On the Integration of Unmanned Aerial Vehicles into Public Airspace

by

Mirmojtaba Gharibi

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Doctor of Philosophy
in
Computer Science

Waterloo, Ontario, Canada, 2020

Examining Committee Membership

The following served on the Examining Committee for this thesis. The decision of the Examining Committee is by majority vote.

External Examiner:	Abdallah Shami Professor, Dept. of Electrical and Computer Engineering, University of Western Ontario
Supervisor(s):	Raouf Boutaba Professor, Dept. of Computer Science, University of Waterloo Steven Waslander Associate Professor, Institute for Aerospace Studies, University of Toronto
Internal Member:	Ali Mashtizadeh Assistant Professor, Dept. of Computer Science, University of Waterloo Samer Al-Kiswany Assistant Professor, Dept. of Computer Science, University of Waterloo
Internal-External Member:	Stephen Smith Associate Professor, Dept. of Electrical and Computer Engineering, University of Waterloo

Author's Declaration

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

Unmanned Aerial Vehicles will soon be integrated in the airspace and start serving us in various capacities such as package delivery, surveillance, search and rescue missions, inspection of infrastructure, precision agriculture, and cinematography.

In this thesis, motivated by the challenges this new era brings about, we design a layered architecture called Internet of Drones (IoD). In this architecture, we propose a structure for the traffic in the airspace as well as the interaction between the components of our system such as unmanned aerial vehicles and service providers. We envision the minimal features that need to be implemented in various layers of the architecture, both on the Unmanned Aerial Vehicle (UAV)'s side and on the service providers' side. We compare and contrast various approaches in three existing networks, namely the Internet, the cellular network, and the air traffic control network and discuss how they relate to IoD.

As a tool to aid in enabling integration of drones in the airspace, we create a traffic flow model. This model will assign velocities to drones according to the traffic conditions in a stable way as well as help to study the formation of congestion in the airspace. We take the novel problem posed by the 3D nature of UAV flights as opposed to the 2D nature of road vehicles movements and create a fitting traffic flow model. In this model, instead of structuring our model in terms of roads and lanes as is customary for ground vehicles, we structure it in terms of channels, density and capacities. The congestion is formulated as the perceived density given the capacity and the velocity of vehicles will be set accordingly. This view removes the need for a lane changing model and its complexity which we believe should be abstracted away even for the ground vehicles as it is not fundamentally related to the longitudinal movements of vehicles. Our model uses a scalar capacity parameter and can exhibit both passing and blocking behaviors. Furthermore, our model can be solved analytically in the blocking regime and piece-wise analytically solved when in the passing regime.

Finally, it is not possible to integrate UAVs into the airspace without some mechanism for coordination or in other words scheduling. We define a new scheduling problem in this regard that we call Vehicle Scheduling Problem (VSP). We prove NP-hardness for all the commonly used objective functions in the context of Jobshop Scheduling Problem (JSP). Then for the number of missed deadlines as our objective function, we give a Mixed Integer Programming (MIP) formulation of VSP. We design a heuristic algorithm and compare the quality of the schedules created for small instances with the exact solution to the MIP instance. For larger instances, these comparisons are made with a baseline algorithm.

Acknowledgements

I am grateful to Prof. Steven Waslander for his support and especially his critical feedback on my writing. I consider these feedback the main learning experience during my PhD. I am grateful to Prof. Raouf Boutaba for his generous support and encouragement and believing in me to accomplish my visions. This has been very nurturing and motivating in my time in the PhD program. I am blessed to have had such an experience.

I would like to thank Yashar Ganjali for valuable feedback on the Internet of Drones project. Also, I would like to thank Claudio Canizares, Ehsan Nasr Azadani, and Amir Khajepour for their support with the traffic model for UAVs project. Also, I am grateful to Stephen Smith and Abbas Mehrabian for fruitful discussions on the vehicle scheduling problem.

I would like to thank my parents and family for helping me through all these years, especially the last couple of years. I am also grateful for my friends in Waterloo who helped make my studies a more joyful experience.

Dedication

This is dedicated to my parents.

Table of Contents

List of Tables x

List of Figures xi

Abbreviations xii

1 Introduction 1

1.1 Internet of Drones . . . 3

1.2 Vehicle Scheduling Problem (VSP) . . . 5

1.3 Traffic flow models for UAVs . . . 8

2 Internet of Drones 13

2.1 Introduction . . . 14

2.2 Relevant Networks . . . 14

2.2.1 Air Traffic Control Network . . . 14

2.2.2 Cellular Network . . . 17

2.2.3 Internet . . . 19

2.3 Architecture . . . 20

2.3.1 Structure . . . 21

2.3.2 Components . . . 22

2.3.3 Layers . . . 24

2.3.4 Cross-cutting features . . . 31
2.4 Operation model . . . 31
2.4.1 Model . . . 31
2.4.2 Interactions with outside . . . 33
2.4.3 Strategies for deployment . . . 33
2.5 Discussion and Future Work . . . 34
2.5.1 Goals, principles, and benefits of my design . . . 34
2.5.2 Routing . . . 36
2.5.3 Congestion control . . . 37
2.5.4 Communication signalling . . . 39
2.5.5 Addressing schemes . . . 39
2.5.6 Drones and minimum performance . . . 40
2.5.7 Security . . . 40
2.5.8 Validation and technical implementation . . . 40
2.5.9 Economics of IoD . . . 41
2.5.10 Legislation . . . 41

3 Vehicle Scheduling Problem **45**
3.1 Introduction . . . 46
3.2 Problem definition . . . 46
3.3 NP-hardness proof . . . 47
3.4 Exact MIP formulation . . . 50
3.4.1 Notations . . . 51
3.4.2 Objective function . . . 51
3.4.3 Constraints . . . 51
3.5 Scheduling algorithms . . . 52
3.5.1 Baseline algorithm: PROXIMITY . . . 52
3.5.2 Heuristic algorithm: DEADLINE & PROXIMITY . . . 52
3.5.3 Complexity . . . 54
3.6 Numerical results . . . 55

4 Traffic Flow Model For UAVs **60**
4.1 Introduction . . . 61
4.2 Model . . . 61
4.2.1 Discussion and design philosophy . . . 62
4.3 Analytical solution . . . 63
4.3.1 Blocking regime . . . 63
4.3.2 Passing regime . . . 65
4.3.3 Stability analysis . . . 65
4.4 Model properties . . . 66
4.4.1 Soundness . . . 66
4.4.2 Passing or blocking behavior . . . 68
4.4.3 Asymptotic behavior in passing regime . . . 69
4.4.4 Asymptotic behavior in blocking regime: linear stability analysis . . . 73
4.4.5 Asymptotic behavior in blocking regime: nonlinear stability analysis 77

5 Conclusion **82**
5.1 Concluding remarks . . . 83

6 Discussion and Future Work **85**
6.1 Discussion & Future work . . . 86
6.2 Notes on Chapter 3 . . . 87
6.3 Notes on Chapter 4 . . . 89

References **90**

APPENDICES **99**

A Helpful lemmas for chapter 3 **100**
A.1 Lemmas and their proofs . . . 100

List of Tables

2.1 Layers in the architecture of IoD 27

4.1 Effect of capacity (κ) on the model's behavior 74

List of Figures

2.1 ARTCCs in the contiguous United States 15
2.2 An illustration of the cellular networks and base stations. 18
2.3 Layers in the architecture of the Internet 20
2.4 An illustration of the airways, intersections, and nodes 23
2.5 The zone graph 24
2.6 The interzone graph 25
2.7 A schematic of the Greater Toronto Area, ZSPs, and IoDSPs 26
2.8 IoD system boundaries 44

3.1 Scheduling subroutine 53
3.2 Scheduling sorting key subroutine 54
3.3 Simulation grid like graph 56
3.4 Simulations results for 25-100 vehicles 57
3.5 Changes of run time with increase in the number of vehicles 58
3.6 Run time results of the heuristic algorithm 59

4.1 Vehicle indexes on the one directional link 61
4.2 Low link capacity and effect on passing vehicles 72
4.3 High link capacity and effect on passing vehicles 73
4.4 Vehicle flow versus density 79
4.5 Time-Space diagram 80
4.6 Minimum and maximum momentary velocity among all vehicles 81

Abbreviations

ACK Acknowledgement 37, 38

ADS Automatic Dependent Surveillance 43

ADS-B Automatic Dependent Surveillance-Broadcast 4, 16, 43

ARC Advisory and Rulemaking Committee 43

ARTCC Air Route Traffic Control Center 15, 16, 38

ATC Air Traffic Control 14–16, 19, 35, 38, 42, 43

ATCSCC Air Traffic Control System Command Center 16, 38

ATM Asynchronous Transfer Mode 36

C2 Control and Communication 42

CAC Call Admission Control 36, 37

COA Certificate of Waiver or Authorization 42

DCAC Distributed Call Admission Control 37

DCV Density/Capacity View 9, 10, 12, 61–63

DHL Dalsey, Hillblom and Lynn 2

ECN Explicit Congestion Notification 38

FAA Federal Aviation Administration 3, 5, 14, 15, 32, 33, 41–44

Fedex Federal Express 2

FVDM Full Velocity Difference Model 10–12

GTA Greater Torontoa Area 22

IDM Intelligent Driver Model 10–12

IFR Instrument Flight Rules 43

IoD Internet of Drones iv, viii, x, 2–5, 8, 14, 16–22, 24, 26, 27, 31, 33–41, 43, 44, 62, 83, 86, 87

IoDSP Internet of Drones Service Provider 22, 26, 32, 34

JSP Jobshop Scheduling Problem iv, 6, 46, 48, 50, 83

M2M Machine to Machine 33

MIP Mixed Integer Programming iv, viii, 7, 8, 46, 50, 55–57, 59, 83, 84

MOBIL Minimizing Overall Braking Induced by Lane change Model 10

MTSO Mobile Telecommunications Switching Office 17, 18

NASA National Aeronautics and Space Administration 3, 11

NP Non-deterministic Polynomial Time 8, 46, 47, 49, 50, 83

NWS National Weather Services 33

OMV One/Multilane View 9–11, 61–63

OVM Optimal Velocity Model 10–12

QoS Quality of Service 38

SAA Sense And Avoid 42

SCM Scalar Capacity Model 61

TCAS Traffic Alert and Collision Avoidance System 40

UAS Unmanned Aircraft System 41–43

UAV Unmanned Aerial Vehicle iv, vii, ix, 2–6, 8–11, 43, 46, 60–63, 84, 86, 87

UTM Unmanned Aircraft System Traffic Management 3

V2V Vehicle to Vehicle 4

VoIP Voice over IP 19

VSP Vehicle Scheduling Problem iv, vii, 2, 5–8, 46–51, 55, 83, 88

VTOL Vertical Take Off and Landing 16, 23, 40

XML eXtensible Markup Language 21

ZSP Zone Service Provider 4, 22, 23, 26–36, 38, 39, 41, 44

Chapter 1

Introduction

With the on-going miniaturization of sensors and processors and ubiquitous wireless connectivity, drones are finding many new uses in enhancing our way of life. There are many applications for drone technology, ranging from the on-demand package delivery, to traffic and wild life surveillance, inspection of infrastructure, search and rescue, agriculture, and cinematography. All drone applications share a common need for both navigation and airspace management. However, this is a field that is still in its infancy and ideas for integration of UAVs in the airspace are just starting to appear [30, 48, 91, 44, 62, 47, 61].

Among these applications, aerial package delivery will most urgently require a robust airspace allocation architecture, as it could result in many thousands of daily flights in the same geographic area, with many potential conflicts between drones navigating along similar or intersecting routes. The benefit to the global logistics network is clear, as drones could usher in a new era of on-demand delivery, and has been shown to be cost-competitive relative to ground-based delivery as well[14], although longer haul transport clearly benefits from bundling onto larger transport vehicles. Amazon states that about 83% of their packages weigh below 2.5 kg [37], a reasonable maximum payload for today's drones. Similarly, the average weight of packages delivered by Federal Express (Fedex) is less than 5kg [27]. In my opinion, this model can provide on-demand, inexpensive, and convenient access to the goods and items already in or near an urban area, including consumer goods, fast-food, medicine, and even on-demand groceries.

Despite a wave of drone package delivery prototype announcements (e.g. Matternet [75], Amazon's prime air [4], Google's project wing [84], and Dalsey, Hillblom and Lynn (DHL)'s Parcelcopter [16]), the prospect of integrating drones in the airspace is still unclear.

I believe to successfully integrate drones into the airspace, various components are needed. We need an architecture that coordinates the access to the airspace and provides different services that are commonly needed by drones' applications such as navigation services. In Chapter 2, I design an architecture called IoD to achieve this purpose. This will define the framework for other problems that need to be solved.

Within IoD, an important problem to solve is that of scheduling drones. In IoD, I advocate for centralized scheduling of drones and it will be an integral part of any system whether they are based on IoD or any other architecture that will be used for integrating drones into the airspace. In Chapter 3, in this regard I formulate an optimization problem called Vehicle Scheduling Problem (VSP) and provide a heuristic algorithm for solving it.

In Chapter 4, I provide a traffic flow model for drones which is motivated by their unique characteristics such as the 3D movement compared to the ground vehicles. The motivation for this problem is two-fold:

- Firstly, it serves as a stable speed assignment scheme to be used for the movements of UAVs over the links. Even though, the speed assignment scheme is for UAVs on a single link as a first step, it is straightforward to generalize it to UAVs on a network in a future work as follows. For each drone, the single link is replaced by the path from the source to destination for that drone. Then, to calculate the speed for the current drone I consider all the drones on that path as well as drones that are about to merge on that path in my speed assignment scheme.

- Secondly, we need computing tools for analyzing the behaviors of drones in the air and use the gained insights to improve the airway structures in the IoD architecture and provide additional capacity or services as the need for them becomes clear. Among these tools are traffic flow models. In the traffic literature for ground vehicles, traffic models have a long history and have been successfully used to analyze various traffic conditions such as formation of congestion and in general the interaction of vehicles with the road network infrastructure.

Lastly, various algorithms are needed as will be discussed in my IoD architecture to make the integration project successful.

I dedicate the next 3 subsections (one for each of these 3 contributions outlined above from Chapter 2-4) to provide more details about each contribution.

1.1 Internet of Drones

The Internet of Drones is an architecture designed for providing coordinated access to controlled airspace for UAVs, often referred to as drones. In this thesis, in Chapter 2, I lay the architecture for generic services that can provide navigation and airspace management for all current and future applications.

To the best of my knowledge, at the time of publication of the research [30] by my coauthors and I, there were not any rigorous publication concerning the architecture of a drone-specific air traffic management system as the technology is still in its infancy. One good starting point is National Aeronautics and Space Administration (NASA)'s Unmanned Aircraft System Traffic Management (UTM) project [62, 61, 47], which organized a symposium to begin preparations of a solution for low altitude traffic management to be proposed to the Federal Aviation Administration (FAA). Related to this effort, both Amazon [5, 6] and Google [35] have published white papers which explore some of the strategies for managing the airspace and coordinating aerial vehicles through onboard

system requirements such as Automatic Dependent Surveillance-Broadcast (ADS-B) and Vehicle to Vehicle (V2V) communication. In all the works cited above, various ideas are presented in an unsystematic way with the aim of using them in the design of an air traffic management system. The interaction between these ideas are not studied. Furthermore, there is not enough structure to explore the connection of these ideas to the existing large scale networks in other areas and to reason about their viability based on the existing experiences in these networks. However, my contribution is to approach the drone airspace management problem by providing a universal architecture and a vocabulary of concepts to describe the IoD. In the future, different IoD systems can be developed based on it with their set of protocols and implementations of the features required by my IoD architecture. I suggest a possible operational model based on my architecture and I discuss the desired goals of the architecture and also the benefits that it provides as well as the subtleties that have to be addressed for any IoD system.

Shortly after my coauthors and I published our preprint[31], authors in [15] published a preprint exploring some of the ideas pertaining to a UAV traffic network, called uNet. In uNet, instead of using a free-flight mode, similar to my architecture as will be explained, the airspace is divided into predefined routes. The authors argue that this provides for less reliance on advanced sense and avoid technologies and the ease of assigning conflict-free routes to the drones using the existing techniques. They consider use of sector-level uNets (sNets) where the traffic in each sector is under the authority of that particular uNet. I have a similar construct in my architecture with different zones where each zone is under authority of one or multiple Zone Service Provider (Zone Service Provider (ZSP)). However, one difference is that in my architecture, more than one ZSP can participate in managing the same zone. Furthermore, I take a systematic approach in defining the layers of the architecture as well as the features that have to be implemented for each layer.

My core contribution in Chapter 2 is formulating a complex and multifaceted problem and showing how on an abstract level, it is related to the vast amount of existing literature on the three existing networks, namely air traffic control, cellular network, and the Internet. I have crafted a blueprint for the implementation of an IoD system based on my IoD architecture. By comparing the challenges that IoD and each of the three named networks address in an abstract way, I have established relationships between existing solutions to the specific problems of IoD, hence creating well formulated open problems for the research community in a diverse range of fields. For instance, on an abstract level all four networks have to route physical objects or data. I have uncovered this connection and others such as congestion control, admission control, and addressing schemes. I have explained the existing strategies and made it clear what prevents a straight forward adoption of them for IoD on some of these matters.

As mentioned before, although there have been numerous announcements in the media on drone applications such as package delivery prototypes, there are not any publication on the architecture for these systems. The FAA's move to address integration of drones in the national airspace[22], in response to a mandate by the US House of Representatives[66] reiterates that IoD is a timely architecture that addresses important questions in this arena. Although there is significant excitement in the industry, to this date, this topic has not received much attention in the academic community. IoD serves as a first step for bringing these important issues to the forefront of academic endeavours and provides the academic community with well-defined problems to tackle. My hope is that an implementation of IoD in the next three to five years will make on-demand package delivery as well as other drone applications possible.

1.2 Vehicle Scheduling Problem (VSP)

VSP is a new scheduling problem that I define in this work. Many of the scheduling problems in the literature are motivated by real life applications. The motivation for defining this new scheduling problem is the impending integration of the UAVs into the airspace and a lack of framework for accommodating them in a scalable way. They will be used in a wide array of applications from search and rescue, to package delivery, traffic enforcement, infrastructure inspection and cinematography. This means in any city, there will be a high amount of congestion that needs to be managed to prevent mid-air collisions as well as to provide an efficient service[30]. While my main motivation comes from the application of UAVs, my scheduling problem is generic and malleable enough to use in other areas as well.

Within the context of IoD, the scheduling algorithms I develop are a first step toward a scheduler inside the zones that will assign the arrival time stamps for intersections in the air in that zone to each UAV. There is more work needed to have a fully operational scheduler within IoD and some necessary next steps are discussed in Chapter 6 for bridging that gap.

On a high level, the problem I try to solve is as follows. We are given a path over a graph for each vehicle. My goal is to minimize the number of tardy vehicles (or any other objective function) subject to the deadlines, minimum and maximum allowed speeds on the links, and the separation time gap needed when entering the nodes (which play the role of intersections). One may wish to formulate the problem as a joint optimization of routing and scheduling whereas in this work I am only interested in the scheduling aspect. This is a valid problem on its own as for various reasons, as the operator, we might not be

authorized to make routing decisions. For instance, the government might restrict UAV flights for a company to only certain paths. Focusing on only the scheduling aspect also has the advantage of allowing for more specialized heuristics given the more structure imposed on the problem.

I first compare VSP with a large class of scheduling problems known as JSP. The Job Shop Scheduling problem comes in many types which are motivated by the real life problems they strive to solve. Therefore, given the vast amount of literature on the subject, a first point of attack will be to model my problem in terms of a variant of JSP. In the classic Job Shop Scheduling problem, we are given a set of jobs, each composed of a chain of operations, and each operation can be performed on a specific machine from the set of all machines. A machine can only process one operation at a time at a specified processing time. At first sight, it seems the nodes in my graph can be simulated by the machines and the separation time is analogous to the processing time. However, upon further inspection, it is not clear how to model the time it takes for a vehicle to reach from one node to another. Of course, this does not seem possible in a straightforward way using the classical type. However, even using variants of JSP with properties including BLOCKING, NO-WAIT, SETUP TIMES, etc. does not seem to represent my problem in an uncomplicated way. In JSP with BLOCKING, machines will hold up the operation and remain busy if the next machine is busy which corresponds to a lack of buffer between machines. In JSP with NO-WAIT, a task cannot wait between machines. In JSP with SETUP TIMES, there will be a time delay to set up a new job on a machine. For a reference to these variants of JSP, look at e.g. [71].

Some researchers have extended JSP and these more or less standardized variants to include transportation times between the machines. That is for a job to start executing in the next machine, it will be delayed by the transportation time between the previous machine and the new one. For instance, see [46] and [79]. However, in these cases, the transportation time is fixed whereas in VSP the velocity over a link can be chosen from a range. Furthermore, in some of these works, one or more robots are used for transportation with empty trips as necessary which again does not have a resemblance to my work [46],[39]. One might propose to extend JSP with transportation robots further to include variable transportation times. However, even with these changes, VSP remains a different problem. For example, if the transportation robot reaches the next machine driving at its minimum speed, to model VSP, the job must be executed on the next machine right away; in concept similar to (but not quite) a NO-WAIT condition. Furthermore, for each vehicle in our problem, we will need a transportation robot which will make it an unusual setting for JSP with transportation robots since the raison d'etre of the problem is to treat these robots as extra bottlenecks whose usage needs careful planning and scheduling. However,

in the proposed setting, there will never be a shortage of them.

Scheduling in computer networks is another area with a vast literature. However, the underlying structure in computer networks is different. Firstly, the velocity of each packet during transmission on a link is constant and equal whereas vehicles can have different velocities over the links. Another difference is that packets might drop (i.e. vanish if needs be), but this is not an option for the vehicles. Furthermore, the bottleneck are the router buffers (similar to the nodes in my work) and the packets spend most of their time in the routers whereas in my case vehicles spend only a minimal amount of time at a node and spend most of their times travelling on the links. These result in drastically different scheduling algorithms and policies which makes it difficult to use them in VSP [69].

In the context of the air traffic management, the problems are formulated differently and the algorithms being used are not directly applicable to my problem. To be more precise, in the context of the air traffic management, the air space is separated into sectors that are basically a volume of airspace and the goal is to avoid over loading each sector by the means of either postponing a flight or changing the calculated route for a flight [3, 65]. Therefore the underlying graph structure in my problem is different.

An area of research that has some similarities is the train scheduling literature. There are similarities and differences with my approach. Most of these models are designed for a single main line with multiple short segments attached to the main line where trains can effectively park and let other trains take over before continuing their travel on the main line. This is the core difference as in my problem, drones are allowed to pass each other on each link and links have unlimited capacity. Apart from this core structural difference, and as a consequence, the ways these problems are formulated are different. In train scheduling, the railway (edges on the railway graph) is segmented into so called blocks. These are treated as bottlenecks whereas in my case the edges are not the bottlenecks. Only the intersections (vertices) of the transportation network are the bottlenecks. One similarity to my approach is the use of an MIP model and a similar idea of treating arrival Time Stamps at a segment as a modelling variable (see [17] for an example). Furthermore, there will be different set of constraints involved as well such as assignment of locomotives and crews which do not have a counterpart for unmanned aerial vehicles[26, 13]. Additionally, the underlying networks encountered in practice are vastly different. While in the railway network, the edges are relatively long and there are few intersections, in the airway network for drones, edges are relatively short, similar to the road network, and there is an abundance of intersections.

My contribution in Chapter 3 can be summarized as follows.

I define a new scheduling problem called Vehicle Scheduling Problem and formulate it in terms of a mixed integer linear programming. My model has applications among other

things to movements of autonomous vehicles over a transportation network; especially autonomous unmanned aerial vehicles. The model is versatile in that I can model vehicles with various minimum and maximum speeds as well as deadlines. Furthermore, I can adjust the safety gap as needed per pair of vehicles for each intersection (that is nodes of the graph).

I then proceed to show the Non-deterministic Polynomial Time (NP)-hardness of VSP for all commonly used objective functions in the context of job shop scheduling problems. These include minimizing

- Makespan: The time the last vehicle exits the graph.
- Total (weighted) completion time: Total or (equivalently) the average travel time with potentially different weights for different vehicles.
- Maximum lateness: The maximum (positive or negative valued) difference between deadline and trip completion of all vehicles.
- Total tardiness: The total time past the deadlines.
- (Weighted) number of tardy vehicles: The number of vehicles that missed their deadlines.

It is possible to provide an MIP of VSP for all these objective functions. To demonstrate that with one particularly important objective function for delivery problems, I pick the objective function of number of tardy vehicles and give an MIP formulation of VSP. To deal with the computational complexity of VSP in this case, I devise a heuristic algorithm in the case where all trips are requested at the same time. I analyze the complexity of my algorithm and compare the solutions yielded from my algorithm to the optimal solution to MIP for a few random instances with a small number of vehicles. I also compare these results to a baseline algorithm that I designed.

Finally, I conclude with a discussion of the shortcomings of my algorithm such as sensitivity to the noise as well as ideas for future improvement.

1.3 Traffic flow models for UAVs

To make integration of the UAVs in the airspace a reality in the realm of IoD, various technical tools are needed, including traffic flow models over a single link. This is my

topic of interest for Chapter 4. Given the target time stamps dictated by the scheduler between intersections in a zone for a UAV, we need a model that flies UAVs between these intersections over a link while respecting the capacity of the link. A microscopic traffic flow model for a single link accepts various parameters about each UAV as an input (e.g. maximum free flow speed) and produces the linear trajectories for these movements.

Furthermore, my traffic flow model can be used as a traffic engineering tool as follows. By revealing the instances of failure in meeting these targets, it gives insight about formation of congestion and information about subjects such as whether the capacity on various links should be decreased or increased to improve the success rate. In their traditional domain of ground vehicles, traffic flow models help with understanding the formation of traffic jams as a result of various flow conditions, driving behaviors, road structures such as on-ramps and off-ramps, etc. They will play an analogous role for UAVs.

Developing microscopic traffic flow modeling for UAVs is a new problem with its unique set of requirements. The closest related research area we can look for solutions is that of traffic flow models for ground vehicles. As we will see, even the limited existing works on UAV traffic flow models are adaptations of ground vehicle traffic flow models. A main characteristic of traffic flow models for ground vehicles is that they structure the road into one or multiple lanes and allow the movement of vehicles in this 2D space [41]. I call this general view of the modelling One/Multilane View (OMV). Within OMV, in the simpler case of one lane, no passing occurs. Most models are first introduced as one lane models and then with the aid of a separate lane changing model are extended to multi-lane models [41, 42, 86].

An OMV-based model is limited in its application to UAVs as their movements are in the 3D space and lanes are not defined. Furthermore, not only the pass planning aspect is ambiguous in the 3D space, but also a low level detail that adds to the complexity of a microscopic model and therefore should be aggregated. This is so since the overall goal is understanding the longitudinal movements of vehicles along the highway. Finally, in OMV models, a velocity will be assigned to each vehicle based on the congestion in their lane. In the same vein, it is ambiguous how the velocity must be determined in the 3D space with no lanes.

The main problem is to formulate a traffic flow model in a 3D space with no lanes for UAVs. I solve this problem by using a concept of a channel in which vehicles move and a density/capacity framework where for a vehicle to move forward, the density (or congestion) in its horizon must be under the set capacity of the channel. That is the velocity of each vehicle is set based on the perceived congestion. I call this general view in modelling, a Density/Capacity View (DCV) as an alternative to OMV. A DCV-based

model also aggregates the pass planning aspect by allowing a vehicle to pass when the congestion is sufficiently low. Furthermore, if instead we use an OMV model to represent the UAV traffic flow and velocity assignment, we effectively limit the movement of UAVs in a way that is artificial. This is due to the fact that while ground vehicles can overtake each other only by moving to the adjacent lanes, we do not have the same channel topology for UAVs in the air. To illustrate the difference, even if we decide to organize the UAV traffic flow in lanes, one might envision many lanes and for a UAV to pass another one, any of these lanes can be used. In other words, any of the lanes are considered adjacent whereas on the road network this is impossible due to the 2D nature of the roads. Therefore, using an OMV model for UAVs will result in reduced traffic flow when used as a velocity assignment scheme.

In this work, the main novelty is to eliminate lanes and formulate a DCV-based microscopic traffic flow model for UAVs with application to ground vehicles as well. Furthermore, my model can exhibit both blocking and passing regimes (analogous to one and multi-lane models) by setting a scalar capacity parameter κ below or above a threshold, respectively. My model is among a few models [38, 63, 92] that can be solved analytically in the blocking regime and piece-wise analytically in the passing regime. In contrast to the existing literature on multi-anticipation [88, 89, 55, 18, 32], my model sets the velocity for each vehicle in a novel way by calculating the overall density in front of each vehicle and imposing a decaying exponential weight on the distances to every vehicle in the front. Finally, I prove various properties for my proposed model, including the stability analysis for the blocking case and the characterization of the asymptotic behavior in the passing case.

In the remainder of this section, I study the related work on traffic flow models in more details. Car following theories model the vehicles' movements on a single lane as they follow each other [41]. There are separate lane changing models such as Minimizing Overall Braking Induced by Lane change Model (MOBIL) [42] or the model in [86] that are used to extend these models to multiple lanes.

Most (if not all) of the modern microscopic models are modelled as either single lane or multi-lane. These include most of the well-known traffic flow models (and their extensions) such as Optimal Velocity Model (OVM) [7], Full Velocity Difference Model (FVDM) [41], Intelligent Driver Model (IDM) [87], and Newell's Car-Following Model [64].

I argued above that pass planning should be aggregated. It is worth noting that in [52], for macroscopic models (with lanes), authors define a rate of lane changing based on macroscopic quantities such as density. In [53], based on the work of [52], authors combine this with a microscopic model together with quantizing the prescribed rate to make it applicable to the microscopic model. However, still the model is essentially OMV-based,

although to some extent the lane changing modeling complexity is avoided.

The literature in the area of UAV traffic flow models is very sparse. I am aware of the following two studies.

To integrate UAVs in the airspace, researchers in NASA [40], propose various structures for the airspace; including a road network like design (below the skyline; that is the tallest building height in a city) similar to my work in [30]. They set certain behavioral rules (i.e. a traffic flow model) for UAVs and accordingly extract the fundamental diagram of flow versus density. However, no stability analysis is done even though it is the standard in the traffic engineering community. Authors perform only a numerical simulation under an acceleration from a standstill, followed by cruising and then braking of the leader on a flight lane. The traffic flow model is an OMV-based multi-lane model similar to that of ground vehicle models. In the model, authors consider the reaction delay. Their traffic flow model is based on a constant gain controller that adjusts the velocity to reach a goal velocity for some required separation. Also, the lane change is done collaboratively utilizing wireless communication between vehicles.

In [8], with the goal of studying the wind effect on the fundamental diagram, the authors extend a car following model by Greenshields et al. [36] to include the wind force. This is a 1-lane model and no stability analysis is performed for the new model beyond what is already done for the original model by the research community.

In the context of ground vehicle traffic flow models, traffic flow theory finds its root in the work of Greenshields in 1930s [36]. Traffic flow models can be classified across different dimensions, such as the aggregation level. Macroscopic models take a high level view of traffic flow similar to the flow of liquids or gases. Quantities of interest are local density, flow, mean speed and variance and their evolution through time [59, 77, 88, 43, 33, 51, 57]. Microscopic models (e.g. see below) to which my model belong such as car-following or cellular automata models describe the interaction of each driver with its environment. In these models, we are interested in quantities such as individual position and speed and perhaps acceleration[88].

Within microscopic models, I categorize the models based on their relevance to my model. In particular, a distinction is made between 1-lane or multi-lane models. Many of the classic models are 1-lane models. Among the classics are OVM [7], FVDM [41], and IDM [87] whereas [58] is a more recent example. However, it is possible to extend these to multi-lane models by use of a lane change model such as MOBIL which dictates the rule of when it is safe and beneficial for a vehicle to change lanes[42].

Another distinction is whether a vehicle takes the optimal velocity in equilibrium instantly similar to my model or gradually. Models with delays are able to demonstrate

delay-induced traffic phenomena at the expense of added complexity. No delay classic models include Reuschel and Pipe's models [76, 72]. Classic models such as OVM[7], FVDM[41], and Newell's Car-Following Model[64] exhibit delay.

Furthermore, one difference is whether the drivers only react to the immediate vehicle in the front or beyond. In particular, in multi-vehicle anticipation models, a few vehicles at the front are considered by the driver for better stability (fewer accidents) [88]. In [89], the authors extend some of the traffic flow models including OVM, FVDM, and IDM by adding multi-vehicle anticipation features. In [55] and [18], the authors extend OVM and Gipps[32].

Additionally, a distinguishing factor is whether the velocity is adjusted based on the time gaps between two vehicles or the space gaps (such as my model). Models such as FVDM [41] and IDM [87] use time gaps whereas OVM [7] and Newell's car following model [64] use space gaps.

I know of very few models that can be solved analytically. A 1-lane model by Hasebe et al. [38] uses the hyperbolic tangent function to relate the distance between only subsequent vehicles to their velocity with exact solution for various delays.

In a highly related work [63], Newell designs a 1-lane model that can be solved analytically. It was later extended by Whitham[92], finding various exact wave solutions, such as periodic and solitary waves. The model assigns the velocity at time $t + \Delta$ to a follower vehicle according to an exponential decay congestion term at time t where Δ is a delay constant. The congestion term is based on only the distance between the follower and the leader. This results in a non-linear differential equation which Newell transforms into a linear form when $\Delta = 0$ and the cars are identical. There are similarities and differences in how this model relates to my work. I used a similar technique to make my differential equations linear. Also, I use an exponential decay scheme, but my formulation is different in that I use all the vehicles in the front and not just the first one. My model is DCV-based and can exhibit passing or blocking behavior according to the set value for capacity whereas this is a 1-lane model. Furthermore, except of having the same horizon for each car, I do not require cars to be identical. Certain details of the models are also different. For example, my model being DCV based, does not have a concept of minimum headway or vehicle length.

Finally, stability analysis is an important part of the study of any traffic flow model. References [88] and [93], establish various needed stability criteria for a traffic flow model.

Chapter 2

Internet of Drones

2.1 Introduction

The Internet of Drones (IoD) is a layered network control architecture designed mainly for coordinating the access of unmanned aerial vehicles to controlled airspace, and providing navigation services between locations referred to as nodes. The IoD provides generic services for various drone applications such as package delivery, traffic surveillance, search and rescue and more. In this chapter, I present a conceptual model of how such an architecture can be organized and I specify the features that an IoD system based on my architecture should implement. For doing so, I extract key concepts from three existing large scale networks, namely the air traffic control network, the cellular network, and the Internet and explore their connections to my novel architecture for drone traffic management. As they were reviewed more in depth in the introduction, the existing efforts on the integration of drones into the airspace lack a systematic view to the problem. They resemble a list of ideas with little organization. In contrast, in this chapter, we tackle this problem by viewing it as a system design problem.

2.2 Relevant Networks

For designing the architecture of the IoD, I study three distinct large scale network structures; namely Air Traffic Control (ATC), cellular network, and the Internet. Each of these networks achieves some of the goals or functionalities I desire for the IoD. In each case, however, their conceptual architecture falls short of providing a thorough solution to the unique challenges of IoD. Hence, the importance of studying these systems is twofold. First, they have valuable lessons about how a scalable and fault tolerant network can be engineered. Second, their differences guide us to IoD's specific challenges which have not been tackled before and are in need of innovative solutions. I describe these structures through a discussion of goals and functionality that are relevant to IoD and the differences with IoD that need to be addressed in my architecture.

2.2.1 Air Traffic Control Network

ATC has strong relevance to IoD as efficiently utilizing the airspace and maintaining collision free navigation is an integral part of any IoD architecture. The functioning of ATC follows similar procedures around the globe. I briefly summarize the components of ATC in the United States. The FAA is in charge of regulations and air safety, and has partitioned

the United States' airspace into 24 areas each managed by one of the 24 Air Route Traffic Control Center (ARTCC) (Fig. 2.1). There are bilateral letters of agreement between any two adjacent ARTCCs on how aircraft must transition from one ARTCC to another. Similarly, within each ARTCC, the airspace is partitioned into between 20 to 80 sectors and each sector is exclusively managed by one controller and the aircraft transitions between sectors are done according to facility directives. The main driver in designating the boundaries of ARTCCs as well as the sectors within each ARTCC is to distribute the load in an equitable way. As it is evident in Fig. 2.1, the high volume of flights in the densely populated east coast translates into a higher number of ARTCCs than the central United States.

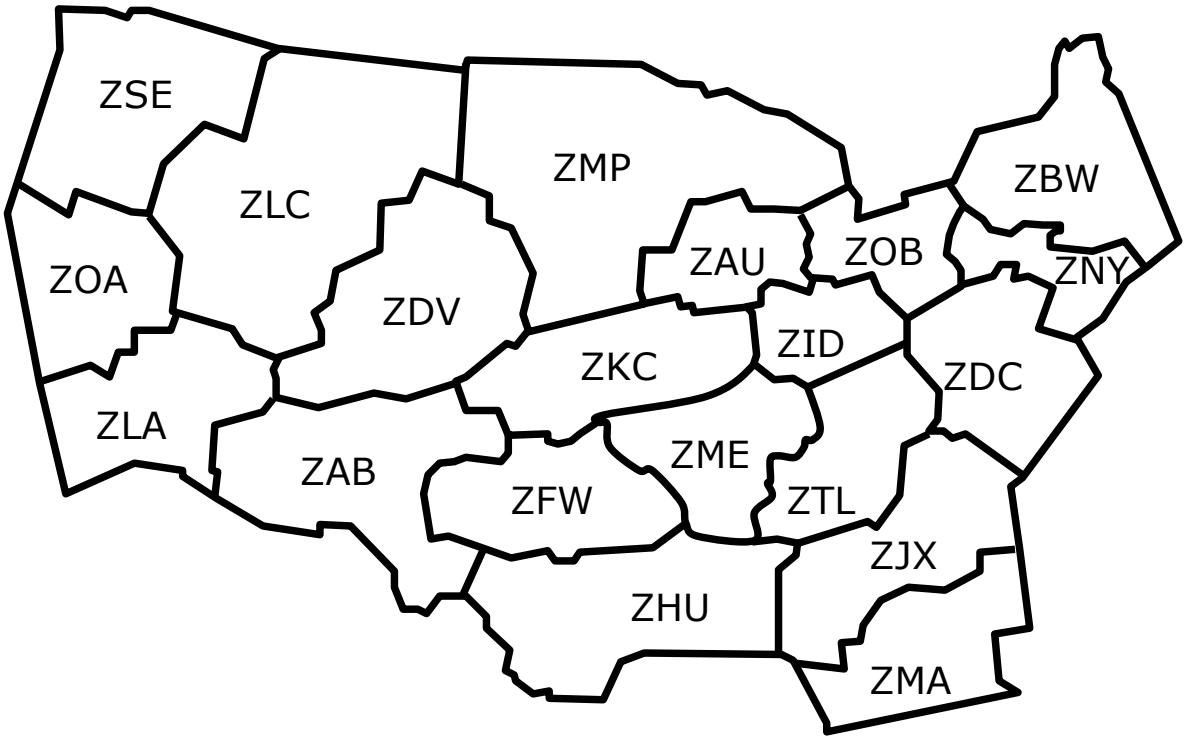


Figure 2.1: ARTCCs in the contiguous United States (recreated from FAA[45]). The zones cover the airspace above and slightly beyond the contiguous United States.

Traditionally, the main role of air traffic controllers was to keep a prescribed separation between all aircraft. However, within the next generation of ATC (NextGen) – a new system with the motivation to address the lack of scalability of the current system, pilots are more autonomous and as a result in charge of their own separation and controllers in-

tervene only when necessary. This is possible due to pilots being equipped with Automatic Dependent Surveillance-Broadcast (ADS-B) technology for navigation and localizing other aircraft in their proximity. ADS-B uses GPS for navigation and broadcasts aircraft position periodically. Use of ADS-B Out (broadcaster only with no receiver) within specific portions of airspace is mandated by 2020 [90]. Difficulty with aircraft localization has been a great problem in aviation, forcing most of the air traffic through certain preferred airways (analogous to the highways on the ground). However, use of ADS-B provides more efficient direct routing within NextGen which allows flying in a straight line from the departure to destination airport (also known as free flight) by providing better situational awareness regarding the congestion. Unlike Internet where if some part of a network exceeds its capacity, it conveniently drops new transferring packets, this is not possible in ATC. Therefore, all ARTCC's and sectors and airports must remain within their capacity which makes in advance reservation necessary. Flight plans are submitted to a central entity called Air Traffic Control System Command Center (ATCSCC) where according to predicted loads, a delay is assigned to each flight to ensure the network will not be oversubscribed. Pilots will receive partial or complete clearance. Once airborne, with the unfolding of how the actual flights progress, additional delays are assigned to the flights. The idea is to apply these delays as early as possible in the flight or before takeoff, rather than near the end where the maneuver space and fuel capacity are limited. These delays can be achieved by ground hold, lowering the cruising speed or by standard holding patterns. These assigned delays are communicated to the sector controllers so they know how long they must keep the aircraft in their designated sector. Interested readers are referred to [65] for a full treatment of air traffic control systems.

There are certain differences between IoD and ATC. As the number of drones scales up to the thousands sharing the limited airspace at any time, use of a centralized entity like ATCSCC for load prediction and assignment is difficult. A helpful approach to deal with this difficulty is to look into the solutions that are more decentralized in their nature. With that volume of flights, separation must be autonomously done by the drones and it is not wise to rely on human interventions for safety management, in contrast to NextGen. The limited airspace of the urban environment can only accommodate drones that have minimum performance requirement which, depending on the situation, can be stringent such as a requirement to execute holding patterns in a small area (ideally hover as in the case of Vertical Take Off and Landing (VTOL) aircraft) and ability to easily land when necessary. This opens up many possibilities within IoD for handling congestion which is not available to the ATC system. Free flight, although a step forward for ATC, is only partially implementable within IoD due to limited urban airspace, obstacles such as buildings and birds and high level of congestion anticipated. In other words, the airspace must be highly

regulated to ensure smooth air traffic flow is achieved.

2.2.2 Cellular Network

In the cellular network, the coverage area is partitioned into most commonly hexagonal cells forming a honeycomb pattern. The communication signals in each cell are sent to and received from the mobile users by a dedicated base station. Each base station uses a certain frequency which is different from the near base stations' frequencies to minimize the interference. The range of signal for each base station determines the size of each cell. Each base station can only carry a certain amount of calls over its frequency channel. As such, the main driver in determining the size of each cell is the expected number of mobile users in the region (Fig. 2.2). Hence the densely populated downtown areas can have many smaller cells whereas in the rural areas, fewer cells with higher range are used. Each of the base stations are connected to a central entity called Mobile Telecommunications Switching Office (MTSO). The MTSO is in charge of periodic localization of the mobile units and assigning a base station to them. Furthermore, it assigns channels to each call and performs the task of handoff or handover which is basically the transfer of responsibility for a moving mobile unit from one base station to the other base station as it enters a new partition. I will later use the same word in the context of IoD. See [74], [34], and [83] for a comprehensive treatment of the cellular network.

Compared to the best effort philosophy of the Internet, in telecommunication, the philosophy is that a call must not be admitted if there are not enough resources to sustain it until its completion. Hence, the handoff process poses a unique challenge as it is not known whether admitting a call in a cell will result in later termination as the mobile unit enters a new cell due to a lack of available channels in the new cell. Since the base stations usually belong to one corporation, the MTSO centrally makes decisions whether to allow access to a user in an effort to minimize the probability of a dropped call. As we will see, a similar problem exists for drones in IoD. It is much less expensive to hold a drone on the ground than to allow it to takeoff and later *ground* it (order it to land) or *hold* it (order to hover or execute holding patterns) due to a lack of resources. Hence, IoD has a design philosophy that is similar to that of cellular telecommunications networks.

There are still various differences between IoD and cellular networks in an abstract level. A subtle difference is that in the case of cellular network, the MTSO does not know which cell will be the next cell a mobile unit will enter after admitting the call in the first place. But in IoD the source and destination is known to a greater extent for a trip by the drones which will allow a more optimized utilization of the network resources. Another

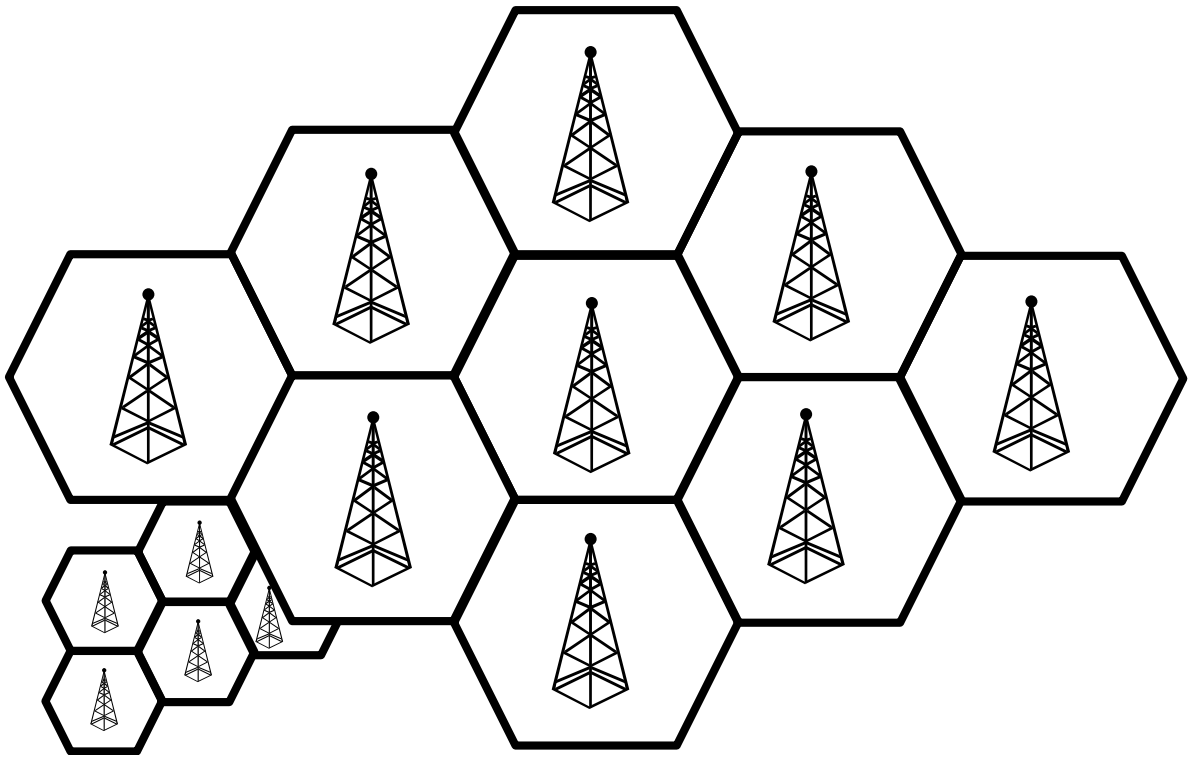


Figure 2.2: An illustration of the cellular networks and base stations.

difference is the central role MTSO plays which is in part possible because each company holds exclusive rights to certain bandwidths in the frequency spectrum. There are at least two reasons why a central design does not seem a good choice for IoD. Firstly, the tasks of IoD are computationally intensive. Hence we have to offload it to many autonomous systems which coordinate with each other. This way we reduce the complexity of the problem while settling for a less optimized solution. Secondly, as mentioned in the cellular network, a portion of the frequency spectrum is allocated exclusively to a corporation which means it has total control over its use. However, in my design for the IoD, each portion of the airspace must be shared by all the companies serving the same airspace and hence the amount of resources available to each company is less predictable. This means flight planning is a more involved task in a trade-off for a more efficient service provider market. I believe the exclusive right to the portions of the spectrum has made the entry of new competitors to the cellular market quite difficult, effectively resulting in a market with only a limited number of providers.

2.2.3 Internet

In the Internet, the goal is to connect networks of computers together, so all the computers on the world-wide network can communicate. The Internet has a layered architecture consisting of five layers as shown in Fig. 2.3. Layering makes it easier to solve the problem that the Internet addresses by separating concerns. Each of these layers is to be thought as a service and upper layers use the services of lower layers. For example, the link layer is concerned solely with the transfer of data on a single communication link or between two adjacent nodes and the physical layer is concerned with the physical means for transferring signals through various mediums, such as air (in case of WiFi) or Ethernet cables. The Internet layer, relying on the connectivity provided by the link layer is concerned mainly with routing or forwarding data packets between any two nodes potentially on two different local networks through the use of standard global addressing as a best effort service rather than a reliable one. This is achieved by routers which locally make a decision about forwarding the data packets they receive to one of the immediately connected networks. Utilizing the universal unreliable connectivity provided by Internet layer, the transport layer is concerned with tasks such as the reliability of transmission and congestion control. Finally the application layer, uses this global and (if needs be) reliable connectivity for various applications like Web, Email, Voice over IP (VoIP), Remote Login, etc. Such a decentralized and deliberately simple architecture has made the Internet a unique engineering feat in that it scaled by many orders of magnitude. Readers can refer to [70] for a comprehensive treatment of the subject of the Internet and to [10] and [11] for discussions of the philosophical guidelines in its design.

There are similarities and differences between the Internet and IoD. Routing is a task performed by both networks. However, the time scale on which the Internet operates is much smaller. In the case of IoD, the longer computation time can allow for the calculation of more optimal routes. Thus, a possibility is to adopt the routing protocols and adjust them accordingly. Another difference is that in the Internet, packets that overload the system can be conveniently dropped since it is buffered and resending it is cheap. In the IoD case, it is not possible to drop drones since they are physical objects and the only option is to remove them from the airspace by ordering them to land and providing resources to them to execute a landing order which is an expensive task. Thus some kind of reservation has to be enforced to ensure the system operates within its capacity to remain economical and viable. Whereas ATC is not a system that scales well, the Internet is designed and shown to scale well and with the expected proliferation of drones, IoD has to be an architecture that can scale. Using the Internet's design guidelines that has afforded it such scalability, such as a decentralized design or providing generic services with the

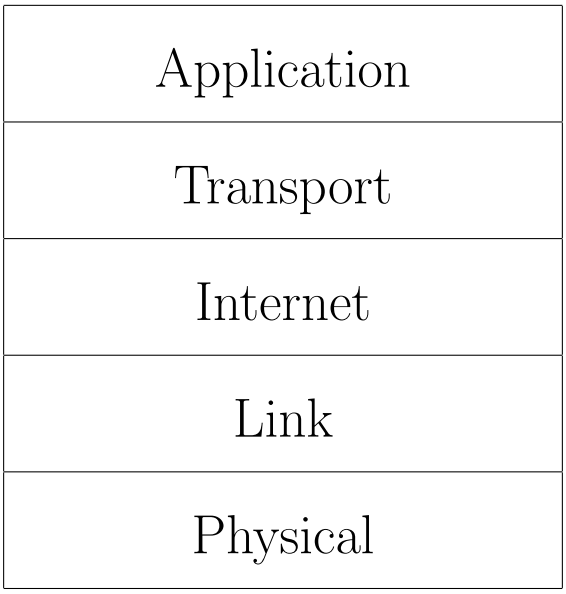


Figure 2.3: Layers in the architecture of the Internet

least amount of assumption about the users of the services is monumental in IoD.

2.3 Architecture

In this section, I explain my architecture in more detail. The purpose of my architecture is to provide extensible generic services to a diverse range of applications, namely navigation service between any two nodes in an efficient and coordinated manner as well as other common or future services such as location aware communication. A need for navigation is the common denominator for drone applications. Serving this need will enable these applications to build on top of the services provided by the architecture. Furthermore, drones are mobile yet tasks are local. In case a pool of worker drones rather than individual drones are responsible for performing these tasks, only the local drones (i.e. those near the task location) should be notified. Hence, providing a mechanism for location aware communication is another common need of the applications as well as other services for which the need will become apparent in the future. Two important concepts to distinguish in my work are that of an *IoD architecture* and an *IoD system*: an architecture gives abstract design and feature requirements that need to be implemented by any system that is based on that architecture whereas a system gives concrete protocols (interfaces

and algorithms) that implement the features required by the architecture. Hence, it is possible to have many IoD systems all based on the same core architecture each with their own advantages and disadvantages. Obviously in any engineering project, not all architectures are viable. Accordingly, at least one working IoD system must implement an IoD architecture to prove it is viable.

2.3.1 Structure

To describe my architecture, first I need to introduce a set of concepts and explain how they are related to each other in my architecture. Words with special meanings for my architecture are italicized and they will form a vocabulary for discussing it.

Airspace is the resource that is utilized by the *drones*. In my architecture, the airspace is structured similar to the roads network in the cities. Drones are only allowed inside the following three: *airways* playing a similar role to the roads, *intersections* formed by at least two airways, and *nodes* which are the points of interest reachable through an alternating sequence of airways and intersections. Each of these three has concrete geometric shape and is guaranteed to be collision free from static structures. Movement of drones inside the airways and intersections is regulated (for example drones must move only in the designated direction(s) of an airway or intersection) whereas inside the nodes, drones are in the *free flight* mode (Fig. 2.4). The airspace is partitioned into *zones* and hence each zone contains its airways, intersections, and nodes. Adjacent zones are reachable from each other through *inbound* and *outbound gates* which are the intersections at the border but they are special in that they belong to both zones. No airway is allowed to cross the border between two zones, unless it is segmented into two airways with a gate at the border joining the airways. The graph that is formed by treating both nodes and intersections (which include gates) as the vertices and airways as the directed edges is called the *zone graph* (Fig. 2.5). A path in the zone graph is called a *pathway*. I use the word *element* to refer to airways, intersections, and nodes. To be reachable, every element has a global address similar to how hosts have a global address on the Internet. If I take the gates as the vertices and connect co-zone gates with directed edges called *transits*, I call the resulting graph the *interzone graph*. Inside each zone, the cost of traveling between any pair of gates is called the *transit cost* where the cost can be time, distance, etc. (Fig 2.6). A path in this graph is called a *route*. For the zone graph, I use the word *progress* within an airway or intersection to state how far the drone has progressed the element according to some progress metric (e.g. distance from the beginning of an airway). In the zone or interzone level, the vertices and edges contain meta data e.g. in the form of *components* and *attributes* as in an eXtensible Markup Language (XML) tag which provide data about the particular vertex

or edge. Among the meta data is the minimum *performance* required from any drone that wishes to travel along the particular element, such as drone range limitations, landing restrictions, and other physical constraints. Meta data may also contain more detailed information about a particular element; for example, the meta data at a node representing a park can have a map of the park which a drone could use upon entry to the node. A portion of airspace is either *public* or *private*. All elements in public and private airspace are considered public and private respectively. For private elements, the *access* rules for drones is specified as meta data, such as which drones are allowed access to them. At the lowest level of abstraction, we deal with *points* in the airspace. The points are uniquely identified using the coordinate system of *(latitude, longitude, altitude)*. For instance, an airway's geometry is understood using points. A path through points is called a *trajectory*. Beware that I do not use the term trajectory in the same way it is used in robotics research where it means a time dependent path.

2.3.2 Components

My architecture comprises of two groups of components: *Zone Service Providers (ZSP)* and *drones*. All ZSPs and drones are connected to the cloud, so communication between any two components is possible.

1) In each zone, any of the ZSPs provides navigation information between any two elements in their designated zone to the requesting drones. The license to operate a specific zone is granted by higher authorities. They establish and enforce the governing laws regarding the airways, intersections and public nodes such as maximum allowed drone capacity or density in them. My architecture is not concerned with how ZSPs are realized, but it is worth mentioning that implementing a ZSP merely as software seems conceivable. I call an organization that offers ZSPs an Internet of Drones Service Provider (IoDSP). Adjacent ZSPs co-manage the gates and coordinate with each other on handoff; that is when a drone crosses the border and the responsibility has to be transferred to a new ZSP. Furthermore, ZSP can order a drone to land or hold its position by hovering or executing holding patterns and I call these actions *grounding* and *holding* respectively. Fig. 2.7 presents a schematic of the Greater Torontoa Area (GTA) in Canada together with the ZSPs deployed in the zones by four IoDSPs.

2) Drones in IoD are the autonomous aerial vehicles which are capable of collision free navigation along a planned route between two nodes and have various performance

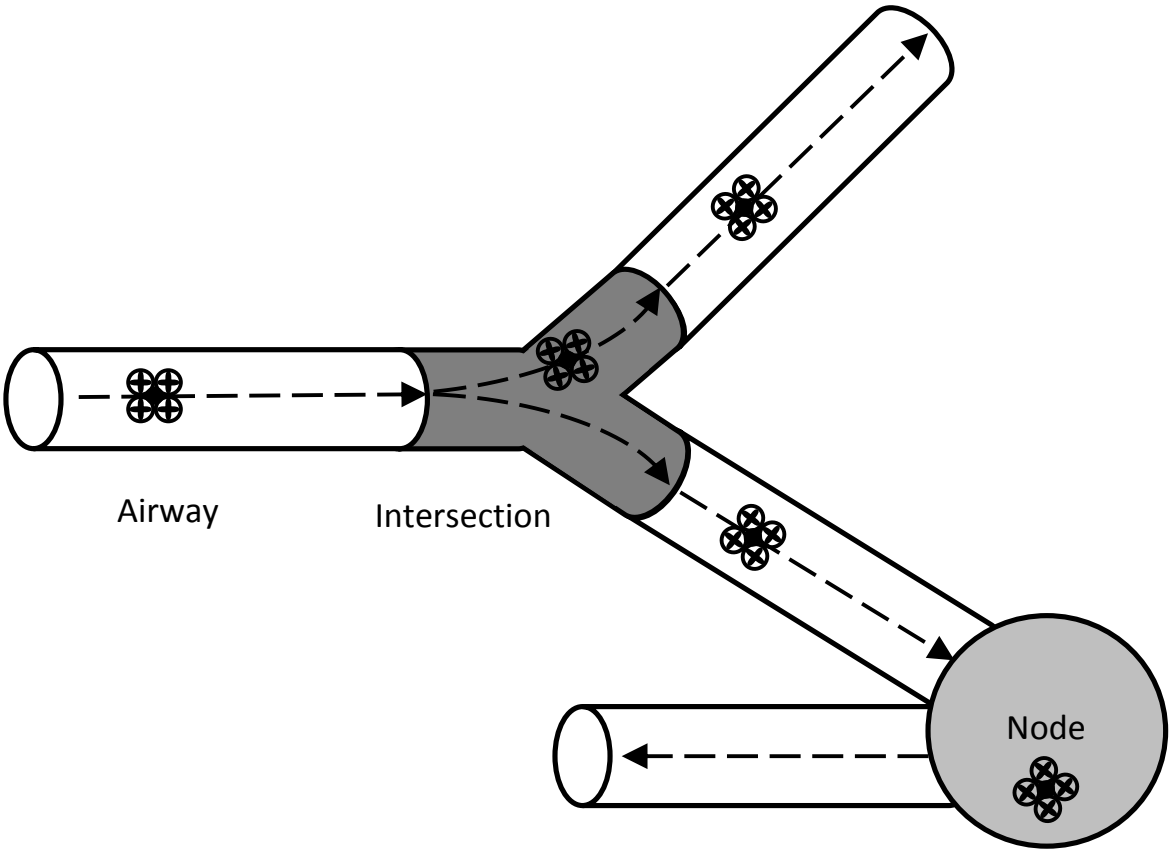


Figure 2.4: An illustration of the airways, intersections, and nodes

characteristics, such as their range, whether they are capable of VTOL and hovering, etc. They broadcast information about their position and their future path which will be used by all ZSPs, not only the particular one serving the drone. Regardless of how ZSPs and drones are implemented, they shall interact with each other through standard protocols. For instance, this allows that two competing firms have two different implementations for their ZSPs and still different drones with different implementations are able to communicate with the ZSPs through the standard protocols. Drones are required to assume fully autonomous operation beyond line of sight operation, be equipped with sense and avoid technology and be capable of emergency landing. Furthermore, specialized airworthiness certification must be considered to establish reliability levels for drones that are comparable to those of commercial aircraft when operating over inhabited areas.

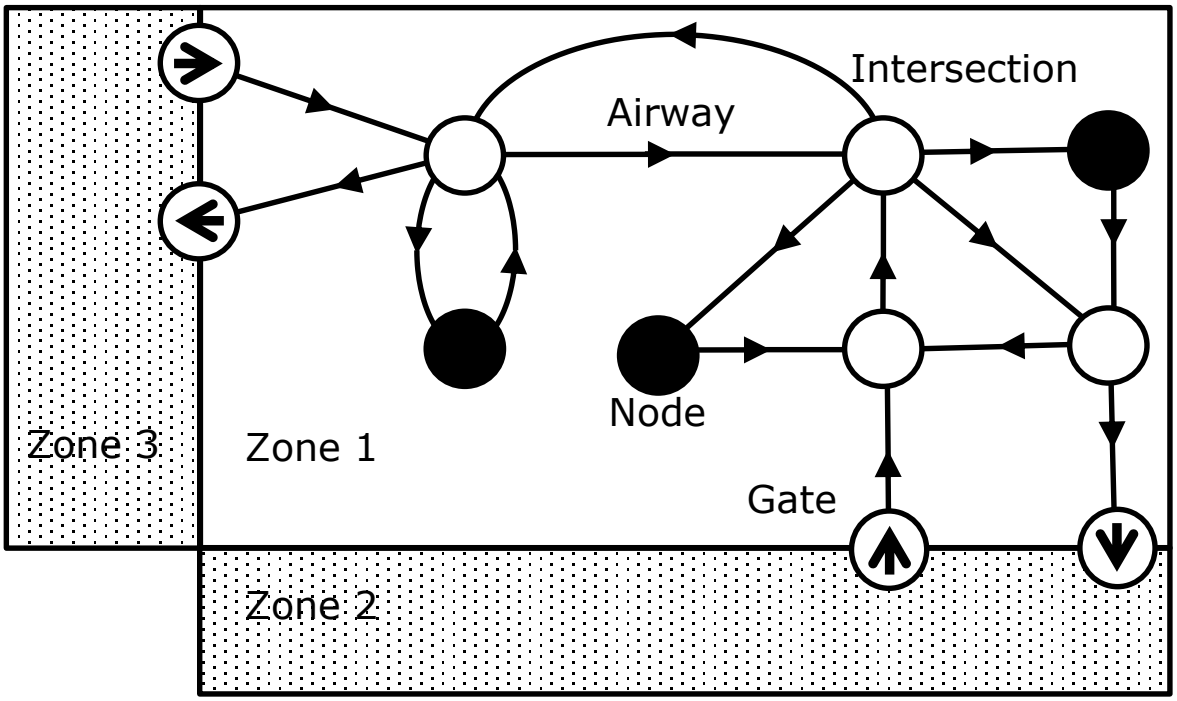


Figure 2.5: The zone graph for zone 1 is shown. Intersections, nodes, and gates are shown with circles and marked accordingly. They constitute the vertices. Airways are shown with arrows and they are the edges of the graph. Most likely, there are many gates between any two zones, but for simplicity I show only two.

2.3.3 Layers

Similar to the Internet, I propose a layered architecture for IoD. Layering provides many benefits such as the separation of concerns, scalability, maintainability of the code base, and flexibility of modifying a layer with minimal changes needed to the other layers. The fundamental goal that the architecture is concerned with is to enable drones to perform various applications by providing common generic services for all applications. Consequently, the architecture has two goals. Firstly, it is to provide guidance to a drone from a source node to a target node and coordinate all drones' access to the airspace as a service to the drone. Secondly, it is to make available an extensible platform for other common current or future services that are needed by applications such as delivery of messages that are intended for a pool of worker drones for an application in a specific zone (an example message is a list of local task requests).

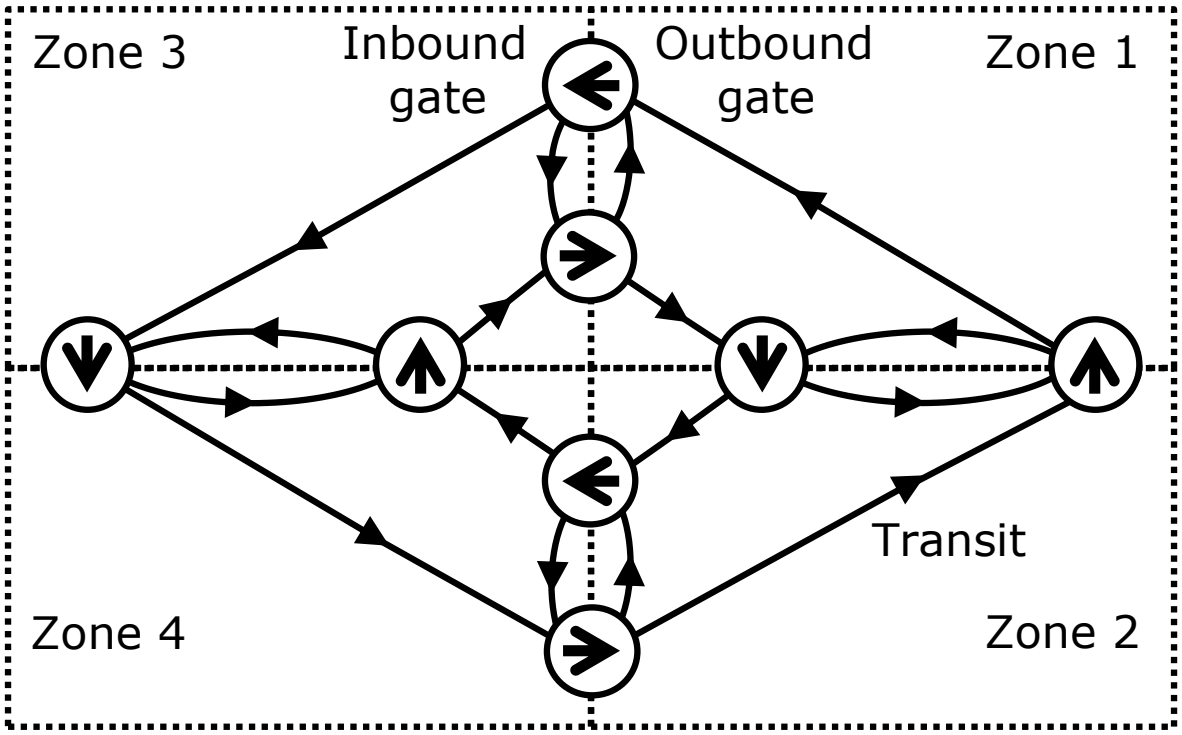


Figure 2.6: The interzone graph for the zones 1-4 is shown. Gates are the vertices of the graph. Transits as edges are representatives of the possibility of a trip from the inbound gate to the outbound gate for the drones. Transit cost can be any cost function associated with the trip between two gates, such as average trip time. Between any two zones, there can be many gates, but for the sake of simplicity I show only two.

The navigation can be reduced to three sub-tasks. Firstly, the drone will have to traverse a path on the interzone graph from the source zone to the destination zone. Secondly, to traverse within each zone, the drone must traverse a path on the airways and intersections of the zone graph. Lastly, a trajectory of points must be chosen which the drone has to follow to stay inside the boundaries of the airways, intersections, and nodes. I tackle each of these tasks in a separate layer. The reason this seems to be a good way of tackling navigation is that having a single giant system with its map and airspace access mechanism is computationally complex and unsustainable, if not impossible. By dividing the problem into smaller sub-problems, each of them becomes more tractable. Therefore, we trade a more optimal solution for a more tractable solution.

As mentioned, there is more needed than just navigation. For example, for a package

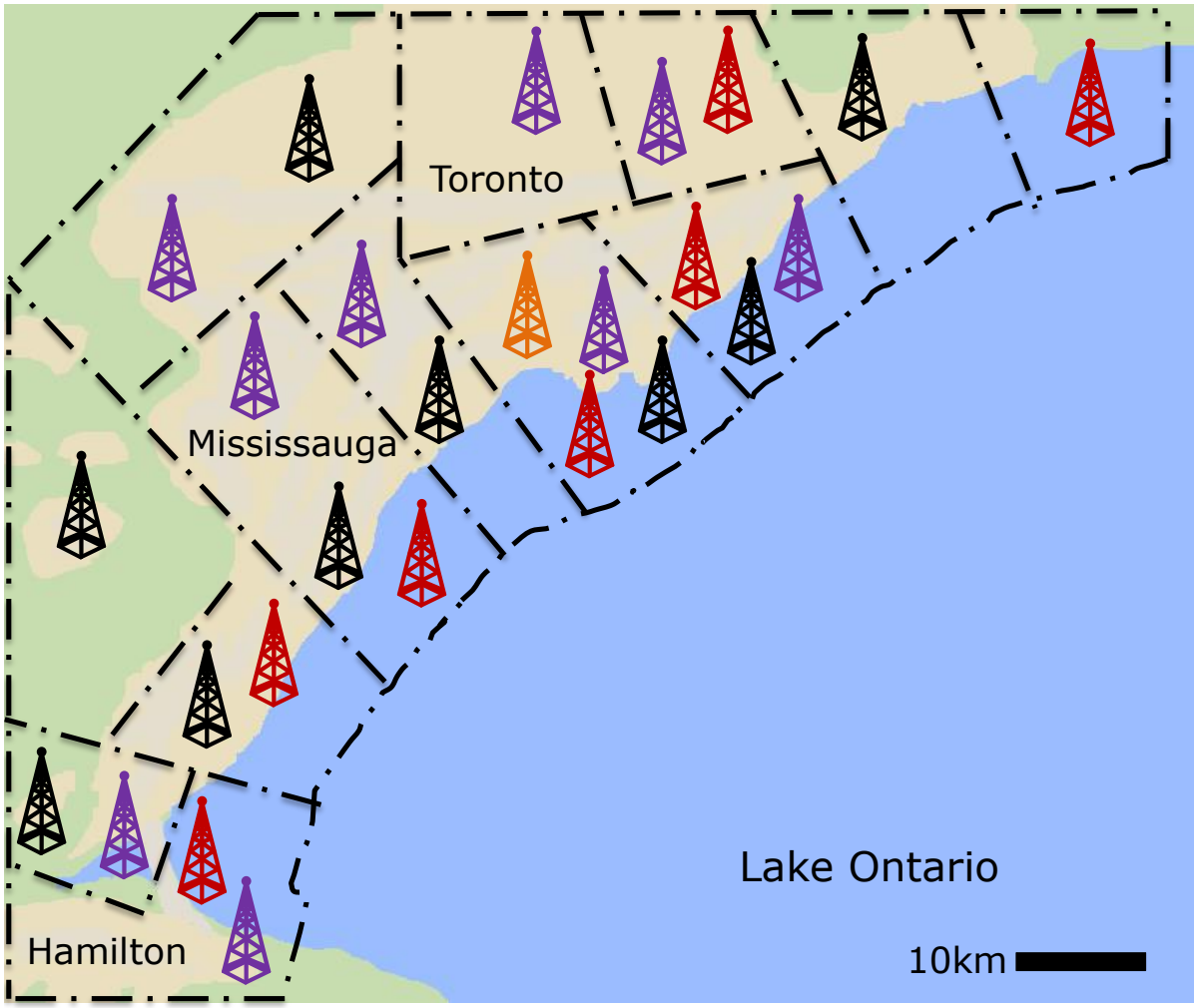


Figure 2.7: A schematic of the Greater Toronto Area and the zones served by the ZSPs deployed by four different IoDSPs each colored differently. Handoff occurs at the boundaries of the zones.

delivery task requested by a grocery store inside some zone, only the drones that are near the store (say in the same zone) should be notified, not all the drones in the realm of IoD. Hence, ZSPs must meet these zone-specific demands through a service layer that is used by all applications. The service layer is extensible to meet the needs for future services as they will become apparent by the common needs of applications.

My architecture consists of five layers as shown in Fig. 2.1. Drones have functions that

Application
Service
End to End (E2E)
Node to Node (N2N)
Airspace

Table 2.1: Layers in the architecture of IoD

fall in all the layers while ZSPs only have functions that fall under the airspace layer up to the service layer. In a strictly layered architecture, each layer provides services that are used by the layer directly above it. As is the case with the Internet (see [70, pp. xvi,xx,xxi,33-36,87,147]), my architecture is a relaxed layered architecture where upper layers can access lower layers and not just the layers directly below them. In effect, layering provides an effective way for logical organization of the architecture and its easy communication to other engineers and should not be treated as a never to be broken rule. The lower layers are not aware of the specifications of the higher layers. The interactions between the layers shall be through standard interfaces. The protocols then are defined between the same layers of two components.

I describe each layer in terms of the *features* it is required to implement to comply with my architecture. This means that any IoD system must implement those features and define specific protocols and interfaces that make access to those features possible. I use capital letters as my convention for the name of the features.

Airspace Layer

The airspace layer is required to implement the following features along with the needed protocols and interfaces for using these features.

MAP: ZSP is required to hold geometric representation of the elements in the zone graph; i.e. the airways, intersections, and nodes.

AIRSPACE BROADCAST AND TRACK: Drones have to broadcast periodically their three dimensional coordinates and their future trajectories. It is conceivable these data are needed for path planning in this layer and indirectly in other layers for calculating the progress.

PLAN TRAJECTORY: ZSP has to provide trajectories to be followed by the drone, so it stays inside the boundaries of airways, intersections, and nodes of the planned pathway.

AIRSPACE PRECISE CONTROL: I envision a possible need for ZSP to request specific maneuvers from a drone such as holding, moving to a new point, or landing at a point. This seems to be a reasonable feature to expect from a universal architecture.

COLLISION AVOIDANCE: In case of dynamic objects such as other drones or birds obstructing the airways or intersections, the drone must avoid colliding with them by overruling the trajectory. The drone must communicate with other drones in proximity through standard protocols for coordinated maneuvers for avoiding collision.

WEATHER CONDITION: ZSP must provide the drones with the weather conditions such as wind speed and temperature, so drones can successfully take these data into account at the time of executing a trajectory.

Node to Node layer

The features required for the node to node layer is as follows.

ZONE GRAPH: ZSP keeps an up to date zone graph that is augmented with the information broadcast from all the drones such as the current airway, intersection, or node of the drones and their future paths as well as their progress within an airway or intersection. ZSP knows how many drones are inside an element and roughly how they are spaced out in an airway or intersection. In the zone graph, the meta data for elements are stored too, such as the minimum performance requirement which is also a function of the weather report and changes in time. Furthermore, ZSP must provide protocols for obtain-

ing the information in the zone graph (e.g. for viewing). Also, it must provide protocols for updating the map, such as identifying certain airways, intersection, or even the complete zone as no fly areas. Also, ZSP must provide protocols for integrating weather reports.

N2N BROADCAST AND TRACK: Drones are required to broadcast their current element, their progress within it in case of airways or intersections, and their future path, and their estimated fuel time left periodically in a way that is accessible to all ZSPs.

PLAN PATHWAY AND CONTINGENCY: A path on the zone graph must be provided by ZSP to a drone that requests a path between any two elements as source and (intermediate) destination in the same zone. The path consists of a sequence of airways and intersections that have to be navigated for the drone to travel from the source to destination. The path does not have to be complete and a partial path for getting closer to the destination is also acceptable. A contingency path must also be provided for example to landing sites which will be used in case the drone cannot continue on its path, such as unexpected fuel shortage or when grounding by ZSP is necessary. ZSP has to take into account the performance characteristics of the drones among other things when allocating a pathway to a drone by verifying the drone meets the minimum performance rating for the paths. Also, various meta data for each element in the path can be disclosed to the drones such as the weather forecasts.

REFUEL: A path to a fuel station node (fuel station can be third party depending on the preference of the drone) should be provided by ZSP to a drone that needs to refuel. ZSP must direct the drones to fuel station nodes that are compatible with them. For example, drones can run on electricity, gas or even hydrogen (in case of fuel cell). When a drone asks for refuelling, ZSP will give a pathway to the proper fuel station accordingly.

N2N PRECISE CONTROL: It must be possible for ZSP to command the drone to hold or to move to an element, or to land at a node.

EMERGENCY: When a drone faces a software or hardware failure, if it is capable enough, it has to broadcast an SOS message to ZSP which must make arrangements such as broadcasting relevant information to all the drones, so other drones change their pathway or hold in their current element. Furthermore, ZSP must detect when a drone abruptly stops broadcasting message and issue the emergency procedures.

CONGESTION NOTIFICATION: Upon request, ZSP must provide congestion report between any two elements inside the zone.

End to End layer

The end to end layer must implement the following features.

INTERZONE GRAPH: ZSP must store the partial interzone graph at the very least. That is, it must have information at least about the the gates and transit costs in its zone (the other end of spectrum is to have complete knowledge about the interzone graph). This gives a partial or local knowledge of the interzone graph which must be learned through different means such as interaction with other zones, or input from administrator. The graph is augmented with the data broadcast from drones, so it is known which drones are inside the zone and of them which are inside the gates and which are transiting between gates. A protocol must be implemented for obtaining the data stored in the interzone graph (e.g. for viewing the graph).

ROUTING: Any two adjacent zones are likely to have several gates connecting them. The ZSPs have to provide drones with one next intermediate gate. The transit cost can be used to provide a shorter route.

HANDOFF: Drones must be able to switch to the new ZSP when entering a new adjacent zone. ZSP must be able to handle the incoming and outgoing drones.

EXPLICIT CONGESTION NOTIFICATION: ZSP has to give explicit congestion notification on any of its gates and transits to at least the ZSPs in the adjacent zones. The algorithm to determine a gate or a transit is congested is up to the implementation by the particular ZSP.

Service layer

The service layer is an extensible layer that currently has the following mandatory feature and can be extended to add more services in the future as needs arise.

ZONE BROADCAST : The main role of the service layer is to provide a common platform where zone-related messages can be broadcast to the drones. For instance, a task request that needs to be performed by a drone in a particular zone can be broadcast to all the

drones in that zone through service layer in ZSP. A particular task can be grocery pick up in a zone. Through encapsulation, the service layer does not understand the content of the message. However, applications by relying on the service layer for receiving the message will make sense of it.

Application layer

There is no feature requirement for the application layer. These are the applications that will be written in the future to use the architecture. The point of having a general airspace navigation and control service along with other services as is provided by the four layers of airspace, N2N, E2E and service is that many application I can conceive of will use these services as a foundation. So by providing it once, I enable the whole range of applications simultaneously, rather than providing a dedicated service to each application.

2.3.4 Cross-cutting features

Any feature listed here cannot be addressed by one single layer and needs to be implemented in several layers.

SECURITY: There are a variety of threats that must be safeguarded against, among them are authentication of drones and ZSPs and other components outside the IoD system, jamming of the broadcast messages, clogging the airspace, and hacking of the drones or ZSPs.

2.4 Operation model

My architecture can lend itself to various operation models. I discuss one seemingly reasonable model here and in remainder of the chapter I assume that we have adopted this model.

2.4.1 Model

Public is the owner of the most of airspace. There are two groups of drone owners. The first group are companies operating fleets of drones and offering various services such as logistics

to users. The second group are individuals with their private drones. Since airspace is a public space, all drones are required to be registered with the government for a license to operate. Interestingly, at the time my coauthors and I were writing the paper related to this chapter (December 16, 2015), FAA published an interim final rule (for a definition, see [67]) that mandates owners of drones with a weight between 250 grams to 25 kg to register it with the U.S. Department of Transportation[25].

The map of where zones are located and the public airways, intersections, and nodes inside each zone is created by the municipalities in consultation with FAA as it is the ultimate aviation authority. Drone operations must be confined to inside of these elements and this must be enforced by the police. Furthermore, unauthorized entry of any drone to the private airways, intersections, and nodes is considered trespassing. Areas that do not fall into any zones are considered unregulated.

Private airspace can be defined in various ways. For example, it can be the airspace directly above a private property and below some elevation level. The municipalities set the boundaries of the private airspaces. As noted before, within private airspace, private elements are located. The owner of the private airspace, if inclined, has to design his/her own map of these elements, according to the constraints set by municipalities. The map is submitted to the municipality for the purpose of integration with the city's map along with consents for releasing the map to one, two or even all IoDSPs. Therefore, one possibility is that a private node be served exclusively by a single IoDSP, a model similar to how a host is connected to the Internet using only one ISP. At the same time any drone company can serve the node so that all the drones are potentially available to the customer resulting in faster service time. Alternatively, all IoDSPs could provide the same services to every nodes and differentiate themselves through better implementations of protocols.

The non-exclusive license for IoDSPs to provide their services within each zone is granted by the municipalities. Airways, intersections, and nodes have to be used according to the policies set by the municipalities such as the maximum drone capacity or density. IoDSPs are obliged to provide service to all drones without discrimination (For example, an IoDSP cannot deny service to a drone in retaliation to the drone using a different IoDSP in the previous zone). These policies must be enforced by the municipalities and the police. More than one IoDSP can operate within the same zone. There is no lower or upper limit on the number of zones within which a company can operate. Any IoDSP can serve any node, as long as the private owner of the airspace has pre-authorized its access to the map through municipalities as mentioned above. And finally, during handoff (i.e. when the drone enters a new zone and the responsibility must be transferred to a new ZSP), drones can choose any ZSP in the new zone.

2.4.2 Interactions with outside

The interactions with outside are orthogonal to my architecture. To decide what entities will use the IoD system and what protocols will be used is a design choice to be made at the time of implementation. However I mention some of the entities that in all likelihood will interact with the system to give a real world picture of how an IoD system might operate.

An example IoD system implements protocols between the ZSPs and US National Weather Services (NWS) to disable and enable parts of the network in an automated way. The US Federal Aviation Administration (FAA) might declare a no-fly zone which is communicated through another protocol to the ZSPs. Other important entities are possibly third party fuel stations. There can be well-defined protocols for negotiating between drones and third party fuel stations (if the drone opts for using them) with the possible role of ZSPs for brokering the messages. Also, for direct messages between a fuel station and drone at the time of docking, Machine to Machine (M2M) protocols can be used due to the low latency that is required for the task. Since both drones and ZSPs are connected to the cloud, users, companies owning the drones, administrators, retailers like grocery stores, etc., can communicate with them through standard protocols like http (Fig. 2.8).

2.4.3 Strategies for deployment

A particularly attractive deployment strategy is the use of already deployed cellular networks. As explained earlier, in the cellular network, each provider partitions its coverage area into cells and places base stations in each of the zones. Since these base stations are already deployed, the physical space is available and they are capable of running the ZSP software. Therefore, they seem well positioned to implement ZSPs and provide wide network coverage for IoD. This strategy becomes even more interesting considering that drones have to use mobile communication which is basically what the base stations provide. This means that ZSPs will provide not only navigation, but the main communication channel for the drones. Since base stations are connected to the cloud, ZSPs can communicate with each other or with other outside entities over the cloud.

2.5 Discussion and Future Work

2.5.1 Goals, principles, and benefits of my design

In my design of the architecture, I have encouraged principles of openness, modularity, and interoperability. To achieve this, I require drones to broadcast their information using standard open protocols to communicate with ZSPs or other drones. Similarly I require interaction between all ZSPs through standard protocols. I believe it is not common or reasonable to expect competing ZSPs share traffic information and other statistics. Hence, by requiring drones to broadcast their current position and future path, I give all ZSPs in the same zone a chance to manage the traffic and have the big picture of the zone.

One immediate benefit of openness, modularity, and interoperability is the lower overall cost for creating the navigation network. Similarly, there will be lower initial investment and lower complexity for the new IoDSPs or drone companies to enter the market. This leads to the organic growth of the network by lowering the barriers to entry and both new and existing firms will benefit from the network effect. A key consequence of these principles is that through innovation, companies will compete in their implementations while they coordinate on the standards. In my architecture, I have tried to require a minimal set of functionalities from IoD systems. My intention is that this leaves the door open for introduction of innovative protocols and algorithms rather than the ones forced by us.

One important aspect in my design is scalability and survivability. From the experience of the Internet, survivability is a prerequisite for scalability. In an expansive IoD system, failures will be commonplace and the IoD system must gracefully survive them. The IoD architecture comprises of many autonomous subsystems that interact with each other only locally. This makes it possible to contain failures as opposed to have them ripple through the entire system and make it unstable. For example, the design of my architecture promotes that only a small portion of a drone's trip be reserved at any time by a local ZSP, since no ZSP has the authority to reserve a path beyond its zone. On the contrary, if the entire trip was reserved and for some reason the drone could not meet its reservation, this would affect the whole system. Furthermore, by relying on autonomous subsystems, the complexity becomes manageable as the size of computational problems that need to be solved will be substantially smaller.

An important goal in my design is to provide generic services which can serve many diverse applications. Furthermore, applications that are not even conceived today are more likely to build on top of generic services than highly specialized ones.

The concept of collision free network of airways and intersections let us circumvent the high cost of 3D mapping of the terrain and the buildings in a city. Basically, instead of guaranteeing a general statement that every possible trajectory in an area is free of obstacles (or even worse not promise that but require drones to avoid it as the only safety measure), we guarantee one example trajectory in the area that is free of obstacles (similar to the road networks). This is a substantially easier task, as mathematicians can attest to when proving a theorem versus providing an example for which the theorem holds. In addition, this enables a higher control over where drones can and cannot be (such as near airports) which is important for safety and security, and noise control (such as near hospitals or residential units). Law enforcement will be possible when drones only operate through predicted routes, as a course violation (such as trespassing in the private airspace) will be easily detected. Furthermore, this results in a more predictable traffic model which provides more organized data for planning, traffic management, and scheduling. It is worth mentioning that in ATC, a standard model for avoiding airborne collisions is the vertical separation of traffic according to direction of flights. In IoD, the model is that each airway and intersection has a specific direction of flight which is meant to achieve the same purpose.

Due to the extreme scarcity of urban airspace and safety critical nature of drone operations, I believe it is necessary to highly regulate its use through a model such as collision free network of airways and intersections as alluded to above. As such, notions like free flight as described in the context of NextGen are unlikely to be practical at least in the urban environment. At the same time I do not take the freedom away completely as free flight is possible inside the nodes by default with the extra flexibility for having collision free maps and other information if needed as the meta-data for the nodes. Nodes can have any geometries which means they have no restriction on the size. Once the drone enters a node, the ZSP no longer provides a specific trajectory as in the case of airways or intersections. ZSP will only provide the meta data about that particular node to the drone. Hence a farming drone is in the free flight mode inside a node representing a farm. If on the other hand, ZSP's help with navigation is needed for a node representing a large national park, the node must be divided into multiple nodes each connected through a network of airways and intersections, but again inside the newly formed nodes, free flight is the mode of operation.

Each drone in an IoD system will be capable of performing one or more applications. In the Internet a user demanding a service accesses a specific host on the network and interacts with a specific application through unique IP and port numbers. However, in the IoD the dominant model is not to make requests to a specific drone directly, rather a pool of drones be ready to accept these tasks broadcast by the ZSP's service layer. This

is analogous to the position of the information-centric networking line of research for the design of the future Internet (see [2] and [94] for recent surveys). The position is that users are mainly interested in the information rather than the host to host connectivity. In this work, I advocate that users are mainly interested in a service such as package delivery or power lines inspection, not the particular drone that performs it or the particular path within which the drone travels to perform the task.

2.5.2 Routing

A fundamental question about how we implement routing is whether we reserve the entire path before the start of the trip such as e.g. in Asynchronous Transfer Mode (ATM) network technologies (see [68] for instance) or metaphorically we start moving while asking (from the ZSPs) for direction. The latter is the approach the Internet takes and I believe this is the superior approach for IoD systems. The main reason is that since drones take a long time to complete their trips, reserving the entire path for them is a wasteful use of the airspace as precise prediction of the future position of a drone in a complex system like IoD is not possible. As a result, in the IoD systems that I advocate, there is no guarantee a drone will complete its trip without being occasionally *grounded* by ZSPs a few times along its path, due to a lack of enough airspace. However, grounding a drone is expensive in terms of energy consumption, travel delay and waste of airspace. The situation is somewhat similar to the cellular network. In both, it takes a long time for a cellphone user or a drone to enter a new cell or zone, respectively. The idea in the cellular network is that it is best to not admit a call, if it has to be dropped later. A similar policy in an IoD system is useful as it is more expensive to ground a drone than to not let it get airborne in the first place. This subject is studied extensively in the area of Call Admission Control (CAC) (See [29] for a survey). However, there are three major differences:

1) In the cellular network, the scheme needed for reserving resources is simpler. Basically one has to ensure that future cells have enough capacity to admit the mobile unit. However, in the case of reserving the zone-graph's elements for the drone, there is more than one way a drone can travel between any two nodes and hence there is more complexity in deciding whether enough resources are set aside for a particular drone or not.

2) On the other hand, in the IoD setting, if nothing unexpected happens, the path a drone will take can be partially or completely known (depending on the particular implementation of IoD system) whereas in the cellular network it is often not known

to which adjacent cell the mobile unit will enter next. Hence, in the IoD, there is less uncertainty over the path.

3) The CAC decisions are made centrally, partially motivated by the fact that in the cellular network, the adjacent cells mostly belong to the same company. However, Distributed Call Admission Control (DCAC) is a possibility, as shown by the seminal papers [56] and [60], where reservations must be made not only in the current cell of a mobile user, but also to a lesser extent in the neighbouring cells and the cells beyond to accommodate the mobile unit as it enters the new cells. If such a reservation is deemed possible after the base stations communicated with each other, then the call will be admitted. Referred to as the shadow clusters concept, it is similar to a quantum wave function which maps the probability of finding an electron in any region in the space where electron is analogous to the mobile unit.

It is conceivable that a similar idea for IoD inspired by DCAC can provide a routing algorithm that grounds very few drones while utilizing the airspace in an efficient way, by not reserving the entire path from the source node to the destination node for them. Otherwise, prediction errors will ripple through the whole system and make it unstable [10]. As noted above, the routing task will be harder in IoD because of the complex structure of the resources, but the lower uncertainty over the drone's path can be useful. Developing such a routing algorithm is an important contribution to IoD.

2.5.3 Congestion control

With the possibility of thousands of drones at flight at any point in time in an urban environment, a main purpose of the IoD architecture is to coordinate access to the airspace. It is instructive to first discuss how the congestion control in the Internet works. The goal in the Internet is to ensure efficient and fair use of bandwidth. There is no central mechanism that in the short run allocates bandwidth to each of the hosts, i.e. the end nodes which are the users of the network. Rather, hosts allocate a fair and efficient amount of bandwidth to themselves in a participatory fashion. They do this by probing the network and refraining to add more loads to it if they realize the network is in a congested state. This is done by analyzing the amount of time it takes for the delivery Acknowledgement (ACK) to be received by the sender (if ever in case of a dropped packet). To probe the network in a decentralized way, the network is driven toward congestion which creates delayed or lost packets which results in delayed or unsent ACKs respectively. From this, the sender realizes it must slow down in sending more packet until the network becomes less congested. This

is an implicit way of inferring congestion. Today, some of the routers in the middle of the Internet are capable of sending Explicit Congestion Notification (ECN) [73] by looking at the number of packets that they have in the queue that are not yet sent. This is a helpful feature, because running a network in a congested mode is not efficient; something that we have to do when the network does not provide feedback, just to be able to implicitly infer congestion.

The congestion status must be known in IoD within each of ZSPs for two reasons. First, running a congested airspace translates into grounding or holding which are both expensive operations. Second, in the Internet, the ACKs happen on the orders of few hundreds of milliseconds. This fast feedback loop allows implicit congestion probing as a viable option. In IoD, probing directly with drones (i.e. by seeing if drones get stuck in the congestion or not) is orders of magnitude slower. Because of especially high cost of congestion for the IoD, I believe I have to require a feature in the IoD architecture for explicit congestion notifications to at least the neighbouring zones in the E2E level. This is not needed in the N2N level, as any ZSP has complete knowledge of the congestion on all the airways, intersections, and nodes due to the broadcasts from the drones. A major difference with ATC is that there is no central controller for the whole network (ATCSCC) that regulates the load on the whole network while each ARTCC only ensures separation, which would have a negative effect on scalability.

In the design of a congestion control algorithm, it is an open research question how to achieve a fair and efficient allocation of the airspace while not overloading any of the elements. A mechanism that exists in the Internet literature to avoid overloading a link is a token bucket scheme (for example see [80]) in which tokens simply represent resources and each party is given a token, only if there is a token left. However, it is not clear how such a mechanism would work for an IoD system as there are more than one ZSPs which can grant access to the same element, and being competitors, it is reasonable to assume they will not share information.

An IoD system must achieve fairness in allocation of the airspace. However, fairness is a subjective term and can lead to different designs depending on how the fairness is defined. Should we give more priority to the faster drones at the expense of slower ones, since they use the airspace for a shorter period or should we allocate the airspace to each drone in an equitable way? In IoD, similar to the Internet, related to the question of fairness is a design that takes into account the Quality of Service (QoS), i.e. the network performance according to various metrics. The interesting fact is that not all the applications have the same needs. For example, a drone that surveys the traffic has to stay aloft for extended time where short interruptions are not necessarily important whereas a drone that delivers a package needs the airspace for a short period of time and has to minimize its delivery

time to meet customers' demands.

2.5.4 Communication signalling

Since drones are wireless and ZSPs have to broadcast, there will be a high amount of communication signalling which can flood the allocated frequency channels. IoD protocols must be designed with respect to the channel capacities as well as the number of drones and ZSP that will use the channel. If a high signalling overhead is inevitable for the functioning of IoD, then communication channels must also be treated as a resource similar to how airways and intersection are treated. Therefore, for the purpose of reserving the airspace for the drone, communication channels should be reserved as well and if any of these resources are not available a reservation should be deemed not possible.

2.5.5 Addressing schemes

Similar to the zone graph elements, drones are in need of global addressing. Whereas airways, intersections, and nodes as well as ZSPs are stationary, drones are mobile. Hierarchical addressing schemes similar to telephone numbers or IP addresses can prove useful for the zone graph elements or ZSPs. However, a particular shortcoming of the current Internet is that when IP was designed, it was assumed that it will work with stationary units. However, with the proliferation of mobile devices, that assumption is no longer valid. It seemed reasonable at the time the Internet was designed to have IP address serve two purposes; i.e. identification and localization. Identification is achieved by requiring every host to have a unique IP address. Localization is achieved by separating the IP addresses into a network portion and a host portion where each network can be part of a larger network; an idea referred to as subnetting. This design choice results in poor performance when the hosts are mobile[70]. Therefore, any addressing scheme for drones should perhaps separate these two functions in some form as is the case with most solutions to mobility on the Internet including Mobile IP and IPv6. A particularly interesting choice would be geographical addressing[81] where each drone is assigned an evolving address according to its current geographical position. For instance, this can provide a finer control over which drones to dispatch for a local task in a zone.

2.5.6 Drones and minimum performance

In an IoD system, an important ability that might be mandated by authorities in high traffic areas such as lower altitude in the urban airspace is the VTOL ability which enables easier grounding or holding by hovering. This can mean that most of the urban airways, intersections, and nodes in the lower altitude may require VTOL whereas in higher urban altitude it may not be required. This is because VTOL drones are highly versatile and can perform tasks in an environment with very little airspace available to them. Most commercial aircraft are each equipped with on-board systems like Traffic Alert and Collision Avoidance System (TCAS) which are designed to decrease the chance of mid-air collision (see [19] for instance). It is a complex system and just to avoid collision between two aircraft, thousands of lines of code are needed. In my case of lower altitude urban airspace, it is reasonable to assume that in often congested area with thousands of drones in flight, to avoid mid-air-collision, aircraft must be able to hover and move vertically to regulate the traffic, similar to the road network and cars which can stop. Drones are ultimately responsible for avoiding collisions mid-air and a TCAS like system for drones without hovering abilities is a major challenge for more than two drones.

2.5.7 Security

Security is not a topic that can be addressed by any single layer. A major challenge in the Internet today is that security is mostly provided by the application layer and there is a lack of in-place security mechanism in the lower layers. The Internet has been exploited for its security vulnerabilities which have led some network researchers to consider the security as one of the main goals in the next architectures for the Internet [28, 78, 9]. Arguably, damages from malicious users are more severe in the case of IoD compared to the Internet and security must be one of the core issues that any architecture for IoD should address. Given the experience from the Internet, I required in my architecture that security be implemented across all the layers, as it is a cross-cutting concern.

2.5.8 Validation and technical implementation

In this chapter, I present a conceptual architecture and an important technical contribution is to instantiate at least one system based on it to validate and demonstrate that my architecture can work in practice. This entails designing protocol suites and interfaces between the layers and implementing the layers with the required features. Any inconsistency or

inefficiency revealed at the time of implementation can be used for later iterations of the architecture. To implement an IoD system there are many non-programming questions that have to be answered, such as the questions discussed about routing and congestion control. This will be the main area of my focus in the future works. Building IoD is a great undertaking which needs the participation of the research community at large. By presenting the architecture in the current stage, useful protocols can be discussed and designed by the research community which can be validated once a simulation as well as a physical platform for IoD is ready. Furthermore, the design of the IoD architecture itself can benefit from the work of the researchers working on diverse range of networks from air traffic control to cellular to the Internet who will apply their knowledge to IoD.

We made some strides toward the validation of IoD by implementing a simulator for various components in IoD. This included simulating the drones, ZSPs, the zone and interzone graphs, and some preliminary simulation of the airspace, N2N, and E2E layer functions. We used a micro-services architecture together with a message broker agent for these components as well as a database (stored in memory for fast access) to ease the transition into a cloud based and decentralized design.

2.5.9 Economics of IoD

From an economic point of view, the operation model and the protocols of the system must provide enough incentives to the stakeholders to pursue the desired actions. It is interesting to study related questions through the lens of game theory and mechanism design.

2.5.10 Legislation

Another major topic is to provide a legal framework for the IoD. One of the main barriers to utilizing the drones today is the lack of legislation that properly address the technology. This is manifested in the recent Public Law 112-95 titled "FAA modernization and reform act of 2012"[66] enacted by US House of Representatives. In the Public Law 112-95, the secretary of Transportation is mandated among other things firstly to develop a comprehensive plan for expediting the integration of civil Unmanned Aircraft System (UAS) into the national airspace system, and secondly create a 5 year roadmap for their introduction. Thirdly, specifically for small unmanned aircraft system, a rulemaking was required that would expedite the start of their civilian operation in national airspace system.

In response, in 2013, FAA along with other governmental agencies jointly published a comprehensive plan for integration of UAS into national airspace system[22]. In this

document, UAS national goals and objectives are described. One of the goals is to make civil visual-line-of-sight operation of small UAS a routine by 2015. Initially this will be outside of class B and C airspace and above urban areas. In accordance with this goal, in February 2015, FAA published a notice of proposed rulemaking [24] that addresses introduction of small UAS (i.e. weighing less than 25kg) into national airspace system. Various safety measures have been proposed such as visual-line-of-sight operations. Flights are restricted to day time at a maximum altitude of 152.4m above the ground. Small UAS cannot operate in class A airspace. However, operation within class B, C, D, and E airspace is possible with permission from ATC. Furthermore, operation in class G airspace does not require a permission from ATC. As mentioned before, for UAS that weighs more than 250 grams, the owner has to register it with FAA [25] for outdoor operation. Another goal set forth in the comprehensive plan is to make routine operation of UAS possible in the national airspace by 2015 for the public organizations and by 2020 for the civilians.

To comply with the public law [66], FAA has published a roadmap for integration of UAS in the national airspace system[21]. Currently for UAS to access the airspace, Certificate of Waiver or Authorization (COA) are needed for public operation and certain airworthiness certificates for experimental civil application as mandated in [66]. Initially FAA plans to accommodate UAS in the near-term (next 5 years), then it transitions into the period of integration (5-10 years) in the mid-term and in the long- term (more than 10 years) it is expected that requirements from UAS will evolve based on the safety requirements from all type of aircraft and is consistent with the timeline for NextGen vision. FAA asserts that for UAS to be allowed access to the national airspace, they must be able to apply and be accepted for standard airworthiness certificate.

A challenging goal for the FAA is to integrate UAS without segregating various types of aircraft. Two important required technologies according to the FAA's roadmap is Sense And Avoid (SAA) and Control and Communication (C2). The SAA is expected to ensure self-separation and at a later stage collision avoidance which needs to be interoperable with other collision avoidance systems as well as compatible with ATC separation services. According to the FAA, third party-communication service providers are used frequently today and it is a routine task for FAA to effectively monitor their performance. The choice of the right type of third party C2 providers is dependent on the choice of UAS architecture. At International Telecommunication Union's World Radiocommunication Conference in 2012, an agreement was reached to dedicate a part of frequency spectrum for exclusive use by UAS. This paves the way for the operation of UAS across international borders and protects UAS from interference from other devices. [21]

According to the roadmap[21], the FAA asserts that unless new classes of airspace are specifically created for UAS, for them to be accepted for integration in the national

airspace system, they must satisfy the following requirements from FAA (with notable exception of line-of-sight small UAS). In addition to airworthiness certificates alluded to above, any UAS must register and execute an Instrument Flight Rules (IFR) flight plan (see [65] for a definition) and be equipped with ADS-B (Out); i.e. the Automatic Dependent Surveillance (ADS) broadcasting component. Furthermore, they have to meet the minimum performance and equipage requirement of the area where the operation takes place. Additionally, each UAS must have a flight crew including a pilot-in-command who is only in charge of only one UAV and fully autonomous operations will not be allowed. Also, minimum required separation must be met in the controlled airspace and ATC will be in charge for separation services for the applicable airspace classes for manned and unmanned aircraft.

In my opinion, the ban on the fully autonomous operation set forth by the FAA in their roadmap takes away the major benefits of any drone architecture, including IoD. Fortunately, the FAA does not rule out the introduction of new classes specifically designated for UAS in their roadmap as mentioned above. Certainly, IoD in its current form is a theoretical framework that is only viable if these new classes are introduced. According to [21], the FAA provides a transparent process for setting regulations which encourages comments from the public as well as other feedback mechanisms for avoiding onerous regulations. In the process of crafting new legislation, the FAA has been soliciting feedback from the UAS community with one example being creation of the Advisory and Rulemaking Committee (ARC) for UAS comprising of members from industry and academia[20]. I am optimistic that the stakeholders will influence the process in a way that new airspace classes are created for UAS rather than what I believe is fitting a fundamentally new technology into a frame that was designed for a different technology.

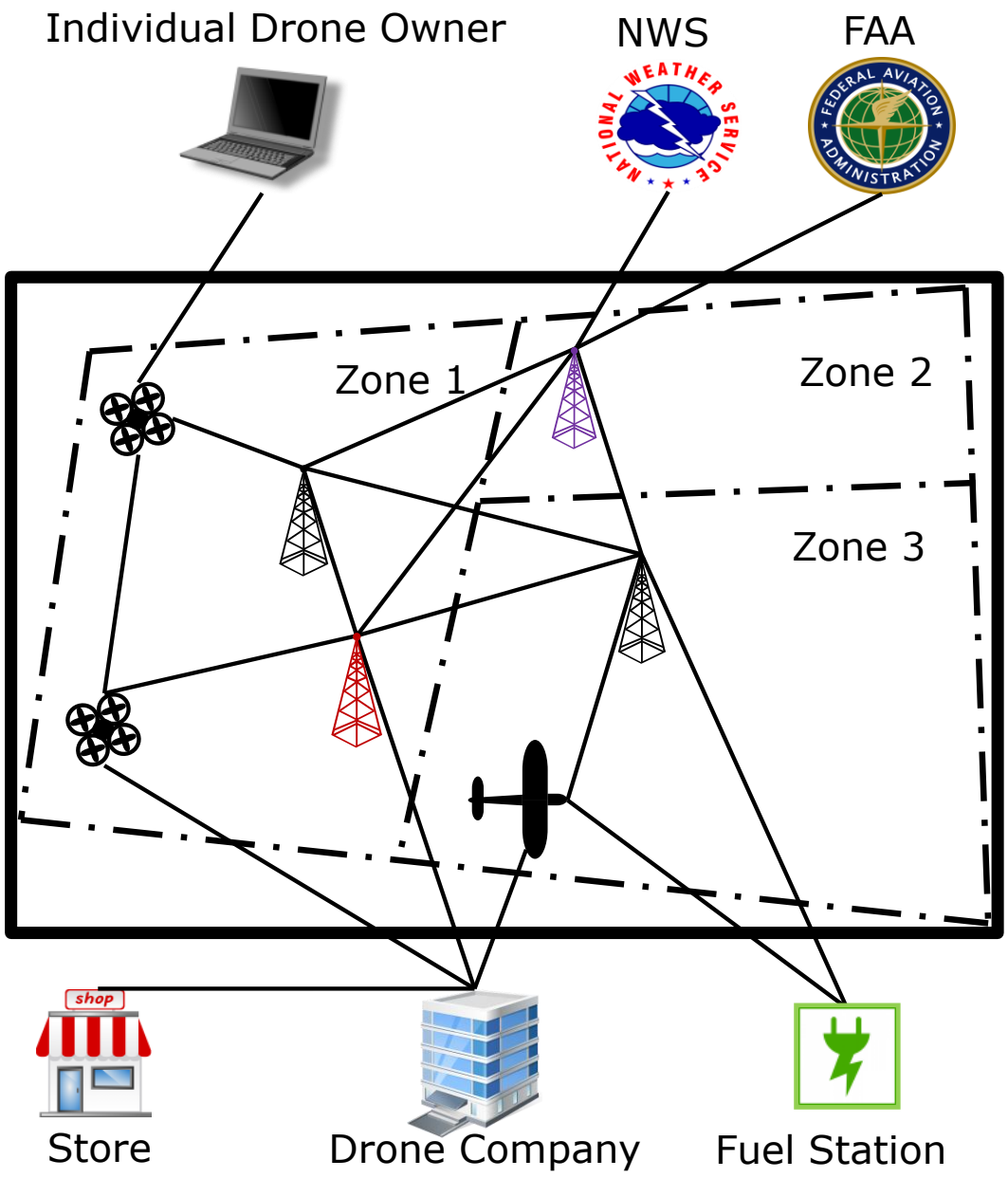


Figure 2.8: Drones and ZSPs are components inside the boundaries of the IoD system as depicted by the box. Outside components such as fuel stations, private or corporate drone owners, governmental organizations such as weather services or FAA interact with drones or ZSPs through standard protocols. Solid lines show some of the possible interactions.

Chapter 3

Vehicle Scheduling Problem

3.1 Introduction

I define a new problem called Vehicle Scheduling Problem (VSP). The goal is to minimize some objective function such as the number of tardy (late) vehicles over a transportation network subject to maintaining safety distances, meeting hard deadlines, and maintaining speeds on each link between the allowed minimums and maximums. I prove VSP is an NP-hard problem for commonly used objective functions that are used in the context of the job shop scheduling. With the number of tardy vehicles as the objective function, I formulate VSP in terms of a Mixed Integer Linear Programming (MIP) and design a heuristic algorithm. I analyze the complexity of my algorithm and compare the quality of the solutions to the optimal solution for the MIP formulation in the small cases. My main motivation for defining VSP is to provide a scheduling framework for the upcoming integration of Unmanned Aerial Vehicles (UAVs) into the airspace. As reviewed in depth in the introduction, our new scheduling problem poses a new and unique problem. In particular, the literature on JSP and its extensions including JSP with transportation robots, as well as the train scheduling do not seem to provide a framework for formulating this problem.

3.2 Problem definition

I first give an informal definition of VSP and then proceed to define it in a more rigorous way.

In VSP, we are given a number of vehicles each of which requests to make a trip at some point in time. Trips take place over a transportation network which is abstracted away as a graph. Each trip consists of a sequence of edges on this graph. We can have unlimited vehicles on each edge travelling at the same time, but when any two vehicles reach a vertex (i.e. intersection), they must be separated by some separation time constant. Therefore, to minimize some objective function (e.g. number of tardy vehicles) we have to schedule vehicles in a way that minimizes the congestion or prioritizes certain vehicles. The velocity of each vehicle on each link must be between a minimum and a maximum velocity. Given the length of the link, one can instead find the minimum and maximum allowed time on the link. In the formal definition, instead of formulating in terms of velocities, I use these time constants. Each vehicle has to meet its trip completion hard deadline. The goal is to assign an arrival time for each vehicle at each vertex on their way (i.e. the schedule) such that the objective function is minimized.

Now, I define VSP in a more rigorous way.

Definition 3.2.1 (Vehicle Scheduling Problem (VSP))**.** An instance of VSP is a 9-tuple $(G, W, \tau_{min}, \tau_{max}, \rho, d, d', S, f)$ as follows. Graph $G = (V, E)$ is a directed connected graph that represents the traffic network. The set $W = \{W_1, W_2, \cdots, W_n\}$ is a set of directed walks on G and $W_j = \left(W_j^1, W_j^2, \cdots, W_j^{q_j}\right) \in V^{q_j}$ is the sequence of vertices vehicle j will visit. Time constants τ_{min} and τ_{max} are the set of minimum and maximum allowed times for a vehicle to reach its next vertex in the walk. Accordingly, for each vehicle j, $\tau_{min,j}$ takes the form $\tau_{min,j} = \left(\tau_{min,j}^1, \tau_{min,j}^2, \cdots, \tau_{min,j}^{q_j-1}\right)$. In a similar way, $\tau_{max,j}$ is defined. The n-tuple $\rho \in \mathbb{R}^n$ denotes the trip request times and hard deadline $d' \in \mathbb{R}^n$ is the maximum allowed delays for completing the walks. Soft deadline $d \in \mathbb{R}^n$ is the maximum allowed delays after which depending on the objective function, a penalty will incur. The set S is a set of elements $s_{j_1,j_2}^{i_1,i_2} \in \mathbb{R}$ that specify the time separation between distinct vehicles j_1 and j_2 when performing the i_1'th and i_2'th step of their walks, respectively. Each element $s_{j_1,j_2}^{i_1,i_2}$ is defined only if vertices $W_{j_1}^{i_1}$ and $W_{j_2}^{i_2}$ are identical.

Subject to the following constraints, the goal is to find a set $t = \{t_1, t_2, \cdots, t_n\}$ where for each vehicle j, $t_j = \left(t_j^1, t_j^2, \cdots, t_j^{q_j}\right)$ assigns to each vertex W_j^i an arrival time stamp while the objective function $f : \mathbb{R}^q \to \mathbb{R}$ is minimized. Note that I define $q = \sum_{1 \le j \le n} q_j$ and f takes the arrival time stamps t_j^i as the input.
Constraints:
Trip request time, trip continuity, and hard deadline:

$$\rho_j \le t_j^1 \le t_j^2 \le \cdots \le t_j^{q_j} \le d'_j \tag{3.1}$$

Minimum and maximum allowable link travel time:

$$\tau_{min,j}^i \le t_j^{i+1} - t_j^i \le \tau_{max,j}^i \tag{3.2}$$

Separation enforcement:

$$|t_{j_1}^{i_1} - t_{j_2}^{i_2}| \ge s_{j_1,j_2}^{i_1,i_2} \tag{3.3}$$

In the definition above, only the vertices are included in the description of the walk and not the edges. In my definition, since edges have unlimited capacity, we do not concern ourselves with the particular edge that is chosen. However, in real life application, the length of a link and the minimum and maximum velocity among other factors might limit us to a particular edge.

3.3 NP-hardness proof

In the general case, VSP is NP-hard. I prove this for various objective functions. First I define the GENERALIZED K-MACHINE UNIT JOB SHOP PROBLEM. For the cases

$K = 3$ and $K = 2$, these problems are already defined in [54] and [49] (with no release dates, deadlines, or no-wait condition), respectively. Since my goal is to reduce this problem to VSP, I do not use the more or less standard notation for JSP since it makes the description of the reduction difficult.

Definition 3.3.1 (GENERALIZED K-MACHINE UNIT JOB SHOP PROBLEM: optimization version)**.** An instance of GENERALIZED K-MACHINE UNIT JOB SHOP PROBLEM is a 7-tuple $(M, J, r, \delta, \delta', \theta, f)$ where M is a set of machines

$$M = \{M_1, M_2, \cdots, M_K\},$$

and J is a set of jobs $\{J_1, J_2, \ldots, J_n\}$. Each job $J_j = \left(M_j^1, M_j^2, \cdots, M_j^{q_j}\right) \in M^{q_j}$ denotes a sequence of machines that will each process job J_j in order, for a duration of one time unit. The n-tuple $r = (r_1, r_2, \cdots, r_n)$ denotes the release dates and $\delta = (\delta_1, \delta_2, \cdots, \delta_n)$ denotes the soft deadlines corresponding to each job, respectively. Hard deadlines δ' are defined similarly. Also, θ is a Boolean parameter that is $true$ if jobs cannot wait between machines, otherwise $false$.

Subject to the following constraints, the goal is to find a set $x = \{x_1, x_2, \cdots, x_n\}$ where for each job j, $x_j = \left(x_j^1, x_j^2, \cdots, x_j^{q_j}\right)$ assigns to each operation J_j^i a starting time on their associated machine while the objective function $f : \mathbb{R}^q \to \mathbb{R}$ is minimized. Note that I define $q = \sum_{1 \leq j \leq n} q_j$ and f takes the arrival time stamps x_j^i as the input.

Constraints:

- Operations done in order, i.e. $x_j^i + 1 \leq x_j^{i+1}$.
- Any machine can process only one operation at a time, i.e. for any distinct j_1, j_2, if there exists i_1, i_2 such that $J_{j_1}^{i_1} = J_{j_2}^{i_2}$, then $\left|x_{j_1}^{i_1} - x_{j_2}^{i_2}\right| \geq 1$.
- No staying on the same machine, i.e. $J_j^i \neq J_j^{i+1}$.
- No job scheduled before its release date, i.e. $x_j^1 \geq r_j$.
- Meeting hard deadlines, i.e. $x_j^{q_j} \leq \delta_j'$.
- No-wait condition, i.e. if θ is $true$, for any j and any i with $1 \leq i \leq q_j - 1$, we have $x_j^i + 1 = x_j^{i+1}$.

I first show there is an efficient reduction from GENERALIZED K-MACHINE UNIT JOB SHOP PROBLEM with an arbitrary objective function to VSP.

Theorem 1. *GENERALIZED K-MACHINE UNIT JOB SHOP PROBLEM has a polynomial time reduction to VSP.*

Proof. The input to GENERALIZED K-MACHINE UNIT JOB SHOP is a 7-tuple

$$(M, J, r, \delta, \delta', \theta, f).$$

The input to VSP is a 9-tuple

$$(G, W, \tau_{min}, \tau_{max}, \rho, d, d', S, f).$$

I generate an input to the latter, for every input to the former. I set G with vertices $V_1, V_2, \cdots, V_K$ to be a complete digraph with K vertices. I establish a one-to-one correspondence between the jobs in J and walks in W as follows. Assuming $|J| = n$ and job J_j has a length of q_j; for $1 \leq j \leq n$ and the job $J_j = (M_j^1, M_j^2, \cdots, M_j^{q_j})$, I create a corresponding walk $W_j = (W_j^1, W_j^2, \cdots, W_j^{q_j})$ where $W_j^i = V_{j'}$ if and only if $M_j^i = M_{j'}$. The sequences ρ, d, and d' will have a length of n and for any j, both $\tau_{min,j}$ and $\tau_{max,j}$ have length $q_j - 1$. Next, I set $\tau_{min,j} = (1, 1, \cdots, 1)$. If θ is $false$ (i.e. jobs can wait), I set $\tau_{max,j} = (+\infty, +\infty, \cdots, +\infty)$, otherwise if θ is $true$ (no-wait), $\tau_{max} = (1, 1, \cdots, 1)$. I set $\rho = r$. For the deadlines, I set $d = \delta$ and similarly $d' = \delta'$. I set all the separation gaps in S to 1. Lastly, I will use the same objective function f in the reduction with arguments changing from x_j^i to t_j^i. This completes the conversion of the inputs. Now, to translate the solution for VSP to GENERALIZED K-MACHINE UNIT JOB SHOP is straightforward. To convert the outputs, for any arrival time stamps t_j^i for $1 \leq j \leq n$, I let $x_j^i = t_j^i$, and with this, the reduction is complete. □

I prove NP-hardness for all commonly used objective functions in the context of job shop scheduling [71].

Corollary 1.1. *VSP with any of the following objective functions is NP-hard.*

Minimizing:

- *Makespan (C_{max})*
- *Total completion time ($\sum C_j$)*
- *Total weighted completion time ($\sum w_j C_j$)*
- *Maximum lateness (lateness can be positive, 0, or negative for each vehicle) (L_{max})*

- *Total tardiness (0 or positive for each vehicle)* ($\sum T_j$)
- *Weighted number of tardy (or late) vehicles* ($\sum w_j U_j$)
- *Number of tardy (or late) vehicles* ($\sum U_j$)

Proof. For any of these objective functions, I refer to a special case of GENERALIZED K-MACHINE UNIT JOB SHOP PROBLEM that is proven NP-hard in the literature. Then, by applying Theorem 1, the NP-hardness of VSP is established for that objective function. In the following, the objective functions related to tardy vehicles correspond to tardy jobs in the context of JSP problem.

Makespan: GENERALIZED K-MACHINE UNIT JOB SHOP is NP-hard according to [54], where $K = 3$ and all jobs are released at time 0 ($r_j = 0$) and there is no deadline ($\delta'_j = +\infty$) and the jobs can wait ($\theta = false$). This problem is referred to as 3-MACHINE UNIT JOB SHOP.

Total tardiness: GENERALIZED K-MACHINE UNIT JOB SHOP PROBLEM with $K = 2$ and the same setting as in the Makespan case above is NP-hard. This problem is referred to as 2-MACHINE UNIT JOB SHOP [49, 85].

Total (weighted) completion time, (Weighted) number of tardy vehicles: Using the same setting as above for 2-MACHINE UNIT JOB SHOP PROBLEM with release dates, this problem is NP-hard. Note that without the release dates, the unweighted problems are polynomially solvable [49, 50, 85].

Maximum lateness: For similar setting as in the case of Total Completion Time, 2-MACHINE UNIT JOB SHOP PROBLEM where jobs cannot wait between two machines ($\theta = true$) is strongly NP-hard [85]. □

3.4 Exact MIP formulation

In the next section, I give an MIP formulation of the VSP where to demonstrate, I use the number of tardy vehicles as my objective function. It is possible to develop MIP formulation for the other objective functions as well by perhaps introducing more variables and doing the necessary minor adjustments.

3.4.1 Notations

I use the same notation from VSP problem and some additional notations as follows.

- Variables $P_{j_1,j_2}^{i_1,i_2}$, $N_{j_1,j_2}^{i_1,i_2}$, decision binary variable $b_{j_1,j_2}^{i_1,i_2}$, and a large fixed number $M_{j_1,j_2}^{i_1,i_2}$ will be used to convert the absolute value constraints to mixed integer linear constraints. The decision binary variable will effectively decide between each pair of vehicles with a conflicting node, which one will have the right of way.
- Variable l_j will designate a late vehicle, X_j will designate how early vehicle j is and a large fixed number M_j will be used to convert the constraints of form $X_j = \max(0, d_j - t_j^{q_j})$ into mixed integer linear constraints. I add variable d_j to designate a soft deadline that a vehicle will strive to meet. On the other hand, the hard deadline d'_j must be met to have a feasible solution.

3.4.2 Objective function

I use the total number of tardy vehicles as my objective function.

$$\min \sum_{1 \le j \le n} l_j. \tag{3.4}$$

3.4.3 Constraints

I use the same constraints as in Definition 3.2.1 except for the constraints that follow. The first constraint is the separation enforcement constraint of Eq. 3.3 where through the standard techniques, the absolute value constraint can be converted to a set of three mixed integer linear constraints.

$$t_{j_1}^{i_1} - t_{j_2}^{i_2} = P_{j_1,j_2}^{i_1,i_2} - N_{j_1,j_2}^{i_1,i_2} \tag{3.5}$$

$$b_{j_1,j_2}^{i_1,i_2} \cdot s_{j_1,j_2}^{i_1,i_2} \le P_{j_1,j_2}^{i_1,i_2} \le b_{j_1,j_2}^{i_1,i_2} \cdot M_{j_1,j_2}^{i_1,i_2} \tag{3.6}$$

$$\left(1 - b_{j_1,j_2}^{i_1,i_2}\right) \cdot s_{j_1,j_2}^{i_1,i_2} \le N_{j_1,j_2}^{i_1,i_2} \le \left(1 - b_{j_1,j_2}^{i_1,i_2}\right) \cdot M_{j_1,j_2}^{i_1,i_2} \tag{3.7}$$

Furthermore, I convert the late vehicle constraints into the acceptable form for a linear integer programming instance using familiar techniques as follows.

$$d_j - t_j^{q_j} \le X_j \tag{3.8}$$

$$0 \leq X_j \tag{3.9}$$
$$X_j \leq M_j \cdot (1 - l_j) \tag{3.10}$$
$$X_j \leq d_j - t_j^{q_j} + M_j \cdot l_j \tag{3.11}$$

3.5 Scheduling algorithms

3.5.1 Baseline algorithm: PROXIMITY

I introduce this baseline algorithm with the purpose of comparing its result to the result from the heuristic algorithm introduced in the next section. My goal is to establish the superiority of the latter algorithm.

The algorithm PROXIMITY resembles the decentralized heuristics used by car drivers in the real world and is as follows. Each vehicle on a link attached to an intersection will access that intersection at the earliest possible time, if and only if it is the closest (in time) vehicle to cross that intersection. An implementation of this algorithm is shown in Fig. 3.1 and Fig. 3.2 with *mode* variable set accordingly.

3.5.2 Heuristic algorithm: DEADLINE & PROXIMITY

My heuristic algorithm is as follows. The algorithm returns the best solution from three independent subroutines.

- PROXIMITY : It is the baseline algorithm introduced earlier.
- ABS DEADLINE & PROXIMITY: It is similar to PROXIMITY with the difference that among vehicles of equal (time) distance, the ones with a lower so called delay slack are prioritized over those with higher values. I calculate the delay slack for a vehicle as the difference between its deadline and the shortest trip time possible. However, if no delay slack is left; that is the calculated delay slack is a negative quantity, I give these vehicles the lowest priorities and among themselves, they will cross the intersection in arbitrary orders.
- REL DEADLINE & PROXIMITY: It is similar to the previous subroutine except that the delay slack is divided by the number of intersections left in the trip.

An implementation for each of these subroutines is shown in Fig. 3.1 and Fig. 3.2 with *mode* variable set accordingly.

Input: paths for vehicles, *mode*
Output: time stamps for each node

1: $timeStampSequence$ = [] *//empty list*
2: $allVehiclesList$ = list of all vehicles
3: sort $allVehiclesLis$t ascendingly using $SortingKey$ for comparison
4: **for all** *vehicle* in $allVehiclesList$ **do**
5: assign earliest possible time stamp to the *vehicle*'s first node and update data structures
6: add time stamp to $timeStampSequence$ (if does not exist) in a sorted increasing order
7: **end for**
8: **while** $length(timeStampSequence) > 0$ **do**
9: $t = timeStampSequence[0]$
10: $curVehList$ = list of all unfinished vehicles with time stamp t for their current node
11: **for all** v in $curVehList$ **do**
12: N = next node in path of v
13: $conflictVehList$ = list of all vehicles with N in their trip
14: $curVehNextNList$ = list of all vehicles with next node N and arrival time stamp t on their current node
15: sort $curVehNextNList$ ascendingly using $sortingKey$ for comparison
16: **for all** $v2$ in $curVehNextNList$ **do**
17: **if** $v2$ has no time stamp for N **then**
18: assign the earliest time stamp for N for $v2$ that satisfies separation constraints imposed by $conflictVehList$ and update data structures
19: add this time stamp to $timeStampSequence$ (if does not exist) in a sorted increasing order
20: **end if**
21: **end for**
22: **end for**
23: delete $timeStampSequence[0]$
24: **end while**

Figure 3.1: A subroutine to calculate the time stamps for vehicles. Depending on our input to the algorithm *mode* variable, the time stamps are calculated using the baseline rule or my heuristic rule.

Input: *vehicle*, *mode*
Output: A two tuple to be used for sorting by first element and then the second element
1: $first$ =time distance to next node
2: **if** *mode* is PROXIMITY **then**
3: *second* = None
4: **else if** *mode* is ABS DEADLINE & PROXIMITY **then**
5: *second* = remaining delay slack if it is ≥ 0 else $+\infty$
6: **else if** *mode* is REL DEADLINE & PROXIMITY **then**
7: *second* = delay slack / number of remaining of nodes if it is ≥ 0 else $+\infty$
8: **end if**
9: **return** (first,second)

Figure 3.2: The *SortingKey* subroutine returns a 2-tuple for each vehicle that determines their local priority based on the chosen rule.

3.5.3 Complexity

The implementation of the three subroutines mentioned above are the same with minor differences in the *SortingKey* subroutine component in Fig. 3.2. This results for the time complexity of all these algorithms to be similar. This is true because with efficient implementation of the data structures, the *SortingKey* has the same time complexity of $O(1)$ in all of the cases. Therefore to obtain the time complexity of all the three subroutines, we need only to obtain the time complexity of the subroutine in Fig. 3.1.

The time complexity of the algorithm will depend on the actual implementation of the data structures which is as follows. In the implementation, I assume I populate these data structures at the point in the algorithm where t is assigned which will take $O(1)$ time on a large enough hash table. I keep the following hash tables:

- Time stamps to list of vehicles: $H_1 : t \mapsto V$
- 2-tuples of current node time stamps and next nodes to list of vehicles: $H_2 : (t, N) \mapsto V$.
- Nodes to the list of time stamps: $H_3 : N \mapsto t$

The most time consuming operations is step 19 in Fig. 3.1 which takes $O(q)$ where q is the total number of time stamps. Assuming D is the degree of the underlying graph, considering the outer loops, the total time spent on this operation is $O(q^2 Dn)$ where n is the number of vehicles.

If we make the further assumption that the length of each path is constant, the complexity of both the baseline and the heuristic algorithm is simplified to $O(n^3)$.

3.6 Numerical results

In this section, I compare the performance of my heuristic algorithm DEADLINE & PROXIMITY to my baseline algorithm PROXIMITY. Additionally, for the smaller cases, I compare the results from both these algorithms to the exact solution obtained from solving the MIP. I consider three metrics in my comparisons, namely, the number of vehicles, the tightness of the soft deadlines, and the run time of the algorithms.

In my setup, I create uniformly at random pairs of source and destination and calculate the shortest paths on the grid like graph G as shown in Fig 3.3. I use the following parameters.

- Separation gap $s_{j_1,j_2}^{i_1,i_2} = 5$ for any j_1, j_2, i_1, i_2.
- Minimum and maximum times on link $\tau_{min,j}^{i} = 50$ and $\tau_{max,j}^{i} = +\infty$ for all i, j.
- Hard deadlines $d'_j = 2.2 q_j \tau_{min,j}$ for each j.
- Trip request time $\rho_j = 0$ for each j.

For my setup I used Gurobi optimization engine version 8.1.1 running on Microsoft Surface Pro 5 with 8GB RAM and 4 Intel(R) Core(TM) i5-7300U CPUs clocking at 2.60GHz. A Gurobi Python 3.6 binding was used to solve the MIP exactly. Also, I implemented the baseline and the heuristic algorithm in Python 3.6.

I create 20 instances of VSP with four different vehicle counts; 25, 50, 75, and 100. For each instance, I experiment with varying levels of tightness of the soft deadline. The fraction of missed deadlines corresponding to each deadline value are reported in Fig. 3.4. The worst run time over all deadlines for each number of vehicles is reported in Fig. 3.5 for the baseline and the heuristic algorithms.

For 25 vehicles, Fig. 3.4a shows the results comparing the percentage of missed deadlines for my heuristic algorithm, the baseline, and the exact solution from MIP. Given the small difference between the result from the baseline algorithm and the exact solution, it is plausible to infer the baseline algorithm is in fact a very effective one in producing schedules with a few number of missed deadlines.

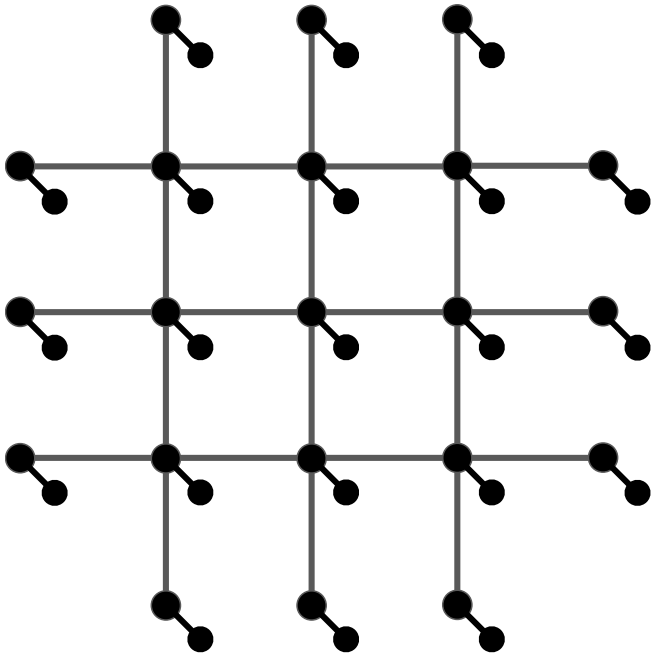
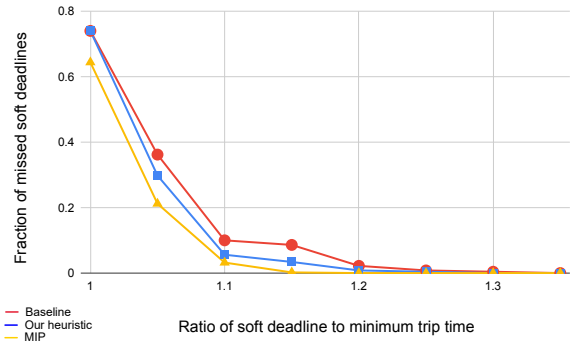


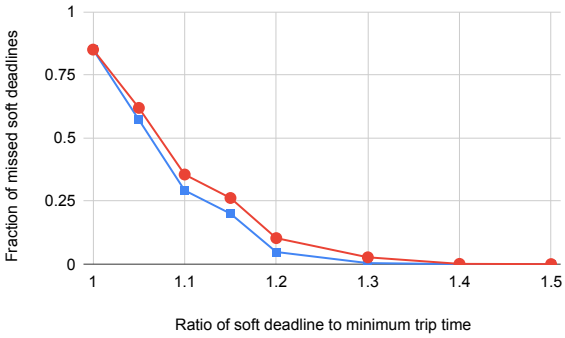
Figure 3.3: Underlying graph for simulation is a 5x5 grid like graph. Each edge represents a bidirectional path with dedicated lanes for each direction.

In the case of 25 vehicles, I am able to solve most of the MIP instances exactly in a reasonable amount of time (less than 1 hour). Since The MIP quickly becomes intractable with increasing the number of vehicles, the 25 vehicles is about the maximum number of vehicles that can be used in my test. The comparison between the run time of my heuristic algorithm to the MIP instance solved by Gurobi is reported in Fig. 3.6 on a logarithmic scale. In most cases, my algorithm is between 1 to 3 orders of magnitude faster, despite the fact that my heuristic algorithm is implemented in notoriously slow Python.

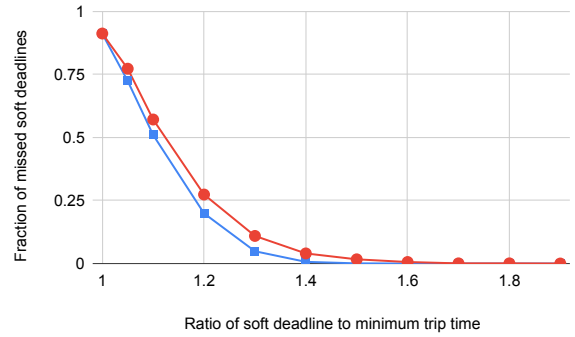
Finally, Fig. 3.5 shows a comparison between the run time of my heuristic algorithm and the baseline algorithm.



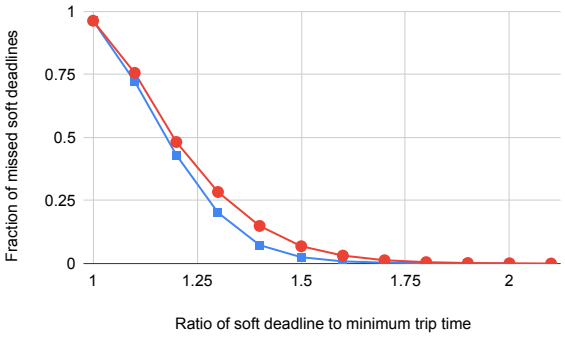
(a) Simulation result for 25 vehicles



(b) Simulation result for 50 vehicles



(c) Simulation result for 75 vehicles



(d) Simulation result for 100 vehicles

Figure 3.4: (a), (b), (c), (d) Simulation results for respectively 25, 50, 75, and 100 vehicles with randomly generated trips comparing the solutions to the baseline algorithm (circle markers) versus my heuristic algorithm (square markers). For the case of 25 vehicles, I include also the exact MIP solution (triangle markers). The vertical axis is the fraction of tardy vehicles averaged over 20 random instances and the horizontal axis is the ratio of the set soft deadlines to the minimum congestion free trip times.

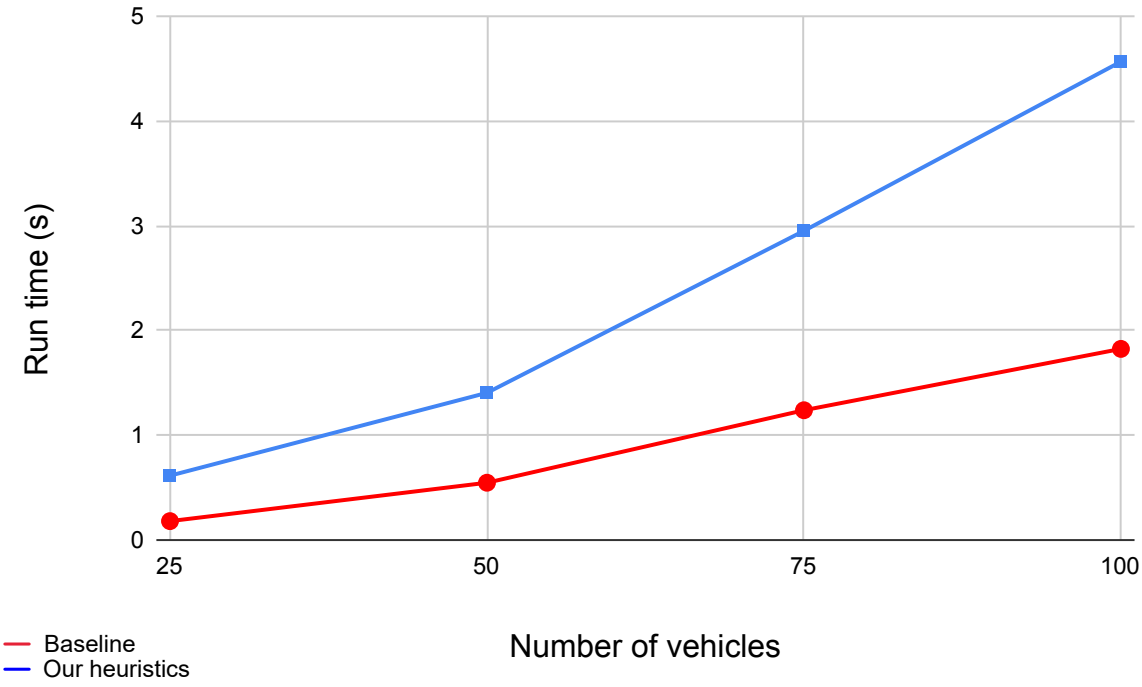


Figure 3.5: This figure shows the average taken over 20 samples of the worst run time among varying levels of soft deadlines. The value of soft deadlines in my test are the same as those appearing in Fig. 3.4. The chart shows how the run time is affected by increasing the number of vehicles from 25 to 100. The square markers represent results from my heuristic algorithm while the circle markers represent the baseline results.

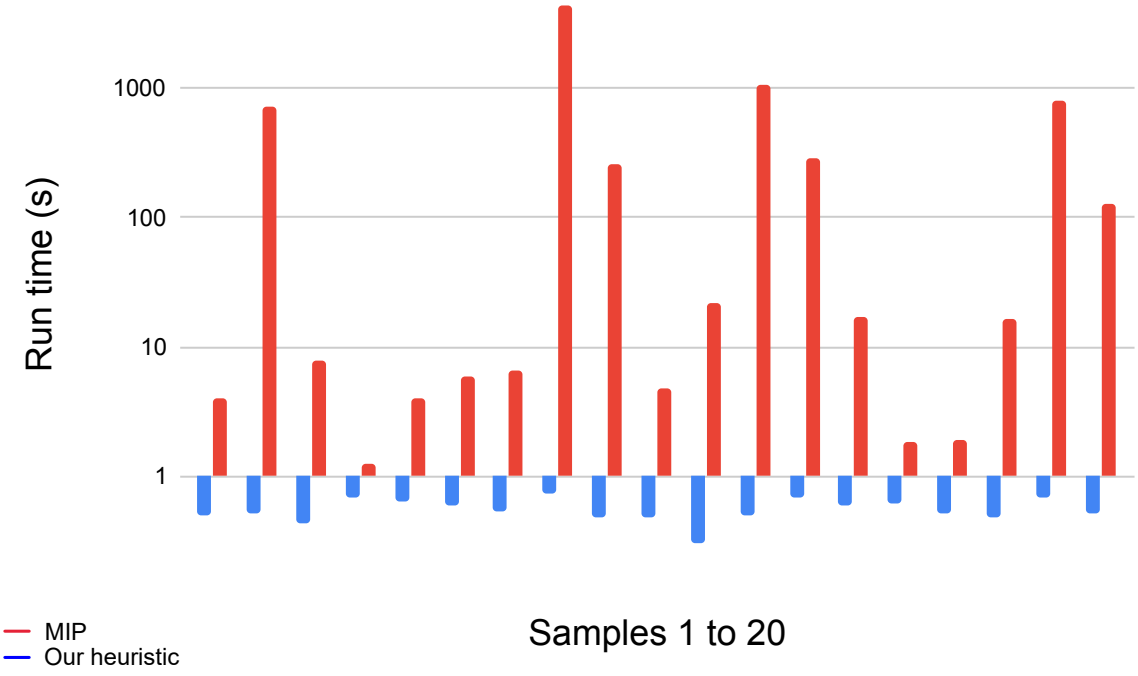


Figure 3.6: This figure shows the run time results of my heuristic algorithm compared to the MIP solver in logarithmic scale for 20 random instances. Each value represents the average of worst run time among various levels of tightness of soft deadlines. These deadline values are similar to those in Fig. 3.4a. Each of these instances utilize 25 vehicles. The red bars which all happen to have a value greater than 1 are the run time from MIP solver and the blue bars under the grid line for 1 are the run time from my heuristic algorithm. As can be seen, in most cases, my algorithm implemented in Python is between 1 to 3 orders of magnitude faster.

Chapter 4

Traffic Flow Model For UAVs

Figure 4.1: Vehicle i's position on the one directional link is shown with x_i. The first vehicles is indexed 0.

4.1 Introduction

In this chapter, I introduce a microscopic traffic flow model called Scalar Capacity Model (SCM) which can be used to study the formation of traffic on an airway link for autonomous UAV as well as for the ground vehicles on the road. Given the 3D nature of UAV flights, the main novelty in my model is to eliminate the commonly used notion of lanes and replace it with a notion of density and capacity of flow, but in such a way that individual vehicle motions can still be modeled. I name this a Density/Capacity View (DCV) of the link capacity and how vehicles utilize it versus the traditional One/Multilane View (OMV). An interesting feature of this model is exhibiting both passing and blocking regimes (analogous to multi-lane or single-lane) depending on the set scalar parameter for capacity. I show the model has linear local (platoon) and asymptotic linear stability. Also, I perform numerical simulations and show evidence for non-linear stability. My traffic flow model is represented by a nonlinear differential equation which I transform into a linear form. This makes my model analytically solvable in the blocking regime and piece-wise analytically solvable in the passing regime. Finally, a key advantage of using my model over an OMV model for representing UAV's flights is the removal of the artificial restriction on passing via only adjacent lanes. This will result in an improved and more realistic traffic flow for UAVs.

4.2 Model

In my model, I consider a sequence of vehicles numbered as 0 up to $N-1$ from the first to the last vehicle travelling along an infinite link. The position of each vehicle is designated by x_i with respect to some chosen origin (Fig. 4.1).

The vehicles adjust their speeds based on the distances to the vehicles in front of them according to some exponential weighting scheme. This model is represented by the non-linear differential equations described by Eq. 4.1, 4.2, and 4.3 as follows.

$$\frac{dx_i}{dt} = V_i \left(1 - \Gamma_i\right) \tag{4.1}$$

$$\Gamma_i = \frac{1}{\kappa} \sum_{0 \leq j < i} \exp\left(\frac{x_i - x_j}{\omega}\right) \tag{4.2}$$

$$\frac{dx_0}{dt} = V_0 \tag{4.3}$$

where the constant V_i is the maximum free flow speed for vehicle i and Γ_i is the congestion factor. The constant ω is called the horizon in front of each vehicle. Once the leading cars are inside this horizon, they will have a substantial effect on slowing down vehicle i, otherwise their effects will be small. Parameter κ is called capacity. Intuitively, κ is roughly the maximum number of vehicles permitted inside the horizon ω. One way to see this is that if all vehicles in front of a vehicle i are located right in front of it, it takes κ vehicles for Γ_i to be 1 in Eq. 4.2 and as a result vehicle i to slow down to 0 velocity (i.e. a perfect jam). However, it is worth mentioning that κ need not be an integer and can take any real positive value.

In accordance with this, in section 4.4.2, I prove that given $0 < \kappa \leq 1$, faster vehicles cannot overtake slower vehicles, corresponding to effectively a 1-lane link (since intuitively only 1 vehicle is allowed in i'th vehicle's horizon as explained above). However, if $1 < \kappa$, faster vehicles might be able to pass slower vehicles if certain conditions are satisfied; corresponding to a multi-lane link. I refer to these two different regimes as *passing* and *blocking* hereafter.

4.2.1 Discussion and design philosophy

In this section, I discuss the model in greater depth as well as some of the details of the model. Some of the main design decisions or features of the model are already presented in the introduction and throughout the chapter and I do not revisit them here.

Originally, in my architecture, Internet of Drones (IoD) [30], I proposed each airway to be a single lane to reduce technological burden on drones to safely execute a passing maneuver. However, it is plausible that as technology matures, allowing passing will increase the efficiency of airway usage. I am interested in both of these cases in this chapter.

I argued earlier that the pass planning should be aggregated. In pass planning, we are dealing with specific maneuvers that happen for a vehicle to change its lateral position (in DCV models) or lane (in OMV models) which has a low relevance to the goal of studying the longitudinal movements. Furthermore, from a technical perspective, passing maneuvers for UAVs is less structured and require a more complex passing model.

One difference between the ground vehicles and autonomous UAVs is the delay aspect. I have assumed the delays for an autonomous vehicle to adjust its velocity according to the traffic condition is negligible. This is not an entirely correct assumption as while it is plausible to assume the perception and reaction time will be very small compared to the human operated vehicles, still there will be a delay component dictated by the mechanical properties of the system and its inertia.

Another design choice that I made was the use of space gaps between vehicles compared to the time gaps. Time gaps seem to be the reasonable choices in cases where there is a high disparity between the maximum velocities of different vehicles. But they also lack a crucial component for use for the airway. Since it is expected that the airway links will be very low altitude, they will be affected by the wind disturbances present in the urban centers. These can displace a UAV by several meters. Therefore, it seems the safest choice is to space vehicles apart enough to safeguard for these disturbances. While time gaps are important as well, we cannot rely solely on them to ensure the safety of flights.

A difference between my model and multi-anticipation models as reviewed in Chapter 1 is how the congestion is calculated. In my model in accordance to DCV, I take into account all the vehicles at the front whereas in multi-anticipation models, given the OMV frameworks, only the vehicles on the same lane are considered.

My model makes it easy to introduce stationary or moving bottlenecks without modifying the model. For example, in the DCV framework, we can adjust the capacity locally by adding dummy vehicles (stationary or moving) whereas in the OMV case, we need to deal with explicit lane closures.

4.3 Analytical solution

I study the passing and blocking regimes separately below.

4.3.1 Blocking regime

I use a differential equation technique to transform the characterizing differential equation (Eq. 4.1) into a linear differential equation. A similar technique was used in [63]. Defining the auxiliary variable

$$z_i := \exp\left(\frac{-x_i}{\omega}\right) \tag{4.4}$$

we will have

$$\frac{dx_i}{dt} = \frac{-\omega}{z_i} \cdot \frac{dz_i}{dt}. \tag{4.5}$$

Replacing z_i in Eq. 4.1 and Eq. 4.2 will yield

$$\frac{-\omega}{z_i} \cdot \frac{dz_i}{dt} = V_i \left(1 - \frac{1}{\kappa} \sum_{0 \leq j < i} \frac{z_j}{z_i}\right). \tag{4.6}$$

After simplifications, we will have

$$\frac{dz_i}{dt} = \frac{-V_i}{\omega} z_i + \frac{V_i}{\kappa\omega} \sum_{0 \leq j < i} z_j. \tag{4.7}$$

Eq. 4.7 creates a set of homogeneous linear differential equations. There is no shortage of ways to solve this set of equations. One particular way which is especially applicable here is to solve a series of first order linear differential equations as follows. First let me define

$$Z_i(t) = \frac{V_i}{\kappa\omega} \sum_{0 \leq j < i} z_j. \tag{4.8}$$

Starting from z_1, it can be solved by solving the following differential equation

$$\frac{dz_1}{dt} = \frac{-V_1}{\omega} z_1 + Z_1(t). \tag{4.9}$$

Since Z_1 is a known function in time, z_1 can be solved easily by the standard methods as it is a first order linear differential equation. As a result, now Z_2 is a known function in time, and similarly z_2 can be solved. Applying this method recursively, the whole set of equations can be solved by solving the resulting first order linear equation for each z_i.

By solving the set of equations using a method like above, in the simple case where all V_i's are unique, the general solution to this set of differential equations can be written as

$$z_i(t) = \sum_{0 \leq j \leq i} c_{i,j} \exp\left(\frac{-V_j t}{\omega}\right) \tag{4.10}$$

where $c_{i,j}$ will be determined using the initial conditions.

In the case where velocities are not unique, the solution looks a bit more involved, but can be expressed in the following way. First let U be the set of smallest indices of

vehicles with unique maximum velocities. Let $m_{i,j}$ be the multiplicity of each velocity V_j for vehicles 0 to i (that is those ahead of vehicle i). Then the solution for z_i will be of form

$$z_i = \sum_{\substack{j \in U \\ j \leq i}} \sum_{0 \leq d < m_{i,j}} c_{i,j,d} \cdot t^d \exp\left(\frac{-V_j t}{\omega}\right) \tag{4.11}$$

and $c_{i,j,d}$ will be determined by the initial conditions.

4.3.2 Passing regime

The same analytical approach of the blocking regime applies to the passing regime. However, after each overtake, we need to solve the differential equations again for the vehicles involved in passing and all the vehicles behind them. Therefore, we need to compute the passing times or in other words the roots to the equations of type

$$x_{i+1}(t) - x_i(t) = 0 \tag{4.12}$$

or equivalently

$$z_{i+1}(t) - z_i(t) = 0. \tag{4.13}$$

The problem is to find the equation that has the smallest passing time and the passing time itself. This is necessary, so the coefficients in the solution can be corrected as soon as a passing occurs.

I have not developed any heuristics for the root finding algorithm, but it seems plausible that an algorithm can generate a short list of candidate equations that are suspected to have the smallest root based on various heuristics such as the distance between two vehicles and the velocity differences among other things. It is then easy to verify whether the obtained passing time is indeed minimal by checking that only one pass has occurred.

4.3.3 Stability analysis

As mentioned, a differential equation technique was used to turn Eq. 4.5 and 4.6 into the linear form of Eq. 4.7. However, since the variables z_i in terms of which Eq. 4.7 is linear are at their cores exponential functions, they can never be 0. This is relevant since the point where all state variables are 0 is the unique equilibrium point for the linear systems of form

$$\frac{dq}{dt} = Aq$$

where

$$det(A) \neq 0.$$

Putting Eq. 4.7 in this matrix format will yield a lower triangular matrix A whose diagonal elements are $\frac{-V_i}{\omega}$ and therefore non-zero. Since in a lower triangular matrix, the eigenvalues are the diagonal elements and no 0 eigenvalue exists in this case, the determinant is non-zero. Therefore, we cannot use my analytical result for the purpose of stability analysis. In the next section, I rely on linearization and numerical simulation to study the stability of the model in the blocking regime.

4.4 Model properties

4.4.1 Soundness

In this section, I prove a few theorems that establish some of the expectations we have from a sound model.

We expect the velocities that are prescribed for each vehicle to be in the direction of the flow; that is non-negative. Here I show that my model never prescribes a negative velocity.

Theorem 2 (Non-negative velocity)**.** *Given vehicle i with maximum velocity V_i in a platoon, $\frac{dx_i}{dt} \geq 0$.*

Proof.
For the sake of contradiction, and without loss of generality, let n be the vehicle closest to the front in a platoon whose velocity will become negative. Call the moment when the velocity becomes zero, $t = t_0$. Taking a time derivative from both sides in Eq. 4.1, we will have

$$\frac{d^2x_i}{dt^2} = \frac{V_i}{\kappa\omega} \sum_{0 \leq j < i} \left(\frac{dx_j}{dt} - \frac{dx_i}{dt} \right) \exp\left(\frac{x_i - x_j}{\omega} \right). \tag{4.14}$$

Evaluating Eq. 4.14 at $t = t_0$, we will get

$$\frac{d^2x_i}{dt^2} = \frac{V_i}{\kappa\omega} \sum_{0 \leq j < i} \left(\frac{dx_j}{dt} \right) \exp\left(\frac{x_i - x_j}{\omega} \right) > 0 \tag{4.15}$$

where the strict inequality holds since no vehicle j with $j < i$ can have a negative velocity due to my assumption and at least the first vehicle has a positive velocity. Since the

derivative of velocity is positive, the velocity cannot become negative.

□

Next, I prove that a vehicle with a smaller maximum velocity cannot pass a vehicle with a larger maximum velocity. One might perceive this is possible if the faster vehicle is subject to more congestion, but I show this will never be the case.

Before presenting the next theorem, let me first define a platoon. A platoon is referred to a group of vehicles that travel together while keeping their distances under some upper bound (i.e. the distance of the first to the last vehicle is always bounded by some constant).

Theorem 3 (No overtaking by slow)**.** *Given vehicles i and $i+1$ in a platoon, if*

$$V_i \geq V_{i+1}$$

and

$$x_i(t_0) > x_{i+1}(t_0),$$

then

$$x_i(t) > x_{i+1}(t)$$

for all $t \geq t_0$.

Proof.
Assume there exists some $t = t_p$, where

$$x = x_i(t_p) = x_{i+1}(t_p).$$

By using Eq. 4.1 for vehicle i at time $t = t_p$, we can rewrite Eq. 4.1 for vehicle $i+1$ as

$$\frac{dx_{i+1}}{dt} = V_{i+1}\left(1 - \left(\frac{1}{\kappa} + \Gamma_i(t)\right)\right) =$$

$$-\frac{V_{i+1}}{\kappa} + \frac{V_{i+1}}{V_i}\frac{dx_i}{dt}. \tag{4.16}$$

From Eq. 4.16 above, it is clear that at $t = t_p$,

$$\frac{dx_i}{dt} > \frac{dx_{i+1}}{dt}.$$

This proves that passing will never be completed.

□

4.4.2 Passing or blocking behavior

First I prove a necessary and sufficient condition for a vehicle to pass another. Then I prove one of my main results that there exists a threshold for κ above which, the model permits passing and below which it is not permitted. This constitutes a regime change in my model.

Theorem 4 (Passing condition)**.** *Given vehicles i and $i+1$ in a platoon with*

$$V_{i+1} > V_i$$

and

$$x_{i+1}(t_p) = x_i(t_p),$$

vehicle $i+1$ will pass vehicle i if and only if at the time of passing t_p the following condition is met

$$\kappa(1 - \Gamma_i(t)) > \frac{V_{i+1}}{V_{i+1} - V_i}. \tag{4.17}$$

Proof.
Eq. 4.2 and theorem 2 imply that

$$0 \leq \Gamma_i(t) \leq 1.$$

I use Eq. 4.1 for vehicle i and Eq. 4.16 for vehicle $i+1$ (which also holds here) in the following. Vehicle $i+1$ will pass vehicle i if and only if we have (at time of passing)

$$\frac{dx_{i+1}}{dt} - \frac{dx_i}{dt} > 0 \Leftrightarrow$$

$$(V_{i+1} - V_i)(1 - \Gamma_i(t)) - \frac{V_{i+1}}{\kappa} > 0 \Leftrightarrow$$

$$\kappa(1 - \Gamma_i(t)) > \frac{V_{i+1}}{V_{i+1} - V_i}. \tag{4.18}$$

□

The next corollary states one of my main results.

Corollary 4.1. *$\kappa \leq 1$ is a sufficient condition for no passing to occur.*

Proof.
Eq. 4.2 and theorem 2 imply that

$$0 \leq \Gamma_i(t) \leq 1.$$

Since $0 < \kappa \leq 1$, we have

$$0 \leq \kappa(1 - \Gamma_i(t)) \leq 1. \tag{4.19}$$

According to theorem 4, a faster vehicle $i+1$ will pass vehicle i if and only if Eq. 4.17 holds. But this will not hold as

$$1 < \frac{V_{i+1}}{V_{i+1} - V_i}.$$

Therefore no passing occurs.

□

4.4.3 Asymptotic behavior in passing regime

We cannot perform a straightforward stability analysis in the passing regime since it is not clear how to conceptualize a reasonable equilibrium point in this case. However, the following theorems will be useful in understanding the passing regime in the asymptotic case.

Theorem 5 (Order stability)**.** *There exists a time T after which the order of vehicles in the system will not change.*

Proof.
The number of possible orderings is fixed. Also, a slower vehicle cannot pass a faster vehicle according to theorem 3. This creates a partial order on the set of ordering configurations. Therefore, at any state, either the system remains in that state forever or will move to a new state according to the partial order with no going back. Since the number of new admissible states is finite, the system will have to stay in one of the states forever after some time T.

□

One might suspect that given enough time, vehicles will be sorted based on their maximum velocities; that is the fastest vehicle will become the first vehicle, the second fastest vehicle will be the second, and so on. But as we will see in Theorem 6, this will not necessarily be the case unless there is a meaningful difference between the velocities of any two

vehicles. Intuitively, this can be understood in the following way; if a highway is congested to some extent and there are two vehicles that have slightly different maximum velocities, it is difficult for the fast vehicle to gain enough speed difference to take advantage of the little space available and overtake the slow vehicle.

Theorem 6 (All fast vehicles pass condition)**.** *For an arbitrary set of N vehicles with the maximum velocities $V_0, V_1, \cdots, V_{N-1}$ and arbitrary initial ordering, as time goes to infinity, vehicles will be sorted via passing according to their maximum velocities, if and only if the following holds*

$$\kappa > \max_{\substack{0\leq i,j\leq N-1 \\ i\neq j}} \left(\frac{V_j}{V_j - V_i} \right). \tag{4.20}$$

Proof.
I first prove given the condition in Eq. 4.20, a sorted order will be achieved. From theorem 5, the final order will be stable. I take the moment we reach the stable state as the origin of time. For the sake of contradiction, assume the stable order is not sorted according to the maximum velocities. Let $i+1$ be the first vehicle with a larger maximum velocity than vehicle i, that is

$$V_{i+1} > V_i.$$

Since the vehicles in front of i are faster than i, as the time goes to infinity, we have

$$\Gamma_i(t) \to 0.$$

So for any sufficiently small ϵ, there exists some t_ϵ such that for

$$t > t_\epsilon \geq 0$$

we have

$$\Gamma_i(t) < \epsilon.$$

For any time $t > t_\epsilon$, We can rewrite Eq. 4.1 for vehicle $i+1$ as

$$\frac{dx_{i+1}}{dt} =$$

$$V_{i+1}\left(1 - \exp\left(\frac{x_{i+1} - x_i}{\omega}\right)\left(\frac{1}{\kappa} + \Gamma_i(t)\right)\right) >$$

$$V_{i+1}\left(1 - \frac{1}{\kappa} - \epsilon\right). \tag{4.21}$$

For vehicle i, we have

$$\frac{dx_i}{dt} \leq V_i.$$

To prove the passing occurs, it is sufficient to show

$$\frac{dx_{i+1}}{dt} \geq V_i + \epsilon'$$

for all $t > t_\epsilon$ and some fixed $\epsilon' > 0$. Eq. 4.20 implies that

$$\kappa > \frac{V_{i+1}}{V_{i+1} - V_i} \implies \frac{1}{\kappa} = 1 - \frac{V_i}{V_{i+1}} - \epsilon'' \tag{4.22}$$

for some $\epsilon'' > 0$. Replacing Eq. 4.22 in Eq. 4.21, yields

$$\frac{dx_{i+1}}{dt} > V_i + V_{i+1}(\epsilon'' - \epsilon) > V_i \tag{4.23}$$

where the last inequality holds for any $\epsilon < \epsilon''$ where ϵ is sufficiently small. Therefore $i+1$ will pass i which will be a contradiction. Therefore, the order is only stable, if it is sorted according to the maximum velocities.

Now, let us assume the set of given maximum velocities are sorted by index so that a larger index corresponds to a larger maximum velocity. To prove the other direction of the theorem, we assume the reverse of Eq. 4.20 holds

$$\kappa \leq \max_{\substack{0 \leq i,j \leq N-1 \\ i \neq j}} \left(\frac{V_j}{V_j - V_i} \right) = \frac{V_j}{V_j - V_{j-1}} \tag{4.24}$$

where without loss of generality, I assumed V_j and V_{j-1} are the two velocities that maximize the middle term. The equality above can be inspected to be true by dividing both the numerator and the denominator in the second term by V_j and observing that only consecutive indexes can result in a maximum.

I construct a non-passing example as follows. Vehicles $j-1$ and j will have maximum velocities V_{j-1} and V_j. All faster vehicles than V_{j-1} will be placed in front of the $j-1$'th vehicle. Furthermore, all slower vehicles than V_j will be placed behind the j'th vehicle. This might induce a change of indices which will be done as needed.

For the sake of contradiction, let us assume the j'th vehicle will pass the $j-1$'th vehicle. At the time of passing, theorem 4 implies

$$\kappa > \frac{V_j}{V_j - V_{j-1}} \tag{4.25}$$

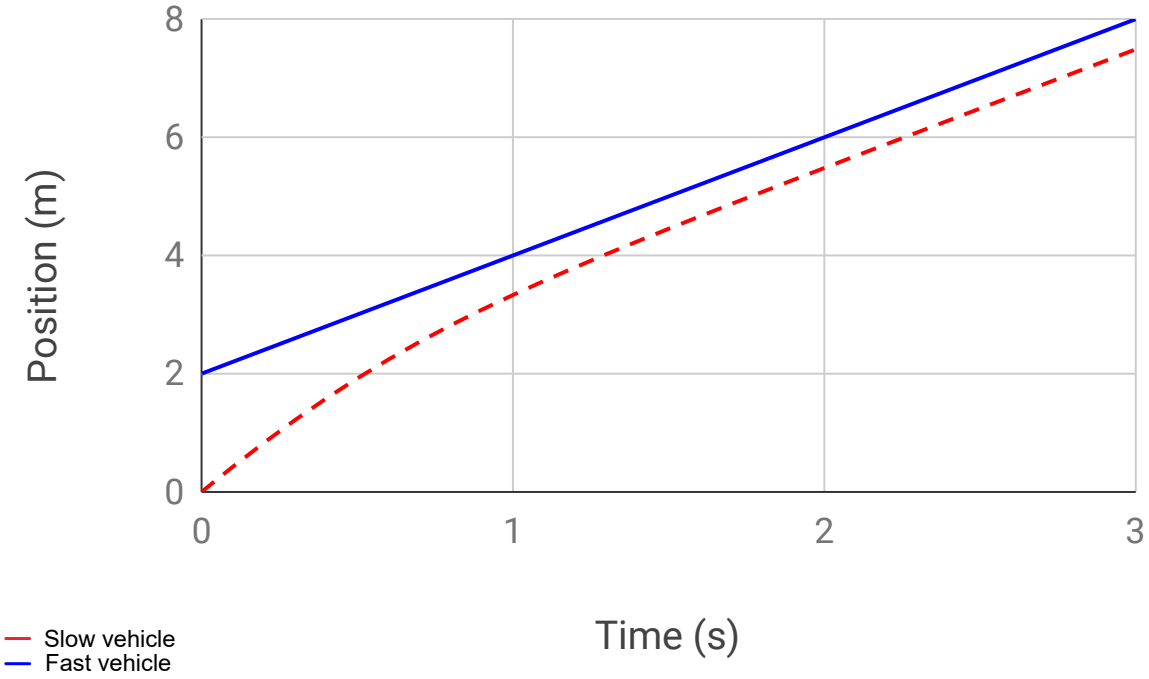


Figure 4.2: In this case, due to the low capacity of the link ($\kappa = 1$) a faster vehicle gets stuck behind a slower vehicle.

given that from Eq. 4.2 and theorem 2 we have

$$0 \leq \Gamma_{j-1}(t) \leq 1$$

which results in a contradiction.

□

I summarize these results together with corollary 4.1 on the effect of κ on how the model operates in Table 4.1. Two example demonstrations of the effect of κ on the passing behavior can be seen in Fig. 4.2 and Fig. 4.3.

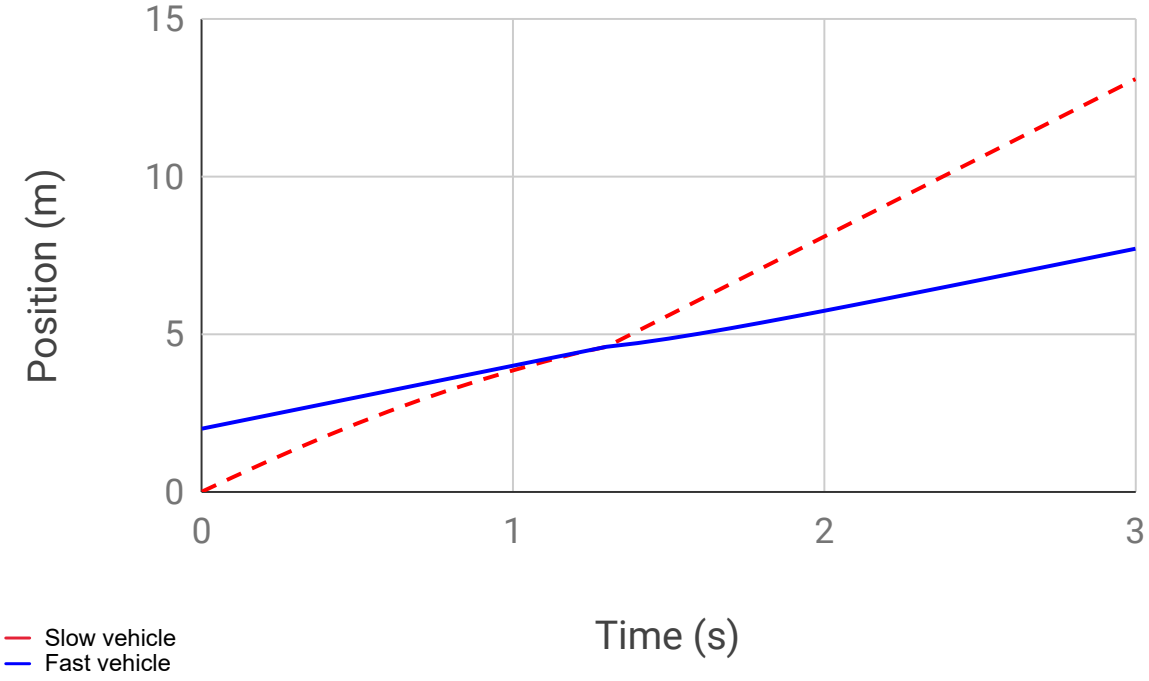


Figure 4.3: In this case, the link has enough capacity ($\kappa = 2$) and a faster vehicle easily passes a slower vehicle.

4.4.4 Asymptotic behavior in blocking regime: linear stability analysis

The standard tool to study the asymptotic behavior in this case is stability analysis. I will study linear stability analysis for vehicles placed on an infinitely long road.

Equilibrium point for the infinite road case

My state variables are the velocities of each vehicle excluding the first vehicle which has a constant velocity of V_0. Assuming we have a sequence of N vehicles such that

$$V_i > V_0$$

for all

$$i \leq N - 1,$$

Table 4.1: Effect of capacity (κ) on the model's behavior

Capacity (κ)	**Model's behavior**
Low: $\kappa \leq 1$	**Blocking regime**: No vehicle can pass
Medium: $1 < \kappa \leq \max_{i,j,i\neq j} \left(\frac{V_j}{V_j - V_i}\right)$	**Passing regime**: Initial position of vehicles determines the final ordering; that is which vehicles will end up passing
High: $\kappa > \max_{i,j,i\neq j} \left(\frac{V_j}{V_j - V_i}\right)$	**Passing regime**: All faster vehicles end up ahead of slower ones

there exists an equilibrium point where all vehicles travel at the same velocity

$$v^{eq} = v_i^{eq} = V_0.$$

Local (platoon) and linear stability analysis

Theorem 7. *In the blocking regime, given a platoon of N vehicles, the Scalar Capacity Model has both local (platoon) and asymptotic linear stability.*

Proof.
I first prove asymptotic linear stability of the model and then the local (platoon) stability will follow as a special case. My state variables are the velocities of all vehicles except the leader V_0 as it is fixed. Without loss of generality, I assume

$$V_i > V_0$$

for all $i > 0$ (otherwise, we do not have a single platoon according to lemma 9 in the Appendix). I define the gap in front of vehicle i and the gap between vehicle n and i, respectively, as

$$s_i := x_{i-1} - x_i, \tag{4.26}$$

$$s_{n,i} := \sum_{i+1 \le j \le n} s_j. \tag{4.27}$$

In the equilibrium we have

$$v_n = v^{eq} = V_n \times \left(1 - \frac{1}{\kappa} \sum_{0 \le i < n} \exp\left(\frac{-s_{n,i}^{eq}}{\omega}\right)\right). \tag{4.28}$$

Now I apply a small perturbation to the velocity of each follower as follows

$$v_i = v^{eq} + u_i(t), \tag{4.29}$$

$$s_i = s^{eq} + y_i(t). \tag{4.30}$$

From Eq. 4.26, Eq. 4.29, and Eq. 4.30, we have

$$\frac{dy_i}{dt} = u_{i-1}(t) - u_i(t). \tag{4.31}$$

For y_i's I define an identity similar to Eq. 4.27 as follows

$$y_{n,i} := \sum_{i+1 \le j \le n} y_j. \tag{4.32}$$

Assuming we kick all the follower vehicles out of equilibrium, for the n'th vehicle we will have

$$v_n = v^{eq} + u_n(t) = V_n \times \left(1 - \frac{1}{\kappa} \sum_{0 \le i < n} \exp\left(\frac{-s_{n,i}^{eq} - y_{n,i}(t)}{\omega}\right)\right). \tag{4.33}$$

After linearization and simplification using Eq. 4.28, we get

$$v^{eq} + u_n(t) = V_n \times$$

$$\left(1-\frac{1}{\kappa}\sum_{0\leq i<n}\exp\left(\frac{-s_{n,i}^{eq}}{\omega}\right)\left(1-\frac{y_{n,i}}{w}\right)\right)\implies$$

$$u_n\left(t\right)=\frac{V_n}{\kappa\omega}\sum_{0\leq i<n}\exp\left(\frac{-s_{n,i}^{eq}}{\omega}\right)y_{n,i}. \tag{4.34}$$

By replacing Eq. 4.31 in Eq. 4.34 and expanding $y_{n,i}$ according to its definition, we will get

$$\frac{dy_n}{dt}=\frac{V_{n-1}}{\kappa\omega}\sum_{0\leq i<j\leq n-1}\exp\left(\frac{-s_{n-1,i}^{eq}}{\omega}\right)y_j$$

$$-\frac{V_n}{\kappa\omega}\sum_{0\leq i<j\leq n}\exp\left(\frac{-s_{n,i}^{eq}}{\omega}\right)y_j. \tag{4.35}$$

We can write Eq. 4.35 for

$$1\leq n\leq N-1$$

for all vehicles in a matrix form as

$$\frac{dY}{dt}=AY. \tag{4.36}$$

By inspection, A is a lower triangular matrix with only negative elements. Since the eigenvalues of a lower triangular matrix are the elements of the diagonal, all the eigenvalues of the matrix are negative. Therefore, according to the linear stability theory, variables y_i are stable with an equilibrium point of all 0s. Hence, a similar thing can be said about u_i. To understand the rate of convergence, I calculate the eigenvalues which are the elements of the diagonal of A. In other words, the eigenvalues λ_n are the coefficients of y_n in Eq. 4.35. By inspection, we have

$$\lambda_n=\frac{-V_n}{\kappa\omega}\sum_{0\leq i<n}\exp\left(\frac{-s_{n,i}^{eq}}{\omega}\right)=\frac{V_0-V_n}{\omega} \tag{4.37}$$

where the last equality is due to Eq. 4.28 and knowing $v^{eq}=V_0$. Therefore, the bigger the difference between V_0 and V_n, the faster the convergence will be to the equilibrium point. The above proved the asymptotic linear stability of the model. Linear (platoon) stability is proven by considering the special case where there is 0 initial perturbation to the position and velocity of vehicles

$$2\leq i\leq N-1$$

and accordingly

$$y_i(t=0)=0,$$

$$u_i(t=0) = 0.$$

□

4.4.5 Asymptotic behavior in blocking regime: nonlinear stability analysis

In this section, I perform the non-linear stability analysis for vehicles placed on a ring road. Note that Eq. 4.2 and Eq. 4.3 are adjusted accordingly to become symmetrical for any vehicle i (i.e. now each vehicle regardless of their numbering is a follower to every other vehicle on the ring road).

Equilibrium point for the ring road

Given a fleet of identical vehicles with maximum velocity V_{max} travelling on a ring road, the exact locations of each vehicle is not important to us. However, their relative distance is important. I take the set of velocities v_i as my state variables. Since the motion equations for all vehicles are symmetrical on the ring road, an immediately obvious equilibrium point is the case where all velocities are identical. This is equivalent to saying that all gaps are identical; that is

$$s^{eq} = s_i = \frac{L}{N} \tag{4.38}$$

where L is the circumference of the ring road and N is the number of vehicles. We define the overall density ρ as

$$\rho = \frac{1}{s^{eq}}.$$

The equilibrium velocity is calculated as follows:

$$v^{eq} = V_{max}\left(1 - \frac{1}{\kappa}\sum_{1\leq j\leq N-1}\exp\left(\frac{-js^{eq}}{\omega}\right)\right) =$$
$$V_{max}\left(1 + \frac{1}{\kappa} - \frac{1}{\kappa}\sum_{0\leq j\leq N-1}\exp\left(\frac{-js^{eq}}{\omega}\right)\right). \tag{4.39}$$

Using the identity for the sum of the geometric series, we obtain

$$v^{eq} = V_{max}\left(1 + \frac{1}{\kappa} - \frac{1}{\kappa}\cdot\frac{1-\exp\left(\frac{-Ns^{eq}}{\omega}\right)}{1-\exp\left(\frac{-s^{eq}}{\omega}\right)}\right) =$$

$$v^{eq} = V_{max}\left(1 + \frac{1}{\kappa} - \frac{1}{\kappa}\cdot\frac{1-\exp\left(\frac{-L}{\omega}\right)}{1-\exp\left(\frac{-1}{\rho\omega}\right)}\right). \tag{4.40}$$

Vehicle flow, velocity, and density are related by

$$Q = v^{eq}\rho$$

which results in the diagram in Fig. 4.4 relating the traffic flow Q to the density ρ. This graph is also one of the fundamental diagrams of a traffic flow model and it gives insight into the macroscopic behavior of my microscopic model. It also gives an intuitive justification for the soundness of my model, since all traffic flow models (including those cited in this work) produce a more or less similar graph. That is the traffic flow increases as the density increases till we reach a peak capacity after which adding any more vehicles will only serve to decrease the flow.

Numerical experiments

In this section, I initiate the system with a variety of conditions and observe whether the system will approach to the equilibrium point. I will use a chosen background density composed with a smaller region of higher density. I will observe how this irregularity will affect the system's stability. My experiments parameters are chosen as follows:

- $L = 1000m$, length of the ring road (m).
- $\omega = 10m$, length of the horizon in front of each vehicle.
- $V_{max} = 6m/s$, for all vehicles.
- $\kappa = 10$, model's capacity.
- $t_{start} = 0s, t_{end} = 500s$, start and finish time of simulation.
- $\rho = 0.5\frac{veh}{m}$, global density of vehicles. In other words, we have 500 vehicles on the ring road (a minimum number of vehicles that is required for a realistic simulation [88]). Note that this does not necessarily translate into an unreasonably high density of vehicles since the vehicles are not directly behind each other in the 3D space. In other words, they can be placed anywhere on the vertical plane that belongs to them.

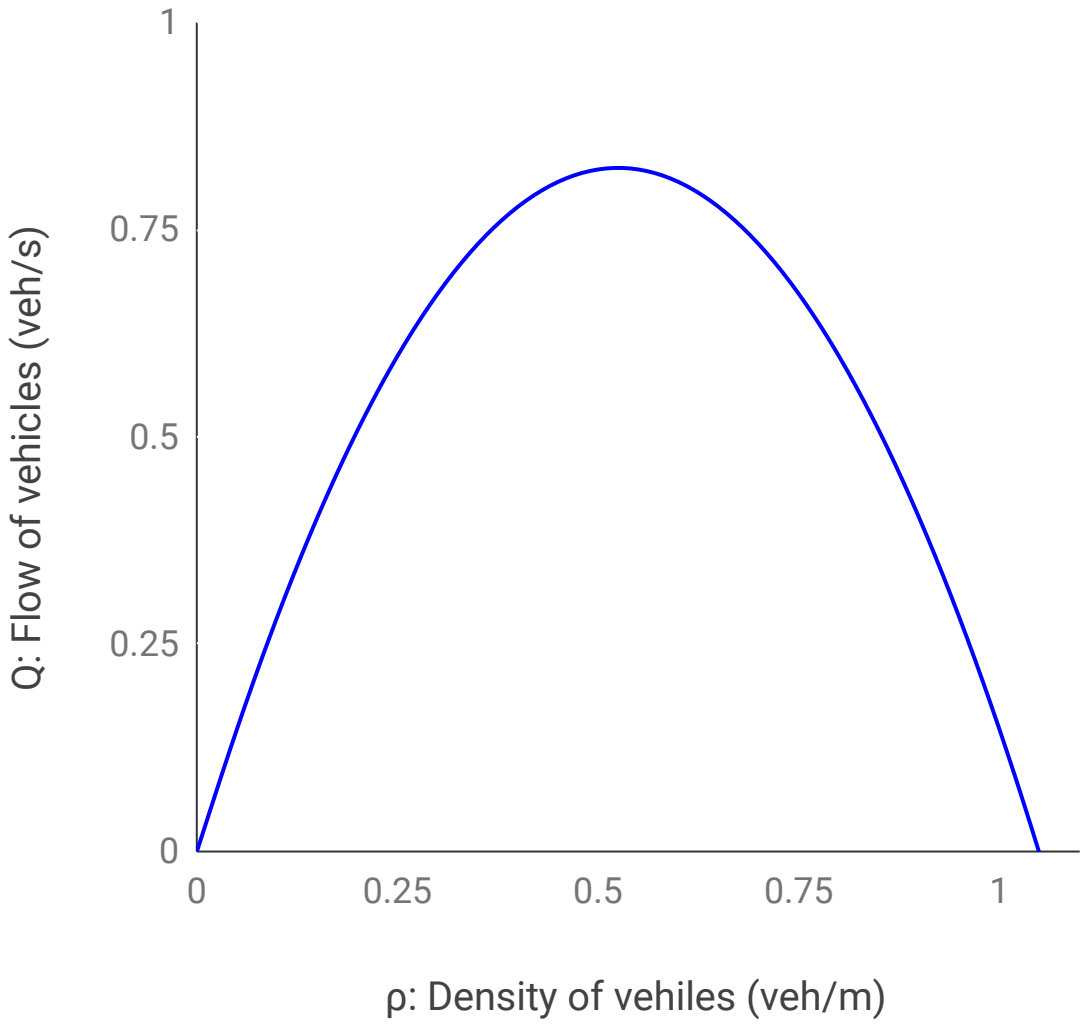


Figure 4.4: Vehicle flow versus density: For my microscopic model, this graph shows the macroscopic relationship between the number of identical vehicles passing a fixed point on a ring road per unit of time and the density of vehicles. As is expected from a traffic flow model, after a peak density matching the available capacity is reached, traffic flow starts deteriorating in the sense that any more vehicles only serves to slow down every vehicle. Before this peak, the traffic is in the free flow regime and then switches to congested.

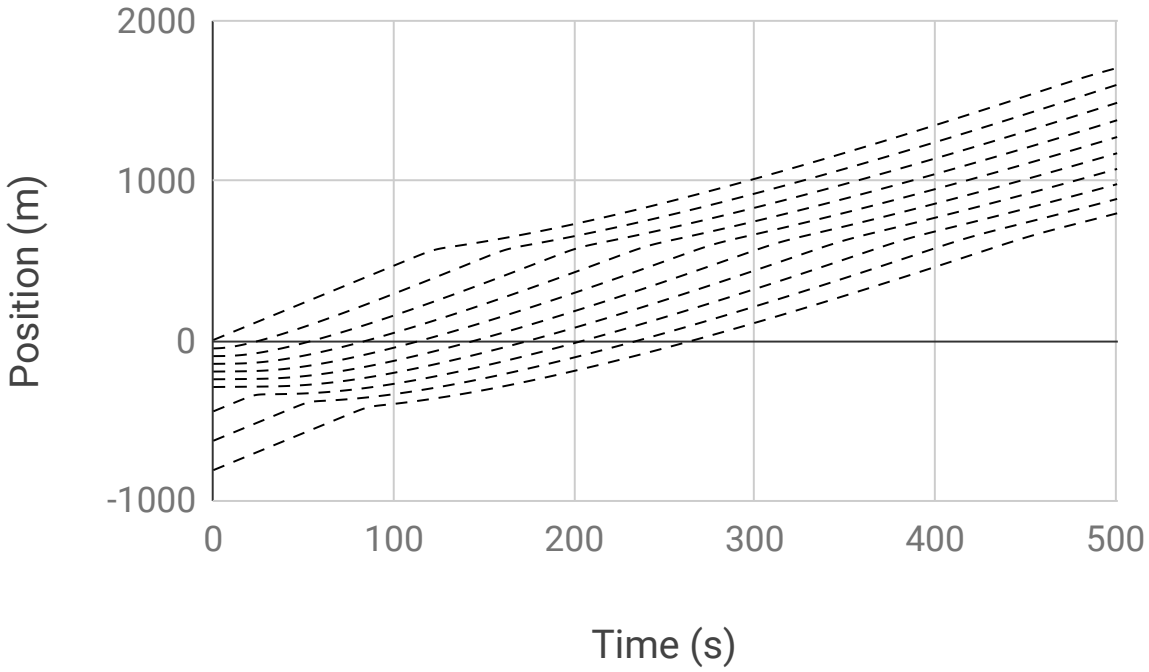


Figure 4.5: Time-Space diagram for every 50'th vehicles: 30% of the ring road has a maximal uniform vehicle density of $1.03\frac{veh}{m}$ and the remaining 70% has a uniform density of $0.27\frac{veh}{m}$. The ring road vehicle density is $0.5\frac{veh}{m}$. Initially, some vehicles are slowed down, but as time goes on, all the velocities converge to the equilibrium velocity.

To produce Fig. 4.5 and 4.6 we distribute the vehicles in two regions. One region consists of 30% of the ring road and has the highest possible uniform density of the vehicles and the remaining vehicles are distributed in the rest of the ring road evenly; so to make the overall density ρ as above. These experiments provide evidence that no matter how far from equilibrium the system is, it will converge to the equilibrium. Also, I performed the same test with the same number of vehicles when 10% or 20% of the ring road had a maximal traffic jam and each case produced essentially the same graphs.

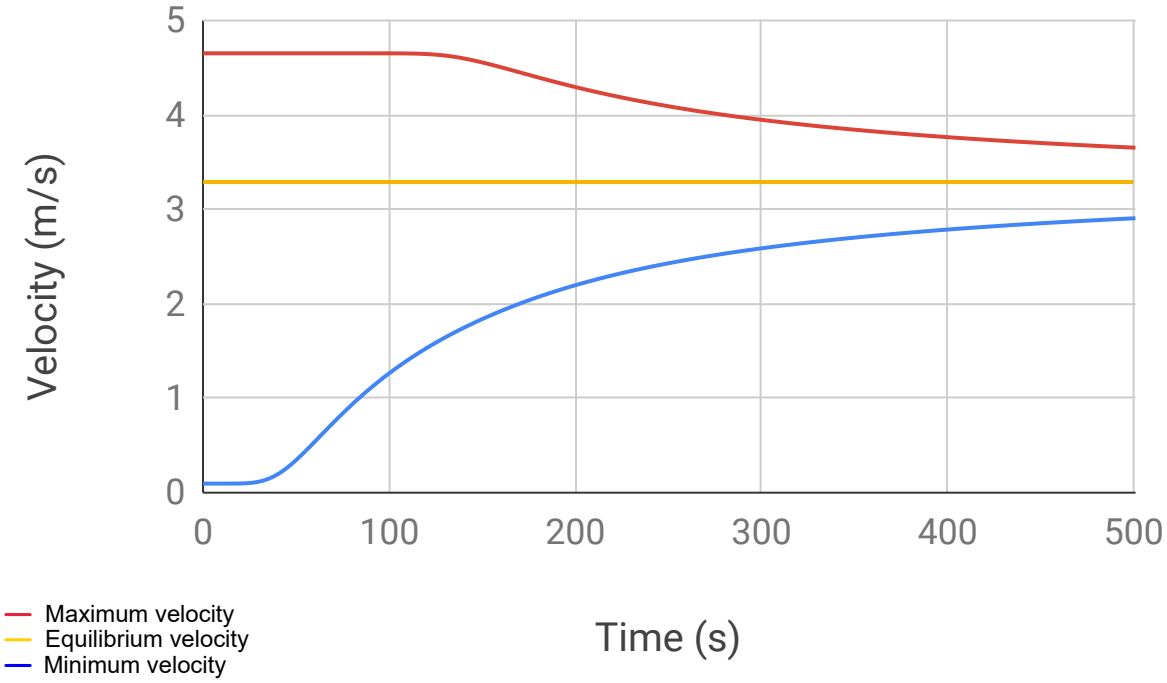


Figure 4.6: Minimum and maximum momentary velocity among all vehicles: 30% of the ring road has a maximal uniform vehicle density of $1.03\frac{veh}{m}$ and the remaining 70% has a uniform density of $0.27\frac{veh}{m}$. The ring road vehicle density is $0.5\frac{veh}{m}$. This graph shows the convergence to equilibrium velocity.

Chapter 5

Conclusion

5.1 Concluding remarks

Many drone applications can benefit from a unified framework that coordinates their access to the airspace and helps them navigate to the points of interest where they have to perform a task. Any architecture poised to provide this service must be scalable and be able to provide it to thousands of drones, which will share the congested and limited urban airspace.

In Chapter 2, I laid out the conceptual foundation for such an architecture by developing a vocabulary of concepts for describing the architecture and identifying the relevant components of it as well as deciding on the boundaries of the architecture. Furthermore, I designed a structure for the airspace and provided strategies for utilizing that structure in the airspace. My design makes it possible to provide generic services that can be used by many applications. To effectively tackle the problem of "how to enable drones to perform tasks", I divided the overall required functionality of the architecture into logical layers. The main sub-problem was the airspace navigation and coordination for various applications as addressed in the first three layers of IoD. I addressed other common services that are needed by the applications such as location aware communication in an extensible service layer. In IoD architecture, I described the features that are required to be implemented in each of these layers by IoD systems. Furthermore, I suggested an operation model that identifies the role of private and public organizations in the governance of IoD. Additionally, I explored and discussed some of the difficulties that have to be addressed for an effective IoD system. In all of this, I used and referred to the wealth of knowledge acquired from three large scale networks, the cellular network, the air traffic control, and the Internet. Finally, I discussed the differences and future works that can benefit from the solutions from the vast existing literature on these three subjects.

In Chapter 3, I introduced a new scheduling problem called Vehicle Scheduling Problem. Given a path between a pair of source and destination for each vehicle over a graph, the goal is to minimize some objective function such as the number of tardy vehicles (i.e. missed deadlines) subject to various constraints as follows. These include, maintaining a safety time gap at conflicting nodes and meeting hard deadlines after trips are requested by vehicles. Furthermore, each vehicle is required to maintain its speed in an allowable range over any link. I established the NP-hardness of VSP for all commonly used objective functions in the context of JSP. Then, I formulated this problem in terms of an MIP where the chosen objective function is the number of tardy vehicles. For the case of simultaneous trip requests, I then devised a heuristic algorithm based on giving priority to vehicles closer to an intersection and with less slack time left. I also devised a baseline algorithm that mimics to some extent the real world traffic, i.e. vehicles closer to an intersection will get

there first. I then performed numerical experiments on random instances of the problem over a grid like graph to compare the obtained objective value from the exact solution to the MIP formulation as well as the baseline algorithm and my algorithm.

In Chapter 4, I introduced a microscopic traffic flow model that can be used to study traffic patterns of unmanned aerial vehicles in the air as they become ubiquitous in the future. The model is equally applicable to the study of the traffic flow of ground vehicles on the road. I advanced the state of art by introducing a scalar capacity parameter for the airway (or roads) rather than the traditional approach of modelling links as 1 lane or multi-lane. This is suited for the study of the 3D nature of UAV flights as opposed to the 2D nature of ground vehicles movements while also resulting in a simpler model for ground vehicles by abstracting away the pass planning aspect. By adjusting the scalar capacity parameter, the model can exhibit passing or blocking behaviors. In the former, vehicles are free to pass each other while in the latter, no vehicle can pass another one similar to a one lane road. My model can be solved analytically for the blocking regime and piece-wise analytically in the passing regime. For the blocking regime I proved linear local (platoon) stability as well as asymptotic linear stability. Also, using numerical simulation, I showed evidence for non-linear stability. For the passing regime, I proved theorems outlining the asymptotic behavior of the model such as whether every faster vehicle gets a chance to pass slower vehicles as time goes to infinity and what the final order of vehicles will be after all the overtakings are completed. Lastly, I proved a main theorem characterizing the transition from blocking to passing as we adjust the scalar capacity parameter.

Chapter 6

Discussion and Future Work

6.1 Discussion & Future work

My work in Chapter 2 opened up many exciting avenues for further research and these were discussed in length and depth in Chapter 2. A major question is what more needs to be done to use the research work done in Chapter 3 and 4 on vehicle scheduling methods and traffic flow model for them to be directly applicable in the context of IoD architecture. I have listed these issues as follows.

- In terms of scheduling UAVs, the work in Chapter 3 is focused on the stationary problem of UAVs scheduling. That is the trip request times are known ahead of time. However, in IoD, UAVs enter and leave the zones and the system at unknown times. Therefore, a first step to close the gap between the scheduling algorithms developed in Chapter 3 and the IoD is to create the online version of these algorithms and study their performance for that purpose. Luckily as explained in the following, it seems both the heuristic algorithm and the baseline algorithm can be adapted to the online version with some changes. Since the static version of both of these algorithms are based on deciding the right of way for vehicles in each intersection, i.e. a local operation, the arrival time of vehicles or the addition of new vehicles in the online version does not interfere with this core function. That is, for each intersection at each time, I rank vehicles for access to the intersection based only on their remaining time distance to the intersection and priorities informed by their deadlines. Hence, there exist no hurdles to consider new vehicles in this ranking routine as well.

- Another issue that needs attention with regard to bridging the gap between the work in Chapter 3 and IoD is the issue of noise. Any scheduling algorithm should allow for errors in arrival times to be of any practical use in IoD. The presence of noise from different sources including the variable wind gusts is an intrinsic property of any IoD system.

- Furthermore, the work in Chapter 3 is best suited for application in scheduling UAVs inside a zone rather than the entire system. That means a routing algorithm is needed to work at the inter-zone level. It is speculative to assume how the routing algorithm will work. However, one potential solution is to book a time window for both the arrival and departure gates for the UAVs. Therefore, the scheduling algorithm, must have the ability to accept or reject the proposed window by the routing algorithm and must integrate the newly accepted drone in its scheduling. Also, the routing algorithm will rely on the intelligence received from the scheduling algorithm to produce reasonable time windows for these gates. Thus, the needed interface and mechanisms must be provided by the scheduling algorithm.

- While the work in Chapter 3 is related to scheduling the arrival times at the nodes (an N2N layer issue), the work in Chapter 4 is related to UAVs flying through the links in a way that is stable and respects the link capacity (an Airspace layer issue). We need both these systems to work in tandem. That is the scheduler sets the target arrival time windows for each node for a UAV and the role of the traffic flow model is to meet those targets. The main inputs to the traffic flow model are the maximum velocity constants for each UAV. Therefore an important missing piece is a module that takes the scheduled times and translate them to appropriate values of maximum velocity constants periodically.

- Another issue that needs to be addressed with regard to work in Chapter 4 is that the speed of each drone is momentarily adjusted to the current state of traffic on the link. This is especially troublesome when a UAV enters a new link. While the congestion in the previous link might have been low and thus the UAV has been flying at high speed, suddenly it might face a high level of congestion and must reduce its speed sharply which might be physically impossible. Therefore mechanisms must be put in the place to smooth the transition between the links.

- Another use for the work in Chapter 4 in the context of IoD is traffic engineering of the airways capacities. Following the previous point, the traffic flow model might fail to meet the targets set by the scheduler, due to some links being persistently overloaded. We will need a module that will process the data about airways utilization rate and the success rate in meeting the arrival targets and translate them into decisions about adjusting the capacity of the airways.

6.2 Notes on Chapter 3

In Chapter 3, I assumed the route for each vehicle is fixed. There are scenarios where this might make sense for example when some UAV companies are allocated various paths in the airspace by the government and all the scheduling must be performed over these preallocated paths. However, in absence of these kind of limitations, an interesting problem to consider for future research is the joint optimization of routing and scheduling.

Another simplified aspect of the problem is the assumption that each link has unlimited capacity for holding vehicles. This is not entirely without merit as the in-flow and out-flow are bounded by various constraints. However, it might be plausible to set a direct capacity limit on a link, especially when the minimum permitted velocity on a link is 0.

With minor modifications, it is possible to expand the DEADLINE & PROXIMITY algorithm and the baseline PROXIMITY to the dynamic VSP to make it more applicable to the real world. It is interesting to see how these two algorithms will compare in that case.

In my numerical result section of Chapter 3, I demonstrated for some special cases the result from my heuristic algorithm. It will be interesting to explore more cases, such as when the trip request times are arbitrary and the minimum speed for vehicles can be more than 0.

A limitation of my work is that in my heuristics algorithm DEADLINE & PROXIMITY, the access rule is based on first the proximity of the vehicles to the intersections and only then the deadlines are used as a tie breaking mechanism. In my setup, the time stamps from the created schedules were all a multiple of an integer (in this case, 5), allowing the tie breaking rule a chance to have an influence. In practice, the tie breaking will likely never be used as the odds of having two vehicles at the exact same distance is close to 0. In other words, in presence of noise, etc., my heuristic algorithm degrades to the baseline algorithm. A simple fix is to use a window of a certain size inside which the deadline is the deciding rule. However, to determine the optimal window size will be the subject of a future work.

Another question with regard to both the baseline and the heuristic algorithms is how irregular lengths for the links will affect the quality of the created schedules. For example, a vehicle might be an intersection away from an intersection of interest, and only a very short distance away. But it will get a lower priority compared to a further away vehicle which is already on a connected link to the intersection. It is an interesting line of research to pursue expanding the pool of eligible vehicles in deciding the "right of way" to include vehicles such as the one in the example above.

Finally, there is an approximation algorithm that is used for Job Shop Scheduling known as Shifting Bottleneck as first appeared in the seminal work [1]. The goal was to minimize the makespan (completion time of the last job to finish). The algorithm works by sequencing each machine as a one machine optimization problem which can be solved efficiently. At any point, a list of sequenced machines and a list of unsequenced machines exist. Based on some criteria about which machine is the next greatest bottleneck, the unsequenced machines are ranked and the machine with the highest rank is chosen to be sequenced next. Based on the results, the already sequenced machines are resequenced one by one till no further improvements can be found in their schedule. Again, based on the outcome of this step, and according to the bottleneck criteria the next machine is chosen and the process is repeated till all the machines are sequenced. In this context, it is an

interesting venue for research to see if any algorithm with a similar idea of detecting the bottleneck vehicles or the bottleneck nodes or even a set of bottleneck neighboring nodes (similar to a zone) can be used to give better schedules.

6.3 Notes on Chapter 4

My work in Chapter 4 leaves many open questions. An important question is whether it is possible to add some mechanism for delay, without losing the closed form analytical solution feature of the model.

Another avenue for research is adding some dummy vehicles to play the role of moving or stationary obstacles for the traffic flow. This works by consuming the capacity of the airway/road dynamically. It organically gives rise to inclusion of obstacles without modifying the model. In the same vein, it is possible to add weights to the exponential congestion term for different vehicles or dummy vehicles (obstacles). Currently in my model, all these weights are equal. The ability to set weights can give us powerful tools for tuning the strength of these obstacles.

References

[1] Joseph Adams, Egon Balas, and Daniel Zawack. The shifting bottleneck procedure for job shop scheduling. *Management science*, 34(3):391–401, 1988.

[2] Bengt Ahlgren, Christian Dannewitz, Claudio Imbrenda, Dirk Kutscher, and Börje Ohlman. A survey of information-centric networking. *Communications Magazine, IEEE*, 50(7):26–36, 2012.

[3] Cyril Allignol, Nicolas Barnier, Pierre Flener, and Justin Pearson. Constraint programming for air traffic management: a survey 1: In memory of pascal brisset. *The Knowledge Engineering Review*, 27(3):361–392, 2012.

[4] Amazon.com Inc. Amazon prime air, 2013 (Accessed on 2020-1-24). URL: http://www.amazon.com/b?node=8037720011.

[5] Amazon.com Inc. Determining safe access with a best-equipped, best-served model for small unmanned aircraft systems, 2015 (Accessed on 2015-12-15). URL: http://utm.arc.nasa.gov/docs/Amazon_Determining%20Safe%20Access%20with%20a%20Best-Equipped,%20Best-Served%20Model%20for%20sUAS[2].pdf.

[6] Amazon.com Inc. Revising the airspace model for the safe integration of small unmanned aircraft systems, 2015 (Accessed on 2015-12-15). URL: http://utm.arc.nasa.gov/docs/Amazon_Revising%20the%20Airspace%20Model%20for%20the%20Safe%20Integration%20of%20sUAS[6].pdf.

[7] Masako Bando, Katsuya Hasebe, Akihiro Nakayama, Akihiro Shibata, and Yuki Sugiyama. Dynamical model of traffic congestion and numerical simulation. *Physical review E*, 51(2):1035, 1995.

[8] Anthony Battista and Daiheng Ni. Modeling small unmanned aircraft system traffic flow under external force. *Transportation Research Record*, 2626(1):74–84, 2017.

[9] Steven Michael Bellovin, David D Clark, Adrian Perrig, and Dawn Song. A clean-slate design for the next-generation secure internet, 2006 (Accessed on 2015-12-15). URL: http://groups.geni.net/geni/raw-attachment/wiki/OldGPGDesignDocuments/GDD-05-05.pdf.

[10] Randy Bush and David Meyer. Some Internet architectural guidelines and philosophy. RFC 3439, 2002.

[11] David Clark. The design philosophy of the DARPA Internet protocols. *ACM SIGCOMM Computer Communication Review*, 18(4):106–114, 1988.

[12] Michael B Cohen, Yin Tat Lee, and Zhao Song. Solving linear programs in the current matrix multiplication time. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, pages 938–942. ACM, 2019.

[13] Jean-Francois Cordeau, Paolo Toth, and Daniele Vigo. A survey of optimization models for train routing and scheduling. *Transportation science*, 32(4):380–404, 1998.

[14] Raffaello D'Andrea. Guest editorial can drones deliver? *Automation Science and Engineering, IEEE Transactions on*, 11(3):647–648, 2014.

[15] Santosh Devasia and Alexander Lee. A scalable low-cost-uav traffic network (unet). *arXiv:1601.01952v2*, 2016.

[16] DHL Express. DHL parcelcopter launches initial operations for research purposes, 2014 (Accessed on 2020-1-24). URL: http://www.dhl.com/en/press/releases/releases_2014/group/dhl_parcelcopter_launches_initial_operations_for_research_purposes.html.

[17] Andrea D'ariano, Dario Pacciarelli, and Marco Pranzo. A branch and bound algorithm for scheduling trains in a railway network. *European Journal of Operational Research*, 183(2):643–657, 2007.

[18] Nils Eissfeldt and Peter Wagner. Effects of anticipatory driving in a traffic flow model. *The European Physical Journal B-Condensed Matter and Complex Systems*, 33(1):121–129, 2003.

[19] FAA. Introduction to TCAS II version 7.1, 2011 (Accessed on 2015-12-15). URL: http://www.faa.gov/documentLibrary/media/Advisory_Circular/TCAS%20II%20V7.1%20Intro%20booklet.pdf.

[20] FAA. Unmanned aircraft systems aviation rulemaking committee, 2011 (Accessed on 2015-12-15). URL: http://www.faa.gov/regulations_policies/rulemaking/committees/documents/media/UASARC-20110617.PDF.

[21] FAA. Integration of Civil Unmanned Aircraft Systems (UAS) in the National Airspace System (NAS) roadmap, 2013 (Accessed on 2015-12-15). URL: http://www.faa.gov/uas/media/UAS_Roadmap_2013.pdf.

[22] FAA. Unmanned Aircraft Systems (UAS) comprehensive plan: A report on the nation's UAS path forward, 2013 (Accessed on 2015-12-15). URL: http://www.faa.gov/about/office_org/headquarters_offices/agi/reports/media/UAS_Comprehensive_Plan.pdf.

[23] FAA. Model aircraft operations, 2015 (Accessed on 2015-12-15). URL: https://www.faa.gov/uas/model_aircraft/.

[24] FAA. Operation and certification of small unmanned aircraft systems, 2015 (Accessed on 2015-12-15). URL: http://www.faa.gov/regulations_policies/rulemaking/recently_published/media/2120-AJ60_NPRM_2-15-2015_joint_signature.pdf.

[25] FAA. Registration and marking requirements for small unmanned aircraft, 2015 (Accessed on 2015-12-15). URL: https://federalregister.gov/a/2015-31750.

[26] Wei Fang, Shengxiang Yang, and Xin Yao. A survey on problem models and solution approaches to rescheduling in railway networks. *IEEE Transactions on Intelligent Transportation Systems*, 16(6):2997–3016, 2015.

[27] FedEx Corporation. Q1 fiscal 2015 statistics, 2015 (Accessed on 2020-1-24). URL: http://investors.fedex.com/files/doc_downloads/statistical/FedEx-Q1-FY15-Stat-Book_v001_t195uu.pdf.

[28] Anja Feldmann. Internet clean-slate design: what and why? *ACM SIGCOMM Computer Communication Review*, 37(3):59–64, 2007.

[29] Majid Ghaderi and Raouf Boutaba. Call admission control in mobile cellular networks: a comprehensive survey. *Wireless communications and mobile computing*, 6(1):69–94, 2006.

[30] Mirmojtaba Gharibi, Raouf Boutaba, and Steven L Waslander. Internet of drones. *IEEE Access*, 4:1148–1162, 2016.

[31] Mirmojtaba Gharibi, Raouf Boutaba, and Steven L. Waslander. Internet of drones. *arXiv:1601.01289*, 2016.

[32] Peter G Gipps. A behavioural car-following model for computer simulation. *Transportation Research Part B: Methodological*, 15(2):105–111, 1981.

[33] Amadou Gning, Lyudmila Mihaylova, and René K Boel. Interval macroscopic models for traffic networks. *IEEE Transactions on Intelligent Transportation Systems*, 12(2):523–536, 2011.

[34] Andrea Goldsmith. *Wireless communications*. Cambridge university press, Cambridge, United Kingdom, 2005.

[35] Google Inc. Google UAS airspace system overview, 2015 (Accessed on 2015-12-15). URL: http://utm.arc.nasa.gov/docs/GoogleUASAirspaceSystemOverview5pager[1].pdf.

[36] BD Greenshields, Ws Channing, Hh Miller, et al. A study of traffic capacity. In *Highway research board proceedings*, volume 1935. National Research Council (USA), Highway Research Board, 1935.

[37] Doug Gross. Amazon's drone delivery: How would it work?, 2013 (Accessed on 2020-1-24). URL: http://www.cnn.com/2013/12/02/tech/innovation/amazon-drones-questions/.

[38] K Hasebe, A Nakayama, and Y Sugiyama. Exact solutions of differential equations with delay for dissipative systems. *Physics Letters A*, 259(2):135–139, 1999.

[39] Johann Hurink and Sigrid Knust. Tabu search algorithms for job-shop problems with a single transport robot. *European journal of operational research*, 162(1):99–111, 2005.

[40] Dae-Sung Jang, Corey A Ippolito, Shankar Sankararaman, and Vahram Stepanyan. Concepts of airspace structures and system analysis for uas traffic flows for urban areas. In *AIAA Information Systems-AIAA Infotech@ Aerospace*, page 0449. 2017.

[41] Rui Jiang, Qingsong Wu, and Zuojin Zhu. Full velocity difference model for a car-following theory. *Physical Review E*, 64(1):017101, 2001.

[42] Arne Kesting, Martin Treiber, and Dirk Helbing. General lane-changing model mobil for car-following models. *Transportation Research Record*, 1999(1):86–94, 2007.

[43] Maurice J Khabbaz, Wissam F Fawaz, and Chadi M Assi. A simple free-flow traffic model for vehicular intermittently connected networks. *IEEE Transactions on Intelligent Transportation Systems*, 13(3):1312–1326, 2012.

[44] Vyacheslav Kharchenko and Volodymyr Torianyk. Cybersecurity of the internet of drones: Vulnerabilities analysis and imeca based assessment. In *2018 IEEE 9th International Conference on Dependable Systems, Services and Technologies (DESSERT)*, pages 364–369. IEEE, 2018.

[45] Charles J Kim. TMA integrated metrics assessment model, 2006 (Accessed on 2015-12-15). URL: http://www.tc.faa.gov/LOGISTICS/GRANTS/pdf/2004/04-G-044.pdf/FINAL%20REPORT%2004-G-044.pdf.

[46] Sigrid Knust. Shop-scheduling problems with transportation. 2000.

[47] Parimal Kopardekar. Safely enabling UAS operations in low-altitude airspace, 2015 (Accessed on 2020-1-24). URL: http://utm.arc.nasa.gov/docs/pk-final-utm2015.pdf.

[48] Anis Koubâa, Basit Qureshi, Mohamed-Foued Sriti, Yasir Javed, and Eduardo Tovar. A service-oriented cloud-based management system for the internet-of-drones. In *2017 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*, pages 329–335. IEEE, 2017.

[49] Svetlana A Kravchenko. Minimizing the number of late jobs for the two-machine unit-time job-shop scheduling problem. *Discrete Applied Mathematics*, 98(3):209–217, 2000.

[50] Wieslaw Kubiak and Vadim Timkovsky. Total completion time minimization in two-machine job shops with unit-time operations. *European journal of operational research*, 94(2):310–320, 1996.

[51] Pushpendra Kumar, Rochdi Merzouki, Blaise Conrard, Vincent Coelen, and Belkacem Ould Bouamama. Multilevel modeling of the traffic dynamic. *IEEE Transactions on Intelligent Transportation Systems*, 15(3):1066–1082, 2014.

[52] Jorge A Laval and Carlos F Daganzo. Lane-changing in traffic streams. *Transportation Research Part B: Methodological*, 40(3):251–264, 2006.

[53] Jorge A Laval and Ludovic Leclercq. Microscopic modeling of the relaxation phenomenon using a macroscopic lane-changing model. *Transportation Research Part B: Methodological*, 42(6):511–522, 2008.

[54] Jan K Lenstra and AHG Rinnooy Kan. Computational complexity of discrete optimization problems. In *Annals of Discrete Mathematics*, volume 4, pages 121–140. Elsevier, 1979.

[55] H Lenz, CK Wagner, and R Sollacher. Multi-anticipative car-following model. *The European Physical Journal B-Condensed Matter and Complex Systems*, 7(2):331–335, 1999.

[56] David Levine, Ian F Akyildiz, Mahmoud Naghshineh, et al. A resource estimation and call admission algorithm for wireless multimedia networks using the shadow cluster concept. *Networking, IEEE/ACM Transactions on*, 5(1):1–12, 1997.

[57] Kun Li and Petros Ioannou. Modeling of traffic flow of automated vehicles. *IEEE Transactions on Intelligent Transportation Systems*, 5(2):99–113, 2004.

[58] Zhaojian Li, Firas Khasawneh, Xiang Yin, Aoxue Li, and Ziyou Song. A new microscopic traffic model using a spring-mass-damper-clutch system. *IEEE Transactions on Intelligent Transportation Systems*, 2019.

[59] Michael James Lighthill and Gerald Beresford Whitham. On kinematic waves ii. a theory of traffic flow on long crowded roads. *Proceedings of the Royal Society of London. Series A. Mathematical and Physical Sciences*, 229(1178):317–345, 1955.

[60] Mahmoud Naghshineh and Mischa Schwartz. Distributed call admission control in mobile/wireless networks. *Selected Areas in Communications, IEEE Journal on*, 14(4):711–717, 1996.

[61] NASA. UTM fact sheet, 2015 (Accessed on 2015-12-15). URL: http://utm.arc.nasa.gov/docs/UTM-Fact-Sheet.pdf.

[62] NASA. NASA UTM 2015: The next era of aviation, 2015 (Accessed on 2020-1-24). URL: http://utm.arc.nasa.gov/utm2015.shtml.

[63] Gordon Frank Newell. Nonlinear effects in the dynamics of car following. *Operations research*, 9(2):209–229, 1961.

[64] Gordon Frank Newell. A simplified car-following theory: a lower order model. *Transportation Research Part B: Methodological*, 36(3):195–205, 2002.

[65] Michael Nolan. *Fundamentals of air traffic control*. Delmar Cengage Learning, Clifton Park, NY, 5 edition, 2010.

[66] House of Representatives. FAA modernization and reform act of 2012, 2012 (Accessed on 2015-12-15). URL: http://www.gpo.gov/fdsys/pkg/CRPT-112hrpt381/pdf/CRPT-112hrpt381.pdf.

[67] Office of the Federal Register. A guide to the rulemaking process, 2011 (Accessed on 2015-12-15). URL: https://www.federalregister.gov/uploads/2011/01/the_rulemaking_process.pdf.

[68] Raif O. Onvural. *Asynchronous transfer mode networks: performance issues*. Artech House, Norwood, MA, 2 edition, 1994.

[69] Larry L Peterson and Bruce S Davie. *Computer networks: a systems approach*. Elsevier, 2007.

[70] Larry L. Peterson and Bruce S. Davie. *Computer Networks: A Systems Approach*. Morgan Kaufmann, Burlington, MA, 5 edition, 2011.

[71] Michael L. Pinedo. *Scheduling: Theory, Algorithms, and Systems*. Springer, 2016.

[72] Louis A Pipes. An operational analysis of traffic dynamics. *Journal of applied physics*, 24(3):274–281, 1953.

[73] Kadangode Ramakrishnan and Sally Floyd. A proposal to add explicit congestion notification (ECN) to IP. RFC 2481, 1999.

[74] Theodore S Rappaport et al. *Wireless communications: principles and practice*. Prentice Hall, Upper Saddle River, NJ, 2 edition, 2002.

[75] Andreas Raptopoulos. No roads? there is a drone for that, 2013 (Accessed on 2020-1-24). URL: https://www.ted.com/talks/andreas_raptopoulos_no_roads_there_s_a_drone_for_that.

[76] Av Reuschel. Fahrzeugbewegungen in der kolonne. *Osterreichisches Ingenieur Archiv*, 4:193–215, 1950.

[77] Paul I Richards. Shock waves on the highway. *Operations research*, 4(1):42–51, 1956.

[78] James Roberts. The clean-slate approach to future Internet design: a survey of research initiatives. *Annals of telecommunications*, 64(5):271–276, 2009.

[79] Johannes MJ Schutten. Practical job shop scheduling. *Annals of Operations Research*, 83:161–178, 1998.

[80] Scott Shenker and John Wroclawski. General characterization parameters for integrated service network elements. RFC 2215, 1997.

[81] Mihail L Sichitiu and Maria Kihl. Inter-vehicle communication systems: a survey. *Communications Surveys & Tutorials, IEEE*, 10(2):88–105, 2008.

[82] Abraham Silberschatz, Greg Gagne, and Peter B Galvin. *Operating system concepts*. Wiley, 2018.

[83] William Stallings. *Wireless communications & networks*. Pearson Prentice Hall, New York City, NY, 2 edition, 2005.

[84] Jack Stewart. Google tests drone deliveries in project wing trials, 2014 (Accessed on 2020-1-24). URL: http://www.bbc.com/news/technology-28964260.

[85] Vadim G Timkovsky. Is a unit-time job shop not easier than identical parallel machines? *Discrete applied mathematics*, 85(2):149–162, 1998.

[86] Tomer Toledo, Haris N Koutsopoulos, and Moshe E Ben-Akiva. Modeling integrated lane-changing behavior. *Transportation Research Record*, 1857(1):30–38, 2003.

[87] Martin Treiber, Ansgar Hennecke, and Dirk Helbing. Congested traffic states in empirical observations and microscopic simulations. *Physical review E*, 62(2):1805, 2000.

[88] Martin Treiber and Arne Kesting. Traffic flow dynamics. *Traffic Flow Dynamics: Data, Models and Simulation, Springer-Verlag Berlin Heidelberg*, 2013.

[89] Martin Treiber, Arne Kesting, and Dirk Helbing. Delays, inaccuracies and anticipation in microscopic traffic models. *Physica A: Statistical Mechanics and its Applications*, 360(1):71–88, 2006.

[90] US Government publishing office. Automatic dependent surveillance broadcast (ADS-B) out performance requirements to support air traffic control (ATC) service, 2010 (Accessed on 2015-12-15). URL: http://www.gpo.gov/fdsys/pkg/FR-2010-05-28/pdf/2010-12645.pdf.

[91] Mohammad Wazid, Ashok Kumar Das, and Jong-Hyouk Lee. Authentication protocols for the internet of drones: taxonomy, analysis and future directions. *Journal of Ambient Intelligence and Humanized Computing*, pages 1–10, 2018.

[92] Gerald Beresford Whitham. Exact solutions for a discrete system arising in traffic flow. *Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences*, 428(1874):49–69, 1990.

[93] R Eddie Wilson and Jonathan A Ward. Car-following models: fifty years of linear stability analysis–a mathematical perspective. *Transportation Planning and Technology*, 34(1):3–18, 2011.

[94] George Xylomenos, Christopher N Ververidis, Vasilios Siris, Nikos Fotiou, Christos Tsilopoulos, Xenofon Vasilakos, Konstantinos V Katsaros, George C Polyzos, et al. A survey of information-centric networking research. *Communications Surveys & Tutorials, IEEE*, 16(2):1024–1049, 2014.

[95] Jingjing Yao and Nirwan Ansari. Qos-aware rechargeable uav trajectory optimization for sensing service. In *ICC 2019-2019 IEEE International Conference on Communications (ICC)*, pages 1–6. IEEE, 2019.

APPENDICES

Appendix A

Helpful lemmas for chapter 3

A.1 Lemmas and their proofs

Lemma 8 (Passing threshold)**.** *Given only two vehicles on the road with*

$$V_1 > V_0,$$

given enough time, vehicle 1 *will pass vehicle* 0 *if and only if*

$$\kappa > \frac{V_1}{V_1 - V_0}.$$

Proof.
Proving vehicle 1 passes vehicle 0 implies

$$\kappa > \frac{V_1}{V_1 - V_0}$$

is a straightforward consequence of theorem 4 since

$$\Gamma_0 = 0$$

at all times including the passing time between vehicles 0 and vehicle 1.

In the other direction, I prove

$$\kappa > \frac{V_1}{V_1 - V_0}$$

implies for all times, that

$$v_1(t) > v_0(t).$$

I first prove the two vehicles will meet as a condition for theorem 4, so we can apply that theorem.

Variable v_1 takes its minimum value $v_{1_{min}}$ when vehicle 0 and 1 are (hypothetically) in the same position, that is $x_1 = x_0$. Therefore, a pass will occur in that case since we will have

$$\kappa > \frac{V_1}{V_1 - V_0}$$

and can use theorem 4,

Therefore, at all other times,

$$v_1 \geq v_{1_{min}} > v_0(t) = V_0$$

.

This implies that there exists a time t_p when the two vehicles will meet, or in other words they are in the same position

$$x_0(t_p) = x_1(t_p).$$

Therefore, according to theorem 4, vehicle 1 will pass vehicle 0.

□

Without loss of generality, assume vehicle n is the first vehicle for which

$$V_n = V_0.$$

In the following lemma, I show if a vehicle n in a sequence of vehicles following a leader with speed V_0 has a maximum speed

$$V_n = V_0,$$

then this will result in creation of two platoons. When

$$V_n < V_0,$$

this is easy to see. But when the maximum speeds are equal, one can see that this still holds. More formally:

Lemma 9 (Platoon splitting)**.** *Given a platoon of n vehicles with stable orders with an extra vehicle n with*

$$V_n = V_0$$

and where vehicle 0 to $n-1$ are in equilibrium, if

$$V_0 < V_j$$

for

$$1 \leq j \leq n-1,$$

then vehicle n is not part of the platoon.

Proof.
From Eq. 4.11 and knowing

$$V_0 < V_j$$

for

$$1 \leq j \leq n-1$$

and $V_0 = V_n$, we have

$$z_n = (c_{n,0,0} + c_{n,0,1} \cdot t) \exp\left(\frac{-V_0 t}{\omega}\right) + \sum_{j \in U - \{0,n\}} \sum_{0 \leq d < m_{n,j}} c_{n,j,d} \cdot t^d \exp\left(\frac{-V_j t}{\omega}\right). \tag{A.1}$$

From the definition of z_0 in Eq. 4.4 we have

$$z_0 = \exp\left(\frac{-x_0}{\omega}\right) = \exp\left(\frac{-x_0(0) - V_0 t}{\omega}\right). \tag{A.2}$$

Now, by using Eq. A.1 and Eq. A.2, we get

$$\lim_{t \to \infty} \exp\left(\frac{x_0 - x_n}{\omega}\right) = \lim_{t \to \infty} \frac{z_n}{z_0} = \infty \implies$$

$$\lim_{t \to \infty} (x_0 - x_n) = \infty \tag{A.3}$$

where the first equality is due to the definition of z_i from Eq. 4.4.

Therefore vehicle n cannot be part of the same platoon of vehicles 0 to $n-1$ since the distance between vehicle 0 and vehicle n will increase with no bound.

□