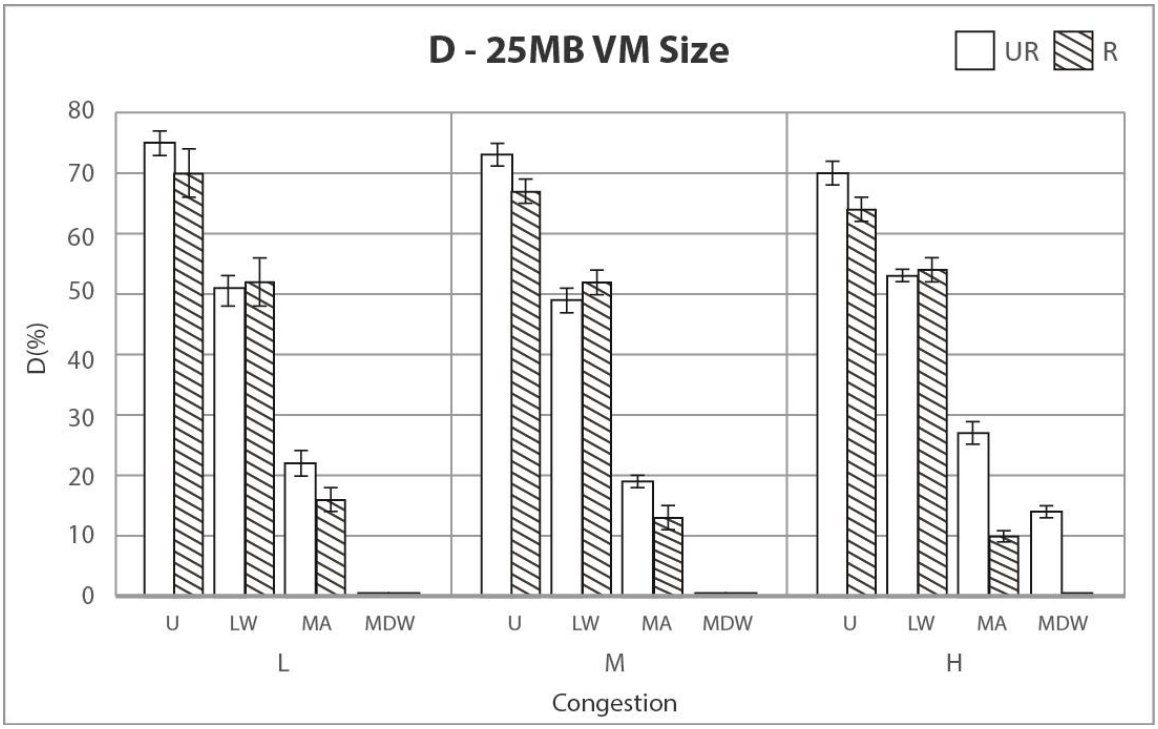

Figure 5.2: Percentage of dropped migrations vs. Congestion – 25MB VM Size

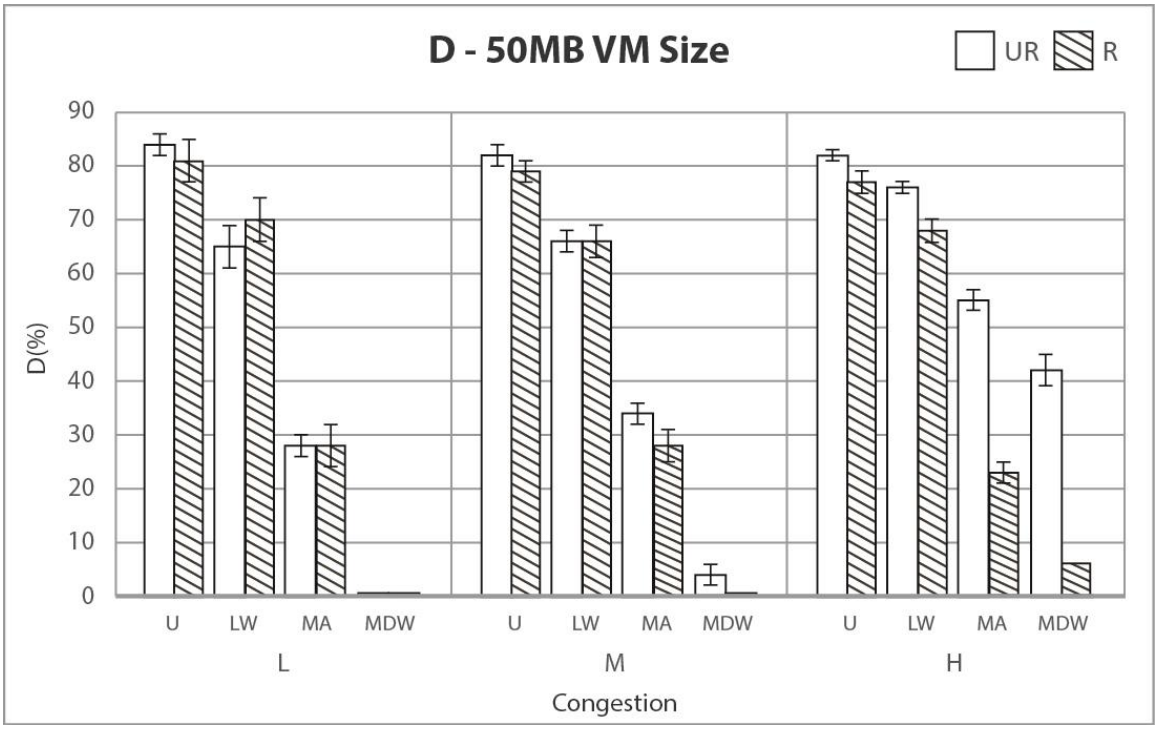

Figure 5.3: Percentage of dropped migrations vs. Congestion – 50MB VM Size

5.3.2 Effect of Varying Congestion Levels and VM Size on the Percentage of V2I migrations (V)

As shown in Figures 5.4 to 5.6, the V2I migrations increase (for the unrestricted scenarios), as the algorithm becomes more advanced. This is because the source vehicles make a more informed decision, now realizing that rather than migrate to a vehicle that will not meet the search criteria, it will instead migrate to the infrastructure. The restrictions imposed drastically decrease the V2I migrations, also giving a more favorable performance, as the algorithm becomes more advanced. Figure 5.4 shows the $\boldsymbol{V}$ for 4 algorithms, for 3 congestions, both Unrestricted (UR) and Restricted (R), at a VM Size of 10MB. Figures 5.5 and 5.6 show the same, for 25MB and 50MB respectively.

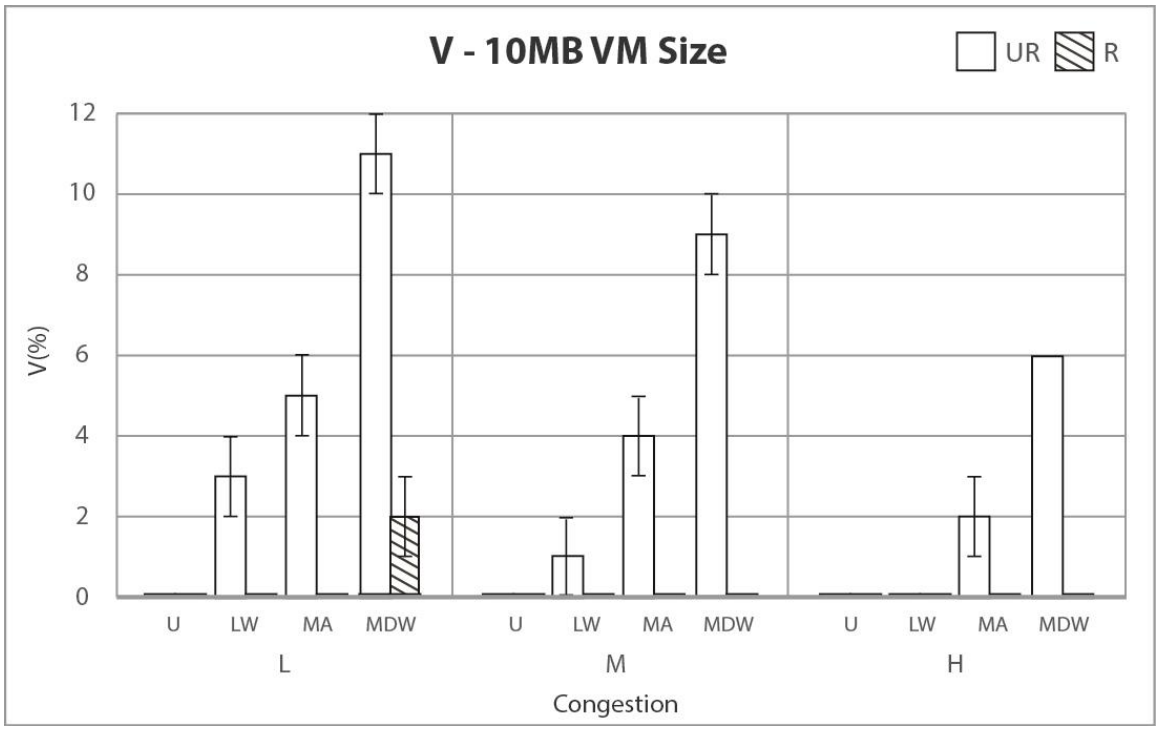

Figure 5.4: Percentage of V2I migrations vs. Congestion – 10MB VM Size

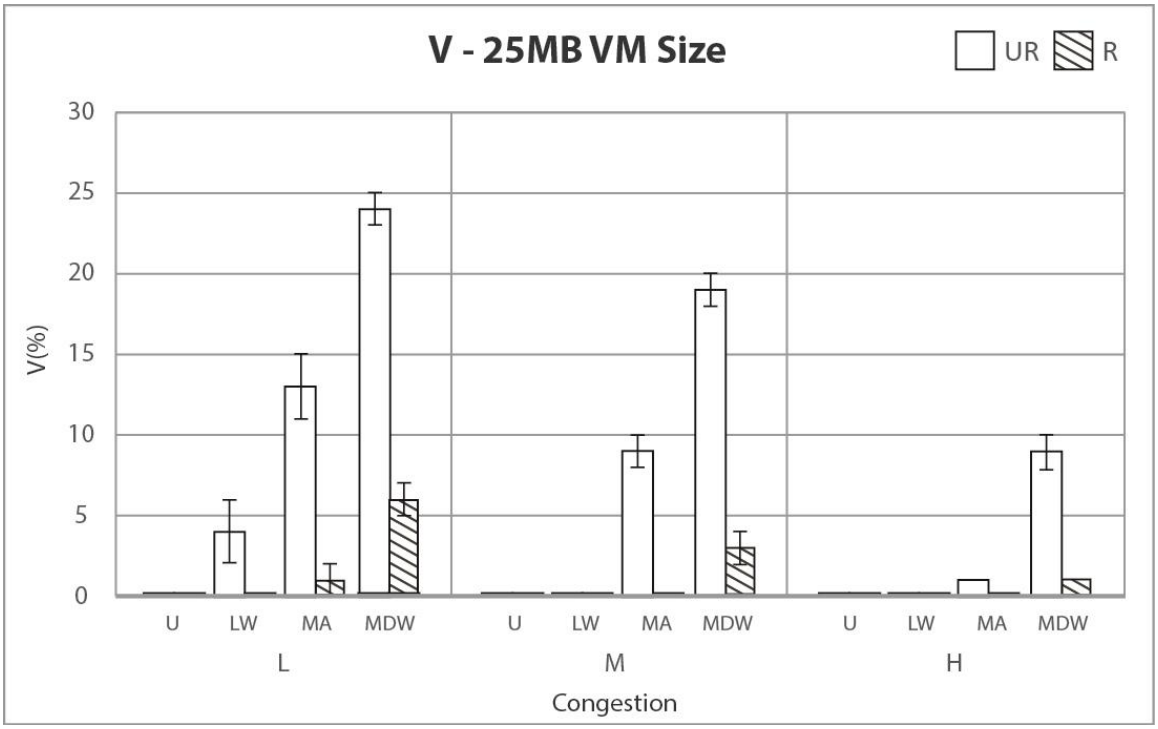

Figure 5.5: Percentage of V2I migrations vs. Congestion – 25MB VM Size

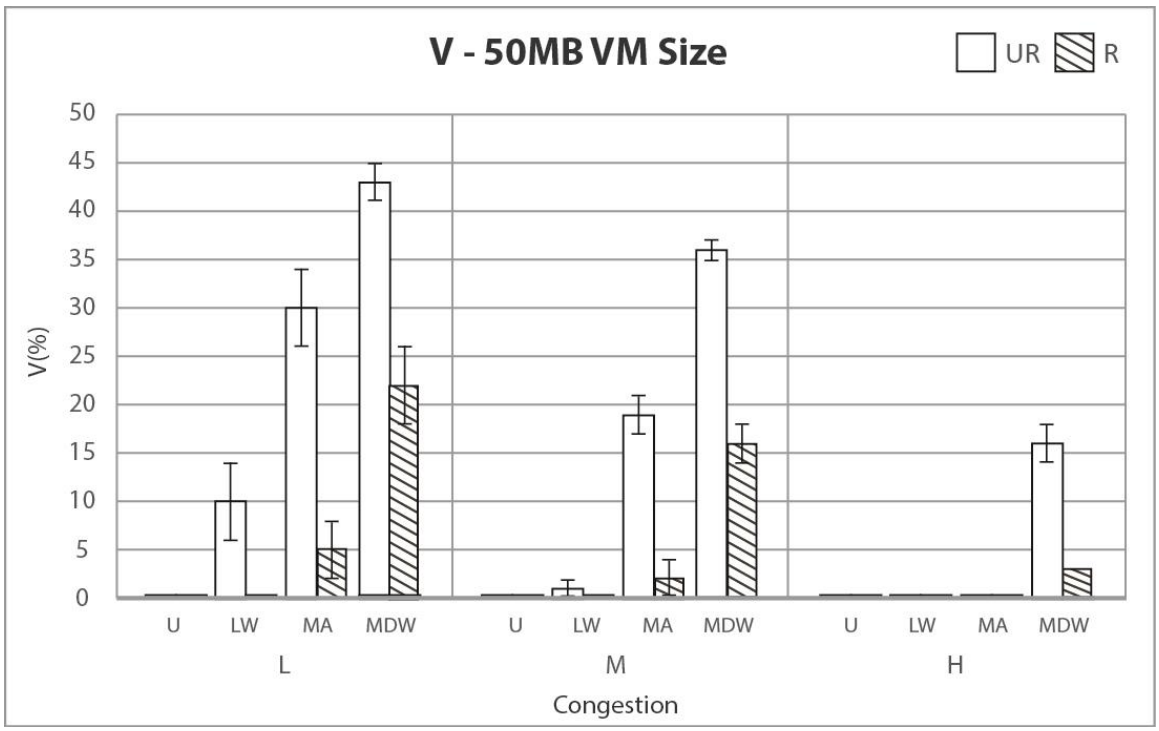

Figure 5.6: Percentage of V2I migrations vs. Congestion – 50MB VM Size

5.3.3 Effect of Varying Congestion Levels and VM Size on the Average Percentage Utilization of Collective Available Resources (*R*)

The resource utilization ***R*** follows a different trend as opposed to the other metrics; it increases as the algorithm becomes more advanced. However, and yet as expected, the imposed restrictions cause the utilization to decrease. This is a tradeoff that is acceptable in order to minimize the two previous metrics, drop percentage and V2I percentage. Figure 5.7 shows the ***R*** for 4 algorithms, for 3 congestions, both Unrestricted (UR) and Restricted (R), at a VM Size of 10MB. Figures 5.8 and 5.9 show the same, for 25MB and 50MB respectively.

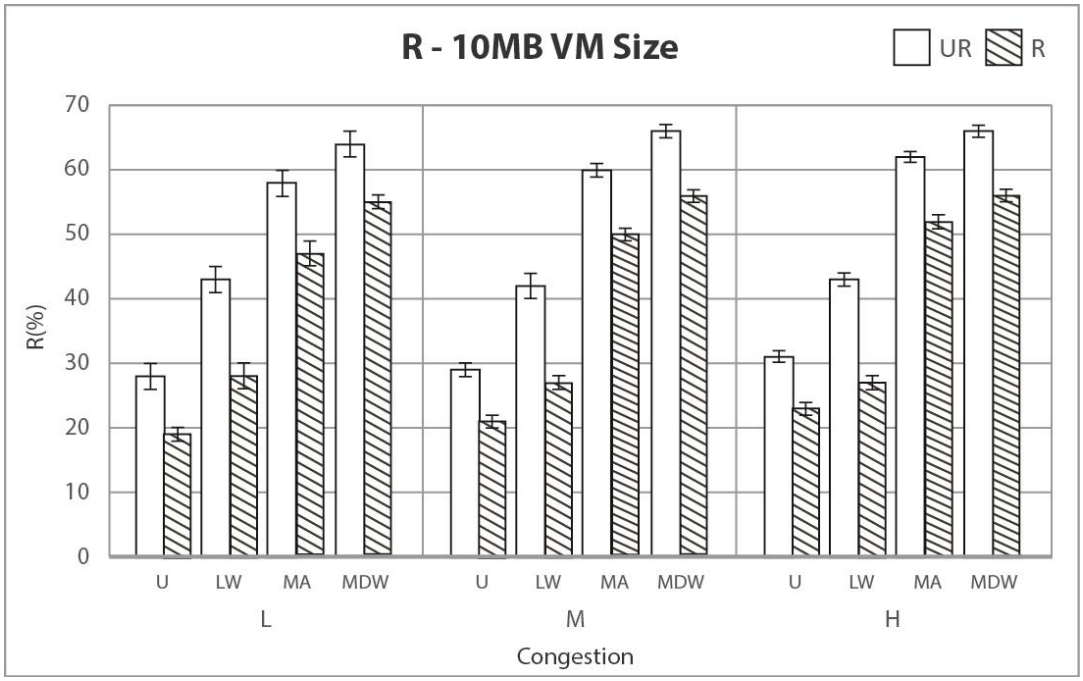

Figure 5.7: Average percentage resource utilization vs. Congestion – 10MB VM Size

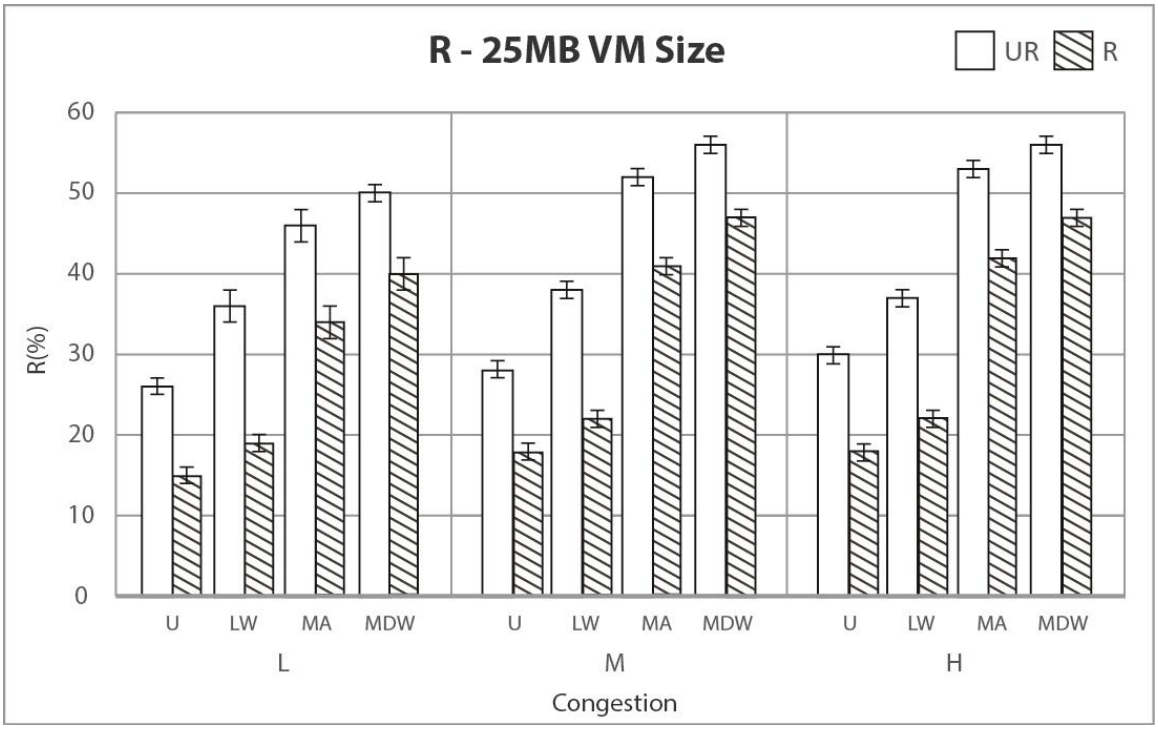

Figure 5.8: Average percentage resource utilization vs. Congestion – 25MB VM Size

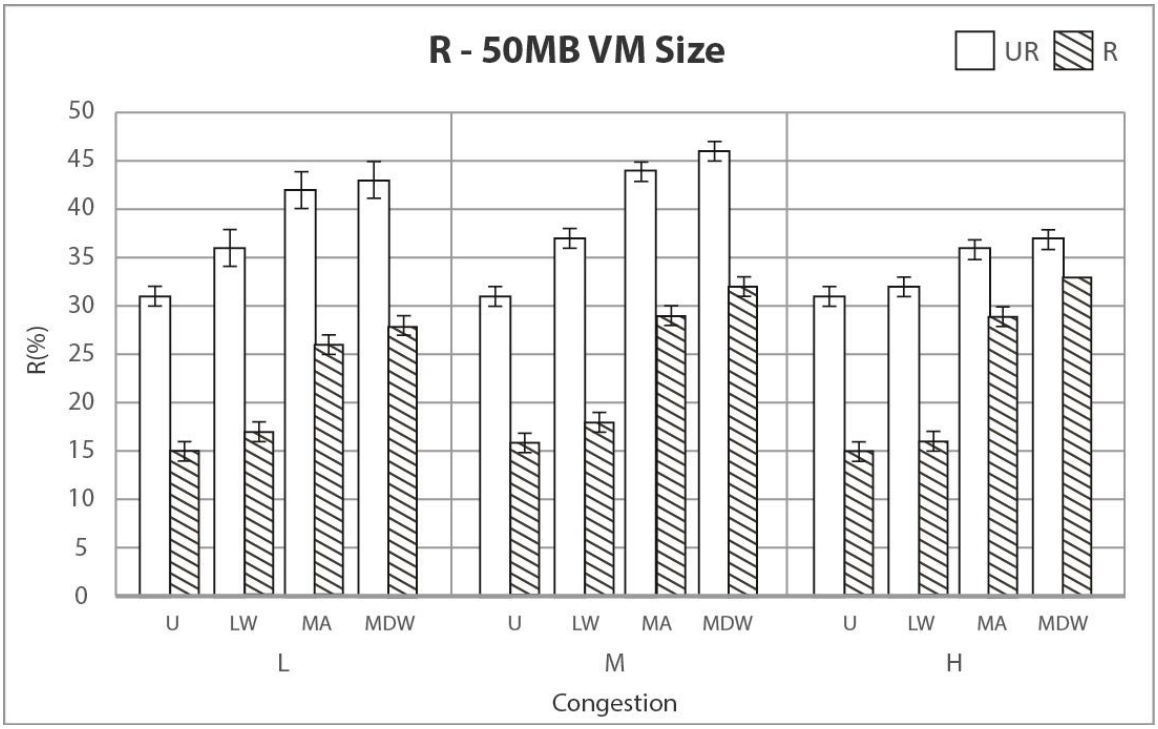

Figure 5.9: Average percentage resource utilization vs. Congestion – 50MB VM Size

5.3.4 Effect of Varying Congestion Levels and VM Size on the Average Percentage Utilization of Available Bandwidth

Figure 5.10 shows the ***UL*** for 4 algorithms, for 3 congestions, both Unrestricted (UR) and Restricted (R), at a VM Size of 10MB. Figures 5.11 and 5.12 show the same, for 25MB and 50MB respectively, and Figures 5.13, 5.14 and 5.15 show the same, but for ***DL***, for 10MB, 25MB and 50MB respectively. The results show very favorable performance with regards to bandwidth utilization (both uplink and downlink). Typical usage of the LTE network (by normal, non-vehicle subscribers) involves mostly downlink communication.

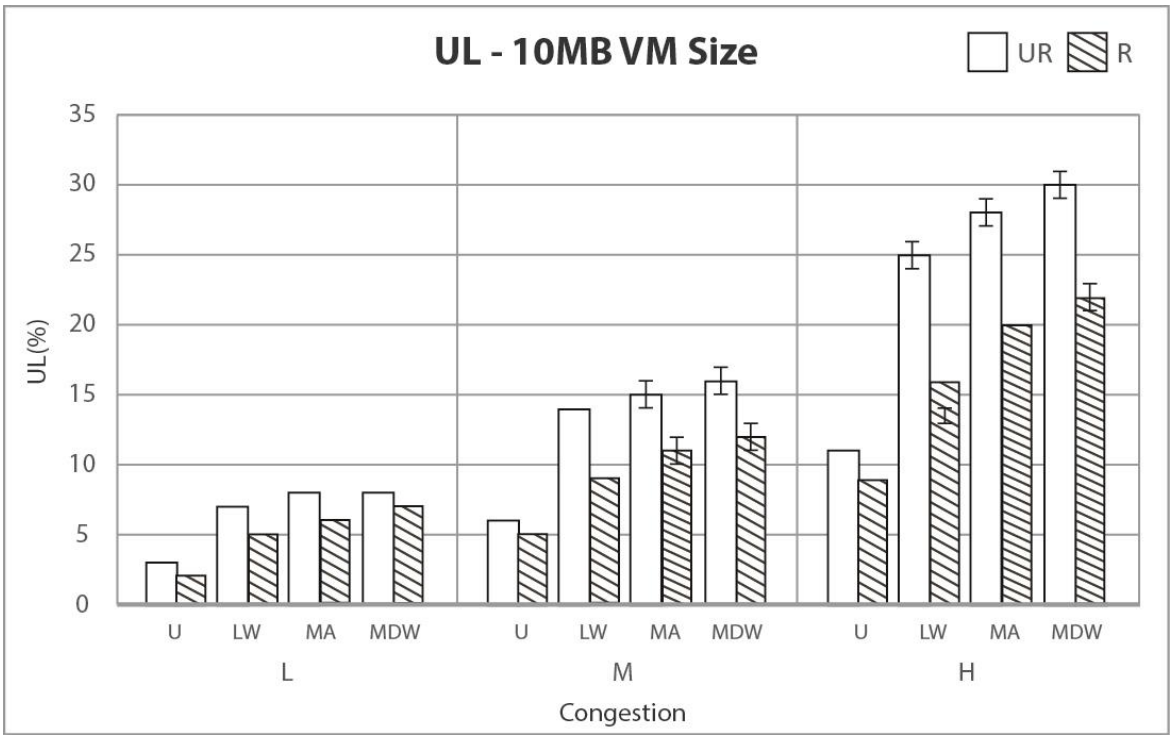

Figure 5.10: Average percentage utilization upload bandwidth vs. Congestion – 10MB VM Size

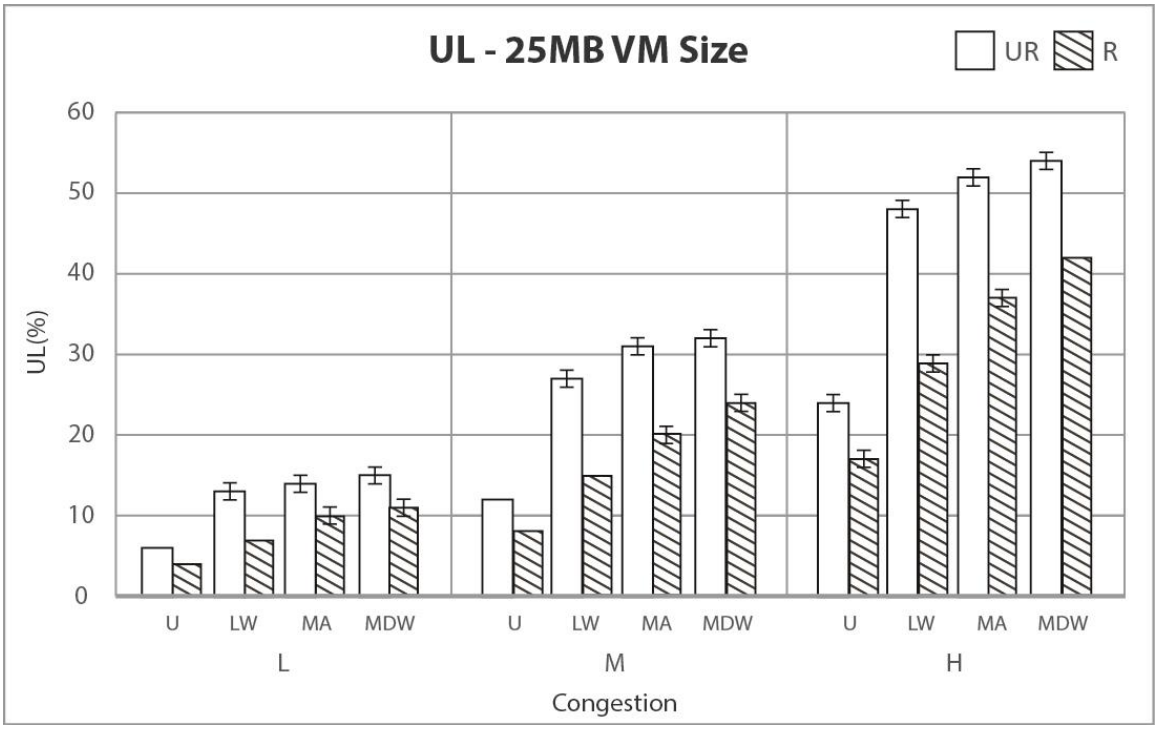

Figure 5.11: Average percentage utilization upload bandwidth vs. Congestion – 25MB VM Size

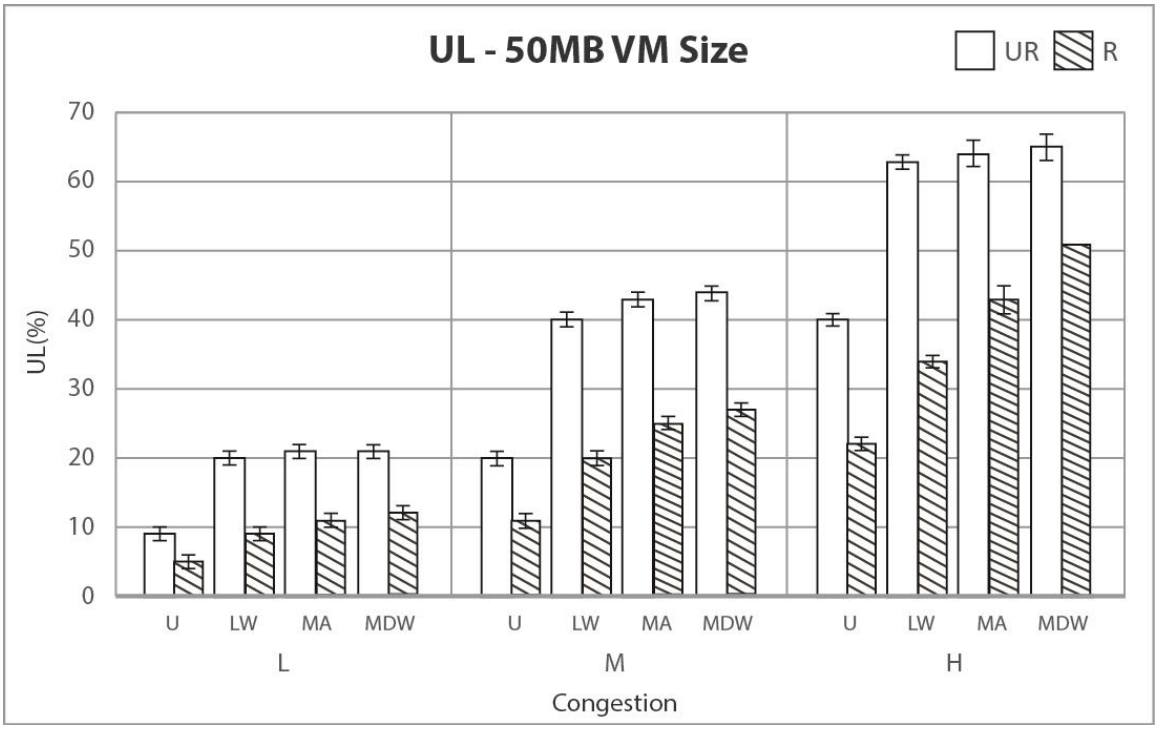

Figure 5.12: Average percentage upload bandwidth vs. Congestion – 50MB VM Size

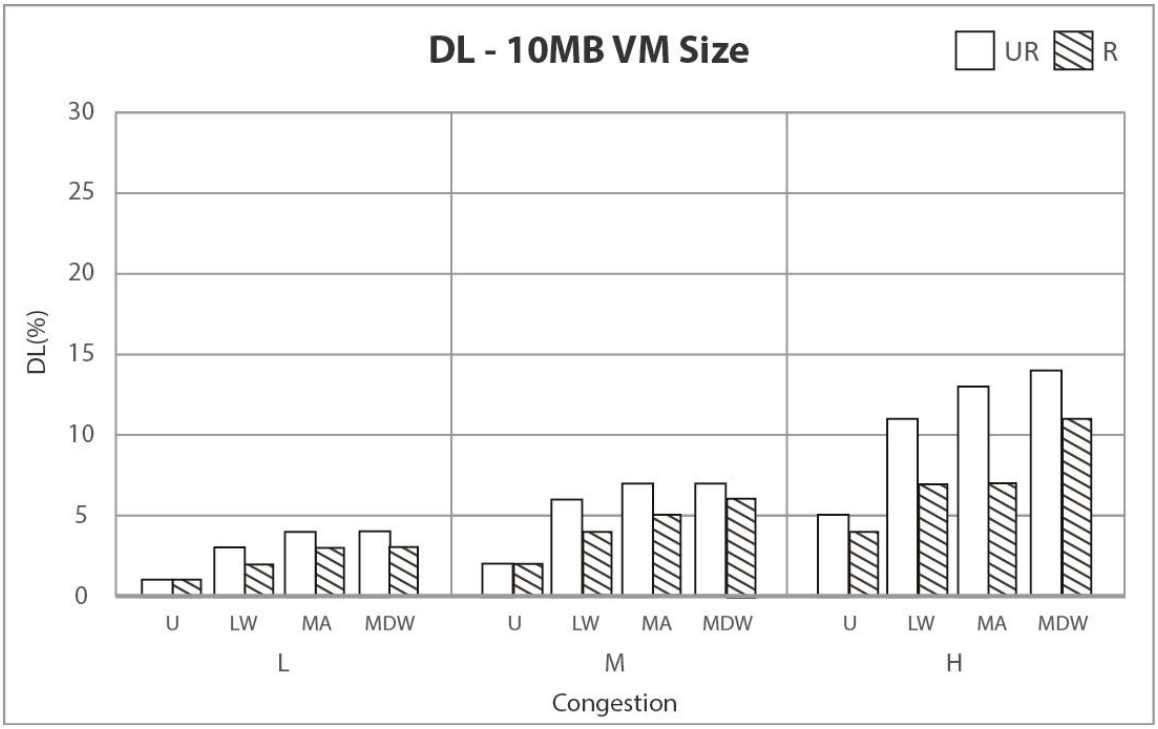

Figure 5.13: Average percentage utilization download bandwidth vs. Congestion – 10MB VM Size

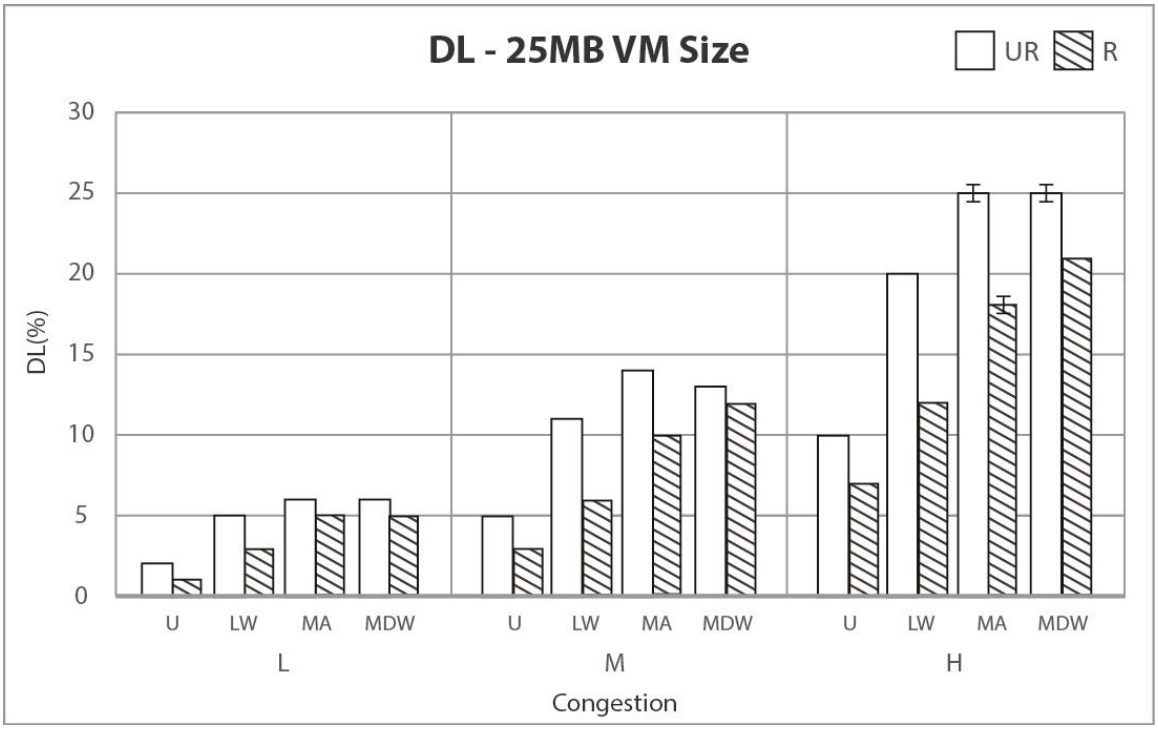

Figure 5.14: Average percentage utilization download bandwidth vs. Congestion – 25MB VM Size

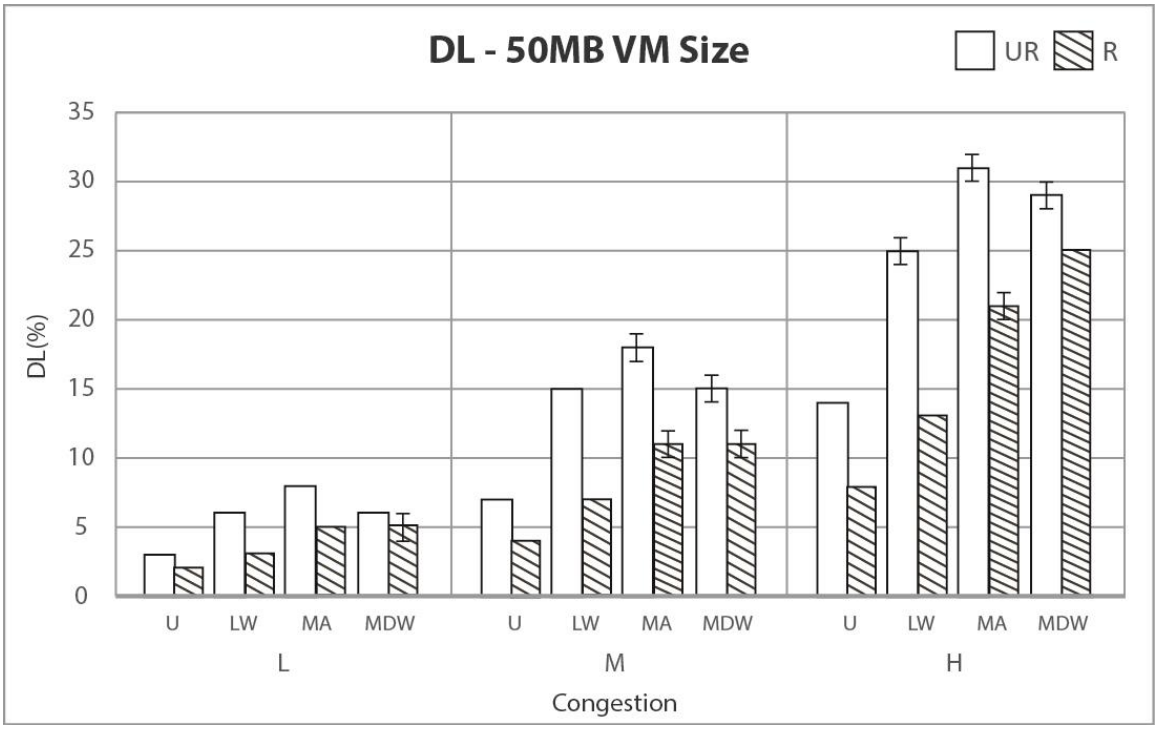

Figure 5.15: Average percentage utilization download bandwidth vs. Congestion – 50MB VM Size

As can be observed from Figure 5.15, the downlink utilization by the vehicular cloud, peaks at 31%. The uplink utilization peaks at around 65%, as shown in Figure 5.12. This has a direct impact on the Quality of Service (QoS) that the LTE network offers. The higher the utilization by the VC, the less bandwidth available for the LTE users. This is both difficult to quantify and highly dependent on the excess bandwidth the network provider allocates to the grid.

In summary, overall, there are several aspects to consider. Generally, a higher congestion is more favorable. The more vehicles available in the grid, the more VMs can be hosted, the more potential candidates for migration are available, with the lowest drop and V2I percentages. It is also notable that lower VM sizes are preferable. While a higher VM size may offer a lower overall resource utilization percentage, this is translated to a greater amount of physical resources

available to the cloud. Selecting the VM size is important to consider as it will most affect the network/bandwidth congestion.

5.4 Non-ideal Communication and Time Safety Factors

While the simulations presented earlier assume ideal conditions, it is important to consider the impact interference and connection instabilities may have on the performance of the VVMM algorithms. A simplified method of incorporating interference and connection imperfections is presented in what follows. This can further be revised to more accurately model the effects of medium congestion and path loss/fading, depending on both the network protocol and the channel quality respectively. However, to observe the performance of the proposed algorithms in non-ideal conditions, the simplified method presented next should suffice.

The probability of unsuccessful packet transmission at time t ($\boldsymbol{P_t}$) is calculated as shown in Equation (5.6). It is important to note that this equation arbitrarily gives equal weighting to both components, and this can be further revised in future research.

$$P_t = P_{max} \cdot \frac{Int_t + CQ_t}{2} \qquad (5.6)$$

where the interference due to concurrent migrations at time t ($\boldsymbol{Int_t}$) is calculated (and normalized) as shown in Equation (5.7).

$$Int_t = \frac{ul_t}{ul_m} \qquad (5.7)$$

The connection quality between source and destination, based on the distance between them, at time t ($\boldsymbol{CQ_t}$) is calculated (and normalized) as shown in Equation (5.8).

$$CQ_t = \frac{SD_t}{SD_m} \tag{5.8}$$

A single scenario, for both unrestricted (UR) and restricted was selected to test the effect of non-ideal conditions: Light congestion and a VM Size of 25MB. When the resource restriction is imposed, 50% is used, just as shown earlier in Table 5.2 for the corresponding scenario.

5.4.1 Simulation Results and Analysis

As can be deduced from Table 5.3, all algorithms are expected to experience increased drop rates, once non-ideal conditions are considered. While this may be disconcerting, it is quite logical and easily rectified. The algorithms were all designed to consider ideal conditions, and as such, the time needed for a vehicle to migrate its load, has no room for packet loss. If even one packet is lost, it guarantees failure of the migration. Note that the developed MATLAB engine models a simplified Transport Control Protocol/Internet Protocol (TCP/IP) because the communication allows packet retransmissions if a packet is lost. However, migrations start at the absolute latest point in time possible, assuming non-ideal conditions. This means that a packet being lost means all other packets will be delayed, even by a very minor time, and the vehicle will have left the grid before transmitting its last packet. To avoid this, vehicles must utilize a *time safety factor*. The simulations were repeated to prove the validity of this solution, and the results are shown in Table 5.3. The results exhibit similar performance trends as previously shown in the ideal conditions.

Future work will study the effect of varying the safety factor in an adaptive manner, as well as refining the modeling of the non-ideal conditions.

Table 5.3 – Simulation Results for Non-Ideal Conditions with Safety Factor – Light Congestion, 25MB VM Size

Algorithm	Restriction	Safety Factor (seconds)	D (%)	V (%)	UL (%)	DL (%)	R (%)
VVMM – U	UR	100	*80.56*	0.00	7.90	2.05	27.22
VVMM – LW	UR	100	*60.21*	0.00	18.96	5.29	35.70
VVMM – MA	UR	100	*44.35*	0.04	18.48	7.08	43.61
VVMM – MDWLAM	UR	100	*36.69*	0.08	16.97	7.15	45.44
VVMM – U	50%	100	*73.11*	0.00	5.16	1.43	15.23
VVMM – LW	50%	100	*55.75*	0.00	11.94	3.00	20.02
VVMM – MA	50%	100	*28.99*	0.00	10.76	4.49	30.09
VVMM – MDWLAM	50%	100	*16.85*	0.00	11.24	5.30	34.51

5.5 Conclusion

The vehicular cloud has many potential applications as well as different approaches with regards to implementation. One such paradigm utilizes the vehicles as hosts for Virtual Machines (VM), effectively constructing a data center, where the resources are mobile. While the conventional cloud problems, such as hotspots or load balancing, are the main triggers for VM migration, vehicular clouds have different triggers for VM migration. Due to the inherent dynamicity of the vehicular environment, vehicles hosting VM loads, may exit the coverage of a cloud area, and as such must migrate their workload to another vehicle that is still accessible to the cloud. We have proposed several migration algorithms for such scenarios, and evaluated their performance via simulations. Simulations show that an unrestricted allotment of VM loads can

result in a significant percentage of unsuccessful migrations, regardless of the performance improvement offered by the various algorithms proposed.

To counteract the issue of unsuccessful migrations, based on the performance of the unrestricted system, certain metrics are used to empirically predict a restriction to be imposed on the system to improve performance. These restrictions are simulated and shown to drastically decrease the percentage of unsuccessful migrations.

The next chapter presents a theoretical approach to addressing the same VVMM problem, using ILP formulations based on the Single Source Shortest Path Problem.

Chapter 6

Multiple-Instance Single Source Shortest Path Formulation

6.1 Introduction

The previous chapter presented an advanced analysis for the heuristics proposed in this thesis, as well as incorporating non-ideal communication in the simulations. However, until this point, the study has been mostly experimental, utilizing simulations. This chapter addresses the problem from a more theoretical perspective. Specifically, this chapter aims to arrive at an Integer Linear Program (ILP) formulation for the problem, to obtain a benchmark for comparison with the heuristics.

The chapter will first discuss potential linear programming models that were considered for the VC problem being addressed. The specific problem selected is then presented and described

in the specific context for the VC. Finally, the detailed ILP formulation is described and solutions are presented in comparison with the heuristics.

6.2 Potential Optimization Problems

Three commonly studied optimization problem forms that can be used in the VC are: the bipartite graph matching problem [LU15][WAN15], the bin packing problem [RAM15][SIN11], and the network flow problem [FEN16][GOL15].

The bipartite graph matching problem involves matching a set/subset of vertices with another set/subset of vertices, where no edges share a vertex. This problem can be applied to the VC in that one set of vertices are VMs (or their hosts) at time t should be matched to a set of vertices (future destinations). However, the lack of sharing vertices potentially introduces the problem of not allowing co-located VMs. The authors of [LU15] utilize the matching problem regarding resource allocation for mobile clouds, and propose an algorithm rather than attempting to solve an NP-Hard problem. The authors of [WAN15] address a more complex multi-objective example of the problem.

It is important to note that this issue is a specific case that falls under the problem of Network Flows. The goal of the network flow problem is maximizing the flow from a source to a destination. This can be a useful model for many engineering problems but there is still a more appropriate model for the VC.

The main target at first was the bin packing problem. Given a set of bins and possible elements to fill the bins, the selection to be made is how to pack, while aiming to minimize the number of bins used to fit the elements. The elements in this case would be the VMs and the bins would be

the vehicles at various instances. A special case that has already been studied for VMs in the cloud computing context has been presented in [SIN11]. The main unique aspect in the VM packing model, is that VMs sharing a physical resource can take up less space than the sum of their individual sizes, due to potential shared pages within the VMs. The authors of [RAM15] also propose a 'sharing-aware' algorithm for VM packing.

In this thesis however, a different optimization problem is selected as the basis for the ILP formulation to be studied. This is presented in the next section.

6.3 Shortest Path Problem

The Shortest Path problem is a common problem involving a directed graph. The most simple form is the Single Source Shortest Path (SSSP) problem. The problem aims at finding the shortest (or minimum weight/cost) path from the source to a specific destination. Studies addressing various ways to tackle the problem are abundant, examples include [DIN15] and [OTT15]. These specific studies investigate slightly modified versions of the problem, such as Dynamic SSSP [DIN15], where edges may dynamically change, as opposed to a static graph. Or, as in [OTT15] the authors evaluate Dijkstra's algorithm in solving many instances of SSSP. This is a very similar issue as will be addressed in this chapter, but slightly simpler.

There are several other forms of the shortest path problem, include Multi-Source Shortest Path (MSSP) [KOU16] and All Pair Shortest Path (APSP) [SUS15].

The MSSP involves a network that has multiple sources that are considered together, targeting various destinations. The APSP is more comprehensive, in that it finds the shortest path between every pair.

6.4 Multiple-Instance Single Source Shortest Path Problem Formulation

As the previous subsections introduced, the SSSP problem is a very common problem that can be solved in polynomial time, (provided there are no negative cycles). In this section, the SSSP is used as the basis for a novel ILP formulation, specific to the problem studied in this thesis: Vehicular Virtual Machine Migration. The goal of the ILP formulation is to provide a theoretical benchmark for the proposed heuristic algorithms (presented in the previous chapters). In order to present a fair comparison, the ILP must accurately represent the simulated system, and if there are any simplifications or assumptions, they must be incorporated into the heuristic model for a fair comparison. This will be discussed in a later section, in more detail. The previous section described the SSSP in general. The reuse of the problem description for VVMM is described in what follows.

In order to simplify the model for ILP, several key points must be mentioned: a) any VM instance is now fixed to a single 50MB VM, b) VMs can be collocated within vehicles but they will be dealt with *individually*, as opposed to previously, when a car's load was dealt with as a grouped unit, c) The resolution for the observation will no longer be set to a second-by-second basis, but will be split into snapshots of 50 second intervals (the time needed for a VM to be migrated at 1MB/s), d) Migrations cannot occur except discretely at the 50 second intervals, and not continuously in between. The reasons for these simplifications will be explained within the problem definition.

The mobility model presented in the previous chapter, is reused with minor modifications to address the simplifications mentioned. The MATLAB generated mobility model is used as the

input to create the ILP formulation. As such, for a fixed observation window, the set of vehicles that will exist at any point within the grid is known before ILP formulation. These vehicles will not necessarily be in the grid at all points in time, but any vehicle will be in the grid at some point during the window. Each vehicle/car c_i will then be split into multiple instances of itself such that c_{ij} represents vehicle i at time j. Doing so for each vehicle, will create a set C which is set to be the vertices/nodes in a graph G. Visually, the graph can be drawn in 2D, with the rows corresponding to the vehicle index and the columns corresponding to the time index t, as shown in Figure 6.1. To the set C, two more rows will be added, I representing the infrastructure and D representing the 'drop' state. A 'unifying destination' U_D is added at the end of the graph and a 'source' vehicle C_{src} is added to the start of the graph. The intervals (or transition from column t to column $t + 1$), for simplicity will be referred to as hops from this point onward.

Edges/arcs are added based on a set of criteria. All edges are directed and transition from one column to the next. As each vehicle's status is known at any point in time, whether it is in the grid or not, edges only exist entering or leaving that vehicle's state at any time t, provided it is in the grid at that time. Furthermore, for any of the infrastructure states or the drop states, edges can only leave that state to transition into a state in the same row (meaning from infrastructure to infrastructure, or drop to drop). The U_D is exempt from all these criteria, at the penultimate interval, each state will have an edge leading to the U_D, thereby terminating the graph at U_D. This can all be observed in Figure 6.1.

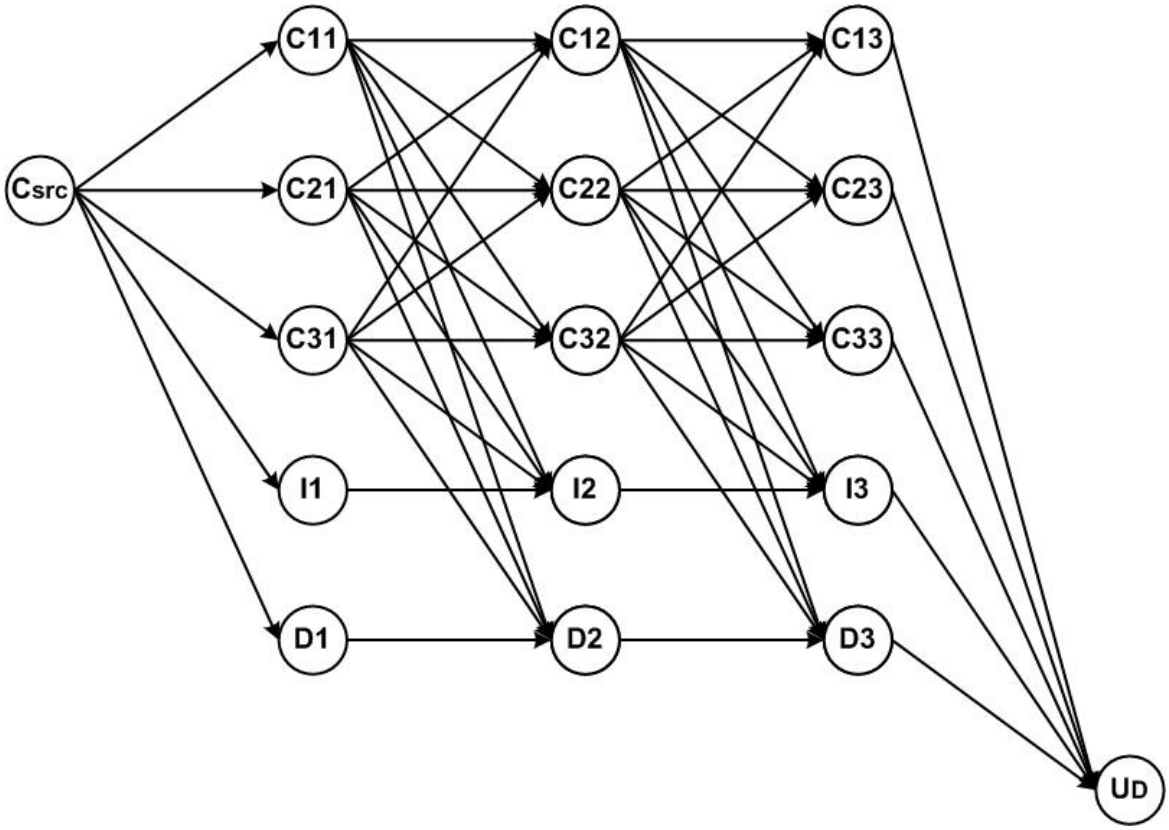

Figure 6.1: Example of Single Source Shortest Path Graph Formulation

The grid as described, with a single source C_{src} is the basis for the SSSP problem, for a single VM, which would be hosted initially at C_{src}. However, the number of VMs to be modeled is not one. While the MSSP concept can be considered, the problem can more easily be described as a Multiple-Instance SSSP (MISSSP) problem. If each VM is considered to be oblivious to all other VMs (including co-located VMs), then the problem can be envisioned as a 'set' of SSSP problems, hence: Multiple-Instance SSSP. However, the key factor here is that the multiple instances, while oblivious to each other, occur in the same grid. This is a main consideration when indexing states and edges. The 'inner grid' (excluding C_{src} and U_D) can be considered identical for each instance, except it is indexed by a further index corresponding to the VM being targeted. This can be seen in the following figure, Figure 6.2, showing a small example of 3 VMs in a 2-

vehicle grid. While v_1 and v_3 are co-located (initially both hosted) at $C_{src} = c_1$, they are dealt with as 2 separate instances of SSSP nonetheless. In the figure, a single edge in each instance is labeled just as an example.

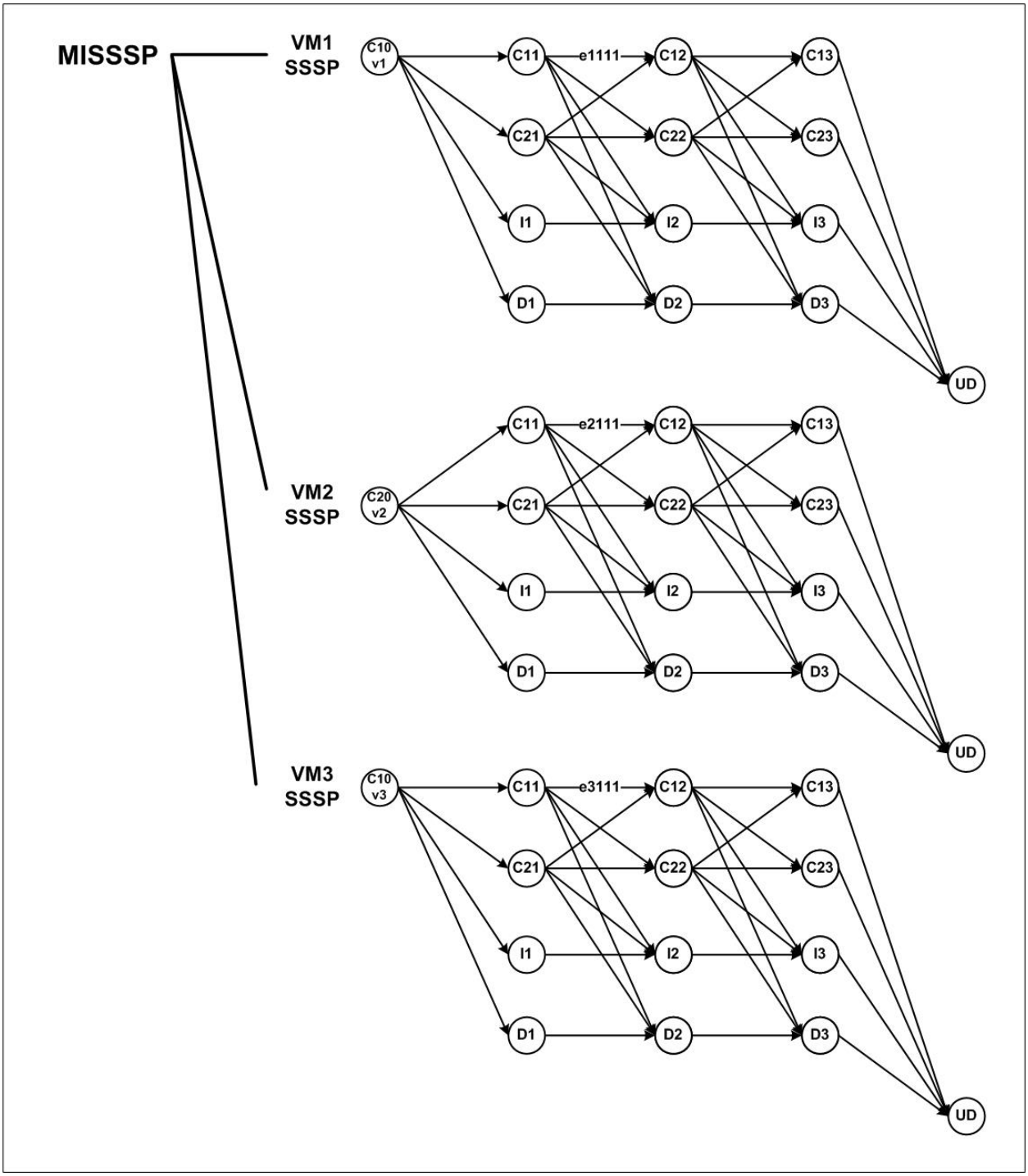

Figure 6.2: Example of Multiple Instance Single Source Shortest Path Graph Formulation

Each edge is then assigned 3 values: edge cost e, bandwidth BW, load L . These are critical in formulating the ILP correctly. The edge cost is a value that will be used in formulating the objective function, while the bandwidth and the load values are used in defining constraints. Edge cost is assigned based on the source vertex and the destination vertex. Edge cost is assigned based on Equation (6.1).

$$e_{vijt} = \begin{cases} 0, & (i = j \ \wedge i \neq \{I, D\} \wedge j \neq \{I, D\}) \vee (j = U_D \wedge i \neq \ U_D) \\ M_c, & i \neq j \wedge i \neq \{I, D, U_D\} \wedge j \neq \{I, D, U_D\} \\ I_c, & j = I \wedge i \neq \{D, U_D\} \\ D_c, & j = D \wedge i \neq \{I, U_D\} \end{cases} \quad (6.1)$$

where:

e_{vijt} cost of transitioning from vertex c_{it}, vehicle i at time t, to $c_{j(t+1)}$, vehicle j at time $t + 1$ (the next column in the graph), for VM v

M_c cost of migration from a vehicle to another vehicle

I_c cost of migrating to the infrastructure, row I at time $t + 1$

D_c cost of a drop (transitioning to the drop row D at time $t + 1$)

Assigning values for M_c, I_c and D_c cannot be arbitrary, but must follow specific criteria to be described later. Using Equation (6.1), edge costs will be zero for any ‘flat’ edges as well as all edges to the U_D. Any transition from vehicle to vehicle will be M_c. Any transition from a vehicle to the infrastructure will cost I_c and will cost D_c for transitioning from a vehicle to the drop row.

Edge bandwidth is assigned based on Equation (6.2).

$$BW_{vijt} = \begin{cases} 0, & j = D \vee i = j \vee (j = U_D \wedge i \neq U_D) \\ B_{avg}, & j = I \vee (i \neq j \wedge i \neq \{I, D, U_D\} \wedge j \neq \{I, D, U_D\}) \end{cases} \quad (6.2)$$

where:

BW_{vijt} bandwidth value for the transition from vertex c_{it}, vehicle i at time t, to $c_{j(t+1)}$, vehicle j at time $t + 1$ (the next column in the graph), for VM v

B_{avg} average bandwidth available for an individual VM (1MB/s)

Using Equation (6.2), the bandwidth will be 0 for all edges other than edges going from a vehicle to a different vehicle or going from a vehicle to the infrastructure.

All edge loads L_{vijt} are assigned values V_{size}, 50MB. This value corresponds to the flow through that edge (should it be selected).

The decision variables for the ILP are $x_{vijt} \in \{0, 1\}$, the value selected as 1 if the corresponding edge is selected to be part of the path for v and 0 if it is not a part of the path. Next, the objective function and constraints are introduced.

6.4.1 Objective Function and Constraints

The ILP formulation necessitates an objective function and a set of constraints. Based on the previous description, if edge costs are assigned correctly, the objective of each instance of the SSSP can simply be combined into one objective.

A single objective for VM v can be formulated as shown in Equation (6.3).

$$minimize: path_v = \sum_{t=1}^{|T|} \sum_{i=c_{1t}}^{c_{(|C|+2)t}} \sum_{j=c_{1(t+1)}}^{c_{(|C|+2)(t+1)}} e_{vijt} \cdot x_{vijt} \tag{6.3}$$

where:

$|T|$ total number of hops

$|C|$ number of vehicles (adding 2 to include I and D)

Note that while this objective exhaustively lists all possible links for the single instance for VM v, edges that do not exist will not be included (such as edges leading into/out of a vertex corresponding to a vehicle that is not in the grid at that time). This will be handled in the implementation of the ILP.

Since the objective function for one instance contains only values and edges exclusive to the instance, and is completely independent of any other instance, the hybrid/combined objective as shown in Equation (6.4), can be used for the MISSSP.

$$minimize: Path_{combined} = \sum_{i=1}^{|V|} path_i \tag{6.4}$$

This objective will present the combined shortest path (or minimum cost path) for each instance. If M_c, I_c and D_c are selected adequately, this minimum path can be claimed as the optimal solution, providing the best path each VM should take, from its corresponding C_{src} to the

U_D. The final hop can be discarded (as it is only a way to ensure that the objective has the freedom to select the penultimate vertex (which is the real final destination), while forcing the VM to move from the start to the end of the graph (thereby modeling all stages of time in the observation window). The selection of M_c, I_c and D_c is based on the following set of inequalities, which can be deduced as shown in Figure 6.3. The figure exhaustively gives the various bounds depending on the possible path types (in descending order as shown in the figure):

No migrations < One V2V migration < Continuous V2V migrations < One V2I migration < V2I migration from the outset < Delayed Drop < Drop from the outset.

$$0 < M_c \tag{6.5}$$

$$|T| \cdot M_c < I_c \tag{6.6}$$

$$|T| \cdot I_c < D_c \tag{6.7}$$

This set of inequalities forces the path selected to prefer no migrations, failing that, minimal migrations, failing that, as delayed a migration to the infrastructure as possible, failing that as delayed a drop as possible. This is in keeping with the goal of maximal data retention within the grid. However, this is all true provided the constraints, presented next, are not violated, otherwise the solution is infeasible.

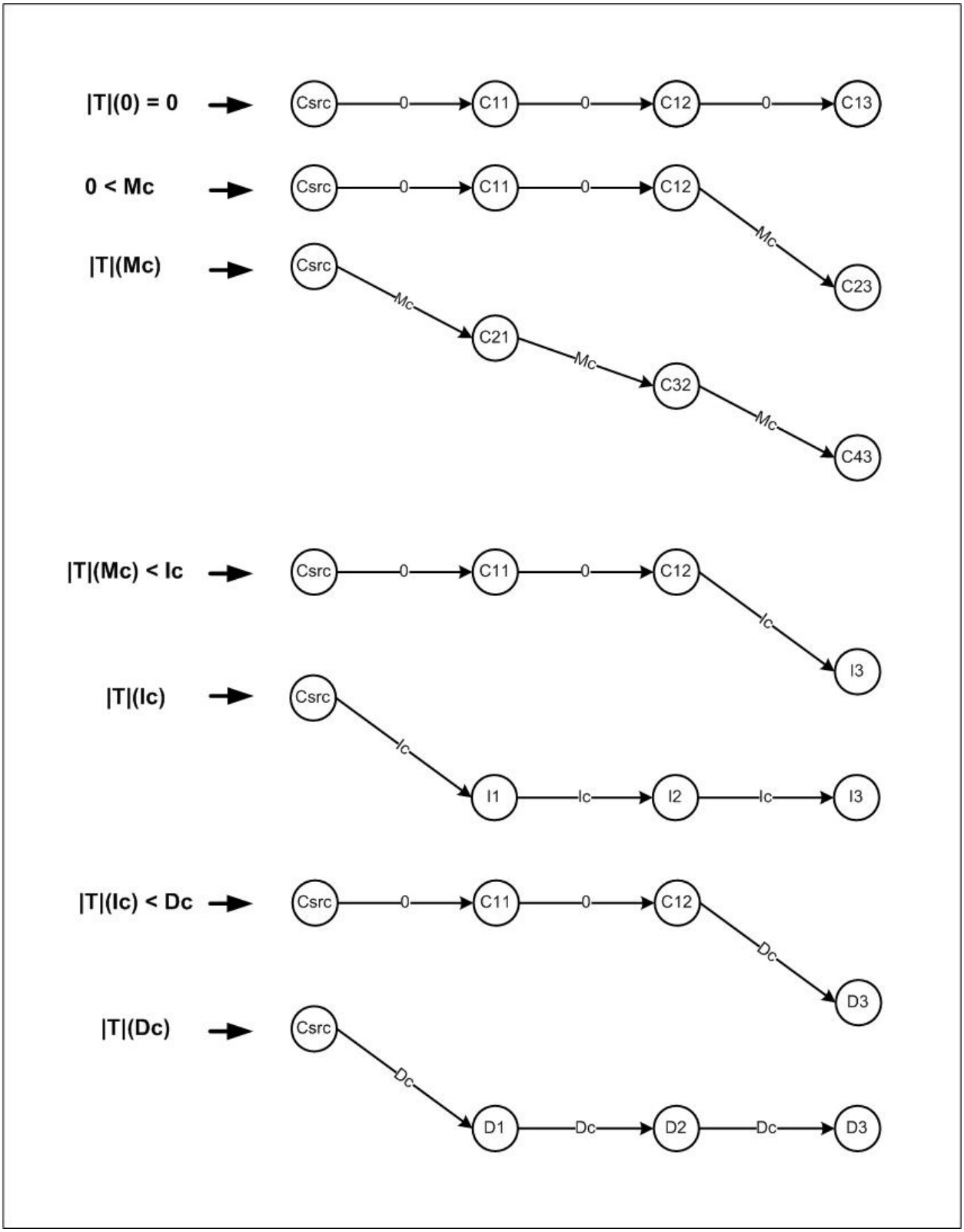

Figure 6.3: Visual Representation of Reasoning behind Selecting Edge Cost Values

There are 4 main sets of constraints: flow, bandwidth, workload and pingpong constraints. For simplicity in the 4 constraint equations, summations over i or j span all *rows*, and summations over v span all virtual machine instances.

The flow constraints pertain to the flow of edges into and out of each vertex. They can be expressed as shown in Equation (6.8).

$$\sum_{i} x_{vci(t+1)} - \sum_{j} x_{vjct} = \left\{ \begin{matrix} 0, & c \neq \{C_{src}, U_D\} \\ 1, & c = C_{src} \\ -1, & c = U_D \end{matrix} \right\}, \qquad \forall\, c, \forall\, t, \forall\, v \qquad (6.8)$$

The flow constraints ensure: a) any edges entering a vertex will leave it, b) the number of edges leaving the source will be one, and c) the number of edges entering the unifying destination will be one.

The bandwidth constraints can be expressed as shown in Equation (6.9).

$$\sum_{v} \sum_{i} \sum_{j} BW_{vijt} \cdot x_{vijt} \leq BW_{max}, \qquad \forall\, t \qquad (6.9)$$

The bandwidth constraints ensure that at any hop, the bandwidth consumed by migrating VMs does not exceed the maximum capacity of the grid BW_{max}. The maximum capacity is dependent on both the specific network protocol and the number of base-stations utilized. In this case, using 3 LTE base stations, as mentioned in Section 4.4, the upload limit of 3 × 50Mbps. Should the protocol or the coverage area (hence the number of base stations) vary, this must be reflected in the maximum capacity. This is a constraint that encompasses all the instances of the SSSP simultaneously. It is very important to consider that this constraint addresses the medium from the perspective of the overall bandwidth offered by the base-stations. Device-to-device

communication (without infrastructure involvement) or the utilization of frequency reuse will require refinement of the constraint to more accurately represent the capacity of the grid, however, this is a stricter constraint and hence will serve as a most strict optimal solution.

The workload constraints can be expressed as shown in Equation (6.10).

$$\sum_{v}\sum_{i} L_{vict} \cdot x_{vict} \leq L_{max}, \qquad \forall\, c, \forall\, t \tag{6.10}$$

The load constraints ensure that at any vertex, the combined load of all VMs in/migrated newly into a vertex does not exceed the maximum capacity of the vehicle L_{max}. This is also a constraint that encompasses all the instances of the SSSP simultaneously.

The pingpong constraints are unique to this specific formulation. A feasible solution cannot allow a VM to be migrated out of a vehicle and then back into the same vehicle at a later time. This constraint spans a row and the goal of the constraint is to prevent more than one entrance into the same row throughout the entire graph. As such the constraint must count all edges entering a row. The 'inner grid' (excluding the source) must discard flat edges in the row, because they represent a vehicle remaining in the same vehicle over time. For the first hop, all edges must be counted, then starting from the next hop, only diagonal edges (or all edges then subtracting all the flat edges). This can be expressed as shown in Equation (6.11).

$$x_{vC_{src}c1} + \sum_{t=2}\sum_{i=1} x_{vict} - \sum_{t=2} x_{vcct} \leq 1, \qquad \forall\, c, \forall\, v \tag{6.11}$$

With these constraints and the objective function as described, it is necessary to run some preliminary tests using the solver (CPLEX) [CPL16], prior to comparing the performance to the heuristics. This will be shown in the next subsection.

6.4.2 ILP Metrics

A C++ driver function was developed to generate the ILP formulation, based on an input generated from the MATLAB mobility model. The driver function takes into account the in/out of grid status of each vehicle at each interval, as well as the host of each VM at the start of the model. It then proceeds to write the objective function, the various constraints and the list of decision variables.

The resulting output of the driver function is then fed into the ILP solver, to obtain the solution. A C++ function was also developed to analyze the solution to obtain the various metrics necessary to compare with the heuristics.

The total number of migrations T_m can be calculated as shown in Equation (6.12).

$$T_m = \sum_v \sum_t \sum_i \sum_j x_{vijt} - \sum_v \sum_t \sum_k x_{vkkt} \tag{6.12}$$

The drop percentage is calculated as shown in Equation (6.13).

$$Drops = \frac{\sum_v \sum_t \sum_i x_{viDt} - \sum_v \sum_t x_{vDDt}}{T_m} \tag{6.13}$$

The percentage of infrastructure migrations is calculated as shown in Equation (6.14).

$$V2I = \frac{\sum_v \sum_t \sum_i x_{viIt} - \sum_v \sum_t x_{vIIt}}{T_m} \quad (6.14)$$

The percentage of Uplink Bandwidth utilization is calculated as shown in Equation (6.15).

$$UL_{BW} = \frac{\sum_t \sum_v \sum_i \sum_j (BW_{vijt} \cdot x_{vijt})}{|T| \cdot BW_{max}} \quad (6.15)$$

The percentage of Downlink Bandwidth utilization is calculated as shown in Equation (6.16).

$$DL_{BW} = \frac{\sum_t \sum_v \sum_i \sum_j (BW_{vijt} \cdot x_{vijt}) - \sum_v \sum_t \sum_i (BW_{viIt} \cdot x_{viIt})}{|T| \cdot BW_{max}} \quad (6.16)$$

Some preliminary runs are presented in the next subsection.

6.4.3 ILP Preliminary Runs

To test the ILP, 2 tests are run. The first test is set up to result in an optimal solution involving no V2I migrations or drops. The second test is set up to result in an optimal solution that should force V2I migrations. The set up involves 2 VMs to be distributed over a grid of 3 vehicles. In the first test, for the duration of the time analyzed, there will always be enough space/vehicles in the grid to host all the VMs. The second test is set up to have all vehicles leave the grid prior to the end of the analyzed time, forcing the only optimal solution to mandate V2I migrations. Table 6.1 and Table 6.2 show the status of each vehicle (1 = in grid, 0 = outside grid) at each time interval for the 1st and 2nd test respectively, while Table 6.3 shows the initial host (vehicle number) for each VM.

Table 6.1 – Vehicle Status at each Time for the 1st Test

Vehicle No.	Status at t = 1	Status at t = 2	Status at t = 3	Status at t = 4
1	1	0	0	0
2	1	1	0	0
3	0	0	1	1

Table 6.2 – Vehicle Status at each Time for the 2nd Test

Vehicle No.	Status at t = 1	Status at t = 2	Status at t = 3	Status at t = 4
1	1	0	0	0
2	1	1	0	0
3	0	0	1	0

Table 6.3 – VM host at Time = 0

VM No.	Host at t = 0	Status at t = 2	Status at t = 3
1	1	To be determined	To be determined
2	2	To be determined	To be determined

The output of the solver can be seen in Table 6.4, and can be visualized in Figure 6.4 for the 1st Test and Figure 6.5 for the 2nd.

Table 6.4 – ILP Solutions for the 1st and 2nd Test

$e_{vc_{it}c_{j(t+1)}t}$					$x_{vc_{it}c_{j(t+1)}t}$	
					1st	**2nd**
E	1	1	1	1	***1***	***1***
E	1	1	2	1	***0***	***0***
E	1	1	4	1	***0***	***0***
E	1	1	5	1	***0***	***0***
E	2	2	1	1	***0***	***0***
E	2	2	2	1	***1***	***1***
E	2	2	4	1	***0***	***0***
E	2	2	5	1	***0***	***0***
E	1	1	2	2	***1***	***0***
E	1	1	4	2	***0***	***1***
E	1	1	5	2	***0***	***0***
E	2	1	2	2	***0***	***0***
E	2	1	4	2	***0***	***0***
E	2	1	5	2	***0***	***0***
E	1	2	2	2	***0***	***0***
E	1	2	4	2	***0***	***0***
E	1	2	5	2	***0***	***0***
E	2	2	2	2	***1***	***1***
E	2	2	4	2	***0***	***0***
E	2	2	5	2	***0***	***0***
E	1	4	4	2	***0***	***0***
E	2	4	4	2	***0***	***0***
E	1	5	5	2	***0***	***0***
E	2	5	5	2	***0***	***0***
E	1	2	3	3	***1***	***0***
E	1	2	4	3	***0***	***0***
E	1	2	5	3	***0***	***0***
E	2	2	3	3	***1***	***0***
E	2	2	4	3	***0***	***1***
E	2	2	5	3	***0***	***0***
E	1	3	4	3	***0***	***0***
E	1	3	5	3	***0***	***0***
E	2	3	4	3	***0***	***0***
E	2	3	5	3	***0***	***0***
E	1	4	4	3	***0***	***0***
E	2	4	4	3	***0***	***0***
E	1	5	5	3	***0***	***0***
E	2	5	5	3	***0***	***0***
E	1	3	3	4	***1***	***0***
E	1	3	4	4	***0***	***0***
E	1	3	5	4	***0***	***0***
E	2	3	4	4	***1***	***0***
E	2	3	4	4	***0***	-
E	2	3	5	4	***0***	***0***
E	1	4	4	4	***0***	***0***
E	2	4	4	4	***0***	***0***
E	1	5	5	4	***0***	***0***
E	2	5	5	4	***0***	***0***
E	1	3	6	5	***1***	-
E	2	3	6	5	***1***	-
E	1	4	6	5	***0***	***1***
E	2	4	6	5	***0***	***1***
E	1	5	6	5	***0***	***0***
E	2	5	6	5	***0***	***0***

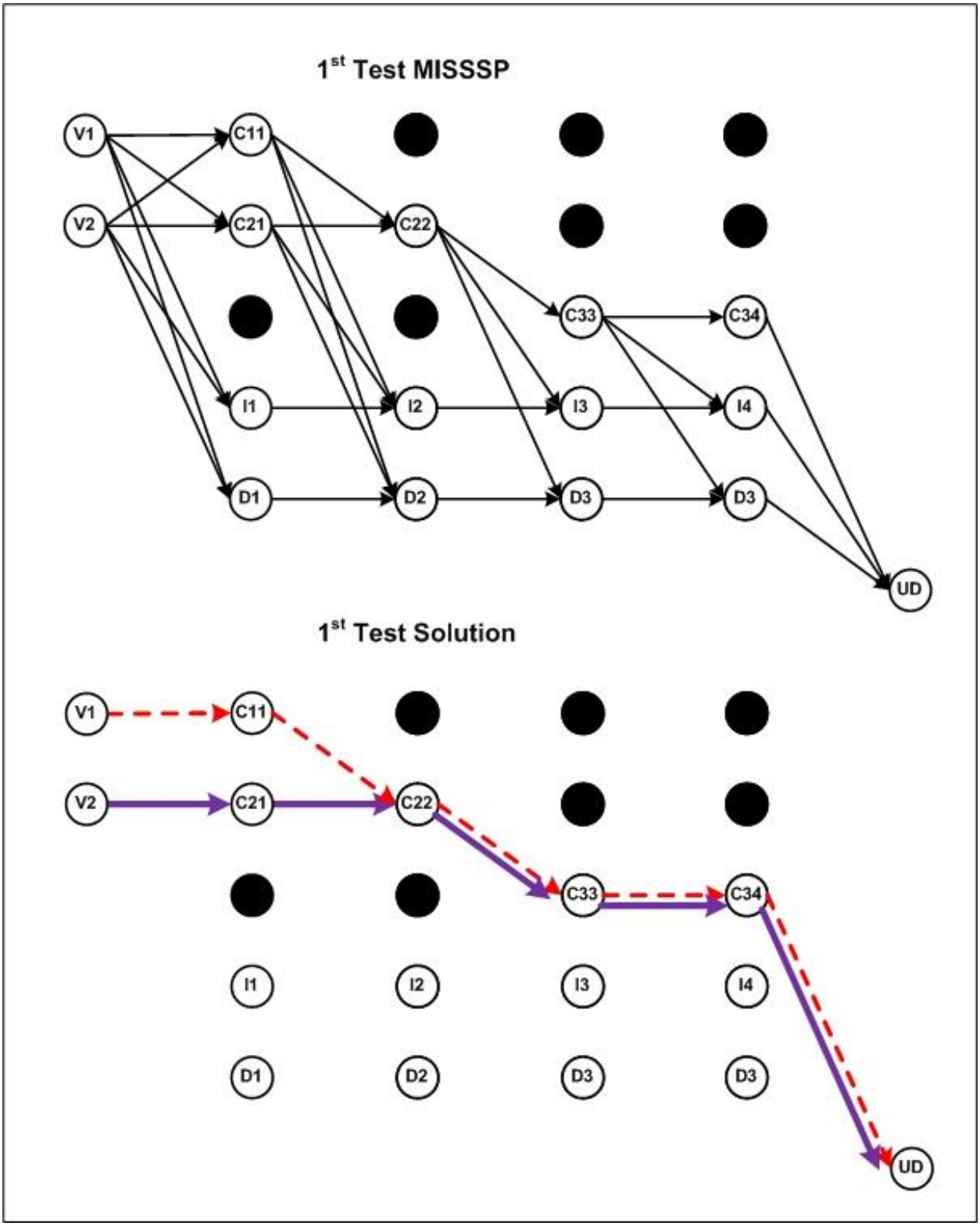

Figure 6.4: MISSSP Graph based on 1st Test and Solution

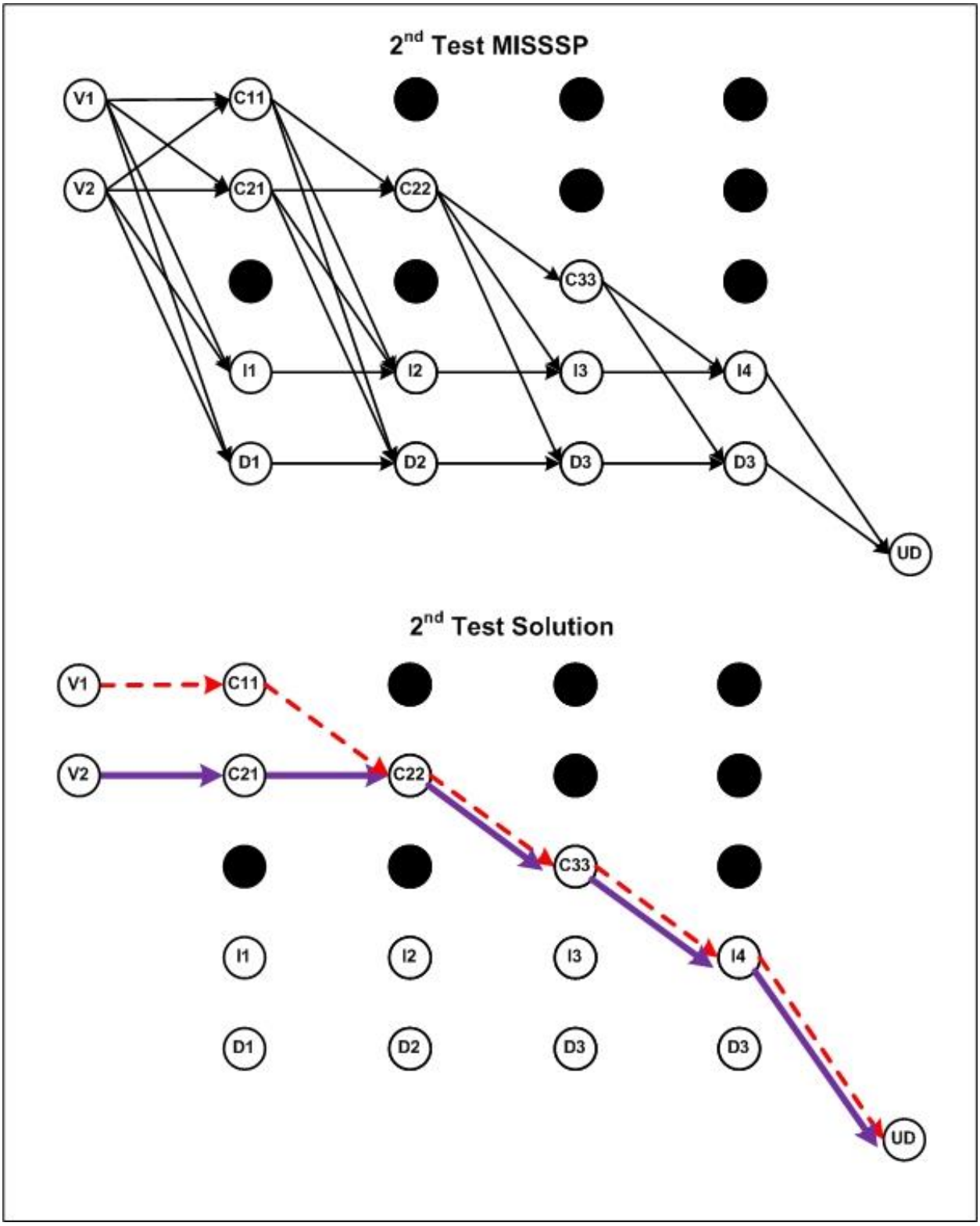

Figure 6.5: MISSSP Graph based on 2nd Test and Solution

For both tests, the optimal sequence of hosts for VM1 and VM2, can be seen by following the red (dashed) and purple (solid) paths respectively. The first test shows both VMs selecting the optimal path, without resorting to V2I migrations. The second test also shows the optimal path for both VMs, resorting to V2I migrations for both, due to the fact that there are no vehicles remaining in the grid prior to the end of the analyzed time. The results of the preliminary tests, as can be seen, show that the ILP formulation can be used, as described, to find the optimal solution for the MISSSP.

6.5 Comparing to Heuristics

As can be deduced from the previously presented formulation, the number of edges grows exponentially as the number of vehicles increases, or the number of hops observed increases, or the number of VMs increases. Without scaling down, one iteration using the values modeled in the previous chapter resulted in a 1.2GB input file for CPLEX. This will be infeasible to model, and as such the scaling down of the model was critical.

The number of hops was decreased to 12, 10 and 8 for high, medium and low congestion respectively. Only 50MB VM sizes were modeled (in an unrestricted mode, now that only single VMs are modeled). Parking and revisiting legs was removed to induce shorter trip times and force more migrations to occur within a shorter window of time. The number of vehicles was decreased to 35, 25 and 15 for high, medium and low congestion respectively. The capacity of a vehicle was also decreased to host a maximum of 3 co-located VMs. Non-ideal communication was removed from consideration to maintain the comparability between the ILP and the heuristic model. The number of iterations was decreased from 30 to 10, per congestion level.

After running, the Drops percentage, the V2I percentage, the UL and DL percentage utilization were measured and compared between the heuristics and ILP. For all runs V2I was found to be 0% so the graphs will only be presented for Drops, UL and DL. These are shown in Figures 6.6, 6.7 and 6.8. Only average values are shown, averaged over the 10 iterations. The runs were also all for an unrestricted (UR) data cap. Note that the 10 iterations per congestion level were run using the same mobility models for both MATLAB and CPLEX, presented with a 95% confidence analysis.

As can be shown from the figures, the ILP sets a clear benchmark for the heuristics showing an improvement in all metrics. However, it is important to note that the simplifications must be taken into account when extending the conclusions beyond the preliminary results present at this stage. Further work must be done to address the simplifications made in the ILP formulation to more accurately compare the ILP to the heuristics presented in the previous chapter.

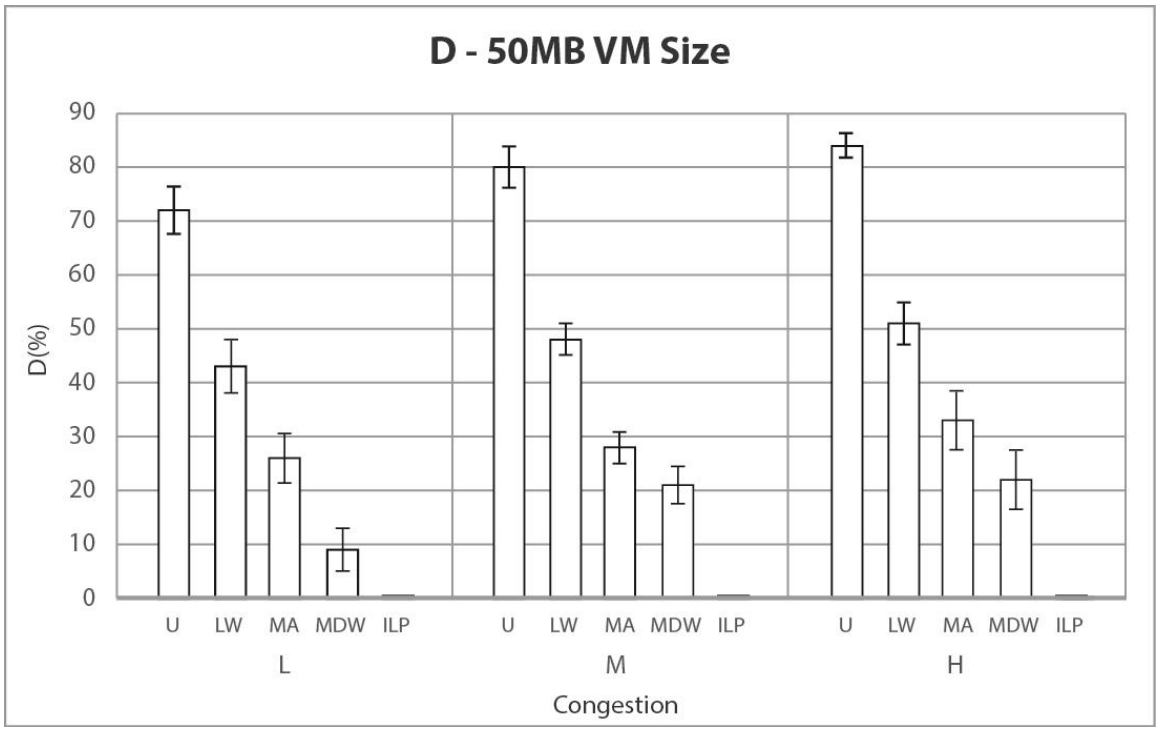

Figure 6.6: Percentage of dropped migrations vs. Congestion – 50MB VM Size

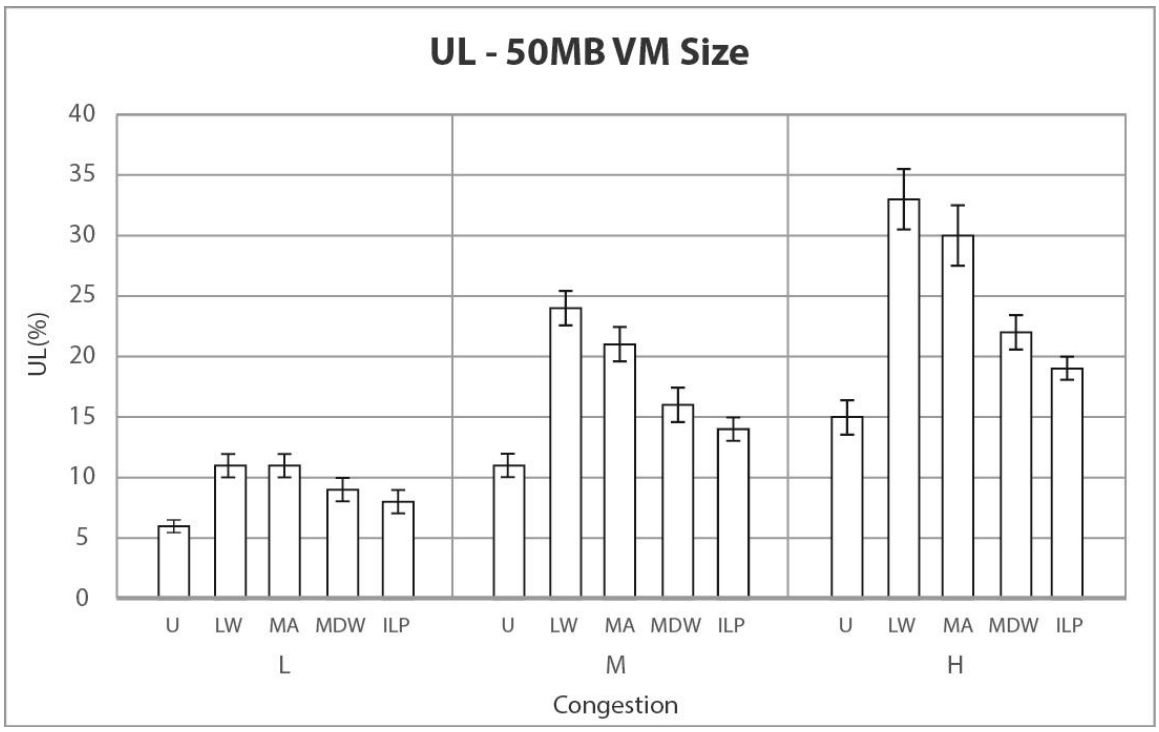

Figure 6.7: Average percentage utilization upload bandwidth vs. Congestion – 50MB VM Size

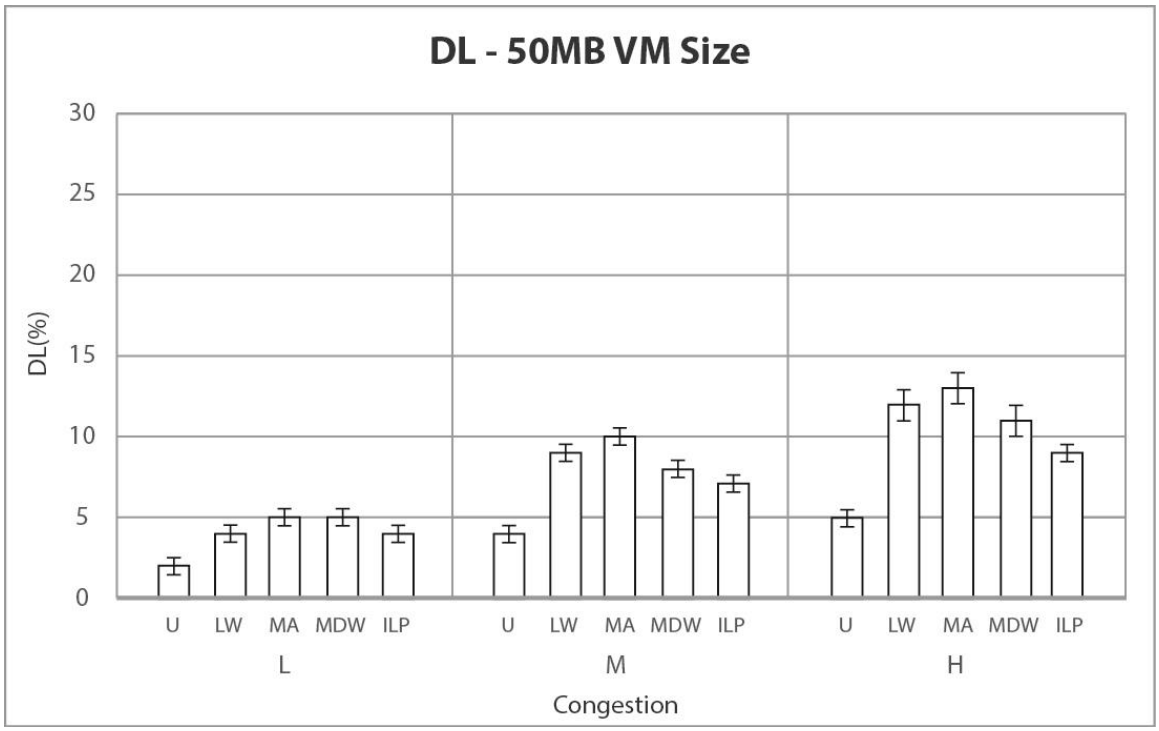

Figure 6.8: Average percentage utilization download bandwidth vs. Congestion – 50MB VM Size

6.6 Summary

This chapter presented an ILP formulation specific to the VC as presented in earlier chapters. While based on the Single Source Shortest Path problem, it further addresses Multiple Instance Single Source Shortest Path (MISSSP) with constraints that encompass all instances simultaneously. Each VM is dealt with on an individual SSSP but the combined group of SSSPs is subject to a set of constraints that ensure the MISSSP model as described does not allow infeasible solutions.

This formulation is shown to be a sufficiently accurate representation of the VC as described earlier, and serves as a clear benchmark for the heuristic VVMM algorithms presented in earlier chapters. However, there is room for further expansion and refinement to the model. This will be included in future research section in the following chapter.

Chapter 7

Conclusions

7.1 Concluding Remarks

This thesis presented a study of Virtual Machine Management for Vehicular and Dynamic Clouds. Of the main issues impeding the realization of the Vehicular Cloud, is the issue of resource availability, as vehicles (hosting virtual machines) move in and out of network coverage. This could result in virtual machines being inaccessible to the cloud subscriber, causing a Service Level Agreement violation. This issue is considered a main trigger for virtual machine migration, as opposed to traditional cloud triggers such as hot/cold spots. In the vehicular cloud, vehicles leaving the grid could be considered analogous to failing physical resources. Virtual machine management and migration schemes specific to vehicular clouds are the main focus of this thesis.

A number of Vehicular Virtual Machine Migration algorithms were proposed, with the goal of reducing the negative impact of vehicle mobility. The migration algorithms were simulated and their performance analyzed based on a number of metrics, within a unique and novel mobility environment developed specifically to test the performance of the algorithms, using a MATLAB engine.

A mobility model was designed, incorporating vehicles with various speeds and realistic mobility patterns, for varying levels of congestion. This mobility model served as the testing environment for the set of migration schemes proposed. Vehicular Virtual Machine Migration (VVMM) is the main algorithm with several variations: VVMM-U – a benchmark algorithm to be used for comparison, VVMM-LW – the first algorithm, incorporating knowledge of the workload of migration destinations, VVMM-MA – the second algorithm, building upon VVMM-LW, incorporating knowledge of the workload and mobility patterns of migration destinations.

A set of metrics selected to analyze the performance of the migration schemes within the mobility environment proposed, including percentage of dropped migrations, successful migrations, and utilization fairness.

The mobility model was then revised to incorporate red lights at intersections, parked cars, more realistic and random mobility patterns, and more accurately modeled congestion levels. The advanced mobility model was used to test refined versions of the migration schemes as well as MDWLAM – a third algorithm (aside from the refined LW and MA), taking into account time needed to migrate cumulative loads on both source and destination. A new set of metrics is proposed to analyze the refined schemes. Finally, the MATLAB engine was revised to

incorporate non-ideal communication factors due to distance between source and destination, as well as network congestion due to concurrent migrations.

Finally, the study presented a theoretical analysis of the problem using integer linear programming (ILP). The problem was modeled using a modification to the Single Source Shortest Path (SSSP) problem, coined Multiple Instance SSSP (MISSSP). A revised objective function subject to a set of comprehensive constraints was presented and modeled in an ILP solver. The optimization problem necessitated several simplifications and assumptions. These assumptions were reincorporated into the heuristic model in the MATLAB engine, to maintain comparability between the heuristics and the results of the ILP solver. The optimization model provides a benchmark to the performance of the heuristics, with a feasible optimal solution of minimum number of dropped migrations. This is an important contribution that serves as a basis for future research points, presented in the next section.

Based on the findings of this thesis, it can be concluded that the issue of virtual machine migration and management for vehicular clouds, and by extension dynamic clouds, can be handled with the proposed algorithms, with a low percentage of unsuccessful migrations. The study leaves room for further improvement upon the suggested methods and a small scale implementation on a physical testbed may be necessary to further confirm the presented results. Furthermore, after some enhancements, the ILP model can be used to compare any heuristic results to an optimal benchmark. It is important to note that, as of the time of submitting this thesis, the issue of VM migration for the VC, with vehicles as VM hosts, is still quite novel. Alternative VM migration and management schemes are difficult to find in the literature, as such even a quantitative comparison of performance with the findings of this thesis, is difficult.

7.2 Future Research

There are several directions that can expand upon this thesis:

- The optimization model can be further expanded upon to increase the resolution of the modeled time window. The assumptions made can be addressed, such as the lack of non-ideal communication, a larger capacity per vehicle. Most importantly, the model can further be refined to minimize complexity, size and memory consumption.
- The heuristic algorithms can address migrations on a VM by VM basis, as opposed to on a per car load basis. This would involve 1:N migrations, and allowing partial migrations, would not necessitate an "everything or nothing" approach to drops / successes.
- A different approach to 1:N migrations can be studied in the context of a redundant migration, to incorporate resilience. A vehicle can attempt to migrate its workload to multiple destinations simultaneously to maximize chances of success. This will involve a higher consumption of bandwidth and the study could attempt to find the optimal number of destinations to achieve a balance between maximizing successful migrations and optimal bandwidth utilization.
- A larger scale of multiple cloudlets can be simulated, to observe how the behavior is affected when vehicles leave one grid and enter another. This could incorporate migrations that were at some point migrated to the infrastructure being returned to another vehicle at a later point, possibly in a different cloudlet.

- Revising the non-ideal communication condition models, to more accurately represent imperfect channels, or more favorably, utilizing a network simulator can provide a more accurate analysis of the communication and network behavior.
- Traffic simulators can be utilized for testing with more accurate and more realistic mobility models.

References

[**3GP12**] 3GPP LTE Standard, 3GPP TS 36.321 V10.6.0 (2012-09) Medium Access Control (MAC) Protocol Specification (Release 10).

[**ABI11**] H. Abid, L. Phuong, J. Wang, S. Lee and S. Qaisar, "V-Cloud: Vehicular Cyber-Physical Systems and Cloud Computing," *Proceedings of the 4th International Symposium on Applied Sciences in Biomedical and Communication Technologies (ISABEL),* Barcelona, Spain, October 2011.

[**ADH16**] T. Adhikary, A. K. Das, M. A. Razzaque, A. Almogren, M. Alrubaian and M. M. Hassan, "Quality of Service Aware Reliable Task Scheduling in Vehicular Cloud Computing," *Mobile Networks and Applications*, vol. 21, no. 3, pp. 482-493, June 2016.

[**AHM15A**] R. W. Ahmad, A. Gani, S. H. Hamid, M. Shiraz, A. Yousafzai and F. Xia, "A survey on virtual machine migration and server consolidation frameworks for cloud data centers," *Journal of Network and Computer Applications*, vol. 52, pp 11-25, June 2015.

[**AHM15B**] F. Ahmad, M. Kazim, A. Adnane and A. Awad, "Vehicular Cloud Networks: Architecture, Applications and Security Issues," *Proceedings of the 8th IEEE/ACM International Conference on Utility and Cloud Computing (UCC)*, pp. 571-576, Limassol, Cyprus, December 2015.

[**AKO10**] S. Akoush, R. Sohan, A. Rice, A. Moore and A. Hopper, "Predicting the Performance of Virtual Machine Migration," *Proceedings of the IEEE International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems*, pp. 37-46, Miami Beach,

FL, August 2010.

[**ALM12**] M. A. Al Mamun, K. Anam, M. F. A. Onik and A. M. Esfar-E-Alam, "Deployment of Cloud Computing into VANET to Create Ad Hoc Cloud Network Architecture," *Proceedings of the World Congress on Engineering and Computer Science,* pp. 210-214, San Francisco, CA, October 2012.

[**ALO15**] M. Aloqaily, B. Kantarci and H. T. Mouftah, "Vehicular Clouds: State of the Art, Challenges and Future Directions," *Proceedings of the IEEE Jordan Conference on Applied Electrical Engineering and Computing Technologies (AEECT)*, pp. 1-6, Amman, Jordan, November 2015.

[**ALS14**] S. Al-Sultan, M. M. Al-Doori, A. H. Al-Bayatti and H. Zedan, "A Comprehensive Survey on Vehicular Ad Hoc Network," *Journal of Network and Computer Applications,* vol. 37, pp. 380-392, 2014.

[**ARI12**] S. Arif, S. Olariu, J. Wang, G. Yan, W. Yang and I. Khalil, "Datacenter at the Airport: Reasoning about Time-Dependent Parking Lot Occupancy," *IEEE Transactions on Parallel and Distributed Systems,* vol. 23, no. 11, pp. 2067-2080, 2012.

[**BAB12**] D. Baby, R. D. Sabareesh, R. A. K. Saravanaguru and A. Thangavelu, "VCR: Vehicular Cloud for Road Side Scenarios," in *Advances in Computing and Information Technology*, N. N. D. C. N. Meghanathan, Ed., Chennai, India, Springer Berlin Heidelberg, pp. 541-552, 2012.

[**BAR16**] B. Baron, M. Campista, P. Spathisa, L. H. Costab, M. D. Amorima, O. C. Duarteb, G. Pujollea and Y. Viniotisc, "Virtualizing vehicular node resources: Feasibility study of virtual machine migration," *Vehicular Communications*, vol. 4, pp. 39-46, April 2016.

[**BAY11**] S. Bayless, "Fourth Generation Wireless - Vehicle and Highway Gateways to the Cloud: An evaluation of Long Term Evolution (LTE) and other wireless technologies' impact to the transportation sector," *U.S. Dept. of Transportation Research and Innovative Technology Administration*, 2011.

[**BIL15**] N. Bila, E. J. Wright, E. De Lara, K. Joshi, H. A. Lagar-Cavilla, E. Park, A. Goel, M. Hiltunen and M. Satyanarayanan, "Energy-Oriented Partial Desktop Virtual Machine Migration," *ACM Transactions on Computer Systems*, vol. 33, no. 1, pp. 30-81, March 2015.

[**BIT12**] S. Bitam and A. Mellouk, "ITS-Cloud: Cloud Computing for Intelligent Transportation System," *Proceedings of the IEEE Global Communications Conference*, pp. 2054-2059, Anaheim, CA, December 2012.

[**BIT15**] S. Bitam, A. Mellouk and S. Zeadally, "VANET-cloud: a generic cloud computing model for vehicular Ad Hoc networks," *IEEE Wireless Communications*, vol.22, no.1, pp.96-102, February 2015.

[**BOS10**] S. Bose, A. Pasala, D. Ramanujam, S. Murthy and G. Malaiyandisamy, "SLA Management in Cloud Computing: A Service Provider's Perspective," in *Cloud Computing: Principles and Paradigms*, R. Buyya, J. Broberg and A. Goscinski, Eds., Hoboken, NJ, John Wiley and Sons, Inc., pp. 413-435, 2010.

[**BOU13**] R. Boutaba, Q. Zhang and M. F. Zhani, "Virtual Machine Migration in Cloud Computing Environments: Benefits, Challenges, and Approaches," *Communication Infrastructures for Cloud Computing.* H. Mouftah and B. Kantarci (Eds.). IGI-Global, USA. pp. 383-408, September 2013

[**BRA07**] R. Bradford, E. Kotsovinos, A. Feldmann and H. Shiöberg, "Live Wide-Area Migration

of Virtual Machines Including Local Persistent State," *Proceedings of the International Conference on Virtual Execution Environments,* pp. 169-179, San Diego, CA, June 2007.

[**CHE12**] J. Chen, W. Liu and J. Song, "Network Performance-Aware Virtual Machine Migration in Data Centers," *Proceedings of the 3rd International Conference on Cloud Computing, GRIDS and Virtualization,* pp. 65-71, Nice, France, July 2012.

[**CHE14**] Y.-S. Chen, C.-S. Hsu and C.-H. Cheng, "Network Mobility Protocol for Vehicular Ad Hoc Networks," *International Journal of Communication Systems*, vol. 27, no. 11, pp. 3042-3063, 2014.

[**CHO11**] L.-D. Chou, J.-Y. Yang, Y.-C. Hsieh, D.-C. Chang and C.-F. Tung, "Intersection-based Routing Protocol for VANETs," *Wireless Personal Communications*, vol. 60, no. 1, pp. 105-124, 2011.

[**CLA05**] C. Clark, K. Fraser, S. Hand, J. G. Hansen, E. Jul, C. Limpach, I. Pratt and A. Warfield, "Live Migration of Virtual Machines," *Proceedings of the Symposium on Networked Systems Design and Implementation,* pp. 273-286, Boston, MA, May 2005.

[**CON08**] H. Conceição, L. Damas and M. B. J. Ferreira, "Large-scale simulation of V2V environments," *Proceedings of the ACM Symposium on Applied Computing,* pp. 28-33, Fortaleza, Brazil, March 2008.

[**COR15**] N. Cordeschi, D. Amendola and E. Baccarelli, "Hard and soft optimal resource allocation for primary and secondary users in infrastructure Vehicular Networks," *Proceedings of the 12th IEEE Consumer Communications and Networking Conference (CCNC),* pp.708-713, Las Vegas, NV, January 2015.

[**CPL16**] "CPLEX Optimizer," International Business Machines (IBM) Corporation, [Online]. Available: http://www-01.ibm.com/software/commerce/optimization/cplex-optimizer/. [Accessed 24/10/2016].

[**DES11**] U. Deshpande, X. Wang and K. Gopalan, "Live Gang Migration of Virtual Machines," *Proceedings of the International Symposium on High Performance Distributed Computing,* pp. 135-146, San Jose, CA, June 2011.

[**DES16**] M. R. Desai and H. B. Patel, "Performance Measurement of Virtual Machine Migration Using Pre-copy Approach in cloud computing," *Proceedings of the 2nd International Conference on Information and Communication Technology for Competitive Strategies (ICTCS)*, Udaipur, India, March 2016.

[**DIN13**] H. Dinh, C. Lee, D. Niyato and P. Wang, "A Survey of Mobile Cloud Computing: Architecture, Applications and Approaches," *Wireless Communications and Mobile Computing,* vol. 13, no. 18, pp. 1587-1611, 2013.

[**DIN15**] W. Ding and K. Qiu, "Dynamic Single-Source Shortest Paths in Erdös-Rényi Random Graphs," *Combinatorial Optimization and Applications*, vol. 9486, pp. 537-550, December 2015.

[**ELR10**] M. El-Refaey, "Virtual Machines Provisioning and Migration Services," in *Cloud Computing: Principles and Paradigms*, R. Buyya, J. Broberg and A. Goscinski, Eds., Hoboken, NJ, John Wiley and Sons, Inc., pp. 123-154, 2010.

[**ELT10**] M. Eltoweissy, S. Olariu and M. Younis, "Towards Autonomous Vehicular Clouds," in *Ad Hoc Networks*, J. Zhi-Zhong, D. Simplot-Ryl and V. Leung, Eds., Springer Berlin Heidelberg, pp. 1-16, 2010.

[**EZI15**] K. Ezirim and S. Jain, "Taxi-cab cloud architecture to offload data traffic from cellular networks," *Proceedings of IEEE 16th International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, pp.1-6, Boston, MA, June 2015.

[**FEN16**] H. Feng, J. Llorca, A. M. Tulino and A. F. Molisch, "Dynamic network service optimization in distributed cloud networks," *Proceedings of the IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pp. 300-305, San Francisco, CA, 2016.

[**GAI16**] K. Gai, M. Qiu, H. Zhao, L. Tao and Z. Zong, "Dynamic Energy-Aware Cloudlet-based Mobile Cloud Computing Model for Green Computing," *Journal of Network and Computer Applications*, vol. 55, pp. 46-54, January 2016

[**GER12**] M. Gerla, "Vehicular Cloud Computing," *Proceedings of the International Workshop on Vehicular Communications and Applications,* pp. 152-155, Ayia Napa, Cyprus, June 2012.

[**GER13**] M. Gerla, J.-T. Weng and G. Pau, "Pics-On-Wheels: Photo Surveillance in the Vehicular Cloud," *Proceedings of the International Conference on Computing, Networking and Communications*, pp. 1123-1127, San Diego, CA, January 2013.

[**GER14**] M. Gerla, E.-K. Lee, G. Pau and U. Lee, "Internet of Vehicles: From Intelligent Grid to Autonomous Cars and Vehicular Clouds," *Proceedings of the IEEE World Forum on Internet of Things*, pp. 241-246, Seoul, Korea, March 2014.

[**GHA15**] P. Ghazizadeh, S. Olariu, A. G. Zadeh and S. El-Tawab, "Towards Fault-Tolerant Job Assignment in Vehicular Cloud," *Proceedings of the IEEE International Conference on Services Computing (SCC)*, pp.17-24, New York, NY, June 2015.

[**GOL15**] A. V. Goldberg, H. Kaplan, S. Hed, and R. E. Tarjan, "Minimum Cost Flows in Graphs

with Unit Capacities," *Proceedings of the 32nd International Symposium on Theoretical Aspects of Computer Science (STACS)*, vol. 30, pp. 406-419, Munich, Germany, March 2015.

[**GU13**] K. Gu, D. Zeng, S. Guo and B. Ye, "Leverage parking cars in a two-tier data center," *Proceedings of the IEEE Wireless Communications and Networking Conference,* pp. 4665-4670, Shanghai, China, April 2013.

[**GUB13**] J. Gubbi, R. Buyya, S. Marusic and M. Palaniswami, "Internet of Things (IoT): A Vision, Architectural Elements, and Future Directions," *Future Generation Computer Systems,* vol. 29, no. 7, pp. 1645-1660, 2013.

[**HAM14**] S. Abdelhamid, H. Hassanein and G. Takahara, "Vehicle as a resource (VaaR)," *IEEE Network*, vol.29, no.1, pp.12-17, January 2015.

[**HE14**] W. He, G. Yan and L. Xu, "Developing Vehicular Data Cloud Services in the IoT Environment," *IEEE Transactions on Industrial Informatics,* vol. PP, no. 99, p. 1, 2014.

[**HUS12**] R. Hussain, J. Son, H. Eun, S. Kim and H. Oh, "Rethinking Vehicular Communications: Merging VANET with Cloud Computing," *Proceedings of the 4th IEEE International Conference on Cloud Computing Technology and Science,* pp. 606-609, Taipei, Taiwan, December 2012.

[**HUS13**] R. Hussain, F. Abbas, J. Son and H. Oh, "TIaaS: Secure Cloud-assisted Traffic Information Dissemination in Vehicular Ad Hoc Networks," *Proceedings of the 13th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing,* pp. 178-179, Delft, Netherlands, May 2013.

[**HUS15**] R. Hussain, Z. Rezaeifar and H. Oh. "A Paradigm Shift from Vehicular Ad Hoc Networks to VANET-Based Clouds." *Wireless Personal Communications*, Springer, pp. 1-28,

2015.

[**HWA11**] K. Hwang, G. Fox and J. J. Dongarra, *Distributed and Cloud Computing: From Parallel Processing to the Internet of Things*, Waltham, MA: Morgan Kaufmann, 2011.

[**JAI84**] R. Jain, C. Dah-Ming and W. Hawe, "A quantitative measure of fairness and discrimination for resource allocation in shared computer system," *Eastern Research Laboratory,* 1984.

[**JAM12**] T. James, "Smart Cars," *Engineering and Technology Magazine,* vol. 7, no. 6, pp. 50-51, 2012.

[**JAW11**] P. Jaworski, T. Edwards, J. Moore and K. Burnham, "Cloud Computing Concept for Intelligent Transportation Systems," *Proceedings of the International IEEE Conference on Intelligent Transportation Systems,* pp. 931-936, Washington, DC, October 2011.

[**JIA13**] B. Jiang, J. Wu, X. Zhu and D. Hu, "Priority-Based Live Migration of Virtual Machine," in *Grid and Pervasive Computing*, J. J. Park, H. R. Arabnia, C. Kim, W. Shi and J. Gil, Eds., Springer Berlin Heidelberg, pp. 376-385, 2013.

[**KAN16**] N. J. Kansal and I. Chana, "Energy-aware Virtual Machine Migration for Cloud Computing - A Firefly Optimization Approach," *Journal of Grid Computing*, vol. 14, no. 2, pp. 327-345, June 2016.

[**KIM16**] R. Kim, H. Lim and B. Krishnamachari, "Prefetching-Based Data Dissemination in Vehicular Cloud Systems," *IEEE Transactions on Vehicular Technology*, vol. 65, no. 1, pp. 292-306, January 2016.

[**KOT09**] G. Kotusevski and K. A. Hawick, "A review of traffic simulation software," Research

Letters in the Information and Mathematical Sciences, Massey University Institute of Natural and Mathematical Sciences, pp. 35-54, 2009.

[**KOU16**] S. Koulouzis, A. S. Belloum, M. T. Bubak, Z. Zhao, M. Živković, and C. T. de Laat, "SDN-aware federation of distributed data," *Future Generation Computer Systems*, vol. 56, pp. 64-76, March 2016.

[**KUM15**] N. Kumar, R. Iqbal, S. Misra and J. J. Rodrigues, "Bayesian Coalition Game for Contention-Aware Reliable Data Forwarding in Vehicular Mobile Cloud," *Future Generation Computer Systems*, vol. 48, pp. 60-72 July 2015.

[**LAN08**] K.-C. Lan and C.-M. Chou, "Realistic Mobility Models for Vehicular Ad hoc Network (VANET) Simulations," *Proceedings of the International Conference on ITS Telecommunications*, pp. 362-366, Phuket, Thailand, October 2008.

[**LEE14**] E. Lee, E.-K. Lee and M. Gerla, "Vehicular cloud networking: Architecture and design principles," *IEEE Communications Magazine*, vol. 52, no. 2, pp. 148–155, February 2014.

[**LIU16**] L. Liu, S. Zheng, H. Yu, V. Anand and D. Xu, "Correlation-based virtual machine migration in dynamic cloud environments," *Photonic Network Communications*, vol. 31, no. 2, pp. 206-216, April 2016.

[**LU15**] Z. Lu, J. Zhao, Y. Wu and G. Cao, "Task Allocation for Mobile Cloud Computing in Heterogeneous Wireless Networks," *Proceedings of the 24th International Conference on Computer Communication and Networks (ICCCN)*, pp. 1-9, Las Vegas, NV, 2015.

[**LU16**] D. Lue, Z. Li, D. Huang, X. Lu, Y. Deng, A. Chowdhary, B. Li, "VC-bots: A Vehicular Cloud Computing Testbed with Mobile Robots," *Proceedings of the First International Workshop*

on Internet of Vehicles and Vehicles of Internet (IoV-VoI), pp. 31-36, Paderborn, Germany, July 2016.

[**MAI16**] V. D. Maio, R. Prodan, S. Benedict and G. Kecskemeti, "Modelling Energy Consumption of Network Transfers and Virtual Machine Migration," *Future Generation Computer Systems*, vol. 56, pp. 388-406, March 2016.

[**MAN12**] R. Mangharam, "The Car and The Cloud: Automotive Architectures for 2020," *The Bridge on Frontiers of Engineering*, vol. 42, no. 4, pp. 25-33, 2012.

[**MAN16**] U. Mandal, P. Chowdhury, M. Tornatore, C. U. Martel and B. Mukherjee, "Bandwidth Provisioning for Virtual Machine Migration in Cloud: Strategy and Application," *IEEE Transactions on Cloud Computing*, vol. PP, no. 99, March 2016.

[**MAR11**] F. J. Martinez, C. K. Toh, J.-C. Cano, C. T. Calafate and P. Manzoni, "A Survey and Comparative Study of Simulators for Vehicular Ad Hoc Networks (VANETs)," *Wireless Communications and Mobile Computing,* vol. 11, no. 7, pp. 813-828, 2011.

[**MAT16**] "MATLAB Software," Mathworks, [Online]. Available: http://www.mathworks.com/products/matlab. [Accessed 24/10/2016]

[**MER13A**] K. Mershad and H. Artail, "Finding a STAR in a Vehicular Cloud," *IEEE Intelligent Transportation Systems Magazine,* vol. 5, no. 2, pp. 55-68, 2013.

[**MER13B**] K. Mershad and H. Artail, "CROWN: Discovering and Consuming Services in Vehicular Clouds," *Proceedings of the International Conference on Communications and Information Technology,* pp. 98-102, Beirut, Lebanon, June 2013.

[**MER13C**] K. Mershad and H. Artail, "A Framework for Implementing Mobile Cloud Services in

VANETs," *Proceedings of the IEEE International Conference on Cloud Computing,* pp. 83-90, Santa Clara, CA, June 2013.

[**MIC14**] "Microsoft Windows," Microsoft Corporation, [Online]. Available: http://http://windows.microsoft.com. [Accessed 28 01 2014].

[**MIS11**] M. Mishra and A. Sahoo, "On Theory of VM Placement: Anomalies in Existing Methodologies and Their Mitigation using a Novel Vector Based Approach," *Proceedings of the IEEE International Conference on Cloud Computing,* pp. 275-282, Washington, DC, July 2011.

[**MIS12**] M. Mishra, A. Das, P. Kulkarni and A. Sahoo, "Dynamic Resource Management using Virtual Machine Migrations," *IEEE Communications Magazine,* vol. 50, no. 9, pp. 34-40, 2012.

[**MOU11**] H. Mousannif, I. Khalil and H. al Moatassime, "Cooperation as a Service in VANETs," *Journal of Universal Computer Science,* vol. 17, no. 8, pp. 1202-1218, 2011.

[**NAR11**] J. Naranjo, J. Zato, L. Redondo, M. Oliva, F. Jimenez and N. Gomez, "Evaluation of V2V and V2I Mesh Prototypes based on a Wireless Sensor Network," *Proceedings of the IEEE Conference on Intelligent Transportation Systems,* pp. 2080-2085, Washington, DC, October 2011.

[**OLA11**] S. Olariu, I. Khalil and M. Abuelela, "Taking VANET to the Clouds," *International Journal of Pervasive Computing and Communications,* vol. 7, no. 1, pp. 7-21, 2011.

[**OLA13**] S. Olariu, T. Hristov and G. Yan, "The Next Paradigm Shift: From Vehicular Networks to Vehicular Clouds," in *Mobile Ad Hoc Networking,* S. Basagni, M. Conti, S. Giordano and I. Stojmenovic, Eds., Hoboken, NJ, John Wiley and Sons, Inc., pp. 645-700, 2013.

[**OTT15**] M. W. Otte, "Modifying Dijkstra's Algorithm to Solve Many Instances of SSSP in

Linear Time," *Univeristy of Colorado Boulder Aerospace Engineering Sciences Faculty Contributions*, December 2015.

[**PAR09**] J.-G. Park, J.-M. Kim, H. Choi and Y.-C. Woo, "Virtual Machine Migration in Self-Managing Virtualized Server Environments," *Proceedings of the International Conference on Advanced Communication Technology*, pp. 2077-2083, Phoenix Park Gangwon-Do, Korea, February 2009.

[**PAS16**] M. Pasha, M. U. Farooq and K. Khan, "A Proof-of-Concept Model for Vehicular Cloud Computing using OMNeT++ and SUMo," *Innovations in Computer Science and Engineering*, vol. 413, pp. 193-198, February 2016.

[**PTV17**] "Software for multimodal traffic simulation: PTV Vissim," PTV Group, [Online]. Available: http://www.ptvgroup.com/en/solutions/products/ptv-vissim [Accessed 20 02 2017].

[**QIN12**] Y. Qin, D. Huang and X. Zhang, "VehiCloud: Cloud Computing Facilitating Routing in Vehicular Networks," *Proceedings of the 11th IEEE International Conference on Trust, Security and Privacy in Computing and Communications*, pp. 1438 - 1445, Liverpool, United Kingdom, June 2012.

[**RAM15**] S. Rampersaud and D. Grosu, "Sharing-Aware Online Algorithms for Virtual Machine Packing in Cloud Environments," *Proceedings of the IEEE 8th International Conference on Cloud Computing (CLOUD)*, pp. 718-725, New York City, NY, August 2015.

[**REF14**] T. K. Refaat, B. Kantarci and H. T. Mouftah, "Dynamic Virtual Machine Migration in a Vehicular Cloud," *Proceedings of the IEEE Symposium on Computers and Communication (ISCC)*, MoCS3.2.1-MoCS3.2.6, Madeira, Portugal, June 2014.

[**REF16**] T. K. Refaat, B. Kantarci and H. T. Mouftah, "Virtual Machine Migration and management for a Vehicular Cloud," *Elsevier Journal on Vehicular Communications*, vol. 4, pp. 47-56, April 2016.

[**RIM09**] B. P. Rimal, E. Choi and I. Lumb, "A Taxonomy and Survey of Cloud Computing Systems," *Proceedings of the 5th International Joint Conference on INC, IMS and IDC*, pp. 44-51, Seoul, Korea, August 2009.

[**SAN08**] J. Santa, A. F. Gómez-Skarmeta and M. Sánchez-Artigas, "Architecture and Evaluation of a Unified V2V and V2I Communication System based on Cellular Networks," *Computer Communications*, vol. 31, no. 12, pp. 2850-2861, 2008.

[**SAN12**] J. Santa, F. Bernal, P. J. Fernández, A. Moragón and A. García, "Continuous IPv6 Communications in a Vehicular Networking Stack for Current and Future ITS Services," *Proceedings of the IEEE Intelligent Vehicles Symposium Workshops*, pp. 1-7, Madrid, Spain, June 2012.

[**SHO16**] M. Shojafar, N. Cordeschi and E. Baccarelli, "Energy-efficient Adaptive Resource Management for Real-time Vehicular Cloud Services," *IEEE Transactions on Cloud Computing*, vol. pp, no. 99, April 2016.

[**SIN08**] A. Singh, M. Korupolu and D. Mohapatra, "Server-Storage Virtualization: Integration and Load Balancing in Data Centers," *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*, pp. 1-12, Austin, TX, November 2008.

[**SIN11**] M. Sindelar, R. K. Sitaraman, and P. Shenoy, "Sharing-aware Algorithms for virtual Machine Colocation," *Proceedings of the 23rd Annual ACM Symposium on Parallelism in*

Algorithms and Architectures (SPAA), pp. 367-378, San Jose, CA, June 2011.

[**SOM08**] C. Sommer and I. D. F. Dietrich, "Realistic Simulation of Network Protocols in VANET Scenarios," *IEEE Communications Magazine*, vol. 46, no. 11, pp. 132-137, 2008.

[**SUS15**] Susmita, "Algorithms of All Pair Shortest Path Problem," *International Journal of Computer Applications*, vol. 120, no. 15, June 2015.

[**TAO15**] J. Tao, Z. Zhang, F. Feng, J. He and Y. Xu, "Non-cooperative Resource Allocation Scheme for Data Access in VANET Cloud Environment," *Proceedings of the 3rd International Conference on Advanced Cloud and Big Data (CBD)*, pp. 190-196, YangZhou, China, November 2015.

[**TOR12**] J. A. Torkestani, "Mobility Prediction in Mobile Wireless Networks," *Journal of Network and Computer Applications*, vol. 35, no. 5, pp. 1633-1645, 2012.

[**TRA06**] F. Travostino, P. Daspit, L. Gommans, C. Jog, C. de Laat, J. Mambretti, I. Monga, B. Van Oudenaarde, S. Raghunath and P. Wang, "Seamless Live Migration of Virtual Machines over the MAN/WAN," *Future Generation Computer Systems*, vol. 22, no. 8, pp. 901-907, 2006.

[**VIG14**] N. Vignesh, R. Shankar, S. Sathyamoorthy and R. V., "Value Added Services on Stationary Vehicular Cloud," *Distributed Computing and Internet Technology*, vol. 8337, pp. 92-97, 2014.

[**VM14**] "VMware," VMware, Inc., [Online]. Available: http://www.vmware.com. [Accessed 28 01 2014].

[**VOO10**] W. Voorsluys, J. Broberg and R. Buyya, "Introduction to Cloud Computing," in *Cloud Computing: Principles and Paradigms*, R. Buyya, J. Broberg and A. Goscinski, Eds., Hoboken,

NJ, John Wiley and Sons, Inc., pp. 3-37, 2010.

[**WAN11**] J. Wang, J. Cho, S. Lee and T. Ma, "Real Time Services for Future Cloud Computing Enabled Vehicle Networks," *Proceedings of the International Conference on Wireless Communications and Signal Processing*, pp. 1-5, Nanjing, China, November 2011.

[**WAN13**] X. Wang, L. X. L. Fan and X. Jia, "A Decentralized Virtual Machine Migration Approach of Data Centers for Cloud Computing," *Mathematical Problems in Engineering*, pp. 1-10, 2013.

[**WAN14**] J. Wan, C. Zou, K. Zhou, R. Lu and D. Li, "IoT Sensing Framework with Inter-cloud Computing Capability in Vehicular Networking," *Electronic Commerce Research*, vol. 14, no. 13, pp. 389-416, November 2014.

[**WAN15**] L. Wang, H. Wu, W. Wang and K. C. Chen, "Socially Enabled Wireless Networks: Resource Allocation via Bipartite Graph Matching," *IEEE Communications Magazine*, vol. 53, no. 10, pp. 128-135, October 2015.

[**WHA14**] M. Whaiduzzamana, M. Sookhaka., A. Gania and R. Buyyab, "A Survey on Vehicular Cloud Computing," *Journal of Network and Computer Applications*, vol. 40, no. 4, pp. 325-344, 2014.

[**WOO07**] T. Wood, P. Shenoy, A. Venkataramani and M. Yousif, "Black-box and Gray-box Strategies for Virtual Machine Migration," *Proceedings of the USENIX Conference on Networked Systems Design and Implementation*, pp. 17-32, Cambridge, MA, April 2007.

[**WOO09**] T. Wood, G. Tarasuk-Levin, P. Shenoy, P. Desnoyers, E. Cechet and M. Corner, "Memory Buddies: Exploiting Page Sharing for Smart Colocation in Virtualized Data Centers,"

Proceedings of the ACM SIGPLAN/SIGOPS International Conference on Virtual Execution Environments, pp. 31-40, Washington, DC, March 2009.

[**WU11**] Y. Wu and M. Zhao, "Performance Modeling of Virtual Machine Live Migration," *Proceedings of the IEEE International Conference on Cloud Computing,* pp. 492-499, Washington, DC, July 2011.

[**XU10**] J. Xu and J. A. B. Fortes, "Multi-Objective Virtual Machine Placement in Virtualized Data Center Environments," *Proceedings of the IEEE International Conference on Conference on Cyber, Physical and Social Computing and the International Conference on Green Computing and Communications,* pp. 179-188, Hangzhou, China, December 2010.

[**YAN12A**] G. Yan, D. B. Rawat and B. B. Bista, "Towards Secure Vehicular Clouds," *Proceedings of the 6th International Conference on Complex, Intelligent and Software Intensive Systems,* pp. 370-375, Palermo, Italy, July 2012.

[**YAN12B**] G. Yan, D. Wen, S. Olariu and M. C. Weigle, "Security Challenges in Vehicular Cloud Computing," *IEEE Transactions on Intelligent Transportation Systems,* vol. 14, no. 1, pp. 284-294, 2012.

[**YE11**] K. Ye, X. Jiang, D. Huang, J. Chen and B. Wang, "Live Migration of Multiple Virtual Machines with Resource Reservation in Cloud Computing Environments," *Proceedings of the 4th IEEE International Conference on Cloud Computing,* pp. 267-274, Washington, DC, July 2011.

[**YU13**] R. Yu, Y. Zhang, S. Gjessing, W. Xia and K. Yang, "Toward Cloud-Based Vehicular Networks with Efficient Resource Management," *IEEE Network,* vol. 27, no. 5, pp. 48-55, 2013.

[**ZAB14**] V. Zaborovsky, V. Muliukha, S. Popov and A. Lukashin, "Heterogeneous Virtual

Intelligent Transport Systems and Services in Cloud Environments," *Proceedings of the International Conference on Networks,* pp. 236-241, Nice, France, February 2014.

[**ZHA12**] X. Zhang, Z.-Y. Shae, S. Zheng and H. Jamjoom, "Virtual Machine Migration in an Over-Committed Cloud," *Proceedings of the IEEE Network Operations and Management Symposium,* pp. 196-203, Maui, HI, April 2012.

Appendix A
Confidence Analysis

Throughout this thesis, various simulations were conducted, incorporating varying types of randomness. Results obtained from these simulations cannot be presented without a statistical analysis. All the results are subjected to a 95% confidence analysis based on the following calculations. A confidence interval can add another perspective to observe for a metric. A simulation set up is run several times, using the same initialization parameters. This allows the various random variables' randomness to impact the metrics. If enough simulations are run, a confidence interval, combined with the mean can be a more representative measure of the metric being analyzed.

In order to obtain the confidence interval for the various results presented in this thesis, the following equations were used.

Let:

X: random variable

μ: Average (mean) of X

σ^2: Variance of X

X_i: i^{th} sample of X simulation

n: Number of samples

x: Sample mean

s^2: Sample variance

The sample mean can be found using Equation (A.1) and the sample variance can be calculated using Equation (A.2).

$$x = \frac{1}{n}\sum_{i=1}^{n} x_i \qquad \text{(A.1)}$$

$$s^2 = \frac{1}{n-1}\sum_{i=1}^{n}(X_i - x)^2 \qquad \text{(A.2)}$$

If a random variable's distribution is unknown, for large sample sizes, the sample mean's distribution will approach a Gaussian distribution, based on the Central Limit Theorem. The sample mean approaches the ensemble mean and sample mean variance becomes a scaled version of the ensemble mean ($E[X] = \mu$ and variance of $x = \sigma_x^2 = \frac{\sigma^2}{n}$). Therefore, the confidence level is defined as the probability that x is below a certain distance from μ:

$$z = \frac{x - \mu}{\sigma_x} \qquad \text{(A.3)}$$

z: is a normal random variable (mean= 0 & variance = 1).

$$P(-z_\alpha < z < z_\alpha) = \alpha \qquad \text{(A.4)}$$

$$P\left[\frac{|x - \mu|}{\sigma_x} < z_\alpha\right] = \alpha \qquad \text{(A.5)}$$

By using $n > 30$ and hence the sample standard deviation s can be used instead of σ as it is difficult to find $\sigma_x = \frac{\sigma}{\sqrt{n}}$. The Gaussian distribution is be used. For the confidence level $\alpha = 0.95$, z_α is calculated.

Appendix B

Elsevier Rights for Reuse

ELSEVIER LICENSE
TERMS AND CONDITIONS

Sep 15, 2016

This Agreement between Tarek K Refaat ("You") and Elsevier ("Elsevier") consists of your license details and the terms and conditions provided by Elsevier and Copyright Clearance Center.

License Number	3923790594435
License date	Aug 07, 2016
Licensed Content Publisher	Elsevier
Licensed Content Publication	Vehicular Communications
Licensed Content Title	Virtual machine migration and management for vehicular clouds
Licensed Content Author	Tarek K. Refaat,Burak Kantarci,Hussein T. Mouftah
Licensed Content Date	April 2016
Licensed Content Volume Number	4
Licensed Content Issue Number	n/a
Licensed Content Pages	10
Start Page	47
End Page	56
Type of Use	reuse in a thesis/dissertation
Portion	full article
Format	both print and electronic
Are you the author of this Elsevier article?	Yes
Will you be translating?	No
Order reference number	
Title of your thesis/dissertation	Virtual Machine Management for Dynamic and Vehicular Clouds
Expected completion date	Sep 2016
Estimated size (number of pages)	125
Elsevier VAT number	GB 494 6272 12

1. The publisher for this copyrighted material is Elsevier. By clicking "accept" in connection with completing this licensing transaction, you agree that the following terms and conditions apply to this transaction (along with the Billing and Payment terms and conditions established by Copyright Clearance Center, Inc. ("CCC"), at the time that you opened your Rightslink account and that are available at any time at http://myaccount.copyright.com).

GENERAL TERMS

2. Elsevier hereby grants you permission to reproduce the aforementioned material subject to the terms and conditions indicated.
3. Acknowledgement: If any part of the material to be used (for example, figures) has appeared in our publication with credit or acknowledgement to another source, permission must also be sought from that source. If such permission is not obtained then that material may not be included in your publication/copies. Suitable acknowledgement to the source must be made, either as a footnote or in a reference list at the end of your publication, as follows:
"Reprinted from Publication title, Vol /edition number, Author(s), Title of article / title of chapter, Pages No., Copyright (Year), with permission from Elsevier [OR APPLICABLE SOCIETY COPYRIGHT OWNER]." Also Lancet special credit - "Reprinted from The Lancet, Vol. number, Author(s), Title of article, Pages No., Copyright (Year), with permission from Elsevier."
4. Reproduction of this material is confined to the purpose and/or media for which permission is hereby given.
5. Altering/Modifying Material: Not Permitted. However figures and illustrations may be altered/adapted minimally to serve your work. Any other abbreviations, additions, deletions and/or any other alterations shall be made only with prior written authorization of Elsevier Ltd. (Please contact Elsevier at permissions@elsevier.com)
6. If the permission fee for the requested use of our material is waived in this instance, please be advised that your future requests for Elsevier materials may attract a fee.
7. Reservation of Rights: Publisher reserves all rights not specifically granted in the combination of (i) the license details provided by you and accepted in the course of this licensing transaction, (ii) these terms and conditions and (iii) CCC's Billing and Payment terms and conditions.
8. License Contingent Upon Payment: While you may exercise the rights licensed immediately upon issuance of the license at the end of the licensing process for the transaction, provided that you have disclosed complete and accurate details of your proposed use, no license is finally effective unless and until full payment is received from you (either by publisher or by CCC) as provided in CCC's Billing and Payment terms and conditions. If full payment is not received on a timely basis, then any license preliminarily granted shall be deemed automatically revoked and shall be void as if never granted. Further, in the event that you breach any of these terms and conditions or any of CCC's Billing and Payment terms and conditions, the license is automatically revoked and shall be void as if never granted. Use of materials as described in a revoked license, as well as any use of the materials beyond the scope of an unrevoked license, may constitute copyright infringement and publisher reserves the right to take any and all action to protect its copyright in the materials.
9. Warranties: Publisher makes no representations or warranties with respect to the licensed material.
10. Indemnity: You hereby indemnify and agree to hold harmless publisher and CCC, and their respective officers, directors, employees and agents, from and against any and all claims arising out of your use of the licensed material other than as specifically authorized pursuant to this license.
11. No Transfer of License: This license is personal to you and may not be sublicensed, assigned, or transferred by you to any other person without publisher's written permission.
12. No Amendment Except in Writing: This license may not be amended except in a writing signed by both parties (or, in the case of publisher, by CCC on publisher's behalf).

13. Objection to Contrary Terms: Publisher hereby objects to any terms contained in any purchase order, acknowledgment, check endorsement or other writing prepared by you, which terms are inconsistent with these terms and conditions or CCC's Billing and Payment terms and conditions. These terms and conditions, together with CCC's Billing and Payment terms and conditions (which are incorporated herein), comprise the entire agreement between you and publisher (and CCC) concerning this licensing transaction. In the event of any conflict between your obligations established by these terms and conditions and those established by CCC's Billing and Payment terms and conditions, these terms and conditions shall control.

14. Revocation: Elsevier or Copyright Clearance Center may deny the permissions described in this License at their sole discretion, for any reason or no reason, with a full refund payable to you. Notice of such denial will be made using the contact information provided by you. Failure to receive such notice will not alter or invalidate the denial. In no event will Elsevier or Copyright Clearance Center be responsible or liable for any costs, expenses or damage incurred by you as a result of a denial of your permission request, other than a refund of the amount(s) paid by you to Elsevier and/or Copyright Clearance Center for denied permissions.

LIMITED LICENSE

The following terms and conditions apply only to specific license types:

15. **Translation**: This permission is granted for non-exclusive world **English** rights only unless your license was granted for translation rights. If you licensed translation rights you may only translate this content into the languages you requested. A professional translator must perform all translations and reproduce the content word for word preserving the integrity of the article.

16. **Posting licensed content on any Website**: The following terms and conditions apply as follows: Licensing material from an Elsevier journal: All content posted to the web site must maintain the copyright information line on the bottom of each image; A hyper-text must be included to the Homepage of the journal from which you are licensing at http://www.sciencedirect.com/science/journal/xxxxx or the Elsevier homepage for books at http://www.elsevier.com; Central Storage: This license does not include permission for a scanned version of the material to be stored in a central repository such as that provided by Heron/XanEdu.

Licensing material from an Elsevier book: A hyper-text link must be included to the Elsevier homepage at http://www.elsevier.com . All content posted to the web site must maintain the copyright information line on the bottom of each image.

Posting licensed content on Electronic reserve: In addition to the above the following clauses are applicable: The web site must be password-protected and made available only to bona fide students registered on a relevant course. This permission is granted for 1 year only. You may obtain a new license for future website posting.

17. **For journal authors:** the following clauses are applicable in addition to the above:

Preprints:

A preprint is an author's own write-up of research results and analysis, it has not been peer-reviewed, nor has it had any other value added to it by a publisher (such as formatting, copyright, technical enhancement etc.).

Authors can share their preprints anywhere at any time. Preprints should not be added to or enhanced in any way in order to appear more like, or to substitute for, the final versions of articles however authors can update their preprints on arXiv or RePEc with their Accepted Author Manuscript (see below).

If accepted for publication, we encourage authors to link from the preprint to their formal publication via its DOI. Millions of researchers have access to the formal publications on ScienceDirect, and so links will help users to find, access, cite and use the best available version. Please note that Cell Press, The Lancet and some society-owned have different preprint policies. Information on these policies is available on the journal homepage.

Accepted Author Manuscripts: An accepted author manuscript is the manuscript of an article that has been accepted for publication and which typically includes author-incorporated changes suggested during submission, peer review and editor-author communications.
Authors can share their accepted author manuscript:

- immediately
  - via their non-commercial person homepage or blog
  - by updating a preprint in arXiv or RePEc with the accepted manuscript
  - via their research institute or institutional repository for internal institutional uses or as part of an invitation-only research collaboration work-group
  - directly by providing copies to their students or to research collaborators for their personal use
  - for private scholarly sharing as part of an invitation-only work group on commercial sites with which Elsevier has an agreement
- after the embargo period
  - via non-commercial hosting platforms such as their institutional repository
  - via commercial sites with which Elsevier has an agreement

In all cases accepted manuscripts should:

- link to the formal publication via its DOI
- bear a CC-BY-NC-ND license - this is easy to do
- if aggregated with other manuscripts, for example in a repository or other site, be shared in alignment with our hosting policy not be added to or enhanced in any way to appear more like, or to substitute for, the published journal article.

Published journal article (JPA): A published journal article (PJA) is the definitive final record of published research that appears or will appear in the journal and embodies all value-adding publishing activities including peer review co-ordination, copy-editing, formatting, (if relevant) pagination and online enrichment.
Policies for sharing publishing journal articles differ for subscription and gold open access articles:
Subscription Articles: If you are an author, please share a link to your article rather than the full-text. Millions of researchers have access to the formal publications on ScienceDirect, and so links will help your users to find, access, cite, and use the best available version.
Theses and dissertations which contain embedded PJAs as part of the formal submission can be posted publicly by the awarding institution with DOI links back to the formal publications on ScienceDirect.
If you are affiliated with a library that subscribes to ScienceDirect you have additional private sharing rights for others' research accessed under that agreement. This includes use for classroom teaching and internal training at the institution (including use in course packs and courseware programs), and inclusion of the article for grant funding purposes.
Gold Open Access Articles: May be shared according to the author-selected end-user license and should contain a CrossMark logo, the end user license, and a DOI link to the formal publication on ScienceDirect.
Please refer to Elsevier's posting policy for further information.
18. **For book authors** the following clauses are applicable in addition to the above:
Authors are permitted to place a brief summary of their work online only. You are not allowed to download and post the published electronic version of your chapter, nor may you scan the printed edition to create an electronic version. **Posting to a repository:** Authors are permitted to post a summary of their chapter only in their institution's repository.
19. **Thesis/Dissertation**: If your license is for use in a thesis/dissertation your thesis may be submitted to your institution in either print or electronic form. Should your thesis be

published commercially, please reapply for permission. These requirements include permission for the Library and Archives of Canada to supply single copies, on demand, of the complete thesis and include permission for Proquest/UMI to supply single copies, on demand, of the complete thesis. Should your thesis be published commercially, please reapply for permission. Theses and dissertations which contain embedded PJAs as part of the formal submission can be posted publicly by the awarding institution with DOI links back to the formal publications on ScienceDirect.

Elsevier Open Access Terms and Conditions

You can publish open access with Elsevier in hundreds of open access journals or in nearly 2000 established subscription journals that support open access publishing. Permitted third party re-use of these open access articles is defined by the author's choice of Creative Commons user license. See our open access license policy for more information.

Terms & Conditions applicable to all Open Access articles published with Elsevier:

Any reuse of the article must not represent the author as endorsing the adaptation of the article nor should the article be modified in such a way as to damage the author's honour or reputation. If any changes have been made, such changes must be clearly indicated.

The author(s) must be appropriately credited and we ask that you include the end user license and a DOI link to the formal publication on ScienceDirect.

If any part of the material to be used (for example, figures) has appeared in our publication with credit or acknowledgement to another source it is the responsibility of the user to ensure their reuse complies with the terms and conditions determined by the rights holder.

Additional Terms & Conditions applicable to each Creative Commons user license:

CC BY: The CC-BY license allows users to copy, to create extracts, abstracts and new works from the Article, to alter and revise the Article and to make commercial use of the Article (including reuse and/or resale of the Article by commercial entities), provided the user gives appropriate credit (with a link to the formal publication through the relevant DOI), provides a link to the license, indicates if changes were made and the licensor is not represented as endorsing the use made of the work. The full details of the license are available at http://creativecommons.org/licenses/by/4.0.

CC BY NC SA: The CC BY-NC-SA license allows users to copy, to create extracts, abstracts and new works from the Article, to alter and revise the Article, provided this is not done for commercial purposes, and that the user gives appropriate credit (with a link to the formal publication through the relevant DOI), provides a link to the license, indicates if changes were made and the licensor is not represented as endorsing the use made of the work. Further, any new works must be made available on the same conditions. The full details of the license are available at http://creativecommons.org/licenses/by-nc-sa/4.0.

CC BY NC ND: The CC BY-NC-ND license allows users to copy and distribute the Article, provided this is not done for commercial purposes and further does not permit distribution of the Article if it is changed or edited in any way, and provided the user gives appropriate credit (with a link to the formal publication through the relevant DOI), provides a link to the license, and that the licensor is not represented as endorsing the use made of the work. The full details of the license are available at http://creativecommons.org/licenses/by-nc-nd/4.0.

Any commercial reuse of Open Access articles published with a CC BY NC SA or CC BY NC ND license requires permission from Elsevier and will be subject to a fee.

Commercial reuse includes:

- Associating advertising with the full text of the Article
- Charging fees for document delivery or access
- Article aggregation
- Systematic distribution via e-mail lists or share buttons

Posting or linking by commercial companies for use by customers of those companies.

20. **Other Conditions**:

v1.8

Questions? customercare@copyright.com or +1-855-239-3415 (toll free in the US) or +1-978-646-2777.