Centralized and Decentralized Approaches to Advanced Air Mobility Traffic Management

by

Christopher H. Chin

B.S. Civil Engineering, University of California, Berkeley, 2018
M.S. in Aeronautics and Astronautics, Massachusetts Institute of Technology, 2021

Submitted to the Department of Aeronautics and Astronautics
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY IN AERONAUTICS AND ASTRONAUTICS

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2024

Authored by:	Christopher H. Chin Department of Aeronautics and Astronautics May 17, 2024
Certified by:	Hamsa Balakrishnan William E. Leonhard (1940) Professor of Aeronautics and Astronautics
Certified by:	Amedeo Odoni T.Wilson Professor Emeritus of Aeronautics and Astronautics
Certified by:	Cynthia Barnhart Provost and Siegel Professor of Management Science and Professor of Operations Research
Accepted by:	Jonathan How R.C. Maclaurin Professor of Aeronautics and Astronautics Chair, Graduate Program

Centralized and Decentralized Approaches to Advanced Air Mobility Traffic Management

by

Christopher H. Chin

Submitted to the Department of Aeronautics and Astronautics
on May 17, 2024 in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY IN AERONAUTICS AND ASTRONAUTICS

ABSTRACT

Advanced air mobility (AAM) is an emerging air transportation concept that leverages new types of aircraft, such as electric vertical take-off and landing (eVTOL) aircraft, to carry passengers or cargo in urban or rural settings. Initial AAM operations will be low-volume, so traffic can be managed with existing air traffic control rules, procedures, and designated routes. However, projected levels of AAM demand will require new traffic management procedures. We focus on three key considerations of AAM traffic management. First, we desire a traffic management system that efficiently utilizes limited airspace and vertiport resources and minimizes delays. Next, given the diverse set of possible AAM applications, we desire a system that is fair across different operators and flights. Finally, we must be cognizant of the level of information sharing required from operators, as AAM operators can have preferences on when, how much, and with whom information is shared.

Current concepts of operations envision a federated architecture in which multiple third-party service suppliers manage AAM traffic rather than regulatory agencies like the Federal Aviation Administration. We first consider how a single service supplier can manage traffic. To maximize efficiency, we start with a centralized optimization that requires operators to share full trajectory information. We consider the trade-off between efficiency and alternative fairness metrics, study the impact of operator preferences for fairness, and evaluate how to handle dynamic demand. We then turn to a decentralized setting since AAM operators may be unwilling or unable to share information with a central traffic manager. We develop a decentralized traffic management protocol that requires less information sharing. We show that the protocol with backpressure prioritization maximizes efficiency in one timestep, even with limited information sharing. We then consider federated airspace configurations where different regions of airspace can utilize different traffic management methods. We show that ideas from the decentralized protocol can be leveraged to coordinate traffic in a federated setting. The methods developed in this dissertation can help service suppliers manage AAM traffic while directly addressing many considerations of AAM operations, like operator fairness and information sharing.

Thesis Supervisor: Hamsa Balakrishnan
Title: William E. Leonhard (1940) Professor of Aeronautics and Astronautics

Acknowledgments

I would like to first express my deep gratitude to Prof. Hamsa Balakrishnan who has been an outstanding advisor for the last five years. Hamsa has opened many doors for me, including the opportunity to work at NASA Ames before starting at MIT. She gave me the opportunity to collaborate with Airbus, the Air Force, JAXA, and other universities on several topics related to air transportation, scheduling, and optimization. Most importantly, she has unfailingly supported me and her other students. It has been a pleasure working with her, as her research insights and support have defined my time at MIT. I would like to thank Prof. Amedeo Odoni for providing crucial advice and comments on this thesis and my research. He provided encouragement at crucial moments and precise feedback when I needed guidance. In addition, his airport systems class was one of the most interesting ones I took. Amedeo's contributions to air transportation, spanning air traffic management and airport planning, are legendary, and I remain inspired by his work. I would like to thank Prof. Cynthia Barnhart for agreeing to serve on my committee despite her extremely busy schedule. She helped me solidify the takeaways and contributions of my thesis. Cindy's contributions to airline scheduling, operations, and large-scale optimization are immense. She encouraged me to think critically about what could be applied from commercial aviation to advanced air mobility and what characteristics make advanced air mobility unique and worth studying.

I want to thank my two readers: Husni Idris and Antony Evans. Husni was my supervisor at NASA Ames and always advocated for more projects and opportunities for me, even when I was just an intern. From securing funding for me to attend the 2019 ATM Seminar in Vienna to introducing me to several of his projects, I knew that Husni was always fully supportive of me. I look forward to hearing about Husni's continued contributions and success. I also had the pleasure of working with Tony while at NASA Ames. He selflessly devoted extra time to a collaboration on accrued delay with Husni and me and always provided detailed, constructive feedback. Tony displayed his creativity, energy, and dedication when we sketched ideas on the whiteboard or hunched over a laptop, trying to make sense of a set of results. He likewise contributed to this thesis by providing crucial perspectives from industry and helping me contextualize my work. I'd like to thank all of the researchers with whom I have had the opportunity to collaborate with, including Maxim E. at Airbus, Michael S. at MIT Lincoln Laboratory, and Eric R. from the US Air Force. A special thank you to Adriana A. from JAXA for initiating our collaboration on disaster response and always providing thoughtful advice, feedback, and encouragement.

I want to thank all current and former lab members of DINaMo. Karthik G. was my first collaborator at MIT and helped me with everything, including coding, conceptualization,

derivations, and writing. I appreciated Karthik's curiosity, which led to insightful questions and late-night breakthroughs. He was also a dream collaborator—an expert in asking the right questions while respecting others and maintaining high spirits. I also had the pleasure of working with Prof. Max Li, who graciously included me in several of his in-progress projects. Max is an expert in knowing which way to guide research and was always generous with his time and encouragement. I want to thank Sandeep B. for also encouraging me and providing detailed feedback whenever I gave a lab presentation. He also introduced me to the sport of squash, which proved to be a useful stress reliever (mostly for him, since he typically won our matches). I want to thank Matt K. and Siddharth N. for their collaboration on the Puckboard project, which included an exciting trip to Oahu. I had the pleasure of working with Akila S. on the JAXA drone disaster response work. I'd also like to thank Victor Q. for collaborating with me on much of the protocol work and Geoffrey D. for including me on much of his drone privacy work. I want to thank Sydney D., Jasmine A., Allan S., Kevin Z., Alissa C., Shashank D., Adina G., Lauren C., Titi F., Joao C., Gerard D., and Lindsey B. for being the best labmates. I want to thank the ICAT friends I made, including Bazyli S., Kevin W., and Tim L.

I count myself very lucky to have such a supportive family. At the top, this includes my late paternal grandmother, who supported my father's move abroad to the US from Malaysia. Growing up, my maternal grandfather and late grandmother made time to treat my siblings and me to lunch regularly. One way in which they showed their love for us was by sharing food and offering seconds, thirds, and fourths. Another expression of their love was their constant words of support and relentless optimism in our futures. My sisters and I are privileged to live in the US thanks to their sacrifices. It is impossible to put into words the amount of support my parents have given me. From driving me to/from after-school sports practices to encouraging my interest in aviation at a young age, thank you to my parents for always being my strongest supporters—I hope to reciprocate in kind. Thank you to my two wonderful older sisters, Dr. Jennifer Chin and Dr. Natalie Chin, for always believing in me and supporting me in whatever I pursued. I am inspired by their dedication and passion for their medical careers, balanced with their ability to live full and exciting lives outside of work. Rather than leverage their positions as older sisters to their advantage, they always looked out for me, even when I played my role as the annoying youngest sibling. Thank you to my fiancée Justina, for always being loyally at my side during the highs and lows of graduate school. Without her, I would not have been able to complete this thesis or my studies at MIT. She always makes me happy, and I am so grateful that we have created many great memories while in Boston, including biking on hot summer nights and hosting Site 4 events. I am so excited for our wedding in Summer 2024. (I will also be more involved with planning now.) Thank you to Justina for moving across the country so we could be together and being the best partner I could have ever hoped for.

Contents

Title page **1**

Abstract **3**

Acknowledgments **5**

List of Figures **11**

List of Tables **13**

1 Introduction **15**
1.1 Civilian Applications of Advanced Air Mobility . . . 17
1.2 Current Regulations . . . 18
1.2.1 Part 107 Regulations [13] . . . 18
1.2.2 Helicopter Routes . . . 20
1.3 Future Concept of Operations . . . 20
1.3.1 FAA UTM Concept of Operations V2 [16] . . . 20
1.3.2 FAA UAM Concept of Operations V2 [17] . . . 22
1.3.3 SESAR U-Space Concept of Operations Vol. 2 [18] . . . 23
1.4 Considerations of AAM Traffic Management . . . 24
1.5 Levels of Information Sharing . . . 25
1.6 Levels of Centralization . . . 28
1.7 Contributions of Thesis . . . 30
1.8 Outline of Dissertation . . . 31

2 Related Literature **35**
2.1 Air Traffic Management . . . 35
2.2 Centralized Traffic Flow Management . . . 38
2.2.1 Ground Delay Programs . . . 38
2.2.2 Traffic Flow Management . . . 39
2.2.3 Fairness and Uncertainty . . . 39
2.3 Decentralized Traffic Flow Management . . . 40
2.3.1 Example: Collaborative Demand Management . . . 40
2.3.2 Congestion Management Protocol . . . 41
2.4 Advanced Air Mobility Traffic Management . . . 42
2.4.1 Market-based Approaches . . . 43

2.4.2 Our Approach . . . 43

3 Centralized AAM Traffic Flow Management 45
3.1 Background . . . 46
3.1.1 Traffic Flow Management Problem (TFMP) . . . 46
3.1.2 Fairness Considerations . . . 48
3.1.3 Operator Preferences and Capabilities . . . 50
3.1.4 Variable File-Ahead Times . . . 51
3.1.5 Contributions and Main Findings . . . 52
3.2 Baseline AAM Traffic Flow Management . . . 54
3.2.1 Notation . . . 54
3.2.2 Formulation . . . 55
3.3 Incorporating Fairness . . . 57
3.3.1 Reversals and Overtaking [47] . . . 57
3.3.2 Time-order deviation [48] . . . 59
3.4 Variable File-Ahead Times . . . 60
3.4.1 Rolling Time Horizon Implementation . . . 60
3.4.2 Pop-up Flights . . . 61
3.5 Experimental Results . . . 63
3.5.1 Scenario Generation . . . 63
3.5.2 Fairness-Efficiency Trade-offs . . . 64
3.5.3 Operator Fairness Alignment . . . 71
3.5.4 Operator Market Share . . . 73
3.5.5 Rolling Horizon . . . 75
3.5.6 Pop-up Flights . . . 79
3.6 Discussion . . . 83
3.6.1 Extensions . . . 84

4 Decentralized AAM Traffic Management Protocol 86
4.1 Background . . . 87
4.1.1 Objectives and Properties of Decentralized Protocol . . . 89
4.1.2 Assumptions . . . 91
4.2 Setup . . . 92
4.2.1 Information-sharing Constraints . . . 95
4.3 Decentralized Protocol . . . 96
4.3.1 Identifying Cycles . . . 99
4.3.2 Computing Backpressure . . . 101
4.3.3 Chains of Flights . . . 102
4.4 Prioritization Schemes . . . 102
4.4.1 Baseline Prioritizations . . . 103
4.4.2 Fairness-Oriented Prioritizations . . . 103
4.5 Experimental Results . . . 106
4.5.1 Scenario Descriptions . . . 106
4.5.2 Intra-operator Deconfliction . . . 108
4.5.3 Fairness and Efficiency of Flight-level Prioritizations . . . 110

4.5.4 Fairness and Efficiency of Operator-level Prioritizations 112
4.5.5 Overview of Results 115
4.6 Efficiency of Backpressure Prioritization 116
4.7 Discussion 121
4.7.1 Extensions 122

5 Federated Airspace Traffic Management **124**
5.1 Background 125
5.1.1 Information Sharing within Regions 126
5.1.2 n-Step TFMP 126
5.1.3 Setup 128
5.2 Procedure for Boundary Flights 131
5.2.1 Federated Airspace Containing More than Two Regions 133
5.3 Experimental Setup 134
5.4 Experimental Results 135
5.5 Discussion 138

6 Conclusion **139**

References **143**

List of Figures

1.1 Representation of file-ahead time in relation to file time and intended departure time. . . . 26
1.2 Examples of sharing different amounts of trajectory information. . . . 27
1.3 Examples of sharing information in a centralized or decentralized manner. . . . 27
1.4 Representative centralized setting. Flights communicate with the central traffic manager. . . . 28
1.5 Representative decentralized setting. Flights communicate with the next sector they intend to enter. . . . 28
1.6 Representative federated setting. The blue arrow represents a flight, which transits multiple regions in a federated architecture. . . . 29

3.1 The purple drone joins the end of the queue at every constrained resource, following a First-Come-First-Serve policy. The dotted purple line denotes the desired trajectory, while the solid purple line denotes the realized one. . . . 49
3.2 Flight trajectories shown from 4 vertiports in a 16 km × 14 km region, with axis ticks along the border denoting 1 km sector boundaries. Purple lines indicate local airports. . . . 64
3.3 Reversals vs. Total Delay Cost (TDC) when incorporating different fairness metrics. The hourly demand level is shown in parentheses. . . . 66
3.4 Overtaking vs. Total Delay Cost (TDC). . . . 69
3.5 Time-Order Deviation vs. Total Delay Cost (TDC). . . . 70
3.6 Operator efficiency and fairness with varying λ^1_{TOD} or λ^1_{rev}, with fixed $\lambda^2_{\text{TOD}} = 3$. Operator 1 and 2 are misaligned when operator 1 prioritizes reversals, aligned when operator 1 prioritizes time-order deviation, and perfectly aligned when $\lambda^1_{\text{TOD}} = \lambda^2_{\text{TOD}} = 3$. . . . 72
3.7 Effect of market share on Time-Order Deviation, with fixed $\lambda^1_{\text{TOD}} = 0.5$, $\lambda^2_{\text{TOD}} = 1.0$. . . . 74
3.8 Reversals vs. Total Delay Cost (TDC), by the length of the rolling horizon. "Myopic" is when flights are planned one by one according to the scheduled time of departure. . . . 76
3.9 Time-Order Deviation vs. Total Delay Cost (TDC), by the length of the rolling horizon. . . . 77
3.10 Runtimes for different rolling horizon sizes. $\lambda_{\text{rev}} = \lambda_{\text{TOD}} = 1$ for all horizon sizes. . . . 79
3.11 Average Total Delay Cost per Flight by Horizon Length and Pop-up Option. 81

3.12 Average Reversals per Flight by Horizon Length and Pop-up Option. 82

4.1 A simple example of the system state at time t, showing the current and intended sectors of all vehicles, with an example of a cycle in green. 94
4.2 Message passing required to identify cycle with example. 100
4.3 Example of system state with cycle in green (with vehicle IDs labeled) and non-cycle vehicles in other colors (with integers representing the backpressure). 100
4.4 Potential queue prioritization schemes for implementing the PRIORITIZEVEHICLE function. 104
4.5 Efficiency and fairness of our flight-level and operator-level protocols for three traffic scenarios. 111
4.6 Operator notions of fairness across prioritizations for Scenarios A-C. The notion of operator fairness is indicated on top of each plot and on the y-axis label. 114
4.7 Abstraction of a set of aircraft in conflict from the protocol (left) into a tree structure (right) 121

5.1 Example configurations of federated airspace with regions controlled with centralized or decentralized traffic management. "P" denotes a region controlled by the protocol, while "O" denotes an optimization region. 125
5.2 Diagram of "n-step TFMP" setup for a given time window between $[t, t+n]$. The extent of the colored arrows indicates the time length of the required information from each category of flight. 127
5.3 Handling boundary flights in a federated airspace setting. The base of solid arrows indicates the current position of flights, while the head represents the desired next sector. Faded arrows indicate information for 2 time steps from the current time. 130
5.4 Federated Airspace Layout with 4 Regions. Orange flights are in a cycle. The highlighted sector e2 determines the one-step actions of boundary flights in e3 and c4. 133
5.5 Delay Comparison of Different Airspace Configurations. "O10" indicates an optimization region of width 10, whereas "P5" indicates a protocol region of width 5. 136
5.6 Delay comparison relative to "1 Region: O10" configuration as crossing proportion varies. Crossing proportion is the proportion of flights that cross from one region to another. 138

List of Tables

3.2 List of Parameters 65
3.3 Change in total delay cost and reversals when incorporating reversals in the TFMP. 82
3.4 Change in total delay cost and TOD when incorporating TOD in the TFMP 83

4.1 Percentage change with intra-operator deconfliction relative to without intra-operator deconfliction for Scenario A. 109

Chapter 1

Introduction

Most civilians interact with aviation through commercial passenger service, allowing for quick transportation worldwide. Civilians may soon interact with aviation more frequently, particularly through smaller aircraft. Recently, uncrewed aerial vehicles (UAVs), commonly known as *drones*, have become more widespread. Some experts believe that drones could replace humans in tasks that are too "dull, dirty, dangerous, or dear" (these are known as the four D's of robotization) [1]. For example, drones could monitor infrastructure like bridges and dams or take aerial photographs after natural disasters. The widespread deployment of drones will require developing an uncrewed aircraft system (UAS), which includes the equipment needed to operate drones safely, and uncrewed traffic management (UTM), which refers to the traffic ecosystem for managing multiple drones. Drones are not the only new class of vehicles entering the public sphere. Electric vertical takeoff and landing (eVTOL) aircraft utilize electric power to hover, take-off, and land. The concept of urban air mobility (UAM) imagines small aircraft (like eVTOLs) carrying passengers and cargo in urban areas. UAM could unlock transportation between places difficult to connect with surface transportation (e.g., due to terrain or traffic congestion). Advanced air mobility (AAM) is a broad term that encapsulates utilizing new vehicles (like drones and eVTOLs) that are potentially autonomous. AAM includes UAM and the suburban/rural counterpart to UAM, regional air

mobility (RAM). There are several applications of AAM, many of which are discussed in Section 1.1. We use *advanced air mobility* (AAM) as a catch-all term for safe, affordable, and automated air transportation for passengers and cargo in urban and rural settings.

Because of the multitude of promising AAM applications, small vehicle traffic in the air may significantly increase. A 2018 study estimated that by 2035, Paris might see as many as 2,500 UAM flights, 16,000 delivery drones, and 60 inspection drones flying each hour of the day [2]. Other urban areas are projected to see a 200-fold increase in the number of flights from drone operations and a 30-fold increase from UAM operations [2], [3]. These operations will be primarily concentrated around dense urban regions near existing airports and will involve significant investments in technology and infrastructure [4].

The increase in traffic demand will inevitably result in congestion in the air and at vertiports (low-footprint airports in urban areas designed to support vertical takeoffs and landings). The entire US National Airspace System (NAS) currently handles about 90,000 flights a day, whereas UAS traffic operations are projected to exceed 2.5 million flights per day in the US [5]. The increase in vehicle- and system-level autonomy will enable the better utilization of limited resources. However, current regulations and processes (described in Section 1.2) may not scale to serve future levels of traffic adequately. Left unchecked, the resultant strain on the air transportation infrastructure could lead to decreased levels of safety and efficiency and increased costs and emissions. Developing an AAM traffic management system and the associated protocols, strategies, and infrastructure is essential for the safe and efficient operations of emerging aircraft. Regulatory agencies like the Federal Aviation Administration (FAA) have developed future concepts of operations (ConOps) for UTM and UAM. These ConOps (described in Section 1.3) serve as our starting point for studying AAM traffic management. The ConOps outline a vision of AAM traffic management and desired properties but do not answer many questions on how AAM traffic will be managed. For example, the ConOps states that third-party service suppliers are to deconflict aircraft (i.e., ensure vehicles remain adequately separated), but how this deconfliction is performed

is left as an open question. The onus is on researchers and stakeholders to develop AAM traffic management solutions. We discuss some considerations of AAM traffic management in Section 1.4 and explain how they motivate our approaches and contributions in Section 1.7.

1.1 Civilian Applications of Advanced Air Mobility

Novel applications of AAM exist across a wide range of industries [6], [7]. Note that drones have several military applications, but we choose to focus on civilian applications in this thesis.

- **Aerial Monitoring & Surveillance**: Some of the first commercial uses of drones involved aerial monitoring of some object of interest. Example monitoring interests include wildlife, infrastructure, construction, and traffic [8]. Drones are less expensive than conventional aircraft to operate and less disruptive to the surrounding environment.

- **Search and Rescue**: In the aftermath of natural disasters, aircraft and helicopters are crucial for searching for survivors and evaluating damage. However, conventional aircraft require ample resources (e.g., airports/heliports, pilots). Drones offer a less expensive alternative that could be deployed faster than conventional aircraft. In addition, drones may be able to access areas too risky for crewed aircraft. The Japan Aerospace Exploration Agency (JAXA) is exploring collaboration between crewed and uncrewed aircraft in disaster response [9].

- **Delivery**: Moving to larger aircraft, drones and eVTOLs could deliver packages, both mundane (e.g., packages delivered to residential addresses) and extraordinary (e.g., organ transplants delivered to rural areas). One notable example is Zipline delivering blood to rural locations in Rwanda [10]. AAM delivery applications are appealing for their speed advantage over conventional aircraft or ground transportation.

- **Air Taxis**: Perhaps no application of AAM has captured the imagination like air taxis. Proponents envision a future where ridesharing services are augmented with UAM vehicles, as air taxis are utilized to fly over surface congestion. It is worth noting that helicopters have historically been deployed for urban transportation with mixed results [11]. Notable barriers include cost, reliability, safety, and noise. After seven years of operations, the San Francisco and Oakland Helicopter Airlines company declared bankruptcy in 1970. New York Airways, which serviced Manhattan, disbanded in 1979 following a string of accidents that resulted in passenger and bystander fatalities. Boston's Air General failed due to low operational reliability in times of inclement weather [12]. Recent examples include Voom (a subsidiary of Airbus), which operated helicopters in congested Sao Paulo, Brazil, but was shut down in 2020. Uber Copter and Blade currently operate helicopter services between Manhattan and JFK International Airport. Proposed AAM vehicles hope to improve upon helicopters in terms of lower costs, improved operational reliability (including operating in limited visibility), higher safety levels, and lower noise.

1.2 Current Regulations

After covering potential applications of AAM, it is informative to consider the constraints of current regulations. In particular, we focus on regulations related to traffic management and vehicle operation. We do not cover other regulatory restrictions, such as aircraft noise or land use (relevant for vertiports) [4].

1.2.1 Part 107 Regulations [13]

The use of drones and unmanned aircraft systems (UAS) in the US is currently regulated by Federal Aviation Administration (FAA) Part 107. There are several restrictions present in Part 107 that currently limit operators and USS. These restrictions include:

- Both recreation and licensed UAS operators are restricted to flying within visual-line-of-sight (VLOS)
- Drones and UAS must avoid manned aircraft.
- Drone pilots cannot operate multiple drones at a time.
- Drones cannot fly over people.
- Drones are restricted to a maximum altitude of 400 feet above ground level (AGL) and cannot fly at night.

Operators can get waivers for many of these requirements if they demonstrate the ability to operate safely. Common waivers include beyond visual-line-of-sight (BVLOS), night, and overflight rules. In addition, drone operators have recently begun to receive Part 135 carrier certificates to fly in the US (most recently Zipline, which operates on-demand delivery drones, previously deployed in rural areas in Africa [14]). These are part of the FAA's BEYOND program to enable BVLOS operations. This certification explicitly permits flights that were previously only granted under waivers. In addition, work is ongoing to certify widespread BVLOS operations with Part 108 regulations.

An additional piece of regulation is the UAS Low Altitude Authorization and Notification Capability (LAANC). It allows UAS operators to request access to controlled airspace near airports at or below 400 feet. Notably, the application and approval process is automated. UAS operators submit requests using an application developed by an FAA-approved UAS Service Supplier (USS); then, the request is checked against FAA-provided airspace constraints, including temporary flight restrictions (TFRs) and Notices to Airmen (NOTAMs). This eliminates the need for UAS operators to communicate directly with air traffic control towers.

1.2.2 Helicopter Routes

The FAA classifies airspace into two categories: regulatory (i.e., controlled) and nonregulatory. UAM flights are expected to traverse controlled Class B, C, and D airspace. Because initial UAM flights will likely have crew onboard, allowing for direct communication with air traffic control, helicopter routes are an attractive, ready-made option for early UAM traffic management. Human-in-the-loop studies showed that current helicopter routes can support limited UAM traffic, but are unlikely to scale to service large amounts of traffic [15]. In addition, many helicopter routes are designed to fly over existing infrastructure like highways, railroad tracks, or rivers to simplify navigation and mitigate noise pollution. As AAM vehicles may be range-limited (particularly early-generation vehicles), more direct routes may be necessary.

1.3 Future Concept of Operations

In the US, the most relevant concept of operations (ConOps) are the UTM ConOps V2 and UAM ConOps V2. In Europe, the Single European Sky ATM Research (SESAR) collaborative project has developed a "U-space" concept of operations.

1.3.1 FAA UTM Concept of Operations V2 [16]

The FAA UTM ConOps V2, released March 2, 2020, focuses on UTM operations below 400 feet above ground level (AGL). It envisions a "community-based traffic management system" where third-party providers facilitate operator cooperation. Several use cases exist for drones, including flights solely within controlled or uncontrolled airspace and flights transiting between controlled and uncontrolled airspace. UAS operators can use *UAS Service Suppliers (USSs)* to support their operations or manage their own operations (i.e., operating as their own USS). There are several motivating reasons for this federated approach, where

federated means a group of systems operating in a standard and connected environment. This allows third-party service suppliers to customize their services for their users. In addition to being more agile and scalable, a federated approach can lead to cost reductions.

The UTM ConOps V2 defines the role of the USS as a communication bridge between stakeholders, providing UAS operators with information needed to plan operations in a volume of airspace. In addition, they are responsible for archiving operations data for regulatory purposes, which shall be made accessible to the FAA if needed. USSs are to "gather, incorporate, and maintain airspace reservations into airspace data repositories that may be accessed by Operators."

The FAA still plays a key oversight role, however, as the Flight Information Management System (FIMS) enables the data exchange of airspace constraints from the FAA to the USS Network and operational data from the USS Network to the FAA. Airspace authorization in controlled airspace can be obtained through LAANC. Operators planning to fly BVLOS are required to share operation intent; manned operators and unmanned VLOS operators are not required to share intent. From an operator's point-of-view, sharing operation intent involves submitting an *Operation Plan* (distinct from flight plan), which is a series of 4D volumes of airspace within which the operation is expected to occur. A single volume can be used, but ideally, the volume is segmented by time. The locations and times of key flight events and flight phases are also preferred. If the operator is self-provisioning services, they are responsible for identifying operational conflicts and strategically deconflicting with others. If an operator is not self-provisioning services, the USS shares operation intent with USS network. Regarding tactical separation, operators are responsible for this, but USSs can help monitor. Operators need to stay within the operation plan (4D volume). In the event of unavoidable non-conformance, the operator/USS notifies the FAA and UTM network.

In the event that deconfliction is needed, airspace volumes can be adjusted in size and shape, and/or volume segment entry/exit times can be modified. Strategic deconfliction is to be used whenever possible, but tactical management is used when necessary or too

time-sensitive. To ensure equity, in airspace with moderate demand, equitable access is "achieved through operator collaboration, efficient airspace design, and FAA rules". The USSs are to propose equitable solutions to competing operators or facilitate negotiations between operators to identify adjustments to minimize volume overlap.

1.3.2 FAA UAM Concept of Operations V2 [17]

The FAA UAM ConOps V2, released April 26, 2023, focuses on urban air mobility yet is very similar to the UTM ConOps V2. The ConOps delineates three types of airspace: traditional air traffic control, UTM (below 400 ft. AGL), and UAM corridors. Providers of Services for UAM (PSUs) assist operators and play a similar role as USSs in UTM. The distinction between PSUs and USSs is not strict, as USSs could expand to become PSUs or vice versa. There are three "stages" of operations described. The first stage is initial operations, which involves low demand and uses existing helicopter infrastructure, including routes and regulations. The second is covered by the ConOps, where vehicles operate in simple UAM corridors with minimal intersections, and separation is done by operators or PSUs. Additionally, vehicles have a pilot onboard. The third is a mature stage, which serves high demand, relying on a network of UAM corridors. Remote pilots or automated vehicles are used in increasing numbers, too. In this dissertation, we consider the second and third stages of maturity, such that existing traffic management methods are insufficient for realized demand levels.

Similar to in UTM, operators are envisioned to provide a flight intent to the PSU, which could be a volume of airspace along with key time-stamps. The UAM corridor could be stratified into layers when demand increases. ATC is to be notified of UAM flights to monitor for non-conformance, i.e., when vehicles stray outside the UAM corridor. The PSU could handle "off-nominal" events where the operator can specify a new operational intent that satisfies constraints, but ATC would intervene in emergencies where vehicles need to exit UAM corridors or perform emergency landings.

The ConOps outlines an estimate of the evolution of the UAM operational environment. In particular, our work is motivated by the different levels of traffic management indicated. NASA also has a similar evolution framework with more levels, but the key ideas remain the same.

- Initial UAM Operations: Demand is low, so no unique UAM structures or procedures exist. Rather, existing air traffic control (ATC) procedures for crewed fixed-wing aircraft/helicopters are used for UAM vehicles.
- Midterm UAM Operations: Demand has increased such that existing ATC procedures need to be modified. UAM vehicles are allocated corridors that are governed by rules determined by stakeholders and approved by the FAA.
- Mature UAM Operations: Demand levels necessitate a new framework for UAM operations, with dedicated corridors, paths, and networks. Traffic management relies extensively on third-party providers of services for UAM (PSUs).

1.3.3 SESAR U-Space Concept of Operations Vol. 2 [18]

The SESAR U-Space ConOps shares many similarities with its FAA counterparts. They describe a U-space service provider (USSP) that performs similar functions as a USS. The ConOps is also flexible in supporting VLOS and BVLOS operations. One key difference is their definition of three types of airspace "volumes": X, Y, and Z. Here, volumes refer to the regions of airspace that AAM vehicles operate in, not the airspace volumes that individual vehicles define preflight in the FAA ConOps.

- **X Volumes:** No conflict resolution service is provided; remote pilots are solely responsible for separation. This is envisioned for low-density, rural areas.
- **Y Volumes:** Only preflight, strategic deconfliction is offered, meaning that operation intents must be shared preflight. Vehicles will be spaced far apart since there is no tactical conflict resolution.

- **Z Volumes:** Both strategic deconfliction and tactical conflict resolution are provided. Separation could be provided by USSPs (in which case the volume is named Zu) or by air traffic controllers (volume is named Za). Z volumes are envisioned for high-density urban areas since the presence of tactical deconfliction allows higher densities than Y volumes.

1.4 Considerations of AAM Traffic Management

Third-party service suppliers: The ConOps illustrate that air navigation service providers (ANSPs), like the FAA, cannot directly manage the high volumes of expected AAM traffic. Moreover, the projected increasing number of AAM operations may exceed the current capabilities of existing ATM infrastructure and workforce resources. Thus, it is proposed that traffic management responsibilities lie with third-party service suppliers. These service suppliers are called USSs in the context of UTM and PSUs in the context of UAM, but they function similarly. These third-party service suppliers perform many functions, including data services, communication services, regulatory compliance, and—most relevant to this thesis—traffic management services. How exactly a third-party service supplier manages traffic remains an open question. We explore centralized and decentralized traffic management methods. In addition, we consider federated airspace configurations where adjacent airspace regions may utilize different traffic management methods.

Efficiency and fairness: In order to scale AAM operations to meet the increasing demand, it is important to operate efficiently and minimize delays (where delay is defined as the difference between desired and actual operation times). Secondly, the equitable and fair allocation of airspace resources is important, especially in the presence of a large number of aircraft operators; yet, fairness is often the first casualty in the quest for efficiency [19], [20]. Fairness is particularly important for AAM applications because of the diverse set of operators that may interact with each other. We explicitly incorporate fairness between op-

erators in our centralized and decentralized traffic methods. Moreover, we evaluate different fairness metrics and the trade-off between efficiency and fairness.

Information sharing: AAM vehicles at any point in time have current information, such as position, velocity, and heading, and future information, such as destination and route. This information acts as input data to any traffic management service. However, AAM operators may be *unable* or *unwilling* to share too much trajectory information. Consider the case of ridesharing operators. An operator may be unable to share destination or route information while it waits for demand to materialize (in the form of customers requesting rides). Alternatively, once a ridesharing operator knows the intended trajectory of a vehicle, it may be unwilling to share this information with a central agent for fear that competitors may learn this information and undercut them on price or fleet deployment. Moreover, the central agent (i.e., the third-party service supplier) may operate flights themselves and thus directly compete with the operator. Now that we've established why information sharing is a concern, we discuss levels of information sharing in the next section.

1.5 Levels of Information Sharing

We define three axes of information-sharing concern: when information is shared, how much information is shared, and to whom information is shared. We describe these three in order, along with accompanying Figures 1.1–1.3.

- *When* information is shared: Operators may want to share trajectory information very early (to "reserve" airspace) or very late (to not reveal sensitive information). Figure 1.1 shows that the file-ahead time is the time difference between the intended departure time and the file time (i.e., when the flight lets others know its intention to depart). Compared to commercial aviation, AAM traffic is expected to be more dynamic. Airlines typically publish their flight schedules months in advance and file their flight plans at least an hour before scheduled departure. By contrast, an AAM

operator may plan a flight (including origin, destination, and route) with a file-ahead time in the order of minutes.

- *How much* information is shared: Operators may only be willing to share partial trajectory information. Thus, we can consider how much information operators share as one level of information sharing. In Figure 1.2, the left-hand side shows the full intended trajectory of a single vehicle, whereas the right-hand side presents two examples of how much information is shared. In settings requiring full trajectory information, the vehicle would provide its intended trajectory as is; however, if less information sharing is permitted, the vehicle can share just one "timestep" of information. When we discretize time into timesteps of a set length, we can allow vehicles to share partial trajectory information (i.e., the blue solid arrow rather than the full trajectory in Figure 1.2).

- *With whom* information is shared: Operators may be willing to share information to a centralized system for traffic management. On the other hand, operators may only want to share information with a limited set of stakeholders (e.g., only with nearby vehicles or airspace sectors). Figure 1.3 presents two examples of whom information is shared with: on the left-hand side, vehicles share information with a central agent, but on the right-hand side, vehicles share information only with their next intended sector. A sector is a small region of airspace, represented in a grid structure in this case.

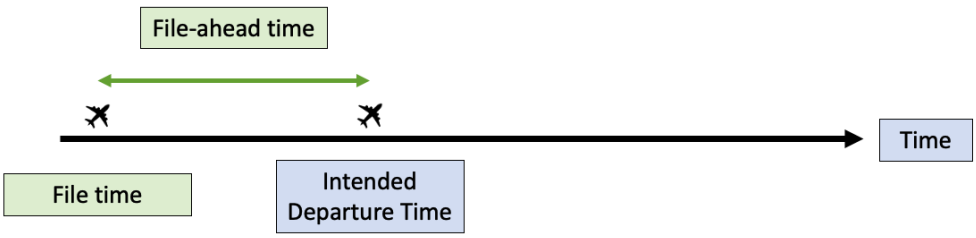


Figure 1.1: Representation of file-ahead time in relation to file time and intended departure time.

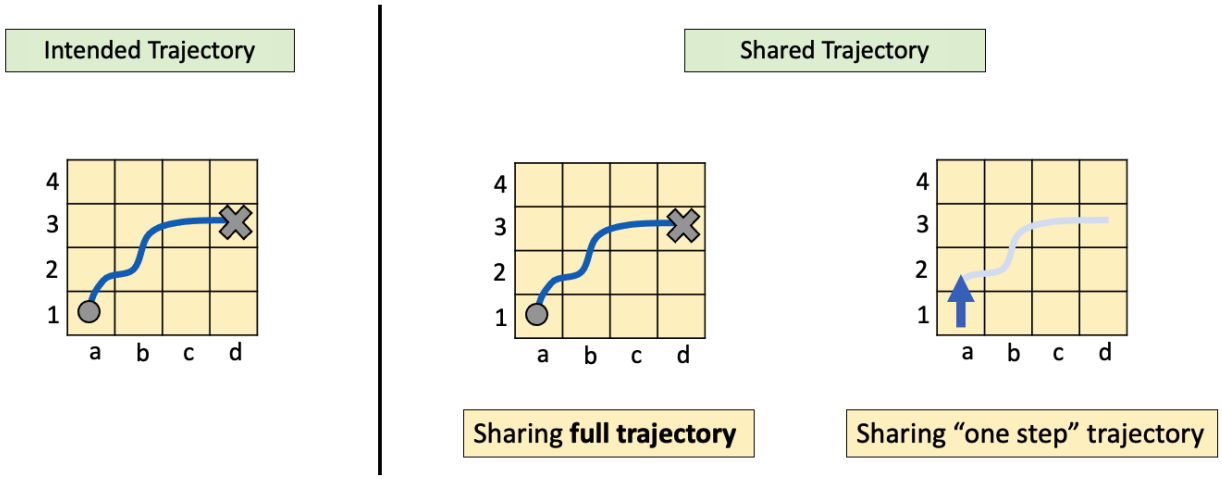


Figure 1.2: Examples of sharing different amounts of trajectory information.

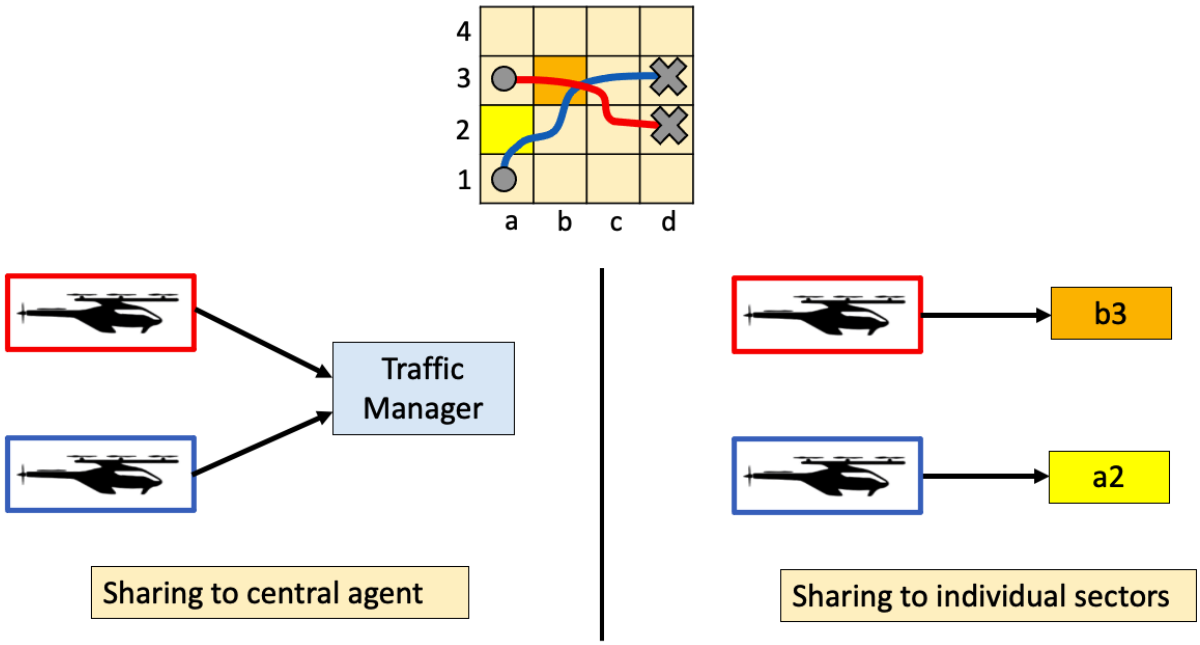


Figure 1.3: Examples of sharing information in a centralized or decentralized manner.

1.6 Levels of Centralization

We have established several considerations of AAM traffic management, including efficiency, fairness, and information sharing. Moreover, we note that third-party service suppliers will be responsible for traffic management while conforming to FAA (or any other ANSP) regulations. Service suppliers must consider the *level of centralization* when determining how to manage traffic. In this dissertation, we consider centralized (Chapter 3), decentralized (Chapter 4), and federated settings (Chapter 5), which we will explain in order.

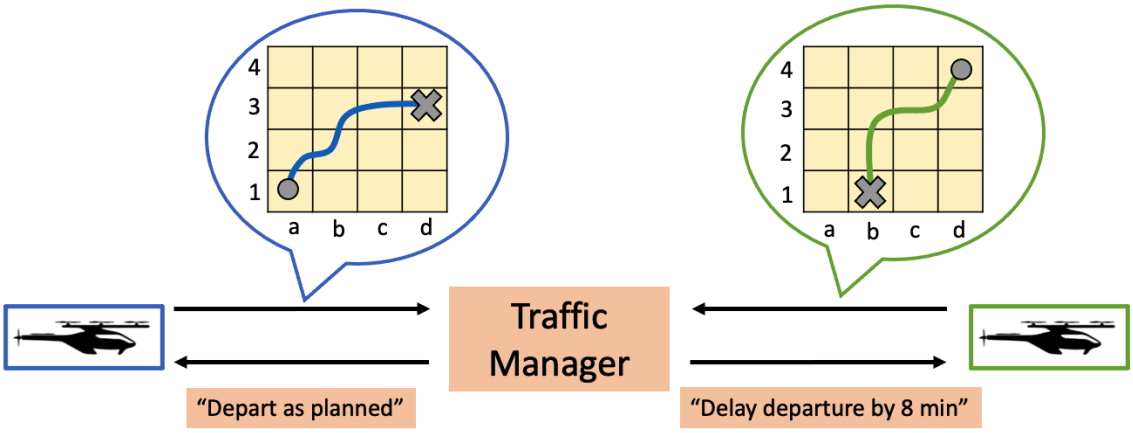


Figure 1.4: Representative centralized setting. Flights communicate with the central traffic manager.

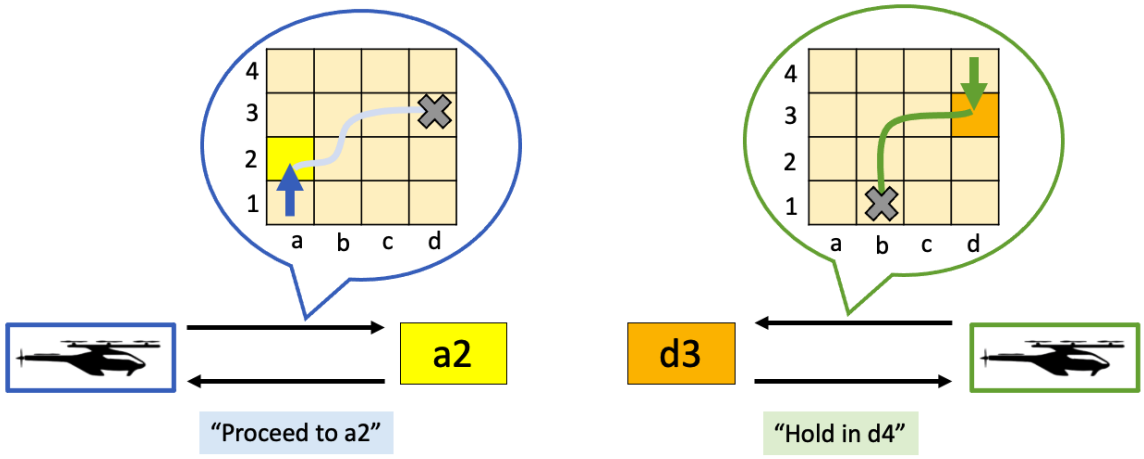


Figure 1.5: Representative decentralized setting. Flights communicate with the next sector they intend to enter.

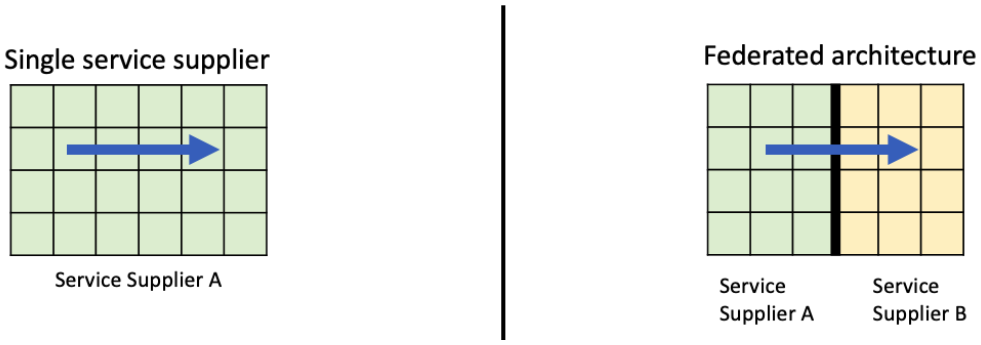


Figure 1.6: Representative federated setting. The blue arrow represents a flight, which transits multiple regions in a federated architecture.

Centralized: The first consideration of AAM traffic management we discussed is efficiency (i.e., minimizing delays). We start with a centralized setting where a traffic manager can leverage its knowledge of all vehicles to minimize system delay. Figure 1.4 displays an example of a centralized setting with two vehicles: a blue and a green one. Both vehicles share their trajectory information with a central traffic manager. This traffic manager aggregates information from all vehicles and issues controlling actions to individual vehicles. For example, it allows the blue vehicle to depart without delay but requires the green vehicle to delay its departure by 8 min. Note that vehicles share *full* trajectory information in this example, but this is not a requirement of centralized settings. We consider a centralized setting in Chapter 3.

Decentralized: While the centralized setting emphasizes efficiency, operators and vehicles may not be comfortable with the requirement to share information with a central agent. Thus, we turn to a decentralized setting to focus on another consideration of AAM traffic management: information sharing. Vehicles do not need to broadcast information to a central agent in a decentralized setting. In Figure 1.5, the blue and green vehicles only need to share information with the sector that they want to enter in the next time step rather than with a central agent. These individual sectors then issue controlling actions to the vehicle, either allowing them to enter or forcing them to hold in their current sector. We develop a decentralized protocol for AAM traffic management in Chapter 4.

Federated: Up to now, we have considered configurations where one third-party service supplier controls the whole area of airspace. Based on the ConOps, it is possible that areas of airspace may be divided between multiple regions. Note that we assume that service suppliers do not control overlapping regions. In Figure 1.6, the left-hand side shows one flight (in blue) traveling in a region with a single service supplier. On the right-hand side, this flight transits the same number of sectors but in a federated architecture with two service suppliers. Service Suppliers A and B may utilize different traffic management methods, which the blue flight must adhere to while inside a given service supplier's region. With multiple service suppliers, we need to consider how to manage flights crossing between regions, which may have different traffic management methods. We consider how to handle crossing flights in a federated setting in Chapter 5.

1.7 Contributions of Thesis

This thesis explores advanced air mobility traffic management in federated settings, including centralized and decentralized settings. We consider the balance between efficiency, fairness, and minimal information sharing. We seek to provide centralized and decentralized approaches to AAM traffic management, which can be adapted for scenarios with different levels of information sharing and centralization, including federated settings. The primary contributions of this thesis are as follows:

1. We evaluate the trade-off between efficiency and fairness and between different fairness metrics when using *centralized* traffic flow management for advanced air mobility.

2. We explore issues of operator fairness and operator file-ahead time in the centralized setting. We demonstrate that when operators are not aligned in how they perceive or value fairness, there is a decrease in the overall fairness of the solution.

3. We design and evaluate a *decentralized protocol* for AAM traffic management, which reduces information-sharing requirements relative to centralized approaches.

4. We show how to incorporate fairness through prioritization schemes in the "rules-of-the-road" decentralized protocol approach. We prove that the protocol is optimal in one time step when using the backpressure prioritization scheme.

5. We present a mechanism for managing traffic that crosses different regions in a *federated* airspace setting where bordering regions of airspace may utilize different traffic management methods.

1.8 Outline of Dissertation

- Chapter 2 details relevant literature on AAM traffic flow management. Since AAM is a relatively new field, we first pull from related centralized and decentralized air traffic flow management work. This includes centralized approaches like the traffic flow management problem (TFMP), which we use in the centralized optimization formulation in Chapter 3.

- Chapter 3 studies AAM traffic flow management in the centralized setting. A centralized setting is appealing to operators if efficiency is the primary goal. We assume that one service supplier controls traffic in a region of airspace. Space is discretized into "sectors" of size 1km $\times$ 1km, and time is discretized into timesteps of 30 s. While centralized traffic flow management does not require it, in Chapter 3, we assume that operators are willing to share full trajectory information with the service supplier in order to minimize delay. Thus, operators share the origin, destination, enroute sectors, and associated scheduled and feasible arrival times at each resource with the third-party service supplier. The service supplier is responsible for assigning ground and airborne delays to flights based on trajectory information that operators submit as well as sector and vertiport capacities. We consider objective functions that balance efficiency and fairness. There are many possible definitions for fairness, so we study three fairness metrics that have been used for air traffic flow management: reversals, overtaking, and

time-order deviation. In the objective function, we include one term for efficiency (total delay cost, where airborne delay cost is more expensive than ground delay cost) and one term for fairness (one of the three fairness metrics at a time). We first evaluate the trade-off between efficiency and fairness and find that fairness can be improved with little loss in efficiency. We also evaluate trade-offs between different fairness metrics (e.g., evaluating the impact on time-order deviation when reversals are incorporated in the objective function). We also consider the impact on efficiency and fairness when operators have different preferences for fairness regarding their choice of fairness metric and their fairness weight (i.e., how much they value fairness relative to efficiency). We find that efficiency and fairness are highest when operator fairness preferences are aligned. Up to this point, we assumed that all flights shared full trajectory information well before their scheduled departure time so that central optimization could solve all flights simultaneously. We relax this assumption and consider cases where a a *rolling horizon* approach is used. This allows flights to submit information later (this relates to *when* information is shared). In a rolling horizon framework, the optimization is solved once for every horizon, and flights are grouped into horizons based on scheduled departure time. We evaluate alternative methods to incorporate dynamic demand. In particular, we consider *pop-up* flights that appear with little lead-time, i.e., flights that share their information very close to their desired departure time.

- Chapter 4 develops a decentralized traffic management protocol for advanced air mobility. The motivation for this is to focus on minimizing the required level of information sharing. Operators may be unwilling or unable to share full trajectory information. Thus, we approach AAM traffic management from the perspective of minimal information sharing between operators and third-party service suppliers and between different airspace sectors. We develop a congestion management protocol. Space is discretized into sectors of unit capacity, and time is discretized into timesteps. Rather than utilize a central solver, vehicles communicate only with the current sector they occupy and

the next sector they wish to occupy in the next timestep. Moreover, we constrain the information shared between airspace sectors—they may only communicate limited information with adjacent sectors. The first step of the protocol is to identify *cycles* that indicate gridlock. For example, a cycle is formed when Flight A wants to go from S1 to S2 and Flight B wants to go from S2 to S1 because either none of the flights can proceed or all of them must proceed, given unit capacity constraints. Another consequence of unit capacity constraints is that when more than one vehicle wishes to enter a sector in the next timestep, the sector must "resolve" this conflict and decide which incoming vehicle to allow to enter. The protocol is deliberate in the *order* in which sectors are resolved, as resolutions at one sector can force other sectors into suboptimal decisions. We use a *backpressure* metric to determine the order in which sectors are resolved. We consider various *prioritization schemes* that a sector can use to resolve its conflict. The backpressure can be used for prioritization to minimize delay. We formalize the notion of *cycles* and *chains* in the protocol to prove that the protocol is efficient and results in a minimal delay solution in one time step when using backpressure prioritization. We explore other prioritization schemes which emphasize fairness, such as schedule reversals, accrued delay, or dominant resource fairness. As with the centralized setting, we evaluate fairness-efficiency trade-offs and show how to incorporate different flight-level or operator-level prioritization. We show that the choice of prioritization method influences efficiency and fairness, and the best method depends on the traffic pattern.

- Chapter 5 utilizes elements from the decentralized protocol developed in Chapter 4 to handle flights crossing between regions in a federated setting. Adjacent regions could be controlled by different service suppliers who utilize different traffic methods. We assume that each region utilizes centralized or decentralized traffic flow management, developed in Chapters 3 and 4, respectively, but the method for handling crossing flights is agnostic to the precise traffic management method within each region. We

show that the protocol can be used to coordinate flights that cross between regions in a federated setting with multiple service suppliers. Moreover, we evaluate the efficiency of several airspace configurations, where the number of regions and type of traffic management within each region varies.

- Chapter 6 provides concluding thoughts on centralized, decentralized, and federated AAM settings and describes areas of future work.

Chapter 2

Related Literature

Traffic is a fact of life for many cities around the world. Managing traffic is essential for ensuring efficiency and safety in moving goods and people. Recently, environmental considerations have increased the importance of traffic management. As early as the Roman Empire, cities deployed traffic management techniques like tolling and time-of-day restrictions [21]. Other traditional traffic management tools for surface transportation include road signs and "rules of the road." The first traffic signal was installed in London, England in 1868 to control the flow of pedestrians and carriages near the Houses of Parliament. However, this signal was not electric, as a nearby police officer had to change the lights manually. In 1914, the first electric traffic signal was introduced in Cleveland, Ohio. The genesis of air traffic control towers followed a similar pattern to traffic signals. Croydon Airport near London was the first airport in the world to have an air traffic control tower in 1920. The first US air traffic control tower was opened in Cleveland in 1930.

2.1 Air Traffic Management

Like surface traffic management, air traffic management is crucial for maintaining safety and efficiency. Commercial aviation has a strong history of safety. There are several safety metrics, but aviation safety has generally increased over time, with some year-over-year

volatility and geographical variation [22]. Commercial aviation is remarkably safe, but passenger and flight delays are not uncommon. According to the Bureau of Transportation Statistics, between 2014 and 2013 (omitting 2020 due to COVID-19 obfuscating delay data), around 16–22% of passenger flights were delayed in the U.S. [23]. Flight delays are costly to both passengers and airlines—a comprehensive 2007 study estimated that flight delays cost airlines 8.3 Billion USD and passengers 16.7 Billion USD in 2007 [24]. Delays are not yet an issue since AAM operations are currently very limited. However, delays could become relevant for AAM operators and users as AAM traffic increases.

Traffic congestion, which leads to flight delays, occurs when demand exceeds capacity (the same can be said about surface transportation). When considering how to handle traffic congestion at airports or in the sky, there are long-, medium-, and short-term approaches [25]. In the long term (on the order of years), we can increase capacity. For example, airports can construct additional runways or taxiways, or ATC procedures can be modified to reduce separation requirements between aircraft. In the medium term (on the order of months to a few years), we can modify demand temporally to reduce the gap between demand and capacity. An example of demand management is *airport slot allocation*, whereby airlines must apply months in advance for a limited number of departure/arrival slots at airports. Airport slot allocation balances scheduled traffic and arrival airport capacity, thereby reducing flight delays [26], [27]. This not only smoothens demand temporally, but also caps aircraft movements at an airport during peak hours. In the short term (on the order of hours), ATC can strategically mitigate delays to try to match demand and capacity. For example, flights can be held on the ground before departure to reduce the airborne delay that flights must incur.

The basic function of air traffic management (ATM) is to prevent aircraft collisions. Effective ATM maximizes the efficiency of airspace utilization, minimizing delays and reducing fuel consumption. ATM can generally be divided into three functions: air traffic control, airspace management, and air traffic flow management. Our work focuses on the last of

these, air traffic flow management for AAM. Air traffic control (ATC) is responsible for guaranteeing safe separation between aircraft and preventing collisions. There is substantial human factors research on air traffic controllers [28]. In addition, there is a growing body of research on remote air traffic control towers, where air traffic controllers are not located at the airports they control [29]. This is particularly useful for lower-traffic-density airports, first introduced in rural Sweden [30]. Airspace management refers to the division of airspace control. Starting from the airport, the airport control towers are responsible for traffic on the ground, landing traffic, or traffic taking off. Moving higher in elevation, the Terminal Radar Approach Control (TRACON) manages airport arrival and departure procedures. Finally, the Air Route Traffic Control Center (ARTCC) is responsible for volumes of airspace containing airways, which are high-altitude established routes. Centers are divided into airspace sectors, which an air traffic controller or several air traffic controllers manage. Traffic levels in sectors can vary spatially and temporally. Researchers have studied the dynamic resizing of sectors and the dynamic assignment of air traffic controllers to sectors [31], [32]. Other work has looked at finding the optimal *sectorization* of airspace to balance ATC workload and minimize sector crossings by aircraft [33]–[35].

Our main area of focus is air traffic flow management (ATFM) functionality for AAM. Air traffic controllers and traffic managers work to minimize system delays for commercial aviation by considering airport and airspace constraints. ATFM measures are typically classified by the time frames in which they are deployed. Strategic measures have planning horizons of 2–8 hours, whereas tactical measures have a planning horizon of less than 2 hours, including real-time planning. ATFM can be divided into centralized and decentralized approaches. In a centralized system, control and decision-making authority are concentrated within a single entity or control center, typically operated by a governmental or regulatory body like the Federal Aviation Administration (FAA). This centralized authority monitors airspace, directs aircraft, and ensures overall safety and efficiency. Clear hierarchical structures, standardized procedures, and robust communication networks often characterize centralized ATFM

systems. We discuss relevant literature on centralized air traffic management in Section 2.2.

Decentralized ATFM distributes decision-making authority across multiple local control centers. For example, in a geographically decentralized system, individual sectors could manage their own traffic with limited coordination with adjacent sectors. Traffic management could even be the responsibility of individual aircraft, as aircraft could autonomously manage their own routes and spacing with minimal intervention from ground-based controllers. Aircraft operators may appreciate the flexibility of decentralized approaches, but these approaches are generally less efficient than centralized methods. We discuss relevant decentralized air traffic management literature in Section 2.3.

2.2 Centralized Traffic Flow Management

2.2.1 Ground Delay Programs

The efficient allocation of constrained airspace and airport resources has been studied extensively. The first area of centralized air traffic management research involves Ground Delay Programs (GDPs). This is centralized because all flights must share trajectory information with a central agency (e.g., the FAA). The trajectory information shared could include the intended flight plan, routing, and schedule. Based on weather conditions, the predicted arrival capacity at an airport may be less than the expected demand. For example, suppose an airport could normally accept 60 landings per hour, but because of fog (common at an airport like San Francisco), the arrival capacity is expected to drop to 40 landings per hour at 8:00 am. If there are 60 flights scheduled to land between 8:00 and 9:00 am, air traffic control would need to delay these flights in the air, either through speed restrictions (i.e., making flights fly slower), path stretching (making flights fly longer paths), or even holding patterns (making flights fly circular patterns or "stacks" while they wait for the runway to clear). These are all forms of airborne delay, which are costly to airlines. GDPs seek to reduce airborne delays by delaying flights at their departure airport (incurring ground de-

lay) to reduce the incursion of costly airborne delays. This has the effect of "smoothening" the arrival demand such that demand between 8:00 and 9:00 am does not drastically exceed capacity. The output of GDP optimization models is new flight departure times to balance demand and airport capacity. Initial studies focused on the optimization formulation for the case of a single airport [36], [37] or groups of airports [38], [39].

2.2.2 Traffic Flow Management

While GDP formulations only consider airport capacities, the air traffic flow management problem (TFMP) also considers en route capacity. Due to weather activity or ATC workforce constraints, airspace sectors may have limited capacities. An airspace sector is a specific portion of airspace assigned to a specific ATC facility. These en route sector capacity restrictions can occur simultaneously with airport capacity restrictions, creating a capacitated network. The TFMP can be modeled as a linear integer program. Optimizing a TFMP is more computationally challenging than optimizing GDPs; however, significant progress has been made in solving this problem [40]–[44]. We adopt the TFMP for an AAM setting in Chapter 3. There are several other traffic flow management strategies. An Airspace Flow Programs (AFP) is the airborne analog to a GDP. In an AFP, flights are held at their origin, rerouted, or speed restricted due to en route constraints at Flow Constrained Areas (FCAs) where traffic needs to be metered [45]. These centralized traffic flow management methods seek to balance demand and capacity strategically. This is typically done by delaying flights to regulate demand and improve efficiency.

2.2.3 Fairness and Uncertainty

In a system with multiple users, balancing efficiency and fairness is important. In the context of ATM, operators and flights expect to be treated fairly regarding delay assignments. The definition of *fairness* can differ based on context and stakeholder. Traditionally, the FAA viewed "first-come, first-served" as the most fair procedure. Alternative approaches consider

fair allocations as ones that adhere to original flight schedules [46]. With regard to jointly allocating airport and airspace resources, notions of fairness include reversals [47], overtaking [47], and time-order deviation [48]. We will discuss these three fairness metrics in Chapter 3. Work has also been done to study airlines trading with each other through at-most, at-least trades [49]. Fairness metrics studied in non-aviation contexts include alpha fairness, proportional fairness, and max-min fairness [50]–[52].

Another area of research in centralized traffic management is decision-making under uncertainty. There is ample uncertainty in air traffic management, including in traffic (when demand will materialize) and weather (directly influencing airport and airspace capacity). Stochastic optimization [53]–[55], robust optimization [56], [57], and chance-constrained programming [58] have been applied to centralized air traffic management.

2.3 Decentralized Traffic Flow Management

2.3.1 Example: Collaborative Demand Management

In the traffic management measures discussed so far, the central agent (e.g., the FAA) dictates actions for individual flights. For example, in traditional ground delay programs (GDPs), the FAA assigned delays to flights with little airline input. Collaborative Decision Making (CDM) is a concept that facilitates collaboration between the FAA and airlines on deciding how to manage traffic [59]–[61]. Airlines can share desired trajectories and schedules with the FAA. When the FAA allocates a set of slots to airlines, each individual airline can choose which flights to assign to slots, subject to other resource constraints. Thus, injecting the CDM concept into GDPs provides a more decentralized approach than traditional GDPs because decision-making no longer lies solely with the FAA. The concept of *Free Flight* furthers decentralization and allows individual aircraft/flights to plan four-dimensional trajectories in real-time, shifting some traffic management responsibilities from ground-based controllers to aircraft [62], [63]. This may be preferable to airlines and airspace

users as they regain more control over their operations. However, the challenge is to maintain efficiency in these decentralized setups.

2.3.2 Congestion Management Protocol

A pertinent example of a decentralized transportation system is surface transportation. Unlike commercial flights, individual vehicles do not need to share their full "trajectory" information with a central agent, allowing them ample freedom to change routes or destinations. Instead, vehicles are expected to adhere to "rules of the road" and protocols, such as first-come, first-served at stop signs. Road congestion is common in cities, so there have been several studies on improving traffic control through connected vehicle technology (e.g., Vehicle to Vehicle (V2V) and Vehicle to Infrastructure (V2I) communication) [64]–[66].

Congestion control protocols have been studied in several contexts, including communication networks [67], surface transportation [68], and air traffic management [69]. Solution approaches proposed range from queue-length management protocols [70] to dynamic traffic routing, demand management [71], backpressure algorithms [72], [73], and optimal network flow management [74]. The idea of decentralized protocols for air traffic management is not a new one [75], [76]; researchers have proposed rules-of-the-road style protocols [77], Markov decision process models [78], and speed control algorithms [79].

Fair congestion control has been studied in the context of routing packets in communication networks [80]. The simplifying assumptions typically made, such as infinite buffers at congested resources or high traffic volumes that can be approximated as fluid flows, are rarely satisfied in air traffic networks, be they conventional aviation or AAM. As a result, fairness in air traffic management has generally been evaluated either through first-come-first-served simulations [81] or in centralized settings with full information sharing [47], [82]. In our AAM traffic management protocol in Chapter 4, we incorporate fairness, reduced information sharing, and cost-aware prioritization schemes into this class of algorithms that have historically focused only on safety and efficiency.

2.4 Advanced Air Mobility Traffic Management

We covered future concepts of operations in Section 1.3; in this section, we discuss research pertaining to AAM traffic management. As with traditional air traffic flow management, AAM traffic management methods span centralized and decentralized methods. One study developed a novel distributed framework using column generation to solve large instances of the centralized air traffic flow management problem, with crewed and uncrewed traffic [83]. Another study uses a Monte Carlo traffic simulator to compare centralized and decentralized approaches for conflict resolution [84]. While they analyzed the frequency of loss of separation events, they did not consider system delay metrics on efficiency and fairness. Another study splits flights into different "layers" of airspace (corresponding to different elevations) using graph coloring techniques to avoid conflict [85], [86]. This division of flights into layers is either done pre-flight (in a centralized manner like a GDP) or en route (in a decentralized manner), with vehicles having the flexibility to jump to adjacent layers temporarily. There are also decentralized approaches that rely on cooperation between aircraft [87] or leverage contract-based reactive synthesis [88].

A body of literature closely related to AAM traffic management is path planning for AAM vehicles. Note that in our work, we take the paths of vehicles as inputs and do not optimize the paths—instead, we focus on assigning delays to vehicles either on the ground or in the air given predetermined paths. However, path planning could be used as an initial step for generating vehicle trajectories in our experiments, and then our traffic flow management methods could be used. The single-agent pathfinding problem is the basic building block for a single vehicle to plan its route between origin and destination subject to obstacles and other en route constraints. AAM paths have a continuous curvature requirement whereby the paths must not be a series of straight line segments but must be "smooth" to be flyable. One approach uses the Rapidly-exploring Random Trees algorithm to generate collision-free waypoints for AAM; then, these paths are smoothed to satisfy the

continuous curvature requirement [89]. Given the complexity of the path-planning problem, computational intelligence methods inspired by nature have been used, including particle swarm optimization [90], [91]. Multi-agent pathfinding involves computing collision-free paths for multiple agents. Conflicts between vehicles can be solved beforehand (i.e., pre-departure) [92] or in an online setting while vehicles are airborne [93]. Other work decomposes the multi-agent pathfinding problem into a sequence of single-agent pathfinding problems to improve computational efficiency [94].

2.4.1 Market-based Approaches

Market-based approaches have been studied for strategic demand management and tactical deconfliction in the aviation context, including airport/airspace slot auctions [95], [96], slot trading during Ground Delay Programs [97], and mobility permits for airspace sector access [98]. More recently, proposals have considered auctions and other market-based mechanisms for AAM airspace use [99], [100]. While auctions have been a controversial topic in conventional aviation, especially in the United States [101], the proposed privatization of AAM service providers [102] could make market-based mechanisms such as auctions feasible ways of allocating and pricing airspace resources. Auctions for congestion management have been studied primarily for road networks, including for congestion pricing in a downtown area [103] and for managing autonomous traffic in an intersection [104]. The latter idea was extended to account for bids from chains of cars with a proportional payment mechanism, along with a "wallet" that controls how cars bid as they traverse their trajectory [105]. Our work on a cost-aware traffic management protocol using second-price auctions for AAM is not included in this thesis but can be found in [106].

2.4.2 Our Approach

In the centralized setting, we adapt the classic TFMP model for AAM traffic and evaluate system efficiency and fairness, including when operators prioritize different fairness metrics.

Recognizing that AAM traffic is more dynamic (i.e., appearing with less lead time than commercial traffic), we evaluate alternative methods to incorporate dynamic demand. In the decentralized setting, we develop an AAM congestion management protocol that minimizes information sharing between vehicles and airspace sectors. As with the centralized setting, we evaluate system efficiency and fairness. We also consider the federated setting, where adjacent regions can be controlled by different service suppliers that utilize different traffic management methods.

Chapter 3

Centralized AAM Traffic Flow Management

This chapter focuses on developing centralized traffic flow management techniques for AAM operations. In high-demand scenarios, centralized resource allocation has the potential to increase overall efficiency and safety. Furthermore, strategically planning ground and airborne delays can reduce the amount of tactical coordination required, thereby improving predictability.

A third-party service supplier may run centralized traffic flow management for AAM operations. They would require operators to share trajectory information with them. Since this is similar to conventional air traffic optimization formulations, we adapt the air traffic flow management problem (TFMP) formulation for AAM traffic management. We evaluate the trade-offs between efficient and fair solutions in a simulated setting. Since operators may have different fairness preferences, we study the impact of different fairness preferences on efficiency and fairness. In contrast to commercial aviation, operations may appear with little lead time (due to the dynamic nature of AAM operations), so we consider how to handle flights with low file-ahead time.

3.1 Background

Our approach to centralized AAM traffic management is primarily motivated by four observations. First, to scale AAM operations to meet the increasing demand, it is important to operate efficiently and minimize delays (in this context, delay can be defined as the difference between the desired and actual operation times). Second, the equitable and fair allocation of airspace resources is important, especially in the presence of a large number of aircraft operators; yet, fairness is often the first casualty in the quest for efficiency [19], [20]. Third, AAM operations are likely to result in competing interests and strategic behavior on the part of the various participants, including first-mover advantages and attempts by aircraft operators to monopolize the airspace. It is essential that the airspace, as a public good, remains accessible to all. Finally, the lessons learned from conventional traffic management can be leveraged in the design of a flexible, future-proof next-generation aviation infrastructure.

The overarching goals of our approach to *centralized* AAM traffic management are to:

1. Explicitly incorporate fairness into traffic flow management decisions while accommodating user-specific fairness requirements.

2. Support a range of aircraft operator preferences and vehicle capabilities, as reflected in their delay sensitivity and mission requirements

3. Support traffic with variable file-ahead times, thereby enabling novel applications such as on-demand mobility, exploration, quick package delivery, etc.

3.1.1 Traffic Flow Management Problem (TFMP)

We build off of the TFMP for our centralized AAM traffic flow management setup. This section is meant to highlight the setup and assumptions of the TFMP—the baseline formulation for the TFMP adapted for AAM (with mathematical notation) is shown in Section 3.2. The TFMP aims to assign delays to a set of flights operating in an airspace to ensure that

capacity restrictions are upheld. The airspace is divided into sectors with time-dependent capacity limits to reduce computation time, and time is discretized into time steps. Each flight has a desired four-dimensional trajectory (4DT), which includes the origin airport, sequence of en route sectors, destination airport, and associated scheduled times at each resource. Each flight has individual characteristics, such as feasible arrival times at each resource, minimum time spent in each sector (based on vehicle capabilities or preferences), and maximum airborne delay (based on energy constraints).

This is a centralized setup because all flights are expected to share their full trajectory information with the central planner. As discussed in Section 1.6, complete information sharing should lead to high efficiency. In this chapter, we assume that flights share full trajectory information; however, we will relax this assumption in Chapter 5 when considering a federated setting. Since operators may desire setups with less information sharing, we consider a decentralized approach in Chapter 4. The central planner (typically an Air Navigation Service Provider like the FAA) then has the authority to assign ground and airborne delays to flights to ensure that airport and sector capacity constraints are not violated. Ground delays are generally preferred over airborne delays since they are less costly and safer. This is particularly true for AAM vehicles, which are expected to have limited energy capacity.

To adapt the TFMP for AAM, we utilize a similar setup with vertiports rather than airports and smaller sectors to correspond to the lower speeds of AAM operations. AAM vehicles also have lower endurance (i.e., shorter ranges) than commercial flights, so we adjust each operation's feasible arrival times and maximum delays accordingly. The information-sharing requirements for flights remain the same: full four-dimensional trajectory information, along with feasible arrival times at each resource, minimum time spent in each sector (based on vehicle capabilities or preferences), and maximum airborne delay (based on energy constraints).

3.1.2 Fairness Considerations

Revising trajectories to mitigate congestion requires assigning delays to each flight (i.e. the aircraft takes off, lands, or accesses a region of airspace at a different time than its operator desires). Depending on the desired trajectory of a flight and the capacities of the airspace sectors it traverses, the amount of delay assigned to each flight can be different. Consequently, the objective of maximizing system efficiency is equivalent to minimizing the sum of delay costs for all the flights. The resulting solution, although efficient, could have an uneven distribution of delays across flights and aircraft operators, raising questions of fairness. Additional considerations include ensuring fairness across flights of differing durations, flights traversing different regions of the airspace, and flights with varying performance capabilities.

When there is only one constrained resource (airspace sector or vertiport), First-Come-First-Served (FCFS) based on the original scheduled arrival times at that resource is a reasonable definition of fairness. However, when a single flight traverses multiple congested resources and FCFS is enforced at each of them, the flight may get excessively penalized as delays from the first resource get compounded further downstream in the trajectory (see example in Figure 3.1).

Other notions of fairness may be preferable in this networked resource allocation setting. A few examples, which we will describe in greater detail later are:

1. As much as possible, maintain the scheduled arrival order of flights at each resource. In other words, if flight f_1 was scheduled to arrive at a resource before another flight f_2, then we prefer that it does so in the final schedule. In this case, a reversal is when f_2 arrives at this resource before f_1. This is unfair to flight f_1 since, in the original schedule, it was supposed to arrive before f_2. We will formalize this metric of fairness as minimizing the *total number of flight-reversals* across all resources.

2. Find the most constrained resource in a flight's trajectory. This resource would introduce the maximum FCFS delay if it were the only constrained resource present in

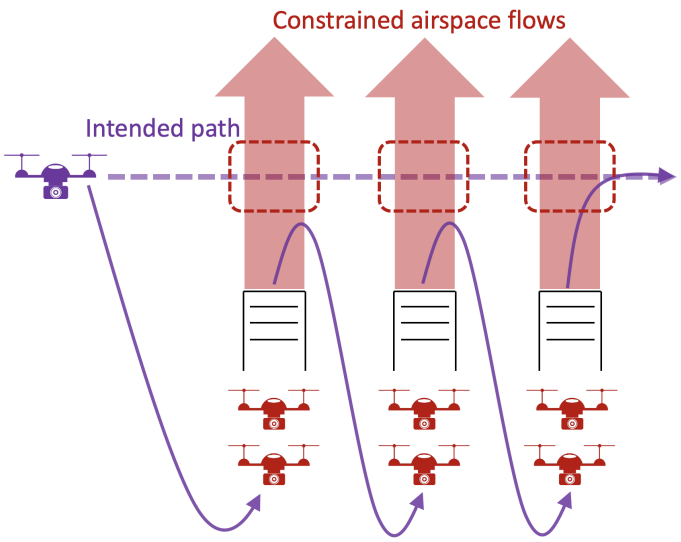


Figure 3.1: The purple drone joins the end of the queue at every constrained resource, following a First-Come-First-Serve policy. The dotted purple line denotes the desired trajectory, while the solid purple line denotes the realized one.

the trajectory. One could argue that this is the minimum delay that a flight on this trajectory ought to endure, and the optimization should try to minimize or at least equalize any delay beyond this minimum value across flights. We will formalize this notion of fairness as minimizing the *time-order deviation (TOD)* for all flights in the schedule.

Several nuances need to be addressed when incorporating fairness into AAM traffic flow management. A direct consequence of having multiple candidate metrics of fairness is that it is not apparent which one is better or more widely acceptable to AAM operators. Although intuition would suggest that enforcing some degree of fairness would result in a loss in system efficiency (i.e., the sum of flight delay costs will increase), the nature of this trade-off is not well understood. Furthermore, higher traffic demand, especially in urban areas, would result in a larger number of congestion points (airspace sectors or vertiports) for each flight trajectory. Without explicit planning, the presence of multiple congested resources can significantly decrease the fairness of the solution. Therefore, we must incorporate fairness early in system design, as concerns of unfairness will only grow with traffic volume.

Fairness in the networked setting has been previously studied in the context of air traffic control flow management [47], [48]. In ATFM, airports typically remain the primary choke points, and the FCFS schedule emerging from a congested airport is usually prioritized over other constraints. It remains to be seen whether AAM operations follow the same patterns or have more choke points.

3.1.3 Operator Preferences and Capabilities

Given the diverse set of AAM operators, we must be cognizant of significant differences amongst different system users. Potential aspects of variability include:

- *Mission requirements:* The *file-ahead time* is how far in advance an operator files a flight plan before the scheduled departure time. AAM traffic management needs to support on-demand operations with short file-ahead times (e.g., package-delivery or UAM) as well as scheduled operations with long file-ahead times (e.g., planned video inspections or fixed-schedule air taxis). Similarly, some missions, such as UAM or package delivery, could require flying the shortest path to a destination, while others (e.g., plume tracking or traffic surveillance) could involve hovering or actively sensing and exploring a region.

- *Vehicle capabilities:* Some vehicles may have a wider range of feasible speeds, resulting in a larger potential for airborne holding. Fixed-wing vehicles are not be able to reduce their speed below a certain critical threshold without the risk of stalling, whereas rotary-wing vehicles (like quadcopters) may be able to hover at a particular location and reduce their lateral speed to zero if needed. Vehicles may also have different endurance times and abilities to operate safely when subject to delays.

- *Preferred metrics of fairness:* Certain aircraft operators, such as those running supply and logistics missions, may have coordinated flight operations. For instance, one vehicle may bring a certain package to a warehouse, and a drone scheduled for later

delivery may transport that package to a customer. For such an operator, reversals in the order of arrivals may be of significant consequence, and they may wish to minimize the number of reversals in their final schedule. On the other hand, a point-to-point UAM operator may only be concerned about their on-time performance relative to competitors. A fair allocation would then be one that results in an equitable distribution of delays for any two flights that use the same set of resources. In such cases, minimizing the TOD might be the preferred metric of fairness rather than minimizing the reversals.

- *Aircraft operator cost/utility functions:* An operator transporting packages might be more sensitive to delays than a leisure drone photographer. Even among the flights managed by the package delivery operator, an aircraft carrying perishable goods may be more delay-sensitive than one carrying non-perishable goods.

Different operator preferences and vehicle capabilities may affect the tradeoffs between fairness and efficiency. Furthermore, the notion of what is considered fair for a pair of operators may be the same (i.e., aligned) or different (i.e., misaligned). This can affect not only the system efficiency but also determine the extent to which efficiency must be compromised to satisfy the fairness preferences of the operators. The challenge lies in developing a framework that can support diverse user preferences and constraints and equitably allocate resources. Our work quantifies the loss in efficiency and fairness for different operator traffic volumes and preferences (delay sensitivity and fairness requirements) using an AAM demand simulator developed by Airbus.

3.1.4 Variable File-Ahead Times

The last feature of our AAM traffic management framework that we wish to highlight is its ability to adapt to temporally dynamic traffic demand. Because of the need to sell tickets to passengers in advance, present-day commercial air traffic is highly predictable, with schedules

published at least a few weeks before departure. This enables the implementation of strategic congestion mitigation strategies such as re-routes and speed or altitude changes for known flight schedules hours in advance of their departure times. However, the file-ahead time for AAM operators (or vehicles) to convey their desire to utilize airspace or vertiport resources may be very short. Low file-ahead time could correspond to two scenarios: missions may be initiated with short lead times (e.g., on-demand UAM or package delivery), or they could be inherently uncertain due to a reactive sensing control loop (e.g., a car chase using a drone, air pollution monitoring, traffic surveillance, etc.). In other words, the file-ahead times could range from a few weeks for scheduled services to a few seconds for reactive sensing missions. Strategic planning requires some look-ahead and visibility into future demand. We address this concern by implementing a rolling time horizon to account for dynamic demand and variable file-ahead times. Some AAM flights may appear as *pop-ups* (flights that have a low file-ahead time such that they were not included in earlier planning horizons). Note that pop-ups are still required to share full trajectory information. We investigate the effect of increasing the proportion of flights that are pop-ups and evaluate two candidate approaches to handle them in our centralized traffic flow management framework.

3.1.5 Contributions and Main Findings

We adopt the air traffic flow management problem (TFMP) formulation for AAM traffic management. We utilize simulations, including trajectory data from an Airbus simulator, to evaluate the trade-offs between efficient and fair solutions in a practical setting. We recognize that operators may have different fairness preferences, so we study the impact of different fairness preferences on efficiency and fairness. AAM operations may appear with low file-ahead times (especially relative to commercial aviation), so we incorporate rolling time horizons and study how to handle *pop-ups* (flights that have a low file-ahead time such that they were not included in earlier horizons).

Some of our main insights are summarized as follows:

1. Although several reasonable notions of fairness can be proposed in a networked setting, they are not all equally easy to achieve in practice. More precisely, we show that some of the metrics may be orthogonal to each other and that optimizing for one may adversely affect another. For example, penalizing time-order deviation in the objective can lead to a decrease in reversals and overtaking; however, penalizing reversals or overtaking generally leads to an increase in time-order deviation.

2. The preferences of different operators can vary in a range of ways (e.g., their preferences may be aligned but differ in magnitude, or they may be misaligned), and the resulting impacts on system efficiency and fairness can differ. More specifically, for the case of two aircraft operators, we find that:

    - The greater the divergence between the fairness preferences of the two operators (both in terms of the fairness metrics and the weights given to them), the greater is the loss in both system efficiency and fairness.
    - When the operator with a weak preference for fairness has a high market share, the fairness of the resulting solution improves for both operators relative to the solution where both operators have an even market share.

3. Centralized AAM traffic management with a rolling horizon framework leads to lower efficiency and higher time-order deviation; however, the number of reversals is similar.

4. Demand with low file-ahead times (*pop-ups*) can be incorporated in the TFMP when using a rolling horizon. Pop-ups can either be inserted or forced to wait until the next horizon. With a high pop-up fraction, using shorter horizons and inserting pop-ups into the schedule is best. With a low pop-up fraction, using longer horizons is acceptable as long as pop-ups are inserted, and shorter horizons perform well when either inserting pop-ups or delaying them until the next planning horizon.

3.2 Baseline AAM Traffic Flow Management

This section presents the main formulation for the baseline centralized AAM traffic flow management problem. We describe some metrics to measure fairness and show how they can be incorporated into the optimization. We first build off the classical traffic flow management problem (TFMP) formulation [41].

3.2.1 Notation

We consider a discrete-time traffic flow management problem. The mathematical notations used in the formulation are described below.

$\mathcal{T}$: Set of time periods $\{1, \ldots, T\}$ of length ΔT

$\mathcal{A}$: Set of all vertiports

$\mathcal{S}$: Set of all airspace sectors

$\mathcal{F}$: Set of all flights

$\mathcal{O}$: Set of all operators

$\mathcal{F}^o$: Set of all flights by operator $o \in \mathcal{O}$

$C(s,t)$: Capacity of sector $s \in \mathcal{S}$ at time t

$A(a,t)$: Arrival capacity of vertiport $a \in \mathcal{A}$ at time t

$D(a,t)$: Departure capacity of vertiport $a \in \mathcal{A}$ at time t

a_f : Scheduled arrival time for flight $f \in \mathcal{F}$

d_f : Scheduled departure time for flight $f \in \mathcal{F}$

$\mathcal{S}^f$: Sequence of sectors in flight f's trajectory

$\mathcal{S}^f_j$: Next sector after j in flight f's trajectory

$\mathcal{P}^f_j$: Sector preceding j in flight f's trajectory

$orig_f$: Origin vertiport for flight f

$dest_f$: Destination vertiport for flight f

$l_{f,s}$: Minimum amount of time that flight f is in sector s

M : Maximum delay for each flight

T_j^f : Set of feasible time periods for flight f to arrive at resource $j \in \mathcal{A} \cup \mathcal{S}$ (vertiport or sector)

$\overline{T}_j^f$: Latest time in the set T_j^f

$\underline{T}_j^f$: Earliest time in the set T_j^f

$w_{j,t}^f$: A binary variable that is 1 when flight $f \in \mathcal{F}$ has arrived at resource $j \in \mathcal{A} \cup \mathcal{S}$ at or before time t

3.2.2 Formulation

We impose a maximum delay across all flights (M) such that no single flight is overly penalized. The value for M is set in advance by the service supplier before running the centralized optimization. M is used to construct T_j^f, the feasible arrival times at each resource. The objective function minimizes the total delay cost (TDC). The expression for (TDC) is assumed to be of the form $\text{TDC} = \beta(\text{GD}^{1+\epsilon} + \alpha\text{AD}^{1+\epsilon})$, where GD is ground delay, AD is airborne delay, β is delay to cost scale factor, and $\alpha \geq 1$ is the ratio of airborne delay cost to ground delay cost. Note that ϵ makes these costs super-linear in the delay duration, as we prefer even distribution of delays across flights over skewed delay distributions. For example, setting ϵ to be a small positive number (≤ 0.05) guides the optimization solver to allocate 2 minutes of delay each for two flights rather than 4 minutes of delay to a single flight, even though the total delay would be the same for both solutions. In other words, this super-linear cost structure breaks ties between solutions that result in the same total delay cost. Without loss of generality, we set $\beta = 1$. Since $\text{TD} = \text{AD} + \text{GD}$, we have:

$$\text{TDC} = \alpha\text{TD}^{1+\epsilon} + (1-\alpha)\text{GD}^{1+\epsilon} \tag{3.1}$$

If a flight departs at time t, then its ground delay (GD) is $(t - d_f)$. Also, if this flight lands

at time t, the total delay (TD) is $(t - a_f)$. Thus, TDC can be rewritten as below.

$$\text{TDC} = \sum_{f \in \mathcal{F}} \Big(\sum_{t \in T^f_{dest_f}} \alpha (t - a_f)^{1+\epsilon} (w^f_{dest_f,t} - w^f_{dest_f,t-1}) - \sum_{t \in T^f_{orig_f}} (\alpha - 1)(t - d_f)^{1+\epsilon} (w^f_{orig_f,t} - w^f_{orig_f,t-1}) \Big) \quad (3.2)$$

The key aspect of the formulation that lends computational tractability to larger-scale problems is the choice of the decision variable $w^f_{j,t}$, which is a binary variable that is non-decreasing in time (Constraints (3.3g) and (3.3h)). Flight f is said to enter a resource i (which could be a vertiport or a sector) *at* time t if $(w^f_{i,t} - w^f_{i,t-1}) = 1$. The following constraints must be satisfied:

$$\sum_{f \in \mathcal{F}:\, orig_f = k} (w^f_{k,t} - w^f_{k,t-1}) \leq D(k,t),\ \forall k \in \mathcal{A}, t \in \mathcal{T} \quad (3.3a)$$

$$\sum_{f \in \mathcal{F}:\, dest_f = k} (w^f_{k,t} - w^f_{k,t-1}) \leq A(k,t),\ \forall k \in \mathcal{A}, t \in \mathcal{T} \quad (3.3b)$$

$$\sum_{f \in \mathcal{F}:\, i \in \mathcal{S}^f, j = \mathcal{S}^f_i} (w^f_{j,t} - w^f_{i,t-1}) \leq C(j,t),\ \forall t \in \mathcal{T} \quad (3.3c)$$

$$w^f_{i,t} = 0,\ \forall f \in \mathcal{F}, t = \underline{\mathrm{T}}^f_{i-1}, i = \mathcal{S} \cup \mathcal{A} \quad (3.3d)$$

$$w^f_{i,t} = 1,\ \forall f \in \mathcal{F}, t = \overline{T}^f_i, i = \mathcal{S} \cup \mathcal{A} \quad (3.3e)$$

$$w^f_{i,t} - w^f_{j,t-l_{f,j}} \leq 0,\ \forall f \in \mathcal{F}, t \in T^f_i, \quad i \in \mathcal{S}^f : i \neq \text{orig}_f, j = \mathcal{P}^f_j \quad (3.3f)$$

$$w^f_{i,t-1} - w^f_{i,t} \leq 0,\ \forall f \in \mathcal{F}, i \in \mathcal{S}^f, t \in T^f_i \quad (3.3g)$$

$$w^f_{i,t} \in \{0,1\},\ \forall f \in \mathcal{F}, i \in \mathcal{S}^f, t \in T^f_i \quad (3.3h)$$

Constraint (3.3a) enforces departure capacity for each vertiport at each timestep by summing across all flights that departed. Constraint (3.3b) does the same for arrivals, and constraint (3.3c) does the same for enroute sectors by tracking the current location of each

flight f based on values of $w^f_{s,t}$ for consecutive s in S_f, the list of sectors that f traverses. If sector capacity is set to one, this is similar to ensuring minimum separation between vehicles, and if sector capacity is greater than one, additional tactical deconfliction may be needed to ensure minimum vehicle-to-vehicle separation is not violated. Constraint (3.3d) ensures that a flight does not reach a sector before the earliest feasible time. Analogously, constraint (3.3e) enforces that a flight must arrive at a sector before the latest feasible time. The minimum time to be spent in each sector is described in Constraint (3.3f).

3.3 Incorporating Fairness

We focus on three candidate notions of fairness, which we describe qualitatively below. We then incorporate them into the baseline TFMP formulation.

3.3.1 Reversals and Overtaking [47]

We start with reversals and overtaking, which are closely related. A reversal is when the order of arrivals of two flights is *reversed* relative to the original schedule, while overtaking is the time duration of the reversal. Suppose the original schedule has Flight A arriving before Flight B at a resource. After flights are delayed due to traffic flow management, if Flight B ends up arriving before Flight A, Flight A would experience a reversal. The time duration between Flight B and Flight A's actual arrival times would be the amount of overtaking. If this time duration is 5 min, it would indicate that Flight B *overtook* Flight A by 5 min. For reversals and overtaking, a fair solution is one in which the relative ordering of arrivals at any resource is preserved according to the original schedule. This original schedule is based on the initial trajectory information shared by flights. We want to minimize *reversals* in the ordering of flight arrivals at a sector or a vertiport relative to the originally scheduled ordering. Reversals are defined between flights. In settings with multiple operators, the definition of reversals does not change, as we still care about reversals regardless of whether

it is between two flights of the same operator, or between flights from different operators. A few additional variables for incorporating reversals are defined below.

R^j : Pairs of reversible flights at resource j

$T^r_{f,f',j}$: Set of time periods common for flights f and f' where a reversal could occur at resource j. These are times that both vehicles could utilize resource j.

λ^o_{rev} : Penalty factor for reversals for operator o

λ^o_{over} : Penalty factor for overtaking for operator o

For reversals, we define a new variable $s_{f,f',j}$ which is 1 if flight f' arrives before flight f at resource j, where f was scheduled to arrive before f', and 0 otherwise. In the objective function, we sum the previously defined TDC with the total number of reversals multiplied by a weight λ_{rev}.

$$\min \quad \text{TDC} + \sum_{o \in \mathcal{O}} \lambda^o_{\text{rev}} \sum_{j \in S, (f,f') \in R^j, f \in \mathcal{F}^o} s_{f,f',j} \tag{3.4}$$

The following constraint must be satisfied:

$$s_{f,f',j} = \max\left(0, w^{f'}_{j,t} - w^{f}_{j,t}\right) \forall t \in T^r_{f,f',j} \tag{3.5}$$

For overtaking, we define a new variable $s^i_{f,f',j}$ which is 1 if flight f' arrives but flight f does not arrive by time $\underline{\text{T}}^f_j + i$ in resource j, where f was scheduled to arrive before f', and 0 otherwise. This new variable s is a binary indicator of whether two flights are in reversed order at a given time. Summing the s variables gives the overtaking value between two flights if their order has been reversed. The objective function looks similar to incorporating reversals, but note that $s^i_{f,f',j}$ is summed over the cardinality of $T^r_{f,f',j}$.

$$\min \quad \text{TDC} + \sum_{o \in \mathcal{O}} \lambda^o_{\text{over}} \sum_{j \in S, (f,f') \in R^j, f \in \mathcal{F}^o}^{|T^r_{f,f',j}|} s^i_{f,f',j} \tag{3.6}$$

The following constraint must be satisfied:

$$s^i_{f,f',j} = \max\left(0, w^{f'}_{j,\underline{T}^f_j+i} - w^{f}_{j,\underline{T}^f_j+i}\right) \tag{3.7}$$

3.3.2 Time-order deviation [48]

In this section, we describe the time-order deviation metric used to quantify fairness. We calculate the first-come-first-serve (FCFS) arrival time $FCFS^f_i$ for each flight f at resource i that it goes through, assuming that i was the only constrained resource. With FCFS, arrival slots are assigned to flights according to the original schedule ordering. For each flight, we then calculate the maximum FCFS delay d^{FCFS}_f.

$FCFS^f_i$: First-come-first-serve arrival time for flight f at resource i assuming that i was the only constrained resource ($i \in \mathcal{S} \cup \mathcal{A}$)

d^{FCFS}_f : Maximum FCFS delay for flight f

$c^f_{TOD}(t)$: Additional delay cost when flight f is delayed for time t

λ^o_{TOD} : Penalty factor for time-order deviation for operator o

Just as with the penalty factor for reversals, λ^o_{TOD} is operator-specific. The intuition behind time-order deviation is as follows. When there are multiple constrained resources, each flight should not expect to incur any less delay than it would incur as a result of only the most constrained resource along its route. In other words, there is a notion of *expected delay*, that every flight is inherently entitled to be assigned, and only delays beyond this expected delay should be equalized among the multiple flights. Thus, for every flight $f \in \mathcal{F}$, the maximum delay that it would have been assigned as a result of only the most constraining resource is

$$d^{FCFS}_f \triangleq \max_{i \in \mathcal{S} \cup \mathcal{A}} FCFS^f_i \tag{3.8}$$

Thus, the modified optimization problem is

$$\min \quad \text{TDC} + \sum_{o \in \mathcal{O}} \lambda^{o}_{\text{TOD}} \sum_{f} \sum_{t=a_f}^{T} c^{f}_{TOD}(t)(w^{f}_{dest_f,t} - w^{f}_{dest_f,t-1}),$$

$$\text{where} \qquad c^{f}_{TOD}(t) = (\max\{0, t - a_f - d^{FCFS}_{f}\})^{1+\epsilon}. \tag{3.9}$$

3.4 Variable File-Ahead Times

The standard TFMP formulation assumes that the demand is deterministic and known well in advance. Given the on-demand nature of many AAM applications, this is not a safe assumption. Whereas commercial aviation flight schedules are known well in advance, AAM demand may materialize with little notice. When flight schedules are known well in advance, the centralized optimization can be run over longer time horizons with more flights to find a globally optimal solution. By contrast, when flight schedules are known with low lead times, the planning horizon is smaller, and it is not possible to proactively plan as many trajectories at once to minimize delay costs and maximize fairness. We expect, though, that there is still a benefit to strategic traffic management over relying solely on a tactical system like First-Come-First-Serve (FCFS). First, we discuss a rolling time horizon implementation that reduces the required file-ahead time for flights. Next, we discuss how to handle *pop-up* flights that have a low enough file-ahead time such that they were not included in the appropriate planning time window.

3.4.1 Rolling Time Horizon Implementation

One way around this challenge is to implement a rolling time horizon. With a rolling horizon of length H (in time periods), we intend to solve the TFMP once for every horizon (i.e., every H time periods). If we solve a horizon at time t, we solve the TFMP for flights with scheduled departure times in the $[t, t + H - 1]$ range. Once a flight is assigned a revised

schedule, it is fixed and acts as a constraint for flights in the next horizon. Each flight must file its flight plan before the start of the time interval that contains its scheduled departure time. For example, if $H = 5$ min, we solve the AAM traffic flow management problem at 6:00, 6:05, 6:10, and so on. Flights departing between 6:05 and 6:10 must file before 6:05. At 6:05, all flights scheduled to depart between 6:05 and 6:10 are considered for planning (from takeoff to landing), and their revised schedule constrains flights and resource capacities in subsequent time-intervals. For comparison, we also solve the baseline in an on-demand fashion such that one flight is scheduled at a time, in order of their scheduled departure time.

3.4.2 Pop-up Flights

A *pop-up* flight is a flight that appears after the time horizon containing its scheduled departure time has been solved. Let r_f denote the time a flight files its trajectory, and let d_f denote its scheduled departure time. The difference between the scheduled departure and filing times is called the *file-ahead time* and is expressed as $\Delta_f = d_f - r_f$. Note that $\Delta_f \geq 0$, since $d_f > r_f$. The starting time of the horizon that contains the scheduled departure time of flight f is denoted as h_f. The rolling horizon implementation thus far works if the following conditions hold for every flight f:

$$h_f = \max(H * i) \mid H * i < d_f, \text{ where } i = 1, 2, 3, ... \tag{3.10}$$

$$\Delta_f > d_f - h_f \tag{3.11}$$

Thus, if a flight wants to depart toward the end of a horizon (i.e, $d_f - h_f$ is large), then its Δ_f must be high.

In some settings, it may be reasonable to require that all flights have sufficient file-ahead times. However, it may be unfair or unreasonable to require flights to file their trajectories before the start of a horizon, particularly for long horizons that require a high Δ_f. We expect that while some flights will have sufficiently high file-ahead times, others will not.

This raises the question of how to incorporate flights with a low file-ahead time in the TFMP framework. We call a flight a *pop-up* if equation (3.10) and (3.11) do not hold. A pop-up flight files its trajectory after the start of the horizon that it wants to depart in. There are several ways to account for this dynamic demand. An extreme way is to abandon the strategic TFMP framework and tactically schedule flights in the order they file, but this would remove the benefits of scheduling multiple flights simultaneously. Instead, we prefer to incorporate the dynamic demand in the existing framework. We consider two such options for handling pop-ups below:

1. **Insert pop-ups:** Consider a pop-up flight f with a departure time d_f such that $h_f < d_f < h_f + H$. By the time flight f files its trajectory, non-pop-up flights scheduled to depart between h_f and $h_f + H$ will have already been scheduled. When a pop-up files its flight plan, we run a one-flight TFMP with vertiport and sector capacities reduced to account for previously scheduled flights. Note that we are not modifying previously scheduled flights, so the TFMP will have to find available capacity for the pop-up. If the maximum delay M for flight f is not high enough and demand is very high, the TFMP may be infeasible. We do not consider such cases.

2. **Delay pop-ups:** Consider the same situation above. Rather than insert the pop-up into the schedule, we delay the pop-up until the next horizon. The scheduled departure time at the origin and the scheduled arrival times at the en route sectors and destination remain the same. However, the pop-up's feasible times (e.g., earliest departure time) are updated to reflect its initial delay because of its shift to the next horizon. The initial pop-up delay is equal to $h_f + H - r_f$ with the following bounds: $0 < h_f + H - r_f < H$. In the worst case, the pop-up will file right after the start of the horizon ($r_f - h_f \approx 0$) and incur a delay close to H.

There are qualitative trade-offs between the two pop-up options. Option 1 (inserting pop-ups) may lead to an inefficient schedule for the current horizon, exacerbating delay in

the next horizon. Option 2 (delaying pop-ups) allows the pop-up to be batched with the flights of the next horizon but forces pop-ups to incur an initial delay before being solved by the TFMP. We will quantify the differences between these two options in the experiments described in the next section.

3.5 Experimental Results

3.5.1 Scenario Generation

We use a package delivery scenario created by Airbus where four operators in Toulouse, France have warehouses on the outskirts of the city and make deliveries in locations randomly distributed around the city [107]. The vertiports are located at the warehouses. Traffic at vertiports is determined through a Poisson process. Each flight has a desired 4D trajectory (three spatial dimensions with time as the fourth dimension). Only the outbound flight segments, from the warehouse to the delivery site, are considered for simplicity. We used two demand scenarios: 50 flights/hour and 25 flights/hour per vertiport. Figure 3.2 shows the scenario with 50 flights/hour.

One of the key requirements of the AAM traffic flow management problem formulation is that time is discretized into timesteps. We rounded sector entry and exit times to the nearest 60 s, while ensuring that each flight spent at least one timestep in each sector. We set a sector time discretization threshold of 3 s and omit a sector from a flight's trajectory if it spent less than 3 s in it. Also, the formulation requires that a flight may only traverse through a sector once. We smoothed the trajectory in eight instances where a flight entered a sector multiple times. For example, a flight that entered Sector A, briefly left for Sector B, then reentered Sector A would be modified to stay in Sector A. This is an edge case of the simulated trajectories that we fix to have smooth trajectories.

An additional factor that we accounted for was maximum battery life, which we assumed to be 20 min. We used the remaining battery life and the unimpeded time-to-destination

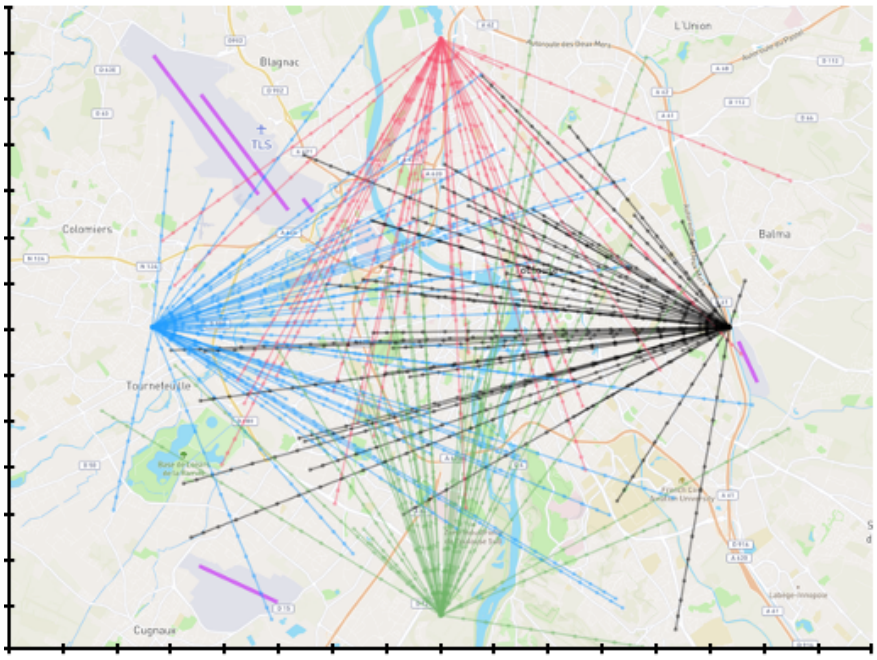


Figure 3.2: Flight trajectories shown from 4 vertiports in a 16 km × 14 km region, with axis ticks along the border denoting 1 km sector boundaries. Purple lines indicate local airports.

to calculate an upper-bound on airborne delay for each flight at each sector. Table 3.2 lists some additional parameters used for the experiment. With the given parameters, vehicle speeds are between 60 and 85 km/h (note the lower speed since these are smaller delivery vehicles). One important parameter is the ratio of airborne delay cost to ground delay cost, represented by α. With higher values of α, airborne delay will be more penalized relative to ground delay. Thus, the total delay (ground + airborne delay) may increase to minimize airborne delay. In contrast, with lower values of α, the solution will approach the minimum total delay solution. We took this ratio to be three based on [108], although different values of α could easily be justified.

3.5.2 Fairness-Efficiency Trade-offs

We seek to evaluate the fairness-efficiency tradeoff when incorporating one of three fairness metrics: reversals, overtaking, or time-order deviation. Recall that the weight that a fairness metric is given is represented by λ^o_{rev}, λ^o_{over}, or λ^o_{TOD}, for reversals, overtaking, and time-order

Table 3.2: List of Parameters

Parameter	Value
Timestep Size	60 s
Sector X-Y Dimensions	1 km $\times$ 1 km
Sector Z Dimension (Height)	65 m
Sector Capacity	1 per sector
Departure Capacity	2 per timestep
Sector Discretization Threshold	3 s
Maximum Battery Life	20 min.
Airborne Delay Cost to Ground Delay Cost Ratio	$\alpha = 3$

deviation, respectively. We vary these values to generate fairness-efficiency curves. We use four main variations of the TFMP objective function: total delay cost only ("baseline") and total delay cost with penalization of reversals, overtaking, or time-order deviation. We use total delay cost as a measure of efficiency (refer to Equation (3.2)). Note that total delay cost is distinct from total delay, as it penalizes airborne delay three times more than ground delay. In this initial round of experiments, we assume that the TFMP is only solved once. This means that there are no pop-ups, as all flights have sufficiently large file-ahead times such that the system is aware of them at the beginning of the experiment.

Incorporating fairness metrics in the objective function results in an inherent tradeoff between fairness and efficiency, measured in terms of the total delay cost. The baseline formulation's objective function is simply the total delay cost without any fairness considerations. Thus, when incorporating fairness metrics in the objective function, the total delay cost either remains the same or increases as the additional terms drive the solution away from the optimal delay cost. In return, we expect fairness to increase. Additionally, we want to evaluate the effect of incorporating one fairness metric in the objective on other fairness metrics.

Figure 3.3 shows the average number of reversals per flight and the total delay cost when minimizing the total delay cost for various scenarios. The objective function in the "Baseline" case minimizes the total delay cost. The objective in the other three cases ("Reversal",

"Overtaking", "TOD") includes total delay cost but also one of the three fairness metrics, where TOD stands for time-order deviation. Recall that λ is the weight given to the fairness term in the objective. Two baselines are shown: the black circles correspond to a high-demand scenario (vertiport demand of 50 flights/hour), and the black squares correspond to a low-demand scenario (25 flights/hour). For each scenario, there is one data point for the baseline case, but several data points for reversals, overtaking, and TOD, corresponding to different λ^o_{rev}, λ^o_{over}, and λ^o_{TOD} values, respectively.

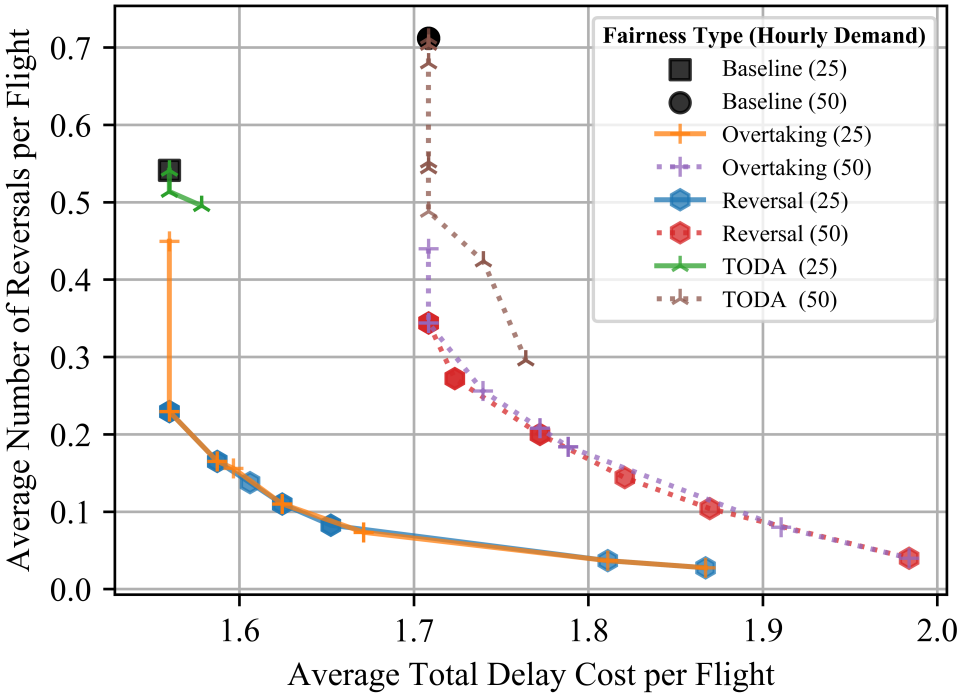


Figure 3.3: Reversals vs. Total Delay Cost (TDC) when incorporating different fairness metrics. The hourly demand level is shown in parentheses.

We first look at the results of incorporating reversals as a fairness metric in the low-demand scenario, shown as blue hexagon points with legend entry "Reversal (25)". As λ_{rev} increases, the number of reversals decreases, and the total delay cost increases relative to the baseline, shown as a black square with legend entry "Baseline (25)". For small λ_{rev} values, reducing the number of reversals with no increase in total delay cost is possible. For example, when $\lambda_{\text{rev}} = 0.4$, the number of reversals per flight decreases to 0.23 (compared to 0.54 in

the baseline) with no increase in total delay cost. With further increases in λ_{rev}, decreases in reversals are smaller and become increasingly expensive in terms of the total delay cost. At $\lambda_{\text{rev}} = 10$, the optimal solution has only 3 reversals (equivalent to an average of 0.03 reversals per flight) but an average delay cost per flight of 1.86 (a 19% increase compared to 1.56 in the baseline). Overall, as λ_{rev} increases, the relationship between reversals and total delay cost resembles exponential decay. This is because to prevent a pair of flights from being reversed, it may be necessary for one flight to incur excess delay. In the absence of limitations on the maximum delay a flight can endure, the number of reversals could be driven to zero at the cost of a very high total delay cost.

In the high-demand scenario, the new baseline, shown as a black circle with legend entry "Baseline (50)", has a higher average number of reversals and average total delay cost than the low-demand scenario, "Baseline (25)". This is expected, as more congestion leads to more flight interactions and potential for reversals. Incorporating reversals in the objective has a similar effect as doing so with lower demand. The trade-off curve has a similar shape, and for very high λ_{rev}, the average number of reversals approaches zero while the average total delay cost increases substantially.

The orange curve with legend entry "Overtaking (25)" and the purple curve with legend entry "Overtaking (50)" correspond to incorporating overtaking in the objective, in the low-demand and high-demand scenarios, respectively. Incorporating overtaking produces nearly identical results as when incorporating reversals, as can be seen with the overlapping lines. In many cases, they have identical optimal solutions, not only with regard to fairness and efficiency but also concerning schedule and delay allocation. This is expected since the two fairness metrics are intertwined, with overtaking measuring the magnitude of time duration that a given pair of flights was reversed. Whereas reversals and overtaking are nearly in lockstep, time-order deviation (shown in green and the dashed brown curves) behaves differently from reversals or overtaking. Note that the first points on the time-order deviation curves overlap with the baselines. As λ_{TOD} increases, incorporating time-order deviation can

decrease the average number of reversals with little to no increase in the total delay cost, especially for the high-demand scenario. (Recall that we are incorporating only one fairness metric at a time in the objective.) However, incorporating time-order deviation does not decrease the average number of reversals as much as explicitly incorporating reversals. For larger λ_{TOD}, the optimal solution does not change, and no further reductions in reversals are apparent. Overall, the takeaway is that a) incorporating reversals in the objective leads to a decrease in reversals at the expense of total delay cost and b) somewhat surprisingly, incorporating time-order deviation also leads to a reduction in reversals, with the expected trade-off between fairness and efficiency (i.e., total delay cost).

Figure 3.4 is similar to Figure 3.3 but displays average overtaking instead of average reversals on the y-axis. As before, we have two baselines in black and curves generated by incorporating one fairness metric at a time with different λ values. Since reversals and overtaking are closely related, it comes as no surprise that the efficiency-fairness trade-offs of both are similar. For very large λ_{rev} or λ_{over}, it is possible to reduce overtaking to near zero, albeit at a great expense to the total delay cost. Incorporating time-order deviation decreases overtaking, just as it did for reversals.

Figure 3.5 shows the average time-order deviation (in minutes) on the y-axis. We first consider how incorporating time-order deviation in the objective affects the average time-order deviation per flight. As λ_{TOD} increases (represented by the green and dashed brown curves), the average time-order deviation decreases, and the average total delay cost increases. The decreases in time-order deviation are modest but more pronounced in the high-demand scenario, for which the tradeoff between the average time-order deviation and the average total delay cost is linear. At $\lambda_{\text{TOD}} = 2$, the average time-order deviation decreases by 4.5%, and the total delay cost increases by 3%. The increase in total delay cost happens despite a reduction in total delay (from 208 min in the baseline to 201 min)—this is because the airborne delay (which is 3x more costly than ground delay) increases. Notably, incorporating reversals or overtaking leads to an *increase* in time-order deviation. This is in contrast to

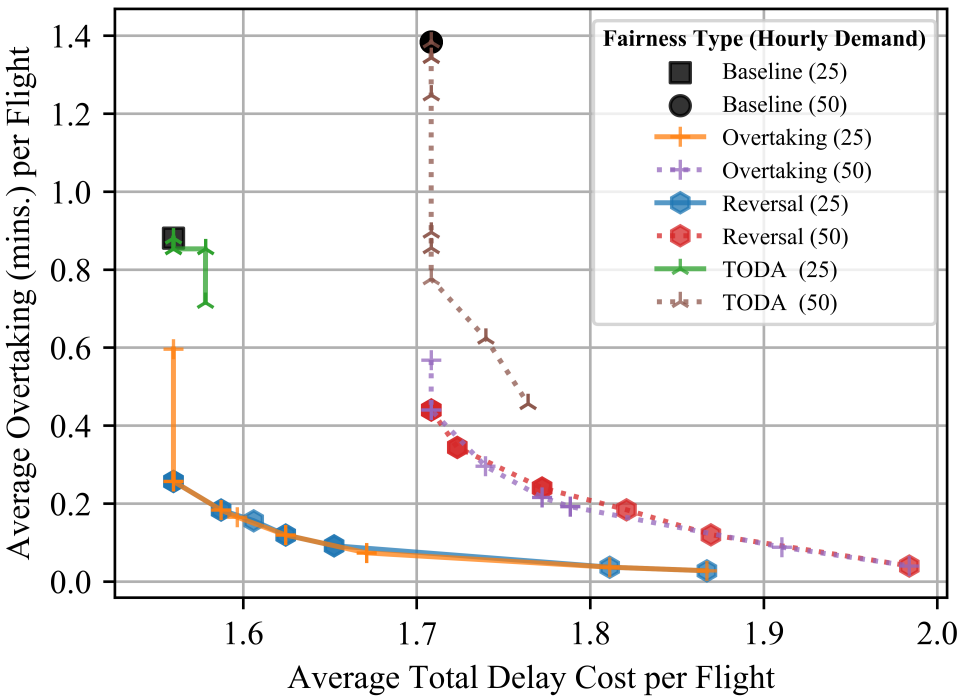


Figure 3.4: Overtaking vs. Total Delay Cost (TDC).

how incorporating time-order deviation led to a *decrease* in reversals or overtaking.

While penalizing reversals or overtaking can drive its value to near zero (as in Figures 3.3–3.4), it is not possible to drive the average time-order deviation to zero, no matter how large λ_{TOD} gets. This is inherently due to the way time-order deviation is defined (3.9). Given a delay allocation, time-order deviation can be reduced by reallocating delay from Flight A to Flight B, where Flight A is a flight with delay *greater* than its maximum expected delay and Flight B is a flight with *less* delay than its maximum expected delay. If all flights are like Flight A and have delay assigned greater than or equal to their maximum expected delay, there are no flights (like Flight B) to reallocate delay to. Instead, time-order deviation can only be decreased by also decreasing total delay. Thus, when all flights have delay assigned that is greater than or equal to their maximum expected delay and the total delay has been minimized, then the time-order deviation is also minimized. This is the case with the high-demand scenario. No flight was assigned delay less than its maximum expected delay, and minimizing total delay rather than total delay cost leads to an optimal solution with

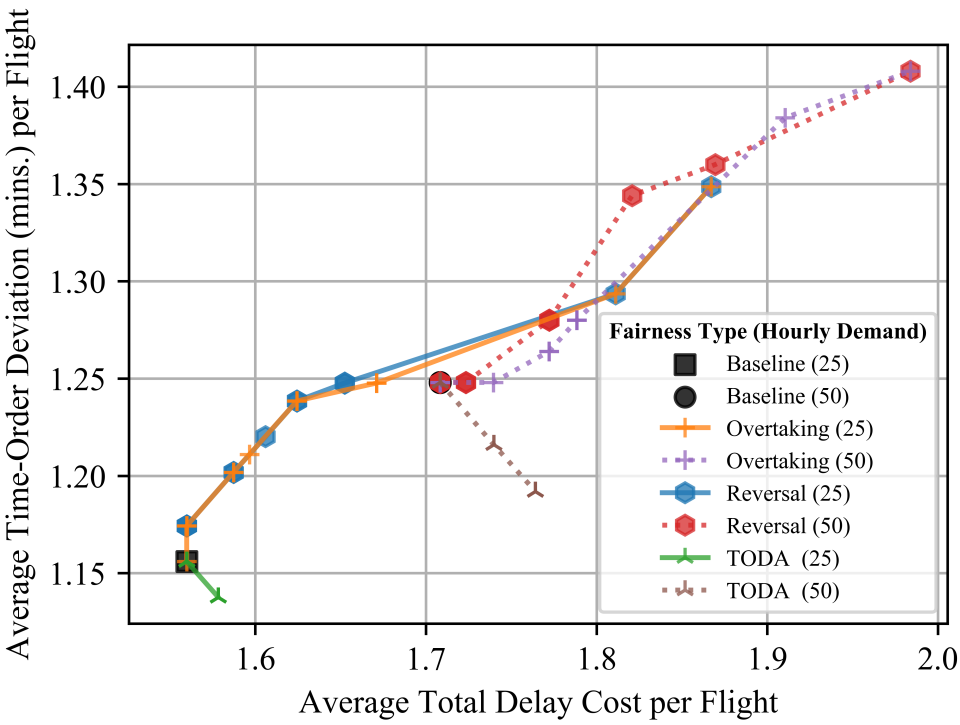


Figure 3.5: Time-Order Deviation vs. Total Delay Cost (TDC).

the same 201 min of total delay seen with $\lambda_{\text{TOD}} = 2$. Incorporating reversals or overtaking results in a 17% increase in average time-order deviation in the low-demand scenario and a 13% increase in the high-demand scenario. In contrast, incorporating time-order deviation can slightly decrease reversals or overtaking.

While the improvement in the average time-order deviation when penalizing time-order deviation may appear modest, there is another benefit. Since the cost coefficient for time-order deviation is a super-linear function, evenly distributed time-order deviation is preferred over lopsided distributions. As such, incorporating time-order deviation also reduces the standard deviation of time-order deviation across flights. As λ_{TOD} increases, the standard deviation decreases; $\lambda_{\text{TOD}} = 2$ results in a 27% decrease in the standard deviation of time-order deviation relative to the baseline. Further, incorporating time-order deviation bounds the loss in efficiency while remaining robust to the choice of λ_{TOD}.

3.5.3 Operator Fairness Alignment

We expect that operators will often have different λ weights of fairness and different notions of fairness. For example, an operator conducting deliveries where the order of operations is very important may care about reversals much more than time-order deviation. The effect of a reversal may propagate downstream to the operators' later operations. On the other hand, an operator passing through multiple constrained resources may care more about time-order deviation (additional delay beyond expected delay) than reversals. We consider a case with two operators where one operator's notion of fairness associated penalty factor is fixed and the other operator's notion of fairness and penalty factor are variable.

The previous section described results when all operators have aligned notions of fairness and the same λ weight of fairness. We refer to this scenario as a *perfect alignment* of fairness. However, operators can have different efficiency-fairness trade-off preferences, as reflected by the λ value or different notions of fairness itself (e.g. reversals or time-ordered-deviation). Whenever two operators have the same notion of fairness, they are said to be *aligned*. Whenever two operators have different notions of fairness (e.g., one prefers to minimize reversals while the other prefers to minimize TOD), they are said to be *misaligned*. In this experiment, Operator 2 has a constant λ weight of fairness for time-order deviation ($\lambda^2_{\text{TOD}} = 3$). On the other hand, Operator 1's notion of fairness and λ weight vary, as shown in Figure 3.6. Three subplots show the change in total delay cost, number of reversals, and time-order deviation for each operator relative to a perfect alignment of $\lambda^1_{\text{TOD}} = \lambda^2_{\text{TOD}} = 3$. In this section, we refer to increases/decreases relative to perfect alignment as "increases" and "decreases", respectively. Starting with the top subplot, we observe that efficiency decreases in two cases: in the misaligned region with the operators having different notions of fairness and in the aligned region with λ^1_{TOD} much higher than λ^2_{TOD}. In the misaligned region, for $\lambda^1_{\text{rev}} \leq 5$ the decrease in total delay cost from prioritizing time-order deviation for Operator 2 is overridden by the worsening of total delay cost due to minimizing reversals for Operator 1.

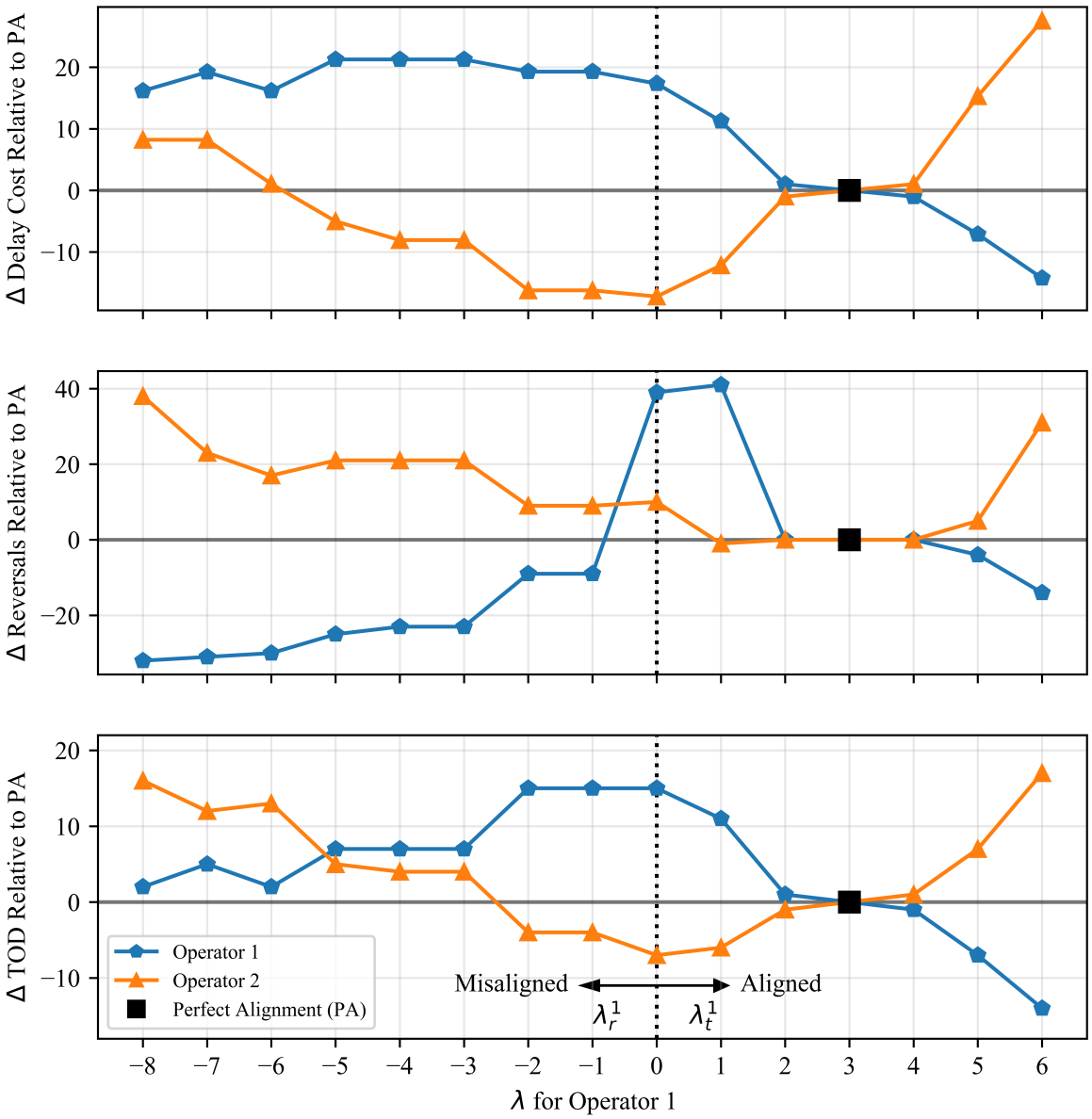


Figure 3.6: Operator efficiency and fairness with varying λ^1_{TOD} or λ^1_{rev}, with fixed $\lambda^2_{\text{TOD}} = 3$. Operator 1 and 2 are misaligned when operator 1 prioritizes reversals, aligned when operator 1 prioritizes time-order deviation, and perfectly aligned when $\lambda^1_{\text{TOD}} = \lambda^2_{\text{TOD}} = 3$.

For $\lambda^1_{\text{rev}} > 5$, total delay cost increases for both operators. In the region near $\lambda^1_{\text{rev}} = \lambda^1_{\text{TOD}} = 0$, delay cost increases for Operator 1 and decreases for Operator 2, but these mostly balance each other out. In the aligned region where Operator 1 has a stronger fairness preference than Operator 2, Operator 1's decrease in total delay cost is outweighed by Operator 2's increase.

As with efficiency, fairness decreases when one operator has a much stronger notion of fairness than the other operator, in both the misaligned and aligned case. Looking at the middle subplot of Figure 3.6, for all misaligned cases, Operator 1's reversals naturally decrease since they are the only operator minimizing reversals, and Operator 2's reversals increase. For $\lambda^1_{\text{rev}} \geq 10$, the increase in Operator 2's reversals outweighs Operator 1's decrease leading to an overall increase in number of reversals, but for smaller λ^1_{rev} the total number of reversals decreases. The increase in Operator 1's (and the system) number of reversals peaks around $\lambda^1 = 0$. This is because Operator 1 has weak fairness preferences for either reversals or time-order deviation, and minimizing either could reduce the number of reversals. Moving onto the bottom subplot of Figure 3.6, in the misaligned region, Operator 1's preference of minimizing reversals increases time-order for both operators. In the aligned region, Operator 1's time-order deviation increases if its fairness weight is lower than Operator 2's but decreases if its fairness weight is higher.

3.5.4 Operator Market Share

Up to this point, we have assumed equal market share between operators. There is concern that operators' market share may lead to unfair delay allocations. Smaller operators may be effectively crowded out by larger operators. As an initial step, we evaluate the effect of market share on system and per-operator efficiency and fairness in a case with two operators. We keep the total number of flights constant and randomly select the appropriate number of flights to switch operators. We test several market share splits in a two-operator scenario.

To test the effect of market share on fairness, we fix the fairness parameters in a two-operator setting. In this scenario, both operators care about time-order deviation, but Operator 1 has a weaker preference for fairness than Operator 2 ($\lambda^1_{\text{TOD}} = 0.5$ and $\lambda^2_{\text{TOD}} = 1$). We then vary the market share of Operator 1 from 0.2 to 0.8 in increments of 0.1, as seen on the x-axis of Figure 3.7. We switch the designated operator of an appropriate number of random operations to create a scenario with any given market share split. We run 50

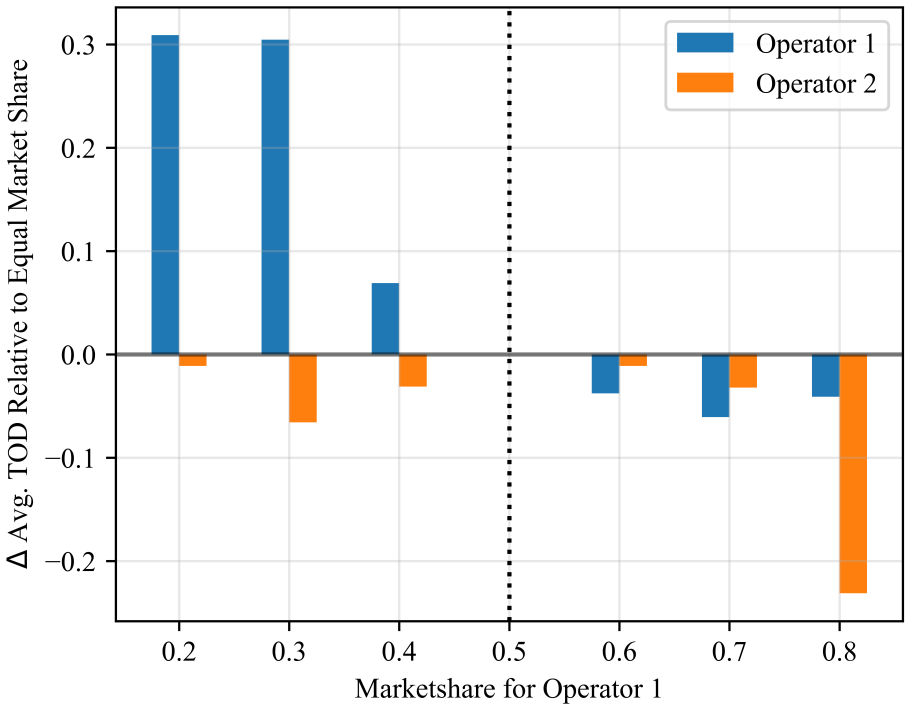


Figure 3.7: Effect of market share on Time-Order Deviation, with fixed $\lambda^1_{\text{TOD}} = 0.5$, $\lambda^2_{\text{TOD}} = 1.0$.

experiments for each market share split, with each experiment having the same number of flights per operator but different flights belonging to each operator. The y-axis of Figure 3.7 shows the change in average time-order deviation relative to average time-order deviation with equal market share on the y-axis. Fairness improves for both operators when the operator with a weak fairness preference (Operator 1) has a higher market share, and the operator with a strong fairness preference (Operator 2) has a low market share. In contrast, the fairness of Operator 1 deteriorates if its market share is reduced.

One way to rationalize this is as follows. If an operator has a very high λ (e.g., Operator 2) but a low market share, it will be relatively easy to accommodate their requests without penalizing other operators. On the other hand, if the market share of this operator was high, then the overall solution would preferentially satisfy the fairness requirements of this operator and might have to impose excessive penalties (in terms of efficiency and fairness) on others.

3.5.5 Rolling Horizon

In this section, we discuss the results when using a rolling horizon of varying size for the high-demand scenario (50 flights/hour). Figure 3.8 shows the total number of reversals vs. the total delay cost for the case with no rolling horizon ("Deterministic") as in the previous sections. This means that all the demand is known in advance, such that all of the flights can be optimized at once. We then consider cases with 15-minute and 5-minute rolling horizons. This means we solve batches of flights based on their scheduled departure time. With a 15-minute rolling horizon, we group flights that are scheduled to depart between $[t, t+15)$ and $[t+15, t+30)$, and so on. For now, we do not consider pop-up flights, meaning we assume that all flights have a file time before the start of their horizon. For example, if a flight wants to depart at $t+5$, it shares its information before t, which is the start of the rolling horizon containing its scheduled departure time. As before, we experimented by incorporating one fairness metric at a time in the objective. We omit points for overtaking since its behavior follows the trends of reversals. Note that fairness is only incorporated among the flights that are planned in a given horizon. For example, when incorporating reversals for a given horizon, we only consider schedule reversals between flights in that horizon. We also show the result for solving the AAM traffic management problem such that one flight is scheduled at a time in order of their scheduled departure time ("Myopic"). This allows for the shortest file-ahead time for each flight, as each flight only needs to share its information immediately before its scheduled departure.

We first look at the impact of the rolling horizon in the baseline case where no fairness metric is incorporated. These points are shown with black shapes. Recall that in our implementation of the rolling horizon, flights from the previous time step cannot be changed, eliminating the ability to shuffle those flights with flights from the current time step. While this lowers the number of reversals, it comes at the expense of total delay cost. Thus, compared to the deterministic baseline, both of the rolling horizon baselines (15-minute and

5-minute horizons, represented by a circle and triangle, respectively) have a lower number of reversals and a higher total delay cost. Flights are planned for the 5-minute rolling horizon with even less information than with the 15-minute rolling horizon; thus, it is not surprising that the total delay cost for the 5-minute rolling horizon is greater than that of the 15-minute rolling horizon. The myopic, on-demand case has, by far, the highest total delay cost, which makes sense given that only one flight is solved at a time, negating much of the benefit of optimization.

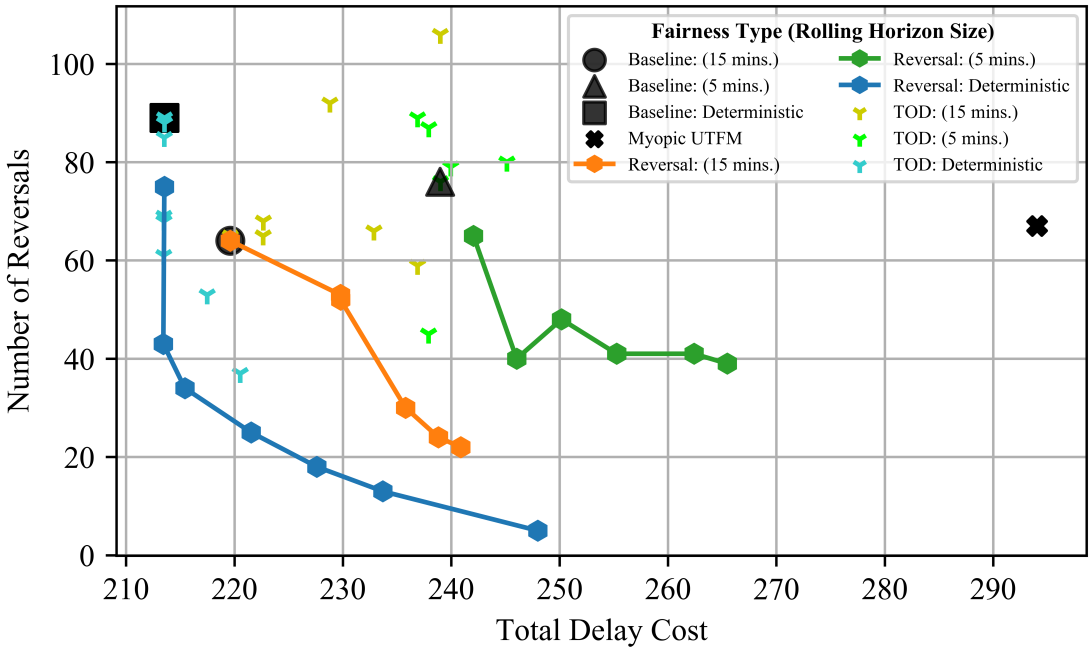


Figure 3.8: Reversals vs. Total Delay Cost (TDC), by the length of the rolling horizon. "Myopic" is when flights are planned one by one according to the scheduled time of departure.

Note that solutions can have the same total delay cost but a different number of reversals. This indicates that the order of flights can be altered without impacting the overall delay. The deterministic case (blue line for reversals and light blue points for time-order deviation) exhibits similar behavior as before. The points for the 15-minute rolling horizon are generated by incorporating one fairness metric (either reversals or time-order deviation) and varying its fairness penalty. We see that incorporating reversals (orange line) in a 15-minute rolling horizon decreases reversals, but not as much as in the deterministic case.

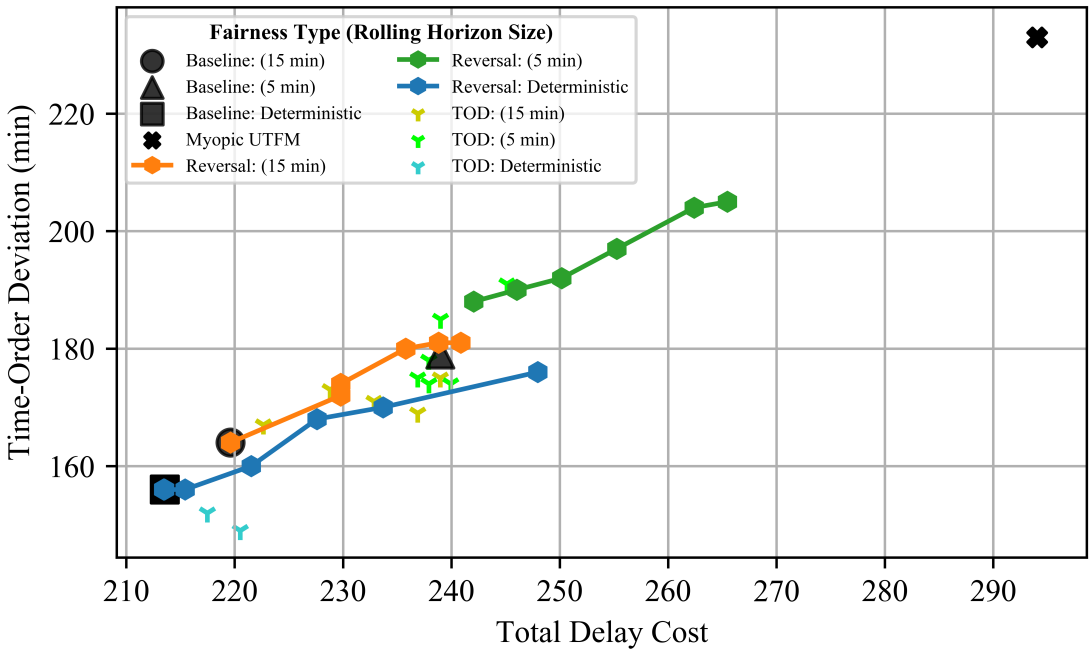


Figure 3.9: Time-Order Deviation vs. Total Delay Cost (TDC), by the length of the rolling horizon.

However, incorporating time-order deviation (yellow points) generally increases the number of reversals with a 15-minute horizon. This is in direct contrast to the deterministic case, where incorporating time-order deviation led to a decrease in reversals. With the 5-minute rolling horizon, incorporating reversals (green hexagon points) follows the expected behavior: decreasing the number of reversals leads to increased total delay cost. Also, the number of reversals plateaus after very little increase in total delay cost. This is likely since, with a shorter horizon, there are fewer flights in each horizon and thus less leeway to adjust schedules to untangle reversals. As with the 15-minute horizon, incorporating time-order deviation in the 5-minute horizon (green points) generally increases the number of reversals. The myopic case again has the highest total delay cost but also has a much higher time-order deviation than all other results.

Figure 3.9 is similar to Figure 3.8 but shows time-order deviation instead of reversals on the y-axis. When comparing the baseline points (in black), time-order deviation increases as the horizon size decreases. As seen before, incorporating reversals increases time-order

deviation, and this trend is seen across all horizon sizes tested. On the other hand, incorporating time-order deviation decreases time-order deviation in the deterministic case (light blue points in the bottom-left of Figure Figure 3.9) and has more mixed results with the 5-minute and 15-minute rolling horizons. An important consideration of the rolling horizon implementation is the runtime, which we define as the computational time to optimize all the traffic demand (flights). As the horizon size increases, more flights are included in each time interval, adding additional variables to the TFMP formulation and leading to a longer runtime. On the other hand, a larger horizon size means that fewer time horizons need to be solved. For example, the simulation lasts 87 min, so with a horizon size of 45 min, just two horizons are needed (and thus, two TFMP optimization problems are solved). In contrast, with a horizon size of 5 min, 18 horizons are needed. In the deterministic setting, only one TFMP problem needs to be solved.

Figure 3.10 shows how runtime varies for different horizon sizes when incorporating either reversals or time-order deviation (TOD). The runtime for the deterministic setting (i.e., all flights are known in advance) is also shown. This corresponds to solving all of the flights at the same time. We generate the lines by varying horizon size. The blue line corresponds to an objective with reversals, whereas the orange line corresponds to an objective with time-order deviation. Thus, there are two "deterministic" points in black, one for each of the fairness metrics. With time-order deviation, total runtime decreases as horizon size increases. With reversals, runtime decreases as horizon size increases from 5 to 25 min but increases thereafter. The TFMP formulation with reversals requires more variables than the formulation with time-order deviation and, therefore, takes longer to solve for a larger number of flights. For all scenarios tested, the total runtime is reasonable (at most about 5 min), considering that the simulation spans nearly 90 min.

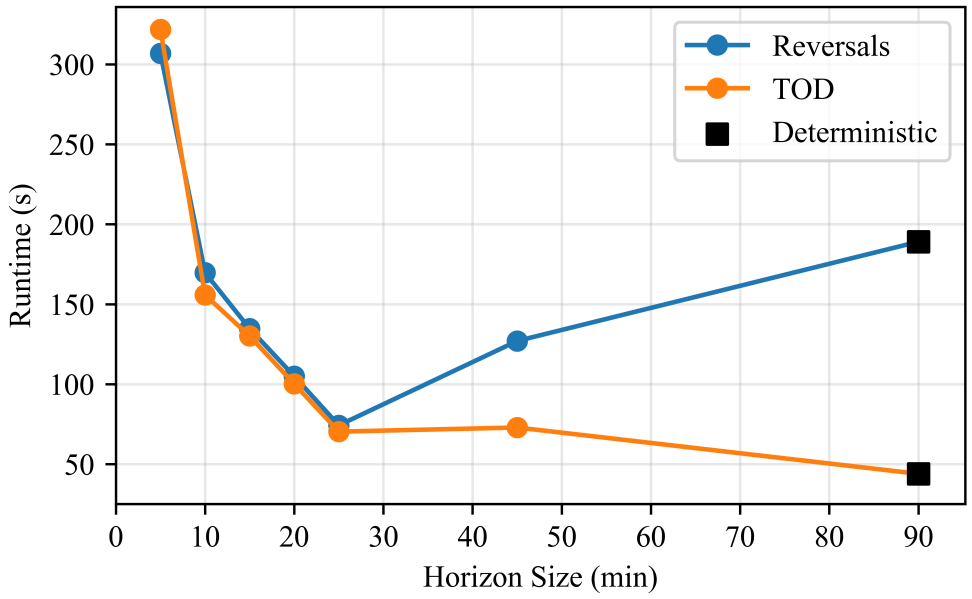


Figure 3.10: Runtimes for different rolling horizon sizes. $\lambda_{\text{rev}} = \lambda_{\text{TOD}} = 1$ for all horizon sizes.

3.5.6 Pop-up Flights

In these sets of experiments, we relax the assumption that there are no pop-ups. We define the proportion of all flights that are pop-ups as the *pop-up fraction*, denoted by p. Given p, we randomly select the appropriate number of flights to be pop-ups. For bookkeeping purposes, flights with departure times that are equal to the start of a horizon are eligible to be pop-ups, but their departure times are shifted to 1 s later. Each pop-up is randomly assigned an integer file time r_f with a discrete uniform distribution on the interval $[h_f \,..\, d_f]$, Recall that h_f is the start time of the horizon containing flight f, and d_f is the scheduled departure time of f. Besides pop-up fraction, there are three other parameters we vary. We have two options for handling pop-ups: Option 1 (inserting pop-ups) and Option 2 (delaying pop-ups). We also vary the horizon length, with larger horizon lengths meaning that fewer planning horizons—with several flights in them—are solved. Finally, we still have the choice of which TFMP objective function to use. Since the selection of pop-up flights is random, we can test scenarios with different sets of pop-up flights. However, when making direct

comparisons between horizon lengths, pop-up options, or TFMP objective function, we use the same random seed so that the same set of flights are pop-ups.

We now turn to the case when we have dynamic demand in the form of pop-up flights. We experimented with several different pop-up fractions, horizon lengths, and TFMP objective functions. Note that in the absence of pop-ups, it is most efficient for the horizon to be as long as possible so that the TFMP has knowledge on as many flights as possible. However, with a larger horizon, flights are forced to file their flight plans earlier to avoid being pop-ups.

The most basic intervention to pop-up flights is to effectively prohibit their existence by telling operators that they must have an early enough file-time such that there are no pop-ups. This is not always practical, as AAM operations are on-demand and may not be able to have an early enough file time. Thus, we tested two approaches to handling pop-ups: Option 1 (insert pop-ups) and Option 2 (delay pop-ups). We are interested in the trade-offs between these different parameters. The following results are with the baseline TFMP objective. As expected, we find that efficiency and fairness decrease as the pop-up fraction increases. Figure 3.11 shows the average total delay cost per flight across the two pop-up options, two horizon lengths (5 and 30 min) in scenarios with pop-up fractions 0.1 and 0.5. Each bar represents an average of 100 runs. We start with a pop-up fraction of 0.1. We see that Option 1 with a 30-minute horizon performed the best overall (it had the lowest average total delay cost per flight). Option 2 with a 30-minute horizon performed poorly because of the high delay assigned to pop-ups that need to wait until the next horizon. With a 5-minute horizon, Option 1 and Option 2 performed similarly to each other, with Option 2 having a very slight edge. Figure 3.12 is arranged similarly to Figure 3.11 but shows average reversals per flight rather than average total delay cost per flight. While inserting pop-ups with a long horizon (Option 1 with a 30-minute horizon) had the best efficiency, delaying pop-ups with a short horizon (Option 2 with a 5-minute horizon) had the fewest reversals per flight.

Moving on to pop-up fraction 0.5, Option 1 with the 5-minute horizon had the lowest

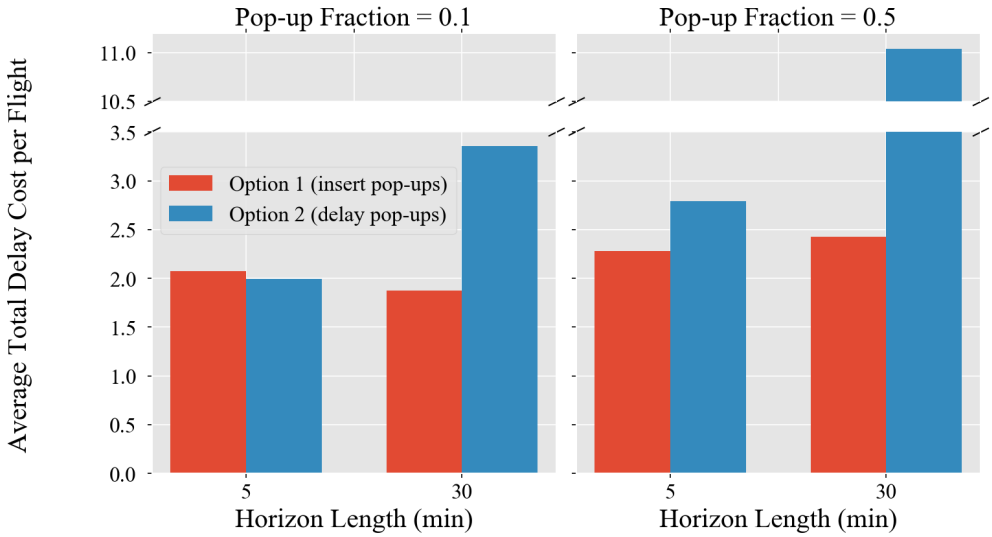


Figure 3.11: Average Total Delay Cost per Flight by Horizon Length and Pop-up Option.

total delay cost and average reversals. In contrast to with pop-up fraction 0.1 and a 5-minute horizon, Option 2 had a higher delay cost than Option 1. In addition, Option 1 with the 30-minute horizon performed worse than with the 5-minute horizon. The best combination was Option 1 with a 5-minute horizon. With more pop-ups, the TFMP horizon needs to be smaller to reduce the number of non-pop-ups that are scheduled before (and thus block) each pop-up. Consider a pop-up with a desired departure time of 9:02 and a horizon with a start time h_f of 9:00. With a 30-minute horizon, non-pop-ups scheduled to depart between 9:00 and 9:30 will be scheduled before the pop-up, but with a 5-minute horizon, only non-pop-ups scheduled to depart between 9:00 and 9:05 will block the pop-up. There is a break in the y-axis of Figure 3.11 and Figure 3.12 because Option 2 with a 30-minute horizon has such a high delay. With pop-up fraction 0.1, we saw that Option 2 does not pair well with a long horizon because of the delay that pop-ups are forced to incur before they are scheduled. This trend is further highlighted with a pop-up fraction of 0.5.

We saw similar trends when the objective incorporated reversals, overtaking, or time-order deviation. Thus, we do not show versions of Figure 3.11 and Figure 3.12 for these objectives. Instead, Table 3.3 shows the effect of incorporating reversals on total delay cost and reversals. Each row corresponds to a combination of pop-up fraction, horizon length,

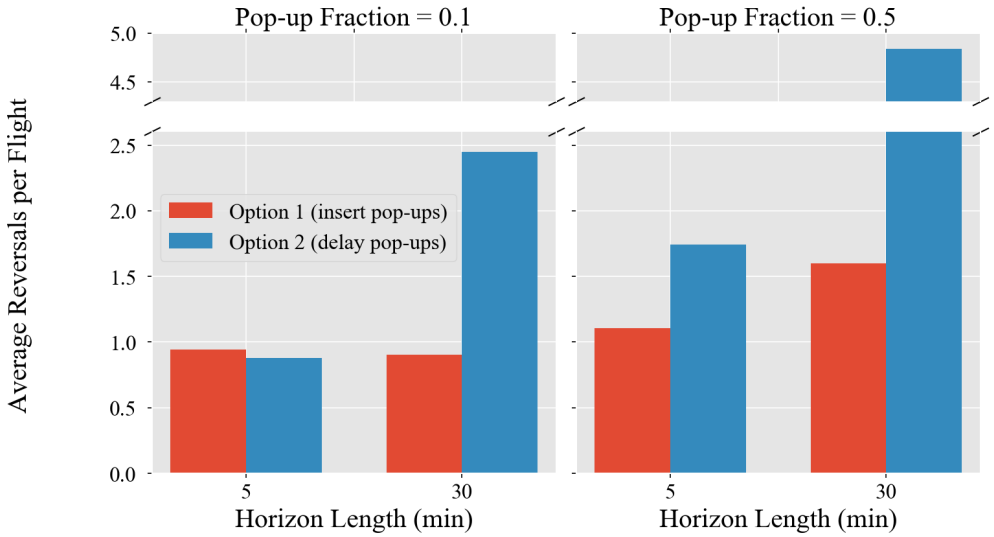


Figure 3.12: Average Reversals per Flight by Horizon Length and Pop-up Option.

and pop-up option. Note that the two combinations that were shown to be impractical are not included (30-minute horizon with Option 2 for pop-up fraction 0.1; 5 or 30-minute horizon with Option 2 for pop-up fraction 0.5). Table 3.4 is similar to Table 3.3 but shows the effect of incorporating time-order deviation in the objective.

Parameters			**Penalizing Reversals**	
Horizon Length (min)	**Pop-up Option**	**Pop-up Fraction**	**% Change Total Delay Cost**	**% Change Reversals**
5	1: Insert	0.1	0.17	-26.8
5	2: Delay	0.1	4.54	-26.6
30	1: Insert	0.1	6.96	-11.8
5	1: Insert	0.5	0.12	-14.9
30	1: Insert	0.5	0.57	-2.34

Table 3.3: Change in total delay cost and reversals when incorporating reversals in the TFMP.

The first takeaway is that even in the rolling horizon setting, incorporating reversals/time-order deviation is effective in improving fairness. Next, we observe that fairness improves by a larger relative amount with 0.1 pop-up fraction than with 0.5 pop-up fraction. All else equal, from an efficiency and fairness perspective, the system benefits from having fewer pop-ups. We also note that we get larger improvements in fairness with a 5-minute horizon

rather than a 30-minute horizon. With a longer horizon, pop-ups are scheduled after more non pop-ups (in Option 1 and Option 2), and forced to delay longer to wait for the next horizon (in Option 2 only).

Parameters			**Penalizing Time-order Deviation**	
Horizon Length (min)	**Pop-up Option**	**Pop-up Fraction**	**% Change Total Delay Cost**	**% Change Time-order Deviation**
5	1: Insert	0.1	3.56	-7.18
5	2: Delay	0.1	4.09	-14.83
30	1: Insert	0.1	4.60	-1.71
5	1: Insert	0.5	3.36	-11.73
30	1: Insert	0.5	2.24	-5.56

Table 3.4: Change in total delay cost and TOD when incorporating TOD in the TFMP

3.6 Discussion

This chapter explored a centralized adaption of TFMP for AAM. One area of focus was the balance between efficiency and fairness of the resultant solution. In particular, we focused on the impact of aircraft operators' preferences and market shares on fairness. We considered three metrics of fairness: reversals, overtaking, and time-order deviation. We found that improving any of these fairness metrics at little cost to efficiency (i.e., little increase in total delay cost) is possible. We also found that while minimizing reversals/overtaking, time-order deviation increases, but when minimizing time-order deviation, the number of reversals/overtaking could also decrease.

In a two-operator setting, system efficiency and fairness are at their best when the two operators have the same notion of fairness and value them similarly. Additionally, fairness improves when the operator with a dominant market share has a weak preference for fairness. When fairness preferences are misaligned (i.e., operators care about different fairness metrics), the larger operator's weak preference for fairness allows the smaller operator to achieve

some fairness (based on their choice of fairness metric). We considered a rolling horizon setting with dynamic traffic demand and found that efficiency and time-order deviation worsen at shorter horizons while the number of reversals is unaffected.

We also tested our formulations in a rolling horizon framework where not every flight files their flight plan sufficiently in advance to be considered in the time horizon that contains their scheduled departure time. We considered two options for handling pop-ups: inserting them into the schedule and delaying them until the next horizon. We found that with a low occurrence of pop-ups, either using longer horizons and inserting pop-ups or using shorter horizons with either pop-up option is acceptable. However, with high occurrences of pop-ups, it is best to use a short horizon and insert pop-ups. In reality, a hybrid approach could be used, depending on the reason that a flight is a pop-up. If a flight is a pop-up due to gaming behavior or negligence, delaying it to the next horizon may be more acceptable. In contrast, inserting it into the current horizon may be preferred if a pop-up flight files late due to changing mission requirements or propagation delay. In our experiments, we found that it was beneficial to incorporate fairness into the rolling horizon framework.

3.6.1 Extensions

An area of future work is to include various uncertainties in the centralized model. This work assumes deterministic capacities and that flights comply with assigned trajectories. We also did not consider rerouting, which is common in commercial aviation and will likely occur in AAM as well. Airborne flights could reroute around congested areas, or flights on the ground could alter their route from the start to reduce the incurred delay. Approaches like trajectory option sets (TOS) exist for commercial aviation wherein each flight submits a set of unique trajectories along with acceptable delays, but such research for AAM is less explored.

In addition, there have been proposals for AAM operators not to submit specific trajectories but rather reserved volumes of airspace. There are several related research questions

on how to manage airspace reservations, requirements, constraints on requests for airspace, and fairness between different-sized operators. Finally, with the rolling horizon framework, pop-ups are at a disadvantage because flights with longer file-ahead times are allocated resources before them. In previous work, researchers have suggested that limiting the time in advance that resources can be allocated/reserved would improve fairness between early-filers and late-filers [81]. There is also the opportunity to consider how to mitigate gaming behavior by AAM operators. For example, operators may submit fake trajectories that they never intend to fly. The design of such a system and the evaluation of its efficiency and fairness remain open challenges.

Chapter 4

Decentralized AAM Traffic Management Protocol

This chapter focuses on developing a decentralized traffic management protocol for AAM. This is in direct contrast to the centralized TFMP approach covered in Chapter 3. We consider a decentralized setting for a few reasons. Operators may be unwilling to share trajectory information with a central agent. In addition, decentralized architectures are more scalable than centralized ones. As before, space is discretized into sectors. We design a decentralized protocol that a third-party service supplier could choose to deploy to distribute traffic management among sectors. We assume that sector-to-sector communication is handled by either service suppliers themselves or a communication service set up by service suppliers, similar to traffic signals in ground transportation.

Note that the protocol can alternatively be adapted to utilize vehicle-to-vehicle communication rather than sector-to-sector communication. This would be suitable for airspace where third-party suppliers have limited oversight. While we will return to this point in Section 4.7, we assume sector-to-sector communication for now.

One way to frame the work in this chapter is that the centralized method in Chapter 3 adopted a traffic management system closely aligned with conventional air traffic manage-

ment. In this chapter, we swing to the opposite end of the spectrum of centralization and consider a minimal information-sharing congestion management protocol. There are parallels to surface transportation (i.e., personal vehicles driven by individuals). Cars benefit from not needing to share information with a central agent and only need to "signal" intent (i.e., share information) for a limited period of time. Yet, it is not quite a free-for-all, as there are understood rules of the road (protocols) that must be adhered to, like traffic signals and merging etiquette. This is preferable to a free-for-all approach without traffic signals or road signs. Thus, we seek to develop an AAM traffic management protocol that mirrors the decentralized nature of surface transportation.

4.1 Background

As advanced air mobility (AAM) applications become more widespread, the skies will become more crowded. It is expected that the number of AAM operations will far exceed that of conventional aviation operations [109]–[111]. The predicted demand for AAM services has motivated large investments in drones and electric-powered Vertical Takeoff and Landing (eVTOL) aircraft [112], [113]. While the focus of research and development efforts has been on the vehicles themselves, there has been limited attention paid to the questions of how these large numbers of vehicles will operate collectively and how the traffic will be managed. Although AAM aircraft will operate a diverse range of missions, we expect that the desired trajectories will strain limited airspace resources, leading to congestion. The consequence will be delays, namely, flights not being able to fly their desired routes at the desired times. Efficient traffic management should minimize such vehicle delays. We define a *conflict* as an event where the number of vehicles that want to access a sector at a given time is greater than its capacity. We aim to design a control algorithm that satisfies sector capacity constraints in a decentralized manner.

Congestion and sector capacity constraints in AAM traffic management can be addressed

with a centralized model like the TFMP in Chapter 3. In this paradigm, the schedules and desired trajectories of all flights being planned are known in advance (as is the case with commercial aviation), and an optimization problem is solved to obtain revised schedules that ensure compliance with airport and sector capacities. This requires flights to submit full trajectories well in advance. As discussed in Chapter 1, AAM operators may prefer to share less information and not to a central agent. There are two distinguishing features of AAM, compared to commercial aviation. The first is that demand is expected to be more dynamic. Airlines typically file their flight schedule up to around a year in advance and file their flight plans at least an hour before scheduled departure. In contrast, an AAM operator, such as a delivery company like Amazon, may plan a flight with a lead time on the order of minutes. Continuing this example, delivery operators may constantly re-plan flight trajectories and destinations as new orders pop-up. To preserve flexibility, operators may only be willing to share partial trajectories. Another reason for sharing partial trajectories is to preserve competitive advantages, as sharing estimates of full trajectories (which includes the destination for a trip) may reveal sensitive information (e.g., areas of high demand for air taxi companies) that competitors can exploit.

The second distinguishing feature of AAM is the projected large scale of operations. Urban areas are projected to see a 200-fold increase in the number of flights due to UAS operations (autonomous drones for package delivery, sensor measurements, surveillance, tracking, etc.) and a 30-fold increase from UAM operations (autonomous, semi-autonomous, or piloted air taxis) [2], [3]. A fully centralized optimization approach may not scale sufficiently to support the expected levels of demand. Therefore, a decentralized approach may be more tractable. In addition, a more interpretable approach like a protocol that defines the "rules of the road" may increase robustness to communication failures, as it would not depend on a single entity performing a large-scale optimization.

4.1.1 Objectives and Properties of Decentralized Protocol

We explore a decentralized congestion management protocol. In addition to being scalable, we designed our protocol to preserve operator privacy by minimizing the amount of information shared. We propose a protocol with a prioritization scheme that tries to meet the following goals:

- Accommodate frequent re-planning: Many AAM applications are on-demand, so we want to allow for frequent re-planning of flight trajectories and destinations.

- Limited information-sharing: We want to preserve operator privacy and flexibility. Thus, we limit the amount of information shared and the number of parties that information is shared with. First, operators should be allowed to share partial trajectories to preserve flexibility and competitive advantages. Second, all vehicles should not be required to share information with a single entity. We use the term "minimal information sharing" to capture both ideas. Specifically, in each time step, operators only need to share their current sector position and the next sector that they want to proceed to in the next time step. In addition, operators do not need to broadcast their information to a central agent; they only need to communicate with their current sector and desired next sector.

- Interpretable: Our approach should be intuitive and interpretable to operators to increase robustness against communication failures. This also increases operator acceptance of delays, which often have large economic consequences, as it is easily understandable that delays are assigned in a fair, systematic manner.

- Fairness and efficiency: We expect a large number of AAM operators competing to provide different types of services, so it is important that they are treated fairly. While there are many definitions for fairness, we use the standard deviation of delays as a metric of fairness.

While the above factors may suggest that the solution is to rely purely on tactical self-separation between aircraft for safety, such an approach can lead to a significant loss in efficiency and even gridlock when traffic density is high. When aircraft perform tactical self-separation, further conflicts could arise. Therefore, we consider traffic management protocols that preserve operator privacy while providing enough structure to mitigate efficiency loss. This approach is similar to how congestion is managed on road networks, where vehicles have freedom of route and destination but must comply with general regulations (e.g., posted signs) and traffic lights.

We evaluate our protocol on two metrics: efficiency and fairness. Operators desire efficiency, measured by low levels of system delays, to increase their commercial viability and profitability. Fairness is also important to flights and operators, as operators expect equal treatment, given the diversity in the types of operations (e.g., consistent use of one airspace sector by a surveillance drone versus the use of multiple airspace sectors by a package delivery drone) as well as size of operators (e.g., large package delivery operators like Amazon prime air versus a hobby photography flight). As discussed in Chapter 3, fairness can have many definitions [49], [114], [115], and there is no consensus on a universal definition for air traffic management. We measure flight-level fairness using the standard deviation of delays across a group of flights. The lower the standard deviation of flight delays, the higher the fairness for that group of flights. Operator-level fairness is more nuanced, so we propose metrics for measuring unfairness in delay assignments across operators.

We design, analyze, and demonstrate through simulations, a congestion management protocol for AAM with the following characteristics:

1. **Avoiding gridlock**: We use the current sector position and desired next sector position of aircraft to create a directed graph. We identify "cycles", which are closed loops on the graph that represent groups of aircraft where either all of them can proceed to their desired next sector, or none of them can. In Section 4.3.1, we show how to identify and prioritize cycles to avoid gridlock, even with limited information sharing.

2. **Efficient sector deconfliction**: When the number of incoming aircraft exceeds sector capacity, a sector needs to be "deconflicted", where the sector decides which aircraft to prioritize and give permission to enter. We use the term "deconfliction" to refer to such strategic deconfliction at least one time step in advance, as opposed to the tactical self-separation of vehicles. There may be multiple sectors that need to be deconflicted. Suboptimal ordering of sector deconfliction could result in conflicting control actions, making the order in which sectors are deconflicted important. In Section 4.3.2, we use a "backpressure" metric (which measures the length of the maximum incoming queue of aircraft) to decide the order in which sectors should be deconflicted. This allows the decentralized protocol to avoid assigning conflicting control actions.

3. **Balancing efficiency, fairness, and operator privacy**: We design a flexible protocol that can incorporate different prioritization schemes. We show that one of these prioritization schemes (backpressure) results in a minimum delay solution for one time step. We also promote operator privacy by minimizing the amount of information sharing required.

4.1.2 Assumptions

We make several assumptions to highlight the intuition behind our approach. We start by assuming perfect state information, full controllability, and no uncertainty. In particular, we assume that vehicles share accurate information to the requisite entity. We also assume that vehicles comply with the decisions of the protocol (remaining in a sector, or proceeding to another). Thus, we do not account for weather disruptions or other sources of uncertainty, like pilot non-compliance. With these assumptions, there is no need to model tactical conflict resolution, as the unit sector capacities are not violated in the simulations. We do not model vehicles violating separation requirements on the boundaries of sectors (e.g., two vehicles close enough to each other to violate minimum separation requirements but in separate sectors and not violating sector capacity constraints). Note that the use of the term

"sector" here is distinct from larger, traditional air traffic control sectors. If we considered uncertainties that may make vehicles not comply with the protocol (either inadvertently or purposefully), sector capacity violations could occur. This would necessitate tactical deconfliction. Since our protocol only considers one time step at a time, system efficiency may not be as susceptible to uncertainties as other methods with longer planning horizons.

We consider a setting where there are no reroutes, and the only congestion management action that can be taken is when to allow a vehicle to enter a sector. Once a vehicle is in a sector, it cannot be forced to leave the sector. We consider a discrete-time setting, where each vehicle can only occupy one sector at any time. In particular, we assume that every vehicle either intends to a) stay in its current sector or b) move to an adjacent sector. In practice, this means that the time discretization is small enough to capture the resource utilization of the vehicles.

We also make simplifying assumptions regarding the sector geometry and capacity. First, we set the capacity of each sector to one. This can be realized in practice by defining a sector as a sufficiently small volume of airspace. Second, in this chapter, we display sectors organized in a grid-like pattern. This is purely for ease of visualization and to build intuition about the flavor of our algorithm. Generally, the sector shapes may be arbitrary, and our protocol only relies on knowledge of the adjacent sectors to any given sector. There are no constraints on the shape of sectors and the number of adjacent sectors.

4.2 Setup

We discretize space into a set of sectors $\mathcal{S} = \{s_1, s_2, \ldots, s_N\}$, represented by a rectangular grid. Recall that the use of the term "sector" here is distinct from larger, traditional air traffic control sectors. Each sector has a capacity of 1. We restrict capacity to 1 to avoid the need for tactical deconfliction within a sector. We also discretize time into time steps. In each time step, flights can move to any adjacent sector. In our examples, each sector

has up to four adjacent sectors, but there is no limit on the number of adjacent sectors. Alternatively, flights may stay in their current sector in the next time step. A flight cannot be forced to leave a sector. At each time step, the protocol decides whether or not to allow a flight into a sector. We assume that vehicles comply with the protocol, meaning they will follow instructions on remaining in a sector or proceeding to another sector. We assume that vehicles move at the same speed of 1 sector in a time step. A vehicle can only occupy one sector at any time. Figure 4.1 illustrates an example of a grid of sectors. We use the following notation in this chapter:

$\mathcal{T}$	=	Set of time periods $\{1, \ldots, t, \ldots, T\}$
$\mathcal{S}$	=	Set of sectors $\{1, \ldots, N_{sectors}\}$
$\mathcal{V}$	=	Set of aircraft $\{1, \ldots, N_{aircraft}\}$
$\mathcal{V}_a$	=	Set of active aircraft, i.e., ready to depart or currently airborne
$C(s,t)$	=	Capacity of sector $s \in \mathcal{S}$ at time $t \in \mathcal{T}$. Sector capacity is set to 1.
$orig(i)$	=	Origin of aircraft $i \in \mathcal{V}$
$dest(i)$	=	Destination of aircraft $i \in \mathcal{V}$
$d(i)$	=	Scheduled departure time of aircraft $i \in \mathcal{V}$
$a(i)$	=	Scheduled arrival time of aircraft $i \in \mathcal{V}$
$x(i,t)$	=	Sector where aircraft $i \in \mathcal{V}$ is located at time $t \in \mathcal{T}$
$\widehat{x}(i,t)$	=	Intended sector at time $t+1$ for aircraft $i \in \mathcal{V}$ based on information at t
$\mathbf{x}(t)$	=	Current sector locations for all aircraft $i \in \mathcal{V}$ at time t
$\widehat{\mathbf{x}}(t)$	=	Intended sectors for all aircraft i at time $t+1$ based on information at t
$\mathcal{G}$	=	aircraft that can proceed to their next sector
$\mathcal{H}$	=	aircraft that must hold in their current sector
$del(i)$	=	Total delay assigned to aircraft i
$del(i,t)$	=	Binary variable representing the delay assigned to aircraft i in time step t

Figure 4.1 presents an example layout of vehicles at a time step. Each grid cell is a

sector with a capacity of 1. Each arrow represents an aircraft: the tails represent their current sector $x(i,t)$ and their heads represent their desired next sector $\hat{x}(i,t)$. Given the unit sector capacity constraint, if more than one aircraft wants to enter the same sector, the protocol must perform *deconfliction*—that is, to decide which aircraft to *prioritize*, or allow to proceed in the direction of the arrow. If an aircraft i is not allowed to proceed to $\hat{x}(i,t)$, it must remain in $x(i,t)$ in time-step $t+1$. This means that $del(i,t) = 1$. In practice, aircraft i could absorb this delay on the ground (if it has not departed yet) or while airborne (through a speed change, path stretch, or hovering maneuver). By contrast, a vehicle with $del(i,t) = 0$ is assigned no delay and can proceed to its next intended sector $\widehat{x}(i,t)$.

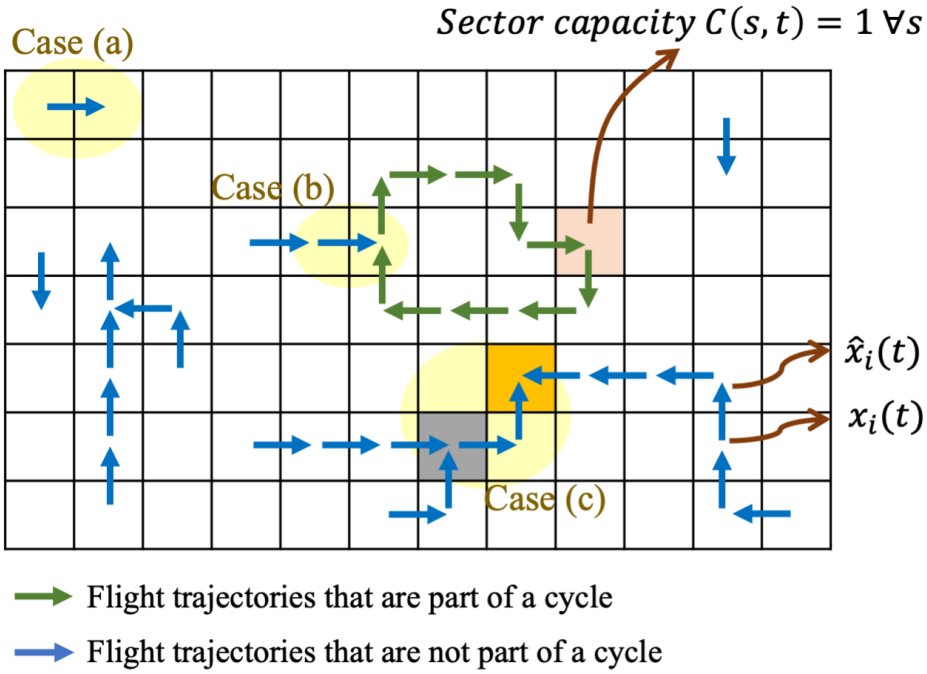


Figure 4.1: A simple example of the system state at time t, showing the current and intended sectors of all vehicles, with an example of a cycle in green.

We make a few observations about this problem setup.

1. If there is a sector into which only one vehicle wants to enter, then the optimal—and trivial—solution would be to set $del(i,t) = 0$ for that vehicle. Case (a) in Figure 4.1 shows an example of this.

2. When we connect the arrows formed by flight trajectories, we may see a closed loop, or *cycle*. Formally, in a cycle with two aircraft i and j, $x(i,t) = \hat{x}(j,t)$ and $\hat{x}(i,t) = x(j,t)$. In a cycle with more than two aircraft, for every aircraft i, there exists an aircraft j where $x(i,t) = \hat{x}(j,t)$ and an aircraft k where $\hat{x}(i,t) = x(k,t)$. An example of a cycle is shown in Case (b)—the flights in green form a closed loop. This means that either all of them are allowed to move or none of them can. Furthermore, there is no feasible way in which any additional vehicle attempting to access the sectors occupied by the cycle can be allowed to do so while the cycle exists because of capacity constraints. For example, the vehicles marked in blue that are incident on the green cycle cannot proceed while the green cycle exists. We address this in the protocol by identifying and prioritizing cycles even with limited information sharing.

3. We need to be careful about the *order* in which we deconflict sectors. For instance, in Case (c), there are two sectors to deconflict: orange and grey. If we deconflict the grey sector first and allow a flight to enter the grey sector, we necessarily have to push the current occupant of the grey sector out. This means that when we go to deconflict the orange sector, its decision will have already been made—it must accept the incoming vehicle coming from the bottom rather than the vehicle from the right. However, it would be better for the system if the orange sector accepted the vehicle from the right, as it would unblock more flights. We address this in the protocol by defining the order in which we deconflict sectors.

4.2.1 Information-sharing Constraints

In each time step, each flight i occupies a sector $x(i,t)$, which we call its *current* sector. Each flight i also has a desired *next* sector $\hat{x}(i,t)$ which it intends to occupy in the next time step. As mentioned in Section 4.1.1, we want to develop a traffic management algorithm that is fair, efficient, and allows for frequent re-planning and limited information sharing. We could solve an optimization problem with $\mathbf{x}(t)$ and $\hat{\mathbf{x}}(t)$ at every time instant t to determine which

aircraft can proceed (i.e., determine $del(i,t)$). However, this requires that all aircraft share their current location and intent with a central authority. This would not achieve our goal of minimizing information sharing.

Thus, we restrict our information sharing as follows. We require that each aircraft i conveys its intent to use sector $\hat{x}(i,t)$ to that sector. If $\hat{x}(i,t) = x(i,t)$, then the aircraft is allowed to stay in that sector. We further allow each sector s to communicate two types of information with all sectors r adjacent to s (there are up to four in a rectangular grid structure). First, they communicate the unique identity of aircraft that want to access sector s (itself). Examples of a unique identity include flight numbers or tail numbers. Crucially, sector s only shares the identity of these aircraft, but not the position. This is necessary for sectors to identify cycles in Section 4.3.1. Second, we allow sector s to signal a scalar value indicative of upstream congestion (i.e., the length of built-up queue) to its neighboring sectors r (used in Section 4.3.2). For example, sector s can convey to sector r that it has a queue of length 7 which is blocked by the aircraft wanting to proceed from s into r, but it does not reveal the location of these 7 aircraft. We assume that all sectors convey this information truthfully. Sector r thus knows that if it allows the vehicle from sector s to enter it, then an additional 7 vehicles in other sectors could proceed. We refer to this set of communication rules between sectors as the *information-sharing constraints*.

4.3 Decentralized Protocol

While being cognizant of the three observations on Figure 4.1, we present the framework for our congestion management protocol, which is run at every time step. We divide our protocol into six steps, with references to the appropriate line numbers in Algorithm 1. Steps 1-3 are completed at the beginning of every time step, and Steps 4-6 are performed for each sector with a conflict.

1. **Initialization** (Lines 1-3). We initialize two lists, a *hold* list $\mathcal{H}$ and a *go* list $\mathcal{G}$. Vehicles

in $\mathcal{H}$ will be forced to hold and stay in the same sector in the next time step, whereas vehicles in $\mathcal{G}$ will be allowed to proceed to their desired next sector. We update the list of *active* vehicles $\mathcal{V}_a$ by removing vehicles that have arrived at their destination and adding vehicles that are scheduled to take off.

2. **Identify and prioritize cycles** (Lines 4-7). From Case (b) in Figure 4.1, we know that cycles need to be identified and prioritized as soon as they appear. Until a cycle is cleared, it will block all sectors that it occupies. We identify vehicles in *cycles* ($\mathcal{V}_c$) and add them to $\mathcal{G}$. To make way for vehicles in $\mathcal{V}_c$, we add vehicles incident on cycles to $\mathcal{H}$ and force them to hold (i.e., their next sector is set to their current sector).

3. **Compute sector prioritization** (Lines 8-9). Now that the cycles have been resolved (for this time step at least), we need to decide the order in which to deconflict sectors. Case (c) in Figure 4.1 shows an example of the dependencies between sectors. We calculate the backpressure at each sector and will deconflict the sector with the highest backpressure first. We will formalize the notion of backpressure, but it provides a measure of the queue build-up incident on a sector.

4. **Loop through sectors** (Lines 10-12). Based on the order determined in step 3, we complete steps 4-6 for each sector. For the highest priority sector yet to be managed, we split the vehicles that want to use this sector in the next time step into two categories: undecided vehicles ($\mathcal{V}_u$) and decided vehicles ($\mathcal{V}_d$). $\mathcal{V}_u$ contains vehicles that the sector is *undecided* on whether to allow them to enter the sector, and $\mathcal{V}_d$ contains vehicles for whom actions are *decided* (i.e., they are in either $\mathcal{G}$ or $\mathcal{H}$).

5. Now, one of the two scenarios will occur:

    a) **Case of capacity exceeds demand** (Lines 14-15). If the sector capacity is sufficiently high to allow all inbound traffic, then we add $\mathcal{V}_u$ to $\mathcal{G}$. Sector capacity is one, so this only occurs when one vehicle wants to enter a sector.

b) **Case of demand exceeds capacity** (Lines 16-20). If there is insufficient capacity to allow all vehicles, then we use one of several prioritization schemes to pick which vehicle gets to proceed. These prioritization schemes can be on a vehicle level or operator level and are described in detail in Section 4.4. These vehicles are removed from $\mathcal{V}_u$ and added to $\mathcal{V}_d$ and $\mathcal{G}$. We keep prioritizing vehicles until all capacity is used or there are no more vehicles in $\mathcal{V}_u$.

6. **Delay all unassigned vehicles** (Lines 24-25). If capacity is fully used and there are still vehicles in $\mathcal{V}_u$, we add all $\mathcal{V}_u$ to $\mathcal{H}$ and force them to hold at their current sector.

Algorithm 1 CONGESTION-MANAGEMENT PROTOCOL($\mathbf{x}, \widehat{\mathbf{x}}$, $\mathcal{V}_a$, $\mathcal{S}$, C)

1: $\mathcal{H} \leftarrow \{\}, \quad \mathcal{G} \leftarrow \{\}$
2: $\mathcal{V}_a \leftarrow \mathcal{V}_a \setminus (i \in \mathcal{V} \mid \mathbf{x}(i) = dest(i))$
3: $\mathcal{V}_a \leftarrow \mathcal{V}_a \cup (i \in \mathcal{V} \mid d(i) = t)$
4: $\mathcal{V}_c \leftarrow$FINDCYCLES($\mathbf{x}, \widehat{\mathbf{x}}$)
5: $\mathcal{G} \leftarrow \mathcal{G} \cup \mathcal{V}_c$
6: $\mathcal{H} \leftarrow \mathcal{H} \cup (i \in \mathcal{V} \mid \exists\, g \in \mathcal{G} \mid \widehat{x}(i,t) = \widehat{x}(g,t))$
7: $\widehat{x}(i,t) = x(i,t)\ \forall\ i \in \mathcal{H}$
8: $B \leftarrow$CALCULATEBACKPRESSURE($\mathbf{x}, \widehat{\mathbf{x}}, \mathcal{V}_a, \mathcal{S}$)
9: SORT $\mathcal{S}$ IN ORDER OF B
10: **for** $s \in \mathcal{S}$ **do**
11: $\quad \mathcal{V}_u \leftarrow i \in \mathcal{V} \mid \widehat{x}(i,t) = s$ **and** $\sim (i \in \mathcal{G}$ **or** $i \in \mathcal{H})$
12: $\quad \mathcal{V}_d \leftarrow i \in \mathcal{V} \mid \widehat{x}(i,t) = s$ **and** $(i \in \mathcal{G}$ **or** $i \in \mathcal{H})$
13: $\quad$ **if** $C(s, t+1) > |\mathcal{V}_d|$ **then**
14: $\quad\quad$ **if** $|\mathcal{V}_u| \leq C(s, t+1) - |\mathcal{V}_d|$ **then**
15: $\quad\quad\quad \mathcal{G} \leftarrow \mathcal{G} \cup \mathcal{V}_u$
16: $\quad\quad$ **else**
17: $\quad\quad\quad$ **while** $C(s, t+1) > |\mathcal{V}_d|$ **do**
18: $\quad\quad\quad\quad p \leftarrow$PRIORITIZEAIRCRAFT($\mathbf{x}, \widehat{\mathbf{x}}, \mathcal{V}_u$)
19: $\quad\quad\quad\quad \mathcal{G} \leftarrow \mathcal{G} \cup p$
20: $\quad\quad\quad\quad \mathcal{V}_u \leftarrow \mathcal{V}_u \setminus p, \quad \mathcal{V}_d \leftarrow \mathcal{V}_d \cup p$
21: $\quad\quad\quad$ **end while**
22: $\quad\quad$ **end if**
23: $\quad$ **end if**
24: $\quad H \leftarrow H \cup i \in \mathcal{V}_u$
25: $\quad \widehat{x}(i,t) = x(i,t)\ \forall\ i \in \mathcal{H}$
26: **end for**
27: **return** $\mathbf{x}, \widehat{\mathbf{x}}$

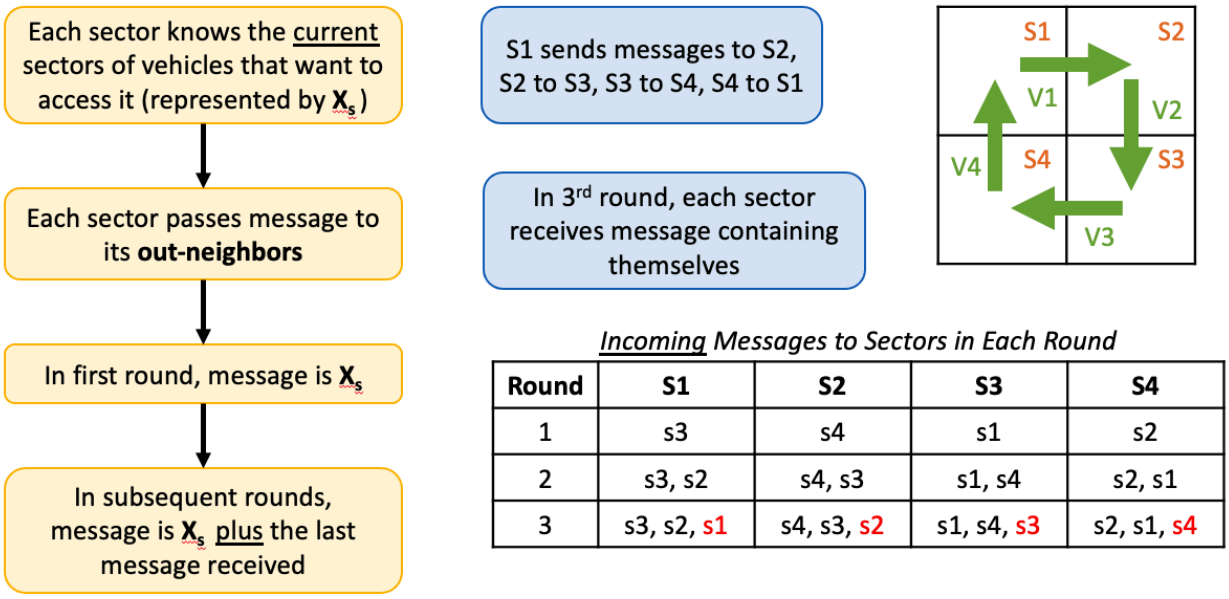
Two main components of this algorithm are the FindCycles and CalculateBackpressure functions, which we will describe next. A key requirement in designing these functions is that they satisfy the information-sharing constraints.

4.3.1 Identifying Cycles

The goal is for sectors to share limited information and identify vehicles incoming into it that are in cycles. We use an adapted Rocha-Thatte cycle detection distributed algorithm [116]. We have a finite directed graph $G := (\mathcal{S}, E)$ where the vertices are the set of sectors $\mathcal{S}$ and the edges are defined with tail $x(i,t)$ and head $\widehat{x}(i,t),\ \forall i \in \mathcal{V}$. Under our assumptions, each sector is only aware of incoming and outgoing vehicles. We use rounds of "bulk synchronous message passing" to identify cycles. For each sector, we define three sets. The first is the set that contains the current sectors of incoming vehicles $X_s = \{x(i,t) \in \mathcal{S}_a \mid \widehat{x}(i,t) = s\}$. Next, we define a sector's in-neighbors as $\mathcal{N}_s^- = \{x(i,t),\ \forall i \in \mathcal{V}_s^-\}$. These are adjacent sectors that want to hand off a vehicle to s. Similarly, we define a sector's out-neighbors as $\mathcal{N}_s^+ = \{\widehat{x}(i,t),\ \forall v \in \mathcal{V}_a \mid x(i,t) = s\}$.

In each round, each sector s passes a message to its out-neighbors $\mathcal{N}_s^+$. That is, messages are passed along the edges E, between sectors. In the first round, this message contains the current sectors of incoming vehicles into s, X_s. In subsequent rounds, each sector appends X_s to each message that they received in the previous round and passes it along. A sector s knows that it is part of a cycle if it sees itself (sector s) in a received message. Consider the example cycle shown in green in Figure 4.2. The vehicles are labeled V1–V4, and they currently occupy sectors S1–S4, respectively. Sector S1 receives a message of "S3" in the first round from sector S4. This is because a vehicle currently in S3 wants to enter S4. Note that the content of the message is the *sector* of the incoming vehicle and not the vehicle name to preserve vehicle privacy. S1 knows there is a vehicle going from S3 to S4, but it does not know its ID. In the next round of message passing, S1 sends a message of "S4, S3" to S2, since it appends the message it received in the first round (S3) to X_s (S4). In the

third round of message passing, all sectors in the cycle receive a message that contains their own sector ID. While the sectors now know the location of the cycles, they do not know the ID of the vehicles that make up the cycle. Once the cycle has been identified, it can now be prioritized. Sectors that contain flights in a cycle allow incoming flights to proceed. We assume that communication between sectors happens nearly instantaneously, such that cycles are identified and resolved in a few seconds. We do not consider technical limitations on communication.



Incoming Messages to Sectors in Each Round

Round	S1	S2	S3	S4
1	s3	s4	s1	s2
2	s3, s2	s4, s3	s1, s4	s2, s1
3	s3, s2, s1	s4, s3, s2	s1, s4, s3	s2, s1, s4

Figure 4.2: Message passing required to identify cycle with example.

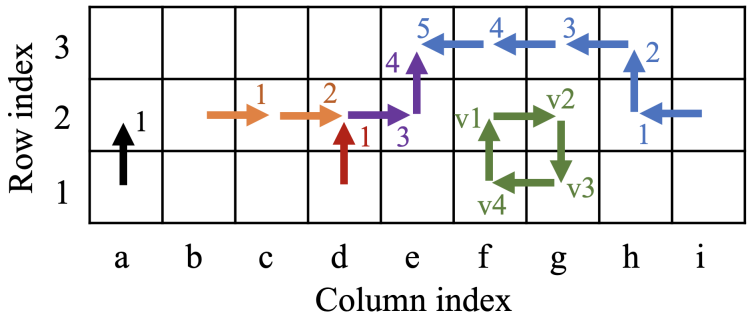


Figure 4.3: Example of system state with cycle in green (with vehicle IDs labeled) and non-cycle vehicles in other colors (with integers representing the backpressure).

4.3.2 Computing Backpressure

After all cycles have been advanced, we use backpressure to determine the order in which to deconflict the remaining sectors $\mathcal{S}$. To motivate why this is necessary, consider what would happen if sector d2 was deconflicted first, followed by e3 in Figure 4.3. Sector d2 may choose to allow the vehicle from d1 to enter. This would block vehicles in b2 and c2 and force the vehicles in d2 and e2 to proceed. Note that the need to force vehicles out of currently occupied sectors would add additional communication overhead. If sector e3 were to be deconflicted next, it would not get to choose between prioritizing vehicles in e2 and f3 because the vehicle in e2 must proceed to avoid gridlock from sector d2's deconfliction. This would lead to a suboptimal solution, as at most 3 non-cycle vehicles (occupying d1, d2, and e2) would be allowed to proceed, compared to possibly 5 non-cycle vehicles (occupying f3, g3, h3, h2, and i2). It would, in fact, be optimal to deconflict sector e3 first, followed by sector d2. Computing backpressure will allow us to do that.

In road networks, backpressure for a traffic movement can be the number of vehicles in a queue [72]. Queues with large build-ups are generally prioritized. We adopt similar logic to determine the order in which sectors are deconflicted. The backpressure at each sector is equal to the maximum number of aircraft that could proceed if the sector allowed an aircraft to enter. To determine its backpressure, each sector needs to collect backpressure values from its in-neighbors. As with cycle detection, each sector s passes a message to its out-neighbors, $\mathcal{N}_s^+$. There are no rounds of messages, however. The message is a modified backpressure metric equal to the maximum number of vehicles that could proceed as a direct consequence of allowing v to proceed to out-neighbor r where $x(i,t) = s$ and $\widehat{x}(i,t) = r$. For example, in Figure 4.3, sector e2 sends a backpressure value of 4 to e3 because at most 4 vehicles could proceed if the vehicle at e2 is permitted to enter e3. Note that sectors that are part of cycles do not pass backpressure messages. To compute backpressure, we start with sectors with no in-neighbors but some out-neighbors (e.g., a1, b2, d1, and i2). They pass

a backpressure value of 1 to out-neighbors. Once a sector has received backpressure values from all of its in-neighbors, it sends the $\max(b_q) + 1,\ \forall q \in \mathcal{N}_s^-$ to all sectors $r \in \mathcal{N}_s^+$. For example, sector d2 sends a backpressure value of $\max(2, 1) + 1 = 3$ to sector e2. This process continues until all sectors with in-neighbors have received a backpressure value. With this heuristic, sector e3 would be deconflicted before d2, allowing the highest number of vehicles to proceed. Note that with backpressure message passing, relaxing information sharing constraints (e.g., allowing non-adjacent sectors to communicate) would not improve the one time step efficiency. That is, sector e3 knowing the position of the vehicle in sector g3 would not improve efficiency, as this vehicle contributes to the backpressure message of 5 received from sector f3.

4.3.3 Chains of Flights

Another pertinent observation from Figure 4.3 is that we can string together the flight trajectories to form *chains* of connected flights. For example, the flights occupying sectors b2, c2, d2, and e2 form a chain of length 5 where the current sector of a flight is the desired next sector of the flight behind it (except for the last vehicle in b2). The chain has a length of 4, which is equivalent to the backpressure message sent by the vehicle in e2. We formally define a chain as an ordered set $L = \{k_1, k_2, \ldots, k_m\},\ k_j \in V$, where $\hat{x}(k_1, t) = s,\ \hat{x}(k_j, t) = x(k_{j-1}, t)\ \forall\, j \in \{2, \ldots, m\}$. Chains will appear again in the next section when we explore the efficiency of backpressure prioritization in Section 4.6.

4.4 Prioritization Schemes

When demand exceeds available capacity, the protocol decides which aircraft to allow to proceed using a prioritization scheme (Step 5b of the protocol). Our protocol's most flexible and modular component is the PrioritizeVehicle function. The key characteristics of a sector-prioritization protocol are deciding the number of queues (e.g., one for each adjacent

sector or one for each operator), the prioritization scheme used within a queue (e.g., time spent in a queue or current delay of a vehicle), and the logic for selecting a queue in case of multi-queue architectures (e.g., round-robin or random). Figure 4.4 presents three candidate queue network architectures. Thus, a PRIORITIZEVEHICLE function requires that we specify the queue architecture (either (a), (b), or (c)), the queue selection logic, and the priority scheme for each queue. We present six implementations of the PRIORITIZEVEHICLE function. We start with two baseline prioritizations, followed by four custom ones.

4.4.1 Baseline Prioritizations

We first present two intuitive baselines to benchmark more complex prioritization schemes. Note that some of the schemes can be implemented differently depending on whether we are focused on vehicle-level efficiency and fairness or operator-level efficiency and fairness.

Random prioritization: We create a merged priority queue and assign a random priority score (α_i in Figure 4.4(b)) to each of the vehicles.

Round-Robin prioritization: We use a sector-specific priority queue with first-come-first-served prioritization (i.e., highest priority for the vehicle that has been in the queue the longest). The queues are selected in a round-robin fashion, with one vehicle allowed per queue per round. Round-robin prioritization can also be performed on an operator level, whereby operator-specific priority queues are selected in a round-robin fashion, as shown in Figure 4.4(c).

4.4.2 Fairness-Oriented Prioritizations

Backpressure prioritization: The backpressure metric is computed for each vehicle. When considering system-level performance, we use a merged priority queue, with the priority score equal to the backpressure for each vehicle. Backpressure prioritization can also be performed on an operator level. In that case, we use a merged operator-specific queue. The queue corresponding to the operator with the highest total backpressure across all of

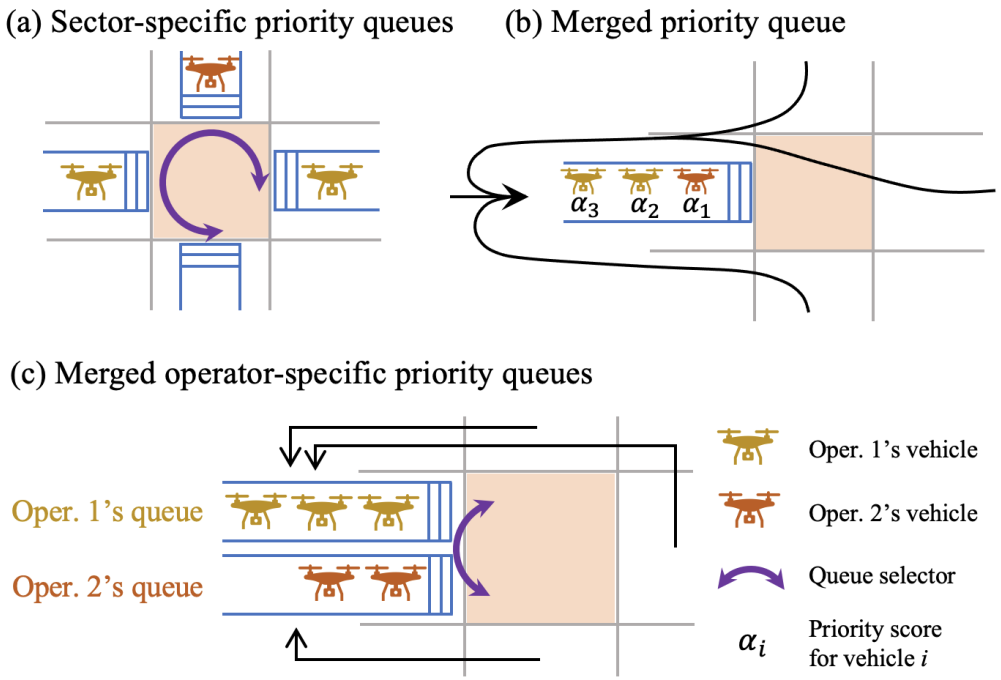


Figure 4.4: Potential queue prioritization schemes for implementing the PRIORITIZEVEHICLE function.

its vehicles is chosen. Intuitively, this prioritization aims to prioritize vehicles with a higher potential to clear upstream congestion. We show that backpressure prioritization minimizes the total system delays at that time step (i.e., minimizes $\sum_i del(i,t)$) among all possible solutions at time t in Section 4.6.

Accrued delay prioritization: Accrued delay is the delay that each vehicle has accumulated up to that point [114]. It can include delays accumulated during the current trip, as well as delays accumulated in previous trips operated by the same vehicle. Accrued delay prioritization orders vehicles based on their accrued delay, in descending order. More formally, this approach uses a merged priority queue, where the priority score for vehicle i, α_i = (Accrued delay)$_i$. The goal is to minimize additional delay for vehicles that have already been delayed. Accrued delay prioritization can be applied at the vehicle level as described above or at the operator level. In this case, a merged operator-specific priority queue architecture is employed, with the queue corresponding to the operator i with higher $\sum_{j \in \mathcal{V}_i} \alpha_j$ being prioritized.

Reversals prioritization: In this approach, a fair solution is one in which the relative ordering of arrivals at any resource is preserved according to the unimpeded schedule. Each vehicle keeps track of how many times it has encountered a resource in which its relative ordering was not preserved; this is called a reversal. For example, if vehicle A was originally scheduled to arrive at a resource before vehicle B, but instead vehicle B arrives before A, we count this as one reversal for A (and zero for B, since it benefited). To determine whether a reversal occurred, vehicles must communicate the original time they intended to utilize a sector. Recall that we assume that vehicles are truthful, including when they report values, such as reversals, used for prioritization. Each vehicle keeps track of how many reversals it has experienced so far along its trajectory, so reversals may not necessarily have occurred at the current sector. Now, we set the priority score for vehicle i, $\alpha_i = (\text{reversals})_i$. For flight-specific prioritization, we use a merged priority queue, and when implementing an operator-specific algorithm, we use merged operator-specific priority queues.

Dominant resource fairness prioritization: Dominant Resource Fairness (DRF) is a solution for fair resource allocation with several desirable properties, like information-sharing incentives and strategy-proofness [115]. For each operator and at each sector, we compute the resource share $R_o(s,t)$ that has been allocated to Operator o at sector s up through time t. When computing resource share, we account for resource allocation in the last T_b time steps, with $T_b = 10$ assumed. If we define $U_o(s,t)$ as the capacity allocated to operator o in sector s at time t, resource share $R_o(s,t) = \sum_{i \in [t-T_b,t]} \frac{U_o(s,i)}{C(s,i)}$. The *dominant share* of an operator is the largest proportion of a resource that it consumes among all of the resources it uses ($D_o(t) = max(R_o(s,t)\ \forall s \in \mathcal{S})$. The dominant resource is the resource corresponding to the dominant share. The assumption with DRF is that equalizing the dominant share among operators is fair. Thus, with each resource allocation, DRF prioritizes the operator with the lowest dominant share. For example, in a scenario where each sector has a capacity of 1, suppose that after 3 min, a vehicle has spent 1 min in Sector A and 2 min in Sector B. Its dominant share would be $max(1/3, 2/3) = 2/3$, and its dominant resource would be

Sector B since it has used Sector B proportionally the most. In DRF prioritization, vehicles with lower DRF are prioritized first since they have not "hogged" any one resource. We consider DRF prioritization on an operator level (not on a vehicle level). We set the priority score for operator o to be $\alpha_o = 1/D_o(t)$, and use a merged operator-specific priority queue.

4.5 Experimental Results

We evaluate our protocol using three traffic scenarios: a) random flight patterns, b) cross-flows, and c) hub-based operations. We also discuss the impacts of intra-operator deconfliction before the protocol is run. Finally, we present the impacts on efficiency and fairness of flight-level prioritization and operator-level prioritization.

4.5.1 Scenario Descriptions

Random flight trajectories (Scenario A): We generate a random scenario with two operators with 62 flights each departing across 50 time steps, indexed by t. The time step size is set to 30 s. We define a 13×13 grid of square-shaped, 1 km × 1 km enroute "high" sectors, and the same number of sectors at ground-level ("low" sectors) to represent vertiports. Each low sector is assigned a random relative weight $\in (0, 1)$. The sum of all relative weights is equal to 1. The origin and destination of each vehicle are randomly chosen based on the relative weights, simulating the fact that certain sectors will be more popular origins/destinations than others. Each vehicle's desired trajectory is assumed to be the shortest path trajectory from its origin to its destination. Further, we assume that each vehicle transitions from the low sector to the high sector immediately after departure and from the high sector to the low sector upon arrival. We randomly sample the departure time based on a bi-modal distribution with the highest peak at $t = 40$ and another peak at $t = 20$. Based on the shortest path trajectory, we identify the sectors that each flight must cross. We assume that 80% of the time, vehicles traverse each sector in one time step, while 20%

of the time, they take two timesteps. Sector capacity remains one. When a vehicle takes multiple timesteps to traverse a sector, it blocks other vehicles from entering the sector. A visualization of the traffic pattern is shown in the left side of Figure 4.5(a). The ticks on the figure's edge denote the sector's size.

Cross-flow flight trajectories (Scenario B): We generate a scenario with two crossing flows. Operator 1 has 60 flights traveling in the east-west direction, whereas Operator 2 has 40 flights traveling north-south. Each operator has five possible origins and five possible destinations to achieve appropriately oriented flows. Assumptions on sector geometry, departure time, and travel time remain the same as in Scenario A. This scenario helps evaluate the performance of our protocol in unbalanced, cross-flow traffic. A visualization of the corresponding traffic pattern is shown on the left side of Figure 4.5(b).

Hub-based trajectories (Scenario C): We use a hub-based package delivery scenario where four operators have warehouses on the outskirts of the city and make deliveries in locations randomly distributed around the city [107]. The demand is generated using a Poisson process. Operator 1 has hubs in the North and West with 66 flights, and Operator 2 has hubs in the South and East with 58 flights. A visualization of the traffic pattern is shown on the left side of Figure 4.5(c).

In all three scenarios, each vehicle is required to transmit its current sector and its desired next sector to its next sector. If queried, vehicles also share fairness metrics, like accrued delay. We assume that each vehicle does not share other information, conforms to its route, and complies with the protocol, holding or proceeding as dictated. Recall that there are six prioritization schemes that we evaluate. Three of the six prioritizations (round robin, accrued delay, reversals) can be applied on a flight-level or operator level; random and backpressure are only applied on a flight level; and dominant-resource fairness (DRF) is only applied on an operator level.

We conduct 100 trials for each prioritization scheme to account for the inherent randomness in the protocols. If all conflicting vehicles either a) belong to the same operator or

b) have the same priority according to the prioritization scheme (e.g., same accrued delay value), then backpressure is used as the tie-breaker for prioritization. We default to backpressure when fairness considerations are equal since backpressure is expected to result in the minimum delay. However, if the fairness metric *and* backpressure of conflicting vehicles are the same, the sector randomly chooses to prioritize one of the vehicles. Such random tie-breaking usually happens in early time steps and can have far-reaching and unpredictable downstream impacts on fairness and efficiency. Thus, we conduct multiple trials for any given prioritization scheme and report the average values in the figures.

4.5.2 Intra-operator Deconfliction

We assume that each operator has the full desired 4-D trajectory (three spatial dimensions plus time) of its vehicles before the start of the simulation. An operator may choose to deconflict its flights before the protocol ("intra-operator deconfliction"). We consider conflicts where more than one vehicle wants to utilize the same sector at the same time (given our unit sector capacity constraint). We only consider delay assignments and not re-routes. We use the traffic flow management problem (TFMP) to model the intra-operator deconfliction. We solve this model by assuming the same capacity constraints but without knowledge of other operators' flights. The objective minimizes the total delay cost for a given operator. The original trajectory of these vehicles is still used to compute delay and fairness metrics. We do this to aggregate the total impact on flights and evaluate the effectiveness of intra-operator deconfliction.

With intra-operator deconfliction by Operator 1, there would be no conflicts (and no need for the protocol) if there were no other operators. With the addition of Operator 2, there will be conflicts, but fewer than there would have been without intra-operator deconfliction since vehicles of Operator 1 will have fewer conflicts among their trajectories. Conflicts between vehicles of Operator 1 may still occur with *intra-operator* deconfliction, due to delays resulting from *inter-operator* conflicts. That is, even though Operator 1 deconflicted its vehicles

in advance, further delays will be required to deconflict with vehicles of other operators, putting them in conflict with their own vehicles that they were initially deconflicted with.

We compared the results of each prioritization scheme with/without intra-operator deconfliction. We present the results on random flight trajectories scenario (Scenario A) with flight-level prioritization. We assume that both Operator 1 and Operator 2 independently conduct intra-operator deconfliction ("with" case), or neither of them do ("without" case). With intra-operator deconfliction, the operators added a total of 47 minutes of delay prior to the start of the simulation. Table 4.1 shows the percentage change in a) total delay, b) standard deviation of flight delay, and c) number of conflicts when applying intra-operator deconfliction. In every prioritization mechanism tested, applying intra-operator deconfliction reduced the number of conflicts by around 30%, but increased total delay and standard deviation by 6-12%. This implies that the delay saved due to the reduced number of conflicts did not make up for the initial 47 minutes of delay applied. Flight accrued delay is particularly affected because intra-operator deconfliction increases accrued delay of some flights before the simulation. Then, the accrued delay prioritization expedites these delayed flights at the expense of others.

Table 4.1: Percentage change with intra-operator deconfliction relative to without intra-operator deconfliction for Scenario A.

Prioritization	Total Delay	Std. Deviation	Conflicts
Round Robin	8.8%	8.2%	-29.8%
Accrued Delay	11.8%	7.8%	-29.5%
Reversals	6.3%	10.5%	-31.0%
DRF	6.1%	11.8%	-27.7%
Backpressure	11.8%	12.8%	-25.7%

It may be counter-intuitive that intra-operator deconfliction deteriorates the final solution. However, intra-operator deconfliction is inefficient in this case because the operator does not have sufficient information about other operators to make informed decisions. These simulations do not represent the approaches to strategic deconfliction proposed in the literature that include discovery and synchronization of intent information from other operators

[20]. Instead, in these simulations operators do not have any incentive to share flight information to other operators because of competitive or privacy concerns. Thus, it is better for operators to send their vehicles out at their originally scheduled times and rely on the deconfliction protocol. In optimization problems, it may be better to solve a global problem rather than several sub-problems. This is the case for these protocols as well.

4.5.3 Fairness and Efficiency of Flight-level Prioritizations

We first discuss the results of flight-level prioritization, shown in the second column of Figure 4.5. From this point on, we assume no intra-operator deconfliction, as we have shown how it reduces system efficiency and fairness. We use mean standard deviation of flight delay (henceforth referred to as "standard deviation") as a metric for fairness, and mean total system delay ("total delay") as a metric for efficiency. (Recall that we show the mean because we ran 100 trials for each prioritization scheme.) We show the results of five prioritization schemes. Note that DRF is not shown in the center panels of Figure 4.5 because we only consider DRF on an operator-level.

We first note that Scenario B has the most intersections of flight trajectories, followed by Scenario C, then Scenario A. Because it has the highest number of conflicts, particularly in the center, Scenario B with cross-flow trajectories has by far the highest delay and standard deviation across all prioritizations, even though it has the fewest number of flights. Scenario C with hub-based trajectories has the next highest total delay and standard deviation, because it has more conflicts than Scenario A.

We now consider the best and worst performers in terms of fairness and efficiency. There is a pattern consistent across all scenarios in terms of relative efficiency and fairness ordering between the prioritization schemes. Prioritizing by accrued delay (red square) leads to the lowest standard deviation across all scenarios. This makes sense given that this approach can balance final delay values by holding flights with little accrued delay and prioritizing flights with high accrued delay. Back-pressure (teal diamond) has the lowest total delay

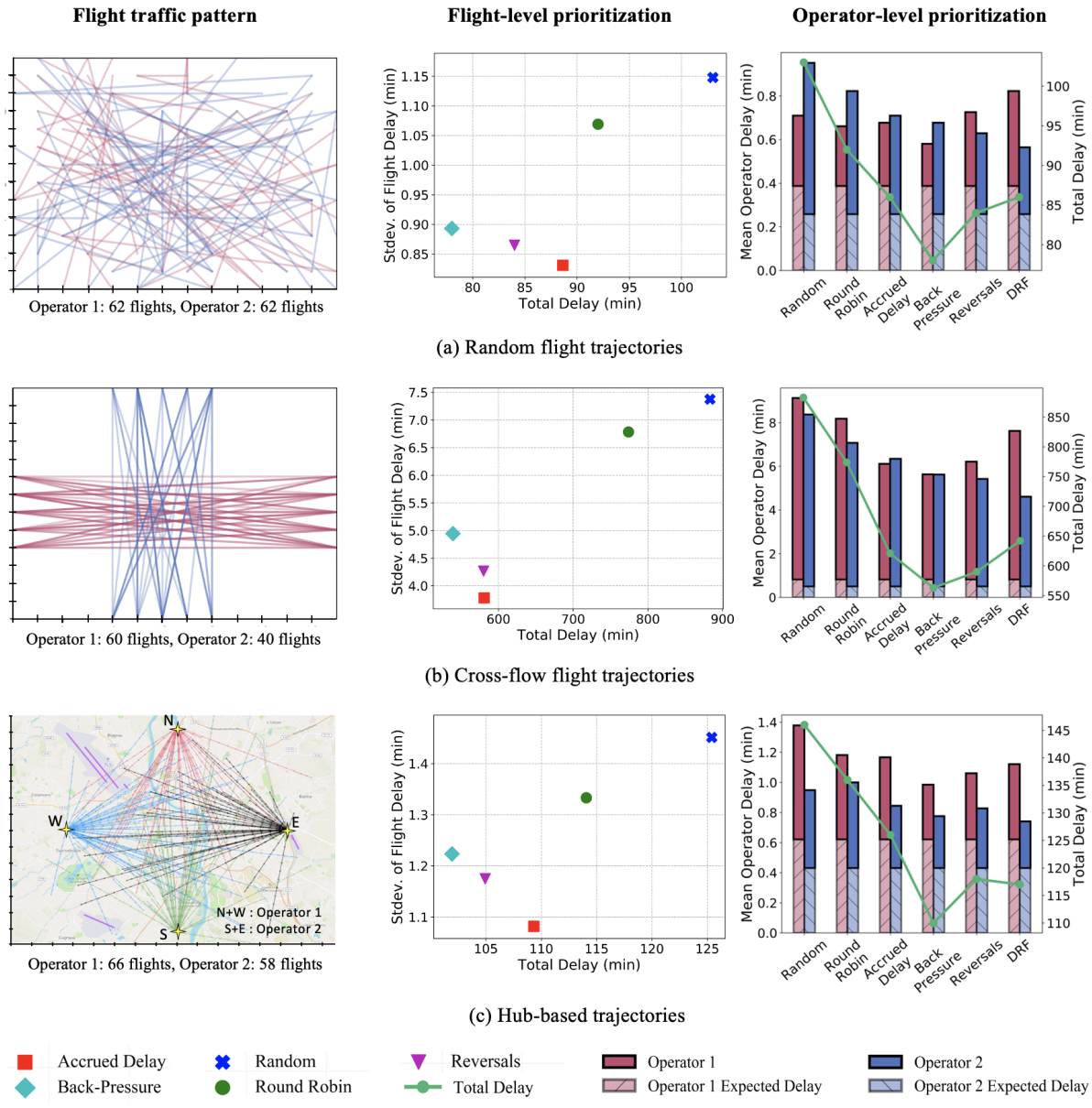


Figure 4.5: Efficiency and fairness of our flight-level and operator-level protocols for three traffic scenarios.

across all scenarios, and therefore highest efficiency, which is expected given its proven properties in other settings. Prioritizing randomly is the least fair and least efficient outcome, which matches our expectations. Round-robin prioritization leads to more fair and efficient solutions than random prioritization, but still worse than the other four prioritizations. Reversals prioritization (purple triangle) is in between backpressure and accrued delay in terms of total delay and standard deviation. We hypothesize that there is a Pareto frontier of different prioritizations that lie between the solutions of backpressure prioritization and accrued delay prioritization, and reversals is one of them.

4.5.4 Fairness and Efficiency of Operator-level Prioritizations

We now consider the performance of operator-level prioritization, shown in the third column of Figure 4.5. We show the results of six prioritization schemes. Note that random and backpressure are shown for reference even though they are prioritized at the flight level. As with flight-level prioritization, we evaluate efficiency with total delay, shown in green on the right-hand side axis. The trends in total delay with operator-level fairness are similar to those with flight-level fairness. Backpressure prioritization remains the most efficient, whereas random and round-robin prioritizations are the least efficient. Reversals, DRF, and accrued delay prioritizations have similar efficiency, with accrued delay having the highest total delay.

We do not show the standard deviation of flight delays for evaluating operator-level fairness, since prioritizations occur on an operator level. We instead show mean operator delay and mean expected delay, which we use to present three possible notions of operator fairness below:

1. **Equal mean operator delay:** The idea is to equalize μ_o^{oper} across operators $o \in \mathcal{O}$.

2. **Equal mean excess operator delay:** The idea here is that equal mean delays for operators might not capture that one operator might have an original schedule with

significantly more inherent delays than the other. For example, in scenario (b), clearly Operator 1 has more flights that cause congestion, even in the absence of Operator 2. We define the mean *expected delay* as the average delay experienced by an operator if it were the only user of the system. Denote by μ_o^{exp} the mean expected delay of operator o. We define the mean *excess delay* as $\mu_o^{exc} = \mu_o^{oper} - \mu_o^{exp}$. Excess delay is philosophically similar to the concept of time-order deviation introduced in [48], but for multiple flights and operators instead of multiple congested resources. Thus, one notion of fairness could be equalizing the excess delay across operators.

3. **Target excess delay ratio:** We may want to link the excess operator delay ratio with the expected operator delay ratio. Suppose $\frac{\mu_1^{exp}}{\mu_2^{exp}} > 1$. For each protocol and scenario, we can construct the relation between the excess delay ratio and expected delay ratio as $\frac{\mu_1^{exc}}{\mu_2^{exc}} = \alpha \frac{\mu_1^{exp}}{\mu_2^{exp}}$. We can choose values of α to target. A value of $\alpha < 1$ implies that the protocol deemphasizes imbalances in the expected delays when assigning excess delays. By contrast, $\alpha = 1$ means that excess delays were assigned in proportion to the expected delays, and $\alpha > 1$ implies that the imbalances in the expected delays were further exacerbated when assigning excess delays.

The fairness of prioritization schemes depends on the metric used. For example, with backpressure prioritization with the cross-flow scenario shown in Figure 4.5(b), Operator 1 and 2 have nearly identical mean delays, satisfying operator fairness notion 1. On the other hand, Operator 2 may feel that this is unfair given that Operator 1 has 50% more flights and 63% more expected delay. Thus, Operator 2 may prefer fairness notions 2 or 3. Note that in all scenarios, Operator 2 has lower mean expected delay than Operator 1 (and in Scenarios B and C, fewer flights). Because DRF tries to equalize resource allocation between operators, in all scenarios, Operator 2 receives the lowest delay with DRF.

Figure 4.6 shows the three notions of operator fairness across the six prioritizations and the three scenarios. Backpressure generally has the most equal mean operator delay and mean excess operator delay. The next best two prioritizations in terms of these two notions

of operator fairness are accrued delay then reversals. DRF has disparate mean operator and mean excess operator delays (particularly in Scenario B), but it has α values closest to 1. This indicates that DRF could be appealing in situations where excess delays should be assigned in proportion to expected delays. Note that in Scenario C, DRF has $\alpha = 1.12 > 1$, indicating that the excess delay ratio exceeded the expected delay ratio.

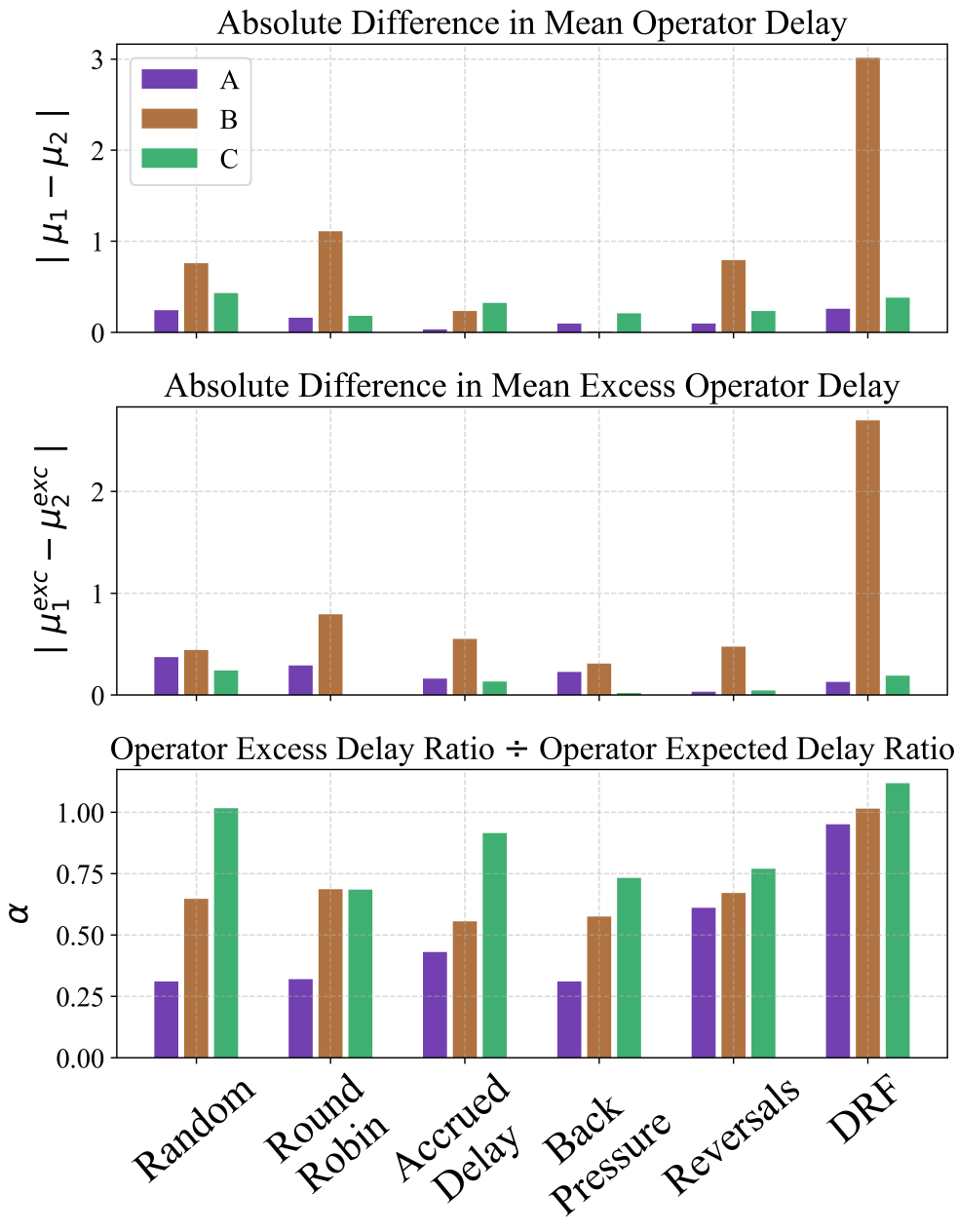


Figure 4.6: Operator notions of fairness across prioritizations for Scenarios A-C. The notion of operator fairness is indicated on top of each plot and on the y-axis label.

An inherent trade-off exists between efficiency (total delay) and operator fairness. For example, backpressure prioritization leads to the lowest delays but also low α values, which may be unfair depending on the target. At the other extreme, DRF has α values close to 1, but higher total delays. On both counts, accrued delay and reversals fall between backpressure and DRF.

4.5.5 Overview of Results

To summarize, we consider three demand patterns: random origin-destination pairs, cross-flow traffic, and hub-and-spoke operations. We evaluate our protocols with and without intra-operator de-confliction, and measure the efficiency and fairness at both flight and operator levels. Our main findings were:

1. Intra-operator deconfliction (i.e., an operator deconflicts its own flights from each other before filing trajectory requests), results in lower system-wide efficiency and fairness. In other words, it is preferable to allow the protocols to perform any necessary deconfliction by considering all operators.

2. Prioritizing flights based on the backpressure metric maximizes efficiency (i.e., minimizes total delays).

3. Prioritizing flights based on the accrued delay metric maximizes one notion of system-wide fairness (i.e., minimizes the standard deviation of flight delays).

4. Dominant resource fairness (DRF) prioritization achieves fairness as defined by a metric that considers *excess* and *expected* operator delays. Furthermore, DRF results in lower delay for the operator whose flights impose less externality (defined as the expected delay when it is the sole operator) on the system.

4.6 Efficiency of Backpressure Prioritization

We now show that the protocol with backpressure prioritization results in an optimal solution for one time step with a sector capacity of one. A *solution* to the problem defines the control decision for each aircraft i: either they can proceed to their desired next sector $\hat{x}(i,t)$, or they must hold at their current sector $x(i,t)$. By optimal, we mean that the protocol results in a minimum delay solution for one time step: $\min \sum_{i \in \mathcal{V}_a} \text{del}(i,t)$. Equivalently, the protocol allows the maximum number of aircraft to proceed from $x(i,t)$ to $\hat{x}(i,t)$.

We first consider aircraft that are in cycles. We define $\mathcal{C} = \{V_{c_1}, V_{c_2}, \ldots, V_{c_n}\}$ as the set of cycles, where V_{c_m} is the set of aircraft in cycle m. In a cycle with two aircraft i and j, $x(i,t) = \hat{x}(j,t)$ and $\hat{x}(i,t) = x(j,t)$. In a cycle with more than two aircraft, for every aircraft i, there exists an aircraft j where $x(i,t) = \hat{x}(j,t)$ and an aircraft k where $\hat{x}(i,t) = x(k,t)$. With unit sector capacity, no two cycles will have overlapping aircraft; thus, we consider one cycle at a time. Let V_c be the set of aircraft in one cycle that we are considering. First, we show that all aircraft in V_c are in $\mathcal{G}$ (list of aircraft that can proceed to $\hat{x}$), or all aircraft in V_c are in $\mathcal{H}$ (list of aircraft that must hold in x).

Lemma 4.6.1. $\forall (j,k) \in V_c$, $j \in \mathcal{G} \iff k \in \mathcal{G}$.

Proof: We show this using proof by contradiction. Assume $j \in \mathcal{G}, k \in \mathcal{H}$. Because $k \in \mathcal{H}$, then aircraft $r \in V_c$ where $x(k,t) = \hat{x}(r,t)$ must be in $\mathcal{H}$. Continue this by induction, until we get to aircraft $r' \in V_c$ where $x(r',t) = \hat{x}(j,t)$. This implies that $j \in \mathcal{H}$, which is a contradiction. □

Lemma 4.6.2. *In an optimal solution, aircraft in cycles should be prioritized (and all assigned to $\mathcal{G}$) before aircraft that are not in any cycles are deconflicted.*

Proof: Define a cycle $V_c \subset V_a$. We denote an arbitrary set of procedures $P : \mathbf{x}(t), \hat{\mathbf{x}}(t) \mapsto \mathcal{G}, \mathcal{H}$ which prioritizes cycles first, and an arbitrary set of procedures $F : \mathbf{x}(t), \hat{\mathbf{x}}(t) \mapsto$

$\mathcal{G}, \mathcal{H}$, where $P \subset F$. We consider two cases and show that in both cases for $(\mathcal{G}_P, \mathcal{H}_P) = P(\mathbf{x}(t), \hat{\mathbf{x}}(t))$ and $(\mathcal{G}_F, \mathcal{H}_F) = F(\mathbf{x}(t), \hat{\mathbf{x}}(t))$, $|\mathcal{G}_P| \geq |\mathcal{G}_F|$.

Case 1: There exists an aircraft i that is trying to utilize a sector currently occupied by the cycle $V_c \subset V_a$:

$$\exists\, i \in V_a \;\; s.t. \;\; \hat{x}(i,t) = x(j,t) = \hat{x}(k,t) \;\; j,k \in V_c \tag{4.1}$$

Using our protocol, $V_c \subset \mathcal{G}_P$, because the cycle is prioritized and all aircraft in cycles are moved. $i \notin \mathcal{G}_P$, because k was chosen to proceed. Using the arbitrary procedure, if $j \in \mathcal{G}$, then $V_c \subset \mathcal{G}$ by Proposition 1.1; if $j \notin \mathcal{G}$, then $V_c \notin \mathcal{G}$ and $i \notin \mathcal{G}$ because $j \in \mathcal{H}$. Thus $|\mathcal{G}_P| \geq |\mathcal{G}_F|$. Regardless of the decision on V_c, aircraft i cannot proceed. Thus, to minimize delay, V_c is assigned to $\mathcal{G}$.

Case 2: In the second case, no aircraft are trying to enter a sector occupied by the aircraft in V_c. P will always have $V_c \subset \mathcal{G}$. By Proposition 1.1, if F chooses $j \in \mathcal{G}$, then $V_c \subset \mathcal{G}$; else if F chooses $j \notin \mathcal{G}$, then $V_c \notin \mathcal{G}$, due to the unit sector capacity constraint. Thus, $|\mathcal{G}_P| \geq |\mathcal{G}_F|$. $\square$

To prove that backpressure prioritization is optimal, the rest of the proof proceeds as follows: we demonstrate a mapping of aircraft (not in cycles) to trees, show that the longest path in a tree is the optimal solution to a conflict, and show that backpressure prioritization finds the longest tree. These statements combine to show that backpressure prioritization finds the optimal path.

Once cycles have been prioritized, any configuration of aircraft in V_a/V_c with $x(i,t)$ and $\hat{x}(i,t)$ can be mapped into a forest (a disjoint collection of trees). Each tree in the forest is an independent subproblem.

Construction: We map sectors to nodes of trees and aircraft to branches of trees. For every aircraft i, we create an edge from child node $x(i,t)$ to parent node $\hat{x}(i,t)$, creating nodes $x(i,t)$ or $\hat{x}(i,t)$ if they do not exist. Because all aircraft in V_c have been prioritized and moved, there are no cycles in the graph. Every node will have at most one parent because

the capacity of one constraint means there is at most one flight occupying each sector (there exists only one aircraft i s.t. $x(i,t) = s$ for some child node s, which means there exists only one $\hat{x}(i,t)$ that is a parent of the child). The root node of the tree represents a sector that has incoming aircraft but no outgoing aircraft.

Independence: Sector nodes q and r are only connected if there exists an aircraft i with $x(i,t) = q$ and $\hat{x}(i,t) = r$. This means that there are often disjoint collections of trees representing independent conflicts, where x and $\hat{x}$ of each tree do not overlap with x and $\hat{x}$ of all other trees. Since, by definition, there are no overlapping nodes or edges between trees, a solution to a subproblem will not impact another subproblem. □

Consider a single subproblem. Define the collection of aircraft in a subproblem as V_s. Then:

Definition 4.6.3. *A solution S to a subproblem centered on the contested sector s (i.e., the root of the tree) is an assignment of all aircraft $i \in V_s$ to $\mathcal{G}$ or $\mathcal{H}$. The chain (an ordered set) $L = \{k_1, k_2, \ldots, k_m\},\ k_j \in V$, where $\hat{x}(k_1,t) = s$, $\hat{x}(k_j,t) = x(k_{j-1},t)\ \forall\, j \in \{2,\ldots,m\}$, is assigned to $\mathcal{G}$ such that $L \in \mathcal{G}$. All other flights are assigned to $\mathcal{H}$: $V_s/L \in \mathcal{H}$.*

Lemma 4.6.4. *Every solution to a subproblem matches to exactly one valid path in the tree starting from the root.*

Proof: We can construct this path by including all nodes connected to the edges created by aircraft in L: a path $h = \bigcup_{k_i \in L} \{x(k_i,t), \hat{x}(k_i,t)\}$. By the capacity 1 constraint, every aircraft k_i maps to exactly one node $s = x(k_i,t)$, so this definition of constructing paths implies that every solution maps to one valid path, and vice-versa.

Next, we prove that every valid solution to a subproblem has a corresponding path in the tree starting from the root. We will show 1) valid solutions to a subproblem are continuous paths in the tree, and 2) these paths must start at the root, excluding the empty path.

1. Valid solutions to a subproblem are continuous paths in the tree. We use proof by contradiction. Assume there is a valid solution S with a corresponding L that maps

to a noncontinuous path in the tree: for some k_i, k_{i+1} there is no edge connecting the nodes in the tree $x(k_i, t), x(k_{i+1}, t)$. This implies that $x(k_i, t) \neq \hat{x}(k_{i+1}, t)$, which contradicts the definition of a solution above.

2. Assume there exists a nonempty path h that doesn't start at the root node s. This implies that the children node of the root are not in $\mathcal{G} : \forall k_i \;\; s.t.\; \hat{x}(k_i, t) = s,\; k_i \notin \mathcal{G}$. Because of this, all nodes $x(k_i, t) \notin h$. This implies that the grandchildren cannot be in $\mathcal{G}$; for a given grandchild $k_j, \hat{x}(k_j, t) = x(k_i, t) \notin h$. By induction, all nodes in the subproblem are excluded from h, which leaves the empty path and is a contradiction. □

Lemma 4.6.5. *A longest valid path in a tree maps to an optimal solution in one time step.*

By the construction of the tree given in Proposition 1.5, each edge represents an aircraft. The optimal solution moves the maximum number of flights, which corresponds to the maximum edges traversed on the tree. Thus, the longest path maps to the optimal solution.

Lemma 4.6.6. *With backpressure prioritization, the protocol results in a solution that maps to the longest valid path in a tree.*

Proof: We show this by constructing labels for the tree that correspond to backpressure metrics in the grid environment, defining the backpressure prioritization method, and then using proof by contradiction to demonstrate that the backpressure method finds the longest valid path.

We define the set N of all non-root sector nodes as $N = x(i, t)\ \forall i \in V_s$. We first add labels $y(j)\ \forall j \in N$ to every node in the tree, as follows:

1. We define the set M of leaf nodes, $M = \{x(l, t)\ \forall l \in V_s \mid x(l, t) \neq \hat{x}(i, t)\ \forall i \in V_s\}$. All leaf nodes $l \in M$ receive a label $y(l) = 0$.

2. For all other non-root nodes $j \in N \setminus M$, we label them $y(j) = \max_{x(j,t)=\hat{x}(i,t)} y(i) + 1$— that is, we label them with the highest child label plus 1.

3. We label the root node s with $y(s) = \max_{s=\hat{x}(i,t)} y(i) + 1$. This edge case is added because the root node doesn't have a corresponding aircraft located in it, i.e. $\nexists\, i \in V_s \mid x(i,t) = s$.

The longest path follows the highest label: the longest path away from the root from any node $x(i,t)$ is strictly upper bound by $y(i)$. Let us define two paths $h*$ and h, where $h*$ is the longest path, and h takes the same partial path $h_{s,i}$ from root node s to node i, but splits into a different edge at node i. Respectively, $h*$ and h lead to nodes $j*$ and j, where $y(j*) \geq y(j)$. First, $|h*| = |h_{s,i}| + y(j*)$, because by definition at every node i there is a child node $j*$such that $y(i) = y(j*) + 1$. Then, $|h*| = |h_{s,i}| + y(j*) \geq |h_{s,i}| + y(j) \geq |h|$. Following the path of the highest label corresponds to the definition of the backpressure prioritization method in Section 4.4 □

Theorem 4.6.7. *The congestion management protocol with backpressure prioritization results in an optimal (minimum delay) solution in one time step.*

Proof: The protocol identifies and prioritizes aircraft in cycles first; then, it finds a solution that maps to the longest valid path in a tree. Taken together, Lemma 1.2 and 1.7 imply that the protocol with backpressure prioritization results in an optimal (minimum delay) solution in one time step. □

We provide the following visual example. The left-hand side of Figure 4.7 shows an example layout of flights for the protocol after cycles have been identified and prioritized. Sector S1 is the root of the tree, and aircraft A is currently in S2 and wants to proceed to S1. In both the layout and tree, we color the aircraft that the protocol would allow to proceed in green. Starting at S1, Aircraft A comes from the sector with the highest backpressure, so it is chosen. At the next decision point, Aircraft C is chosen over Aircraft L, as it comes from a sector with higher backpressure. Note that the branch represented by Aircraft C leads to a deeper leaf (S13) than the Aircraft L branch (S9). After all of the sectors have been deconflicted, the aircraft in $\mathcal{G}$ are {A, B, C, D}.

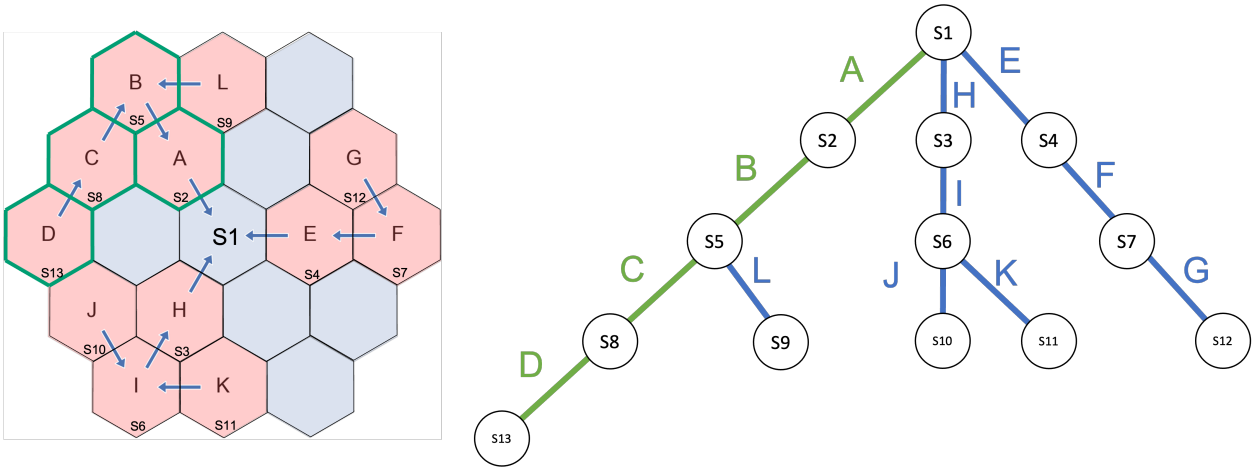


Figure 4.7: Abstraction of a set of aircraft in conflict from the protocol (left) into a tree structure (right)

We have shown that the protocol maps a layout to trees, and within each tree, a path from the root to a leaf with maximum depth is found. Branches (aircraft) in the path are allowed to proceed, while others are forced to hold. By doing so, we also maximize the number of aircraft that are allowed to proceed. Naturally, this problem could be formulated as a one time step optimization with the objective of minimizing delay. We confirmed that our protocol with backpressure prioritization leads to the same optimal delay value by comparing it with an optimization formulation in simulation. However, there are several trade-offs between the protocol and a one time step optimization: the protocol has more communication overhead but is more decentralized, which enhances privacy.

4.7 Discussion

We explored the concept of reduced information sharing and fair congestion management algorithms for efficient advanced air mobility operations. Such algorithms are critical to the development of a traffic management infrastructure that can support dynamic, low lead-time operations such as drone deliveries and urban air mobility. Our key contribution lies in developing reduced-information decentralized protocols that avoid gridlocks, a frequent pitfall of approaches that do not rely on centralized coordination. Our protocol is also flexible and

supports a wide variety of user-specified priorities to help achieve the desired balance between efficiency and fairness at an operator- or system-wide level. We also tested our protocol on three simulated demand scenarios, with several assumptions on vehicle truthfulness and compliance. We also studied other prioritization schemes in simulation and showed that our protocol balances efficiency and fairness, depending on the choice of prioritization scheme. Our work explores the interplay between information exchange, efficiency, and fairness in traffic coordination. In this regard, our analysis is also of relevance to the problem of coordination and control of road traffic, especially with the increasing deployment of connected autonomous vehicles. We also developed cost-aware traffic management protocol based on the second-price auction [106]. In this work, instead of just using pre-calculated metrics like backpressure or accrued delay, we allow individual vehicles to provide bid values for their movement (this allows for informed prioritization between flights).

4.7.1 Extensions

This chapter assumed sector-to-sector communications, but the protocol can be adapted for vehicle-to-vehicle (V2V) communication. We go through the information sharing of the protocol in order and discuss how to adapt it for (V2V) communication:

- Sharing of intent: We previously assumed that vehicles alerted a sector s if it wanted to utilize it in the next timestep. Instead of alerting sector s, vehicles could instead alert flights within a certain range of their intent. This way, conflicts can be identified.

- Cycles: It is crucial to identify cycles and then allow flights in cycles to proceed to avoid gridlock. Rather than have sectors perform rounds of message passing, vehicles can pass messages to each other. This will require vehicles to know who wants to enter their current sector and who currently occupies their intended next sector. The content of the message remains the same: the current sector of vehicles $j \in \mathcal{V}$ that want to access the current sector of vehicle i.

- Backpressure: As with cycles, the content of backpressure messages remains the same, but vehicles can communicate with each other rather than sectors. Namely, when a vehicle receives two backpressure messages from two incoming vehicles (or in-neighbors), it chooses the highest backpressure metric.

- Fairness: When deciding who gets to utilize a sector, vehicles must agree on a prioritization scheme. These prioritization schemes will require vehicles to share information such as backpressure or schedule reversals (which depends on their original schedule). Assumptions on truthfulness would need to be made, or some mechanism to prevent gaming behavior would be needed.

Other future work could incorporate more sources of uncertainty, like weather disruptions and vehicle non-compliance. Operators and flights can easily manipulate the protocols by misreporting their current state or strategically filing for particular routes or at specific times to minimize their overall delays. Identifying robust, strategy-proof, and incentive-compatible congestion management protocols is therefore of practical interest. Another area of research is providing operators with more control of, for example, the prioritization schemes used, thereby increasing the acceptability of the protocol to operators. Our protocol focused on deconflicting one time step at a time; it is worth considering extensions to handle multiple time steps and partial trajectory information. Given the combinatorial set of outcomes when considering multiple time steps, it will be difficult to guarantee efficiency over multiple time steps. Finally, adding more realism by incorporating reroutes and simultaneously handling both on-demand and scheduled aircraft operations will improve the practical application of the proposed approaches.

Chapter 5

Federated Airspace Traffic Management

We presented centralized and decentralized methods for AAM traffic management in Chapters 3 and 4, respectively. So far, we have assumed that either centralized or decentralized methods uniformly control an entire block of airspace. However, larger regions of airspace may utilize a *federated* airspace configuration. In a federated setup, different third-party service suppliers control adjacent regions of airspace. Moreover, adjacent regions may choose different traffic management methods. This is particularly true across regulatory boundaries (e.g., city limits) where different regulators may have different AAM traffic management requirements. We assume that the third-party service supplier can choose centralized traffic management or the decentralized protocol. In regions where operators are willing to share information with a central agent for the sake of efficiency, centralized traffic management can be used. On the other hand, a decentralized protocol can be used when operators are reluctant to share information with a central agent and for more than one time step. This chapter considers how to manage traffic in *federated* airspace configurations where different service suppliers control different regions of airspace. Of particular interest are *boundary flights*, which cross between adjacent regions. We show how adapting principles from the protocol allows for seamless federated airspace traffic management.

5.1 Background

Figure 5.1 shows representative examples of possible airspace configurations. The top left configuration displays a reference configuration where the entire airspace is controlled by centralized traffic management. The "O" indicates that the region is controlled by centralized optimization. The other three configurations represent federated setups, with the dark black lines indicating borders between adjacent regions. The top right configuration contains two regions controlled by centralized traffic management. The bottom left configuration also has two regions, but one is controlled by the decentralized protocol (indicated with "P"). The bottom right configuration has three regions: a protocol region in the middle between optimization regions on either side. Like before, we assume that all sectors have unit capacity. We need to tackle several issues, including the different levels of information sharing in each region and the coordination of flights transitioning from one region to another.

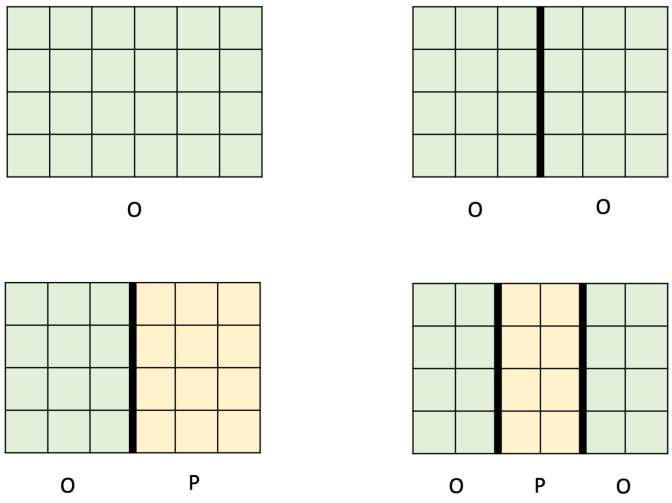


Figure 5.1: Example configurations of federated airspace with regions controlled with centralized or decentralized traffic management. "P" denotes a region controlled by the protocol, while "O" denotes an optimization region.

5.1.1 Information Sharing within Regions

In the federated airspace setting, we assume that information sharing *within regions* follows the methods outlined in Chapters 3 and 4. Recall the different information-sharing considerations associated with our centralized or decentralized methods.

- When information is shared: With the TFMP, flights must share their trajectory information before their scheduled departure time. To avoid being classified as a *pop-up*, they must share their information before the TFMP is solved. This time varies depending on the setup on how often the TFMP is solved. With the protocol, flights share their desired next sector, which they want to access in t, at $t-1$.

- How much information is shared: With the TFMP, we previously assumed in Chapter 3 that flights shared their full trajectory information (including origin, en route sectors, and destination). Here, we will relax this assumption and allow regions to choose how much information is shared. We call this the n-step TFMP, which will be covered in Section 5.1.2. With the protocol, one step of information is shared at a time corresponding to the desired next sector at $t+1$.

- To whom information is shared: With the TFMP, flights must share their information with the third-party service supplier managing the airspace. With the decentralized protocol, vehicles communicate with their current sector and desired next sector. If the vehicles handle the protocol themselves (i.e., without sector-to-sector communication), then vehicles communicate with each other instead of with sectors.

5.1.2 n-Step TFMP

We key in on the second item listed above, namely, how much information is shared. In Chapter 3, we assumed that flights shared their full trajectory information with the third-party service supplier. However, operators may prefer to share less information, even in a

centralized setting. Operators may not be uncertain about their full trajectory or may be unwilling to share this information with others. The centralized formulation is amenable to flights sharing partial trajectories. We call this the "n-step TFMP" since operators are required to share n time steps of information. Going forward, we shorten "time steps" to "steps" unless otherwise noted.

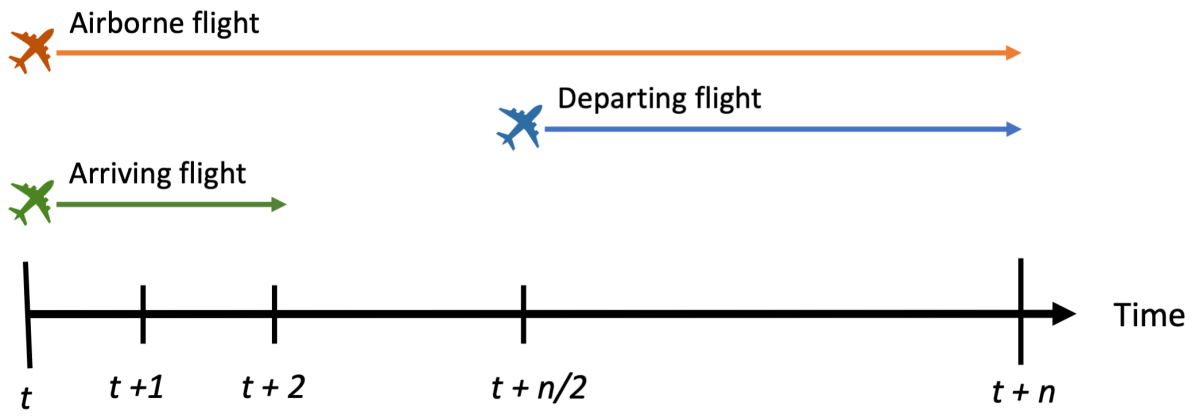


Figure 5.2: Diagram of "n-step TFMP" setup for a given time window between $[t, t+n]$. The extent of the colored arrows indicates the time length of the required information from each category of flight.

Recall that with a rolling time horizon implementation, we solve the TFMP at regular time intervals. Here, we continue solving the TFMP at regular time intervals but drop the requirement in Section 3.4.1 for flights to share their full trajectory information. Figure 5.2 shows an example of three types of flights in an n-step TFMP setup, where $n = 8$. We make the following assumptions:

- We use a rolling horizon setup where the TFMP is solved every n steps. Thus, in Figure 5.2, the rolling horizon is between $[t, t + n]$.
- Flights provide *up to* n steps of information. The arrows going to the right from each flight indicate the required length of information sharing.
  - If a flight is scheduled to arrive at its destination (example in green in Figure

5.2) within the time window, it shares all of its remaining trajectory information. This is less than n steps of information. The flight shares its trajectory in $[t, t+2]$ in this example.

  - If a flight wishes to depart in the middle of the time window (example in blue in Figure 5.2), it will share trajectory information between its scheduled departure time and $t+n$. The flight shares its trajectory in $[t+4, t+8]$ in this example. This is less than n steps of information, but in the next time window, it will have the opportunity to share up to n steps of information if it remains airborne.
  - Airborne flights (example in orange in Figure 5.2) share n steps of information by the horizon's start. This is in contrast to flights departing or arriving within the time window, which share less than n steps of information.

- Pop-up flights are still relevant. The blue departing flight would be a pop-up if it failed to provide the required trajectory information by t. Likewise, airborne and arriving flights must share information by t.

5.1.3 Setup

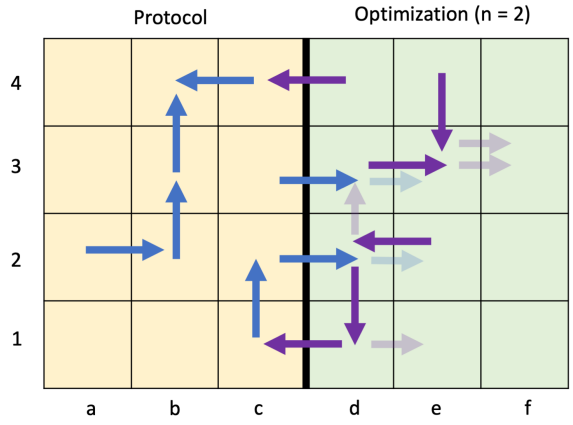
We now consider the procedure for handling federated airspace configurations. Flights are likely to pass through multiple regions in a federated airspace configuration. We need to consider how to handle these transitions. Figure 5.3 displays a representative example of coordinating flights transiting regions. A decentralized protocol controls the left-hand-side region, whereas the right-hand-side region is controlled by a 2-step optimization. Within the optimization region, flights typically share 2 steps of information. However, recall from Section 5.1.2 that flights arriving and departing within a time window may share less than n steps of information if they are not active during the entire duration of a time window. We start with the configuration at the beginning of a time step, as shown in Figure 5.3a. The arrows are color-coded based on their current region: blue arrows are currently in the

protocol region, whereas purple arrows are in the optimization region. As before, the solid arrows indicate positional information of flights: the base of the arrow represents the current sector, and the head of the arrow represents the desired next sector. The 2-step optimization allows flights to share 2 steps of information, so the solid arrow indicates the current sector and next sector, while the shorter, faded arrows indicate the next sector and sector after the next sector. For example, there is a purple flight currently in sector e2 that would like to proceed to d2 in the next time step. The same flight intends to continue to sector d3 two steps from now. Note that all of the flights currently in the optimization region (in purple) share two steps of information in Figure 5.3a, but this need not be the case for flights departing or arriving in the time window. (They may only need to share one step of information.) When considering federated airspace scenarios, we are most concerned with *boundary flights* that wish to cross from one region into another. An example of a boundary flight is the flight currently in $c2$ in the protocol region that wants to cross into $d2$ in the optimization region.

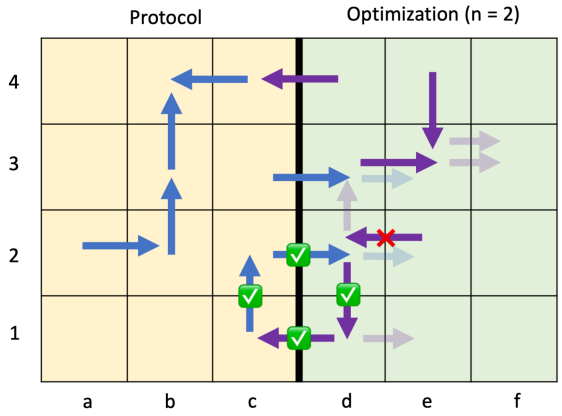
Information Sharing for Boundary Flights: Recall that the decentralized protocol utilizes one step of information, whereas the centralized optimization uses n steps of information. From a flight's perspective, it adheres to the information-sharing requirements of its next region.

- Boundary flights entering protocol region: Flights share one step of information (e.g., the flight in d4 in Figure 5.3a alerts c4 that it wishes to utilize it). They also share backpressure and participate in message passing of the identity of flights behind it (i.e., in-neighbors of its current sector) to identify cycles.

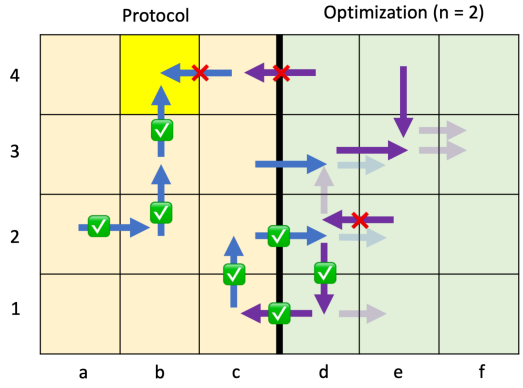
- Boundary flights entering optimization region: Flights share n steps of information (e.g., the flight in c2 heading to d2 shares [d2,e2] with the optimization region). As with flights entering a protocol region, flights entering an optimization region need to share the identity of flights behind it to identify cycles. They may also need to share backpressure information if the optimization region requests it (as per Section 5.2).



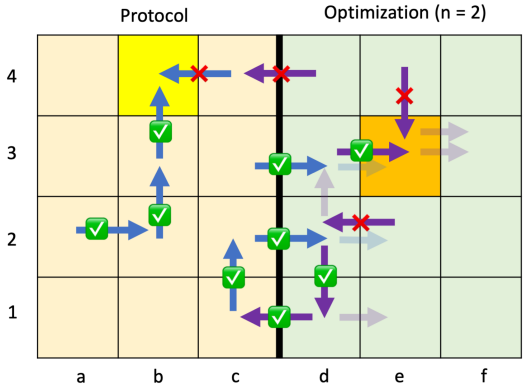
(a) Configuration at beginning of time step



(b) Clearing cycles



(c) Flights entering protocol region



(d) Flights entering optimization region

Figure 5.3: Handling boundary flights in a federated airspace setting. The base of solid arrows indicates the current position of flights, while the head represents the desired next sector. Faded arrows indicate information for 2 time steps from the current time.

5.2 Procedure for Boundary Flights

We follow the procedure below to manage boundary flights in federated airspace configurations.

1. At the beginning of each time step, flights communicate with the requisite set of next sectors they wish to utilize. For flights in or entering a protocol region, this is just one sector, and for flights in or entering an optimization region, flights communicate with up to n sectors.

2. Regions are responsible for identifying incoming boundary flights. If there are no boundary flights for a given step for either region, the regions can manage traffic independently, and the procedure for boundary flights is not necessary.

3. If there are boundary flights, they must share information according to the guidelines of the region they wish to enter.

4. **Boundary flights in cycle:** Identify all flights in cycles. This is done through the message passing described in 4.3.1. If any of the boundary flights are in a cycle, they *must* be allowed to proceed to their next sector. That is, the region it wishes to enter may not hold this flight in its current sector because doing so would cause gridlock. Figure 5.3b shows an example of an inter-region cycle that is allowed to proceed.

5. **Boundary flights entering protocol**: Consider boundary flight $i \in F$ where F is the set of all flights and flight i wishes to enter sector s in a protocol region (i.e., $\hat{x}(i,t) = s$). If there is no flight currently in s and no flight wishing to enter s (i.e., $\nexists\ j \in F \mid \hat{x}(j,t) = s \vee\ x(j,t) = s$), allow the boundary flight to proceed. If there is a flight j currently in s, proceed up the *chain* of flights containing j until the sector at the head of the chain is reached. Deconflict this sector using the protocol to determine whether this boundary flight can proceed. In Figure 5.3c, the flight in d4 is *not* allowed

to proceed to c4 because of the protocol's decision in b4. If there is no flight in s but there is a conflict at s, deconflict sector s using the protocol to determine whether this boundary flight can proceed.

6. **Boundary flights entering optimization:** Follow the same procedure as for a boundary flight entering a protocol region—proceed up the chain of flights until the head of the chain is reached. In Figure 5.3d, the flight at c3 is allowed to enter and proceed to d3 based on the decision made at the orange sector e3 to allow the flight in d3 to proceed to e3.

    - Note that the optimization region may need less than n steps of information from a boundary flight if the current time is in the middle of a time window. For example, if $n = 2$ and the current time window is $[0, 2]$ but the current time (t) is 1, then incoming boundary flights only provide one step of information.
    - If the current time is in the middle of a time window and there are incoming boundary flights into the optimization region or outgoing boundary flights that got held, the optimization region may need to re-solve flights in this time window. For example, suppose that $n = 2$, the current time is $t = 1$, and the current time window is $[0, 2]$. A boundary flight f entering (perhaps because it is part of a cycle) may occupy the intended sector of another flight g. The optimization region must allow f to enter, so it has to re-solve to consider the downstream effects on g and other flights.

7. Now that all boundary flights' go/hold decisions are determined, each region can manage its traffic for this step independently. The protocol remains the same in the protocol region as in Chapter 4, with the decision of the boundary flights predetermined. In the optimization region, the TFMP can be solved as in Chapter 3 with the one step decisions of the boundary flights fixed.

5.2.1 Federated Airspace Containing More than Two Regions

Cycles: It is possible (but rare) for cycles to span more than two regions. The orange flights form a cycle that covers all four regions in Figure 5.4. Regardless of the number of regions a cycle crosses, the message passing will allow for the detection of the cycle.

Boundary flights: In the procedure outlined above, the one-step actions of boundary flights depended solely on the region that the boundary flight wished to enter. However, this is not always the case in configurations with more than two regions. For example, in Figure 5.4, the top-right region (Optimization 1) does not have authority over whether the flight in c4 can enter, even though Optimization 1 contains the sector d4 which the flight wants to enter. This is because of the *chain* of flights that extends into the bottom-right region (Protocol 2).

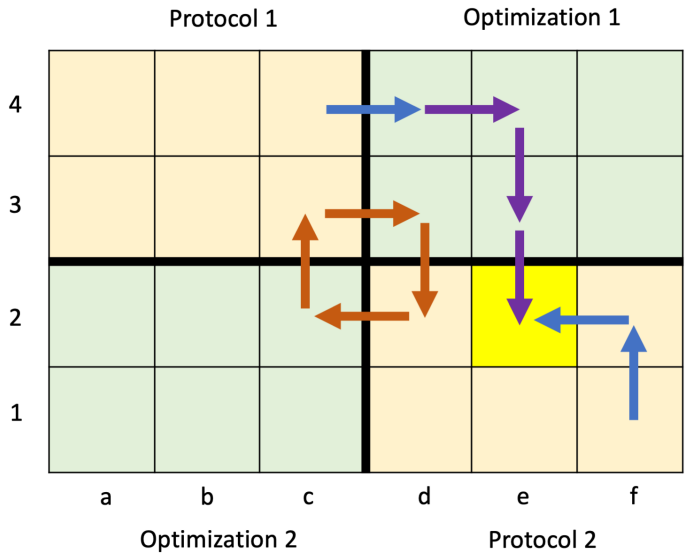


Figure 5.4: Federated Airspace Layout with 4 Regions. Orange flights are in a cycle. The highlighted sector e2 determines the one-step actions of boundary flights in e3 and c4.

When Optimization 1 is aware of the boundary flight in c4, it needs to first determine if it has enough knowledge to accept or reject the flight. In federated airspace configurations with two regions, Optimization 1 has enough knowledge. However, in configurations with more than two regions, Optimization 1 needs to see if the flight in c4 forms a chain with

any boundary flights exiting Optimization 1. In this case, the boundary flight in e3 wishing to enter e2 forms the head of the chain containing flights in [e3,e4,d4,c4]. Thus, Protocol 2 deconflicts the highlighted sector e2. This has downstream effects on the boundary flights in e3 and c4.

5.3 Experimental Setup

We use a similar experimental setup to Chapter 4 with simulated traffic of 250 flights with random origin and destination pairs across 60 time steps. The airspace is arranged in a 10 $\times$ 10 grid of square cells, each with a unit capacity. Example airspace configurations with different dimensions are shown in Figure 5.1. The results are averaged over 100 scenarios. The resulting delay is sensitive to various factors, including the airspace configuration (i.e., how many regions and the type of traffic control used), the optimization time step length, and the proportion of flights that cross between regions vs. stay within one region. We consider the following airspace configurations:

- 1 Region, Protocol: The entire airspace is assigned to one continuous region. The decentralized protocol is used for traffic management. Thus, flights share one time step of information at a time. This should be the least efficient (highest delay) configuration.
- Single Window TFMP: The entire airspace is assigned to one continuous region, which uses centralized traffic management (the TFMP). We call it a single window TFMP because all flights are expected to share their full trajectory information. Thus, the TFMP is solved only once, and the delays of all flights is prescribed at once with the one solve. This should be the most efficient (lowest delay) configuration, since it requires the most information sharing.
- 1 Region, O10: The entire airspace is assigned to one continuous region, which uses centralized traffic management. However, flights are not required to share full information; instead, the n-step TFMP is used. Flights need to share up to n time steps of

information in each time window, and n is varied. The "O10" part indicates that the entire width of the airspace is controlled by optimization.

- 2 Regions, O5|O5: The airspace is divided vertically into two regions, both of which use centralized traffic management. The width of each airspace is 5 cells (hence the "O5" notation), so the total size of each region is 5×10. The n-step TFMP is again used. We assume that the regions use the same n as each other. Boundary flights appear when a flight wants to cross from one region into another in a time step.

- 2 Regions, O5|P5: The airspace is again divided into two regions, but this time the right-hand side region is controlled by the decentralized protocol rather than optimization.

- 3 Regions, O3|P4|O3: From left to right, there are three regions, an optimization region of width 3, a protocol region of width 4, and an optimization region of width 3.

5.4 Experimental Results

Figure 5.5 plots the total delay vs. the optimization time step length for the airspace configurations described above. The total delay is the sum of delays of all flights (equivalently, the sum of delays across regions). The "1 Region: Protocol" and "Single Window TFMP" configurations provide upper and lower benchmarks, respectively. These benchmarks are not impacted by the x-axis of optimization time step length. As expected, the "Single Window TFMP" has the lowest delay, whereas the "1 Region: Protocol" has the highest, given that it utilizes the least amount of information. The remaining four configurations are plotted against optimization time step length. For any optimization region in the configuration, we assume it utilizes an n-step TFMP with n equal to the value indicated on the x-axis. We first note that delay generally decreases as the optimization time step length n increases. This is because flights share more information so regions can make more informed delay

decisions. Increasing n is most effective for the "1 Region: O10" configuration since it has the largest continuous optimization region. In contrast, the federated layouts see diminishing improvements in delay as n increases, particularly between $n = 8$ and $n = 16$, where there are negligible benefits to increasing n. We also see that the federated layouts have higher delays. As the number of regions increases (from one to two to three), delay increases. Moreover, the federated layouts with more optimization regions perform better, again due to the increased information sharing. The most direct comparison is "2 Regions: O5|O5" vs. "2 regions: O5|P5" where the former has a lower delay across all n than the latter.

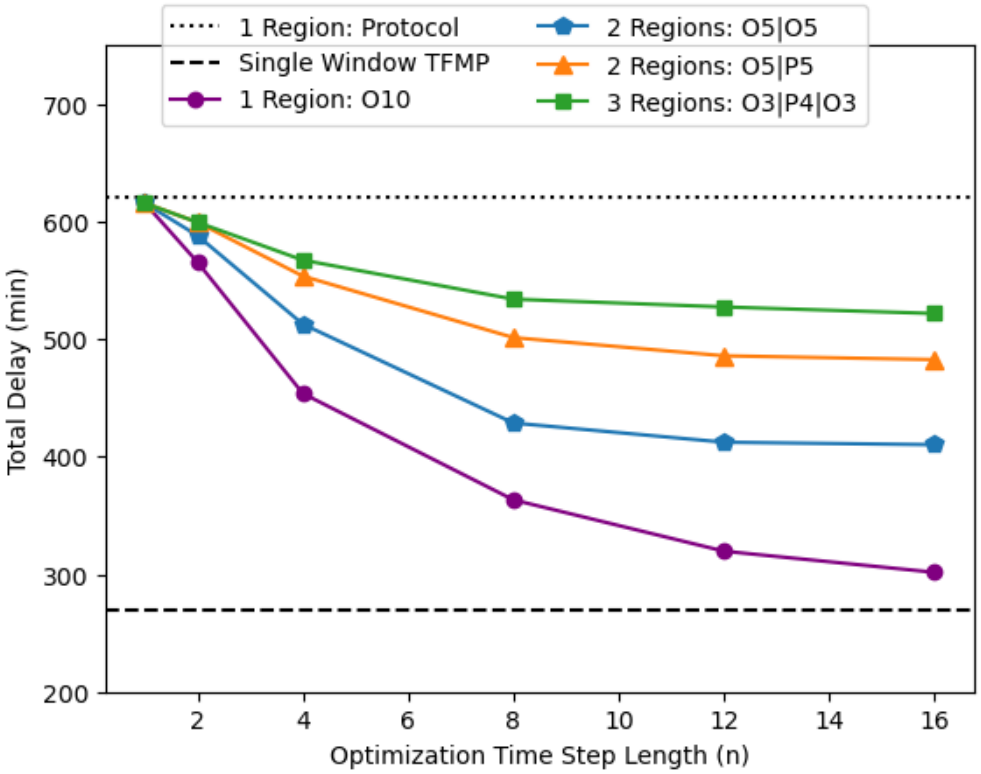


Figure 5.5: Delay Comparison of Different Airspace Configurations. "O10" indicates an optimization region of width 10, whereas "P5" indicates a protocol region of width 5.

The loss in efficiency with federated airspace compared to configurations with one region is due to boundary flights that cross from one region to another. An optimization region is aware that a crossing flight is approaching it up to n steps in advance. However, while the crossing flight is in another region, the optimization region has no control over the flight.

When the crossing flight appears as a boundary flight, the optimization region has to adapt to this new arrival. Figure 5.6 tests the sensitivity of delay to the *crossing proportion*, or proportion of flights that cross from one region to another. We show delay relative to the "1 Region: O10" configuration for three configurations: "2 Regions: O5|O5", "2 Regions: O5|P5", and "1 Region: Protocol". These configurations are arranged similarly to in Figure 5.5 where the two regions are adjacent to each other, from left to right. We use a similar process of generating origin-destination pairs randomly, except this time we utilize the crossing proportion. When the crossing proportion is 0.25, this indicates that 25% of flights cross from one region to another (i.e., from left to right or right to left, since the regions are adjacent to each other). For these crossing flights, we pick the origin randomly and then choose the destination to be in the other region. For example, suppose the 10×10 grid is arranged in the x-y plane where x and y are both in $[0, 9]$. If an origin is [1,3], then the destination would have $x > 4$ and $y \in [0, 9]$ (i.e., the destination would be in the right-hand-side region since the origin is in the left-hand-side region). We fix $n = 8$, so in optimization regions, flights share 8 steps of information.

The configuration with two optimization regions ("2 Regions: O5|O5") has a lower delay than the configuration with a protocol and optimization region ("2 Regions: O5|P5"). This follows the trend seen before in Figure 5.5. As the crossing proportion increases, the delay difference between the federated two region layouts and the baseline single region ("1 Region: O10) configuration increases. That is, having more crossing flights increases delay for the federated setups. This effect is more pronounced for the "2 Regions: O5|O5" configuration than the "2 Regions: O5|P5" configuration since optimization regions are more affected by boundary flights. With a crossing proportion of 1.0 (i.e., all flights cross from one region to another), there is still a benefit to using some optimization relative to the "1 Region: Protocol" configuration. However, it is best for crossing proportion to be minimized in federated layouts in order to have delay values closer to the delay from the one optimization region configuration.

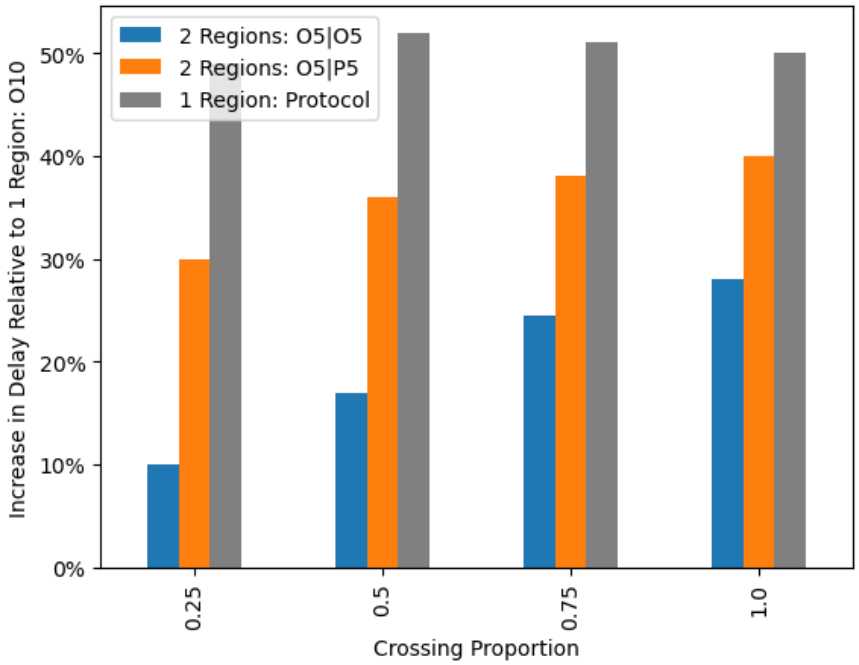


Figure 5.6: Delay comparison relative to "1 Region: O10" configuration as crossing proportion varies. Crossing proportion is the proportion of flights that cross from one region to another.

5.5 Discussion

In this chapter, we explored federated layouts of airspace. We consider airspace divided into multiple regions, where each region can use previously discussed centralized optimization or decentralized protocol traffic management methods. A key challenge is handling flights that cross from one region to another in federated setups. We compared the delay performance of various setups and showed that having fewer regions leads to lower delays. This is because of the delay burden of coordinating boundary flights. In addition, there is a trade-off between the increased information sharing required by centralized traffic management results with its lower delays compared to the decentralized protocol. We also performed sensitivity analysis on the proportion of crossing flights and found that delay increases as the proportion of crossing flights increases. This suggests that when federated airspace layouts are used, it is best to structure regions to minimize crossing flights. This can vary based on traffic layouts and patterns.

Chapter 6

Conclusion

In this dissertation, we considered centralized and decentralized methods for advanced air mobility traffic management. We first adopted the traffic flow management problem (TFMP) from commercial aviation in Chapter 3. This requires that flights share complete trajectory information with the central traffic manager. We considered trade-offs between efficiency and fairness and analyzed various scenarios where operator preference for fairness varied. We performed a sensitivity analysis on aircraft operators' fairness preferences and market share on fairness. In a two-operator setting, system efficiency and fairness are at their best when the two operators have the same notion of fairness and value them similarly. Additionally, fairness improves when the operator with a dominant market share has a weak preference for fairness.

Recognizing that demand may appear with little lead time (i.e., with a short file-ahead time), we tested a rolling horizon framework. With the rolling horizon, flights still share their entire flight trajectory, but they do not need to share this information as early. To be included in their horizon, they must submit their trajectory before the start of the time horizon that contains their scheduled departure. We also considered two options for handling pop-up flights: inserting them into the schedule or delaying them until the next time horizon. We found that with a low occurrence of pop-ups, either using longer horizons and inserting

pop-ups or using shorter horizons with either pop-up option is acceptable. However, with high occurrences of pop-ups, it is best to use a short horizon and insert pop-ups.

We presented an opposite approach in Chapter 4, developing a decentralized AAM traffic management protocol. One advantage of this approach is the minimal information-sharing requirements. This may be beneficial to AAM operators who may want to preserve privacy and flexibility. Specifically, our protocol uses a rules-of-the-road, one time step approach with unit capacity. We leverage cycle detection to avoid gridlock and use the backpressure metric for sectors to self-organize and deconflict in the optimal order. We incorporate fairness through prioritization schemes, including fairness metrics used in Chapter 3, like reversals and time-order deviation. We also incorporate unique fairness metrics for the protocol, including accrued delay and dominant resource fairness. To show that our protocol with backpressure prioritization results in a minimum delay solution in one time step, we showed how the protocol can be converted into a graph structure. With the graph, we leverage chains and cycles to show that backpressure prioritization will choose the longest chain at each decision point.

Multiple service providers may provide traffic management services in adjacent regions, so we consider a mixed airspace setting where traffic management could be either centralized or decentralized. We show how concepts from the decentralized protocol are instrumental in facilitating coordination between adjacent regions, particularly for boundary flights transiting between regions.

Some future directions of research are listed below:

- **Protocol Extensions**: The decentralized protocol leverages sector-to-sector communication, but it may desirable to utilize vehicle-to-vehicle communication (V2V) instead. We presented the intuition behind how to convert the protocol into a V2V setting, but additional work could be done to test this version of the protocol while considering sensitivity to communication speed. Since we wanted the protocol to be self-sustaining and not require additional deconfliction, we restricted the protocol to

unit capacity. However, the protocol could be extended to handle multi capacity. An individual sector with a capacity greater than 1 could be virtually split into single capacity units, or perhaps some network flow or graph theory approach could be used. In addition, we constrained our protocol to one time step of information sharing. Multi-time step information sharing generally requires a centralized system to leverage the extra information, but a protocol may be able to improve efficiency and fairness by gathering more steps of information from operators.

- **Pricing**: Third-party service suppliers will likely charge airspace users for traffic services. Besides traffic management services, they could also assist with route planning and weather monitoring. There is ample opportunity to study economic models of pricing airspace access. For example, when multiple operators want to utilize the same airspace, service suppliers could set up an auction for airspace. Alternatively, they could set up a pricing scheme that regulates demand. This raises interesting questions on fairness. Specifically, under a pay-to-play scheme, it is necessary to ensure that smaller operators get fair access to airspace.

- **Interactions between Service Suppliers**: In this dissertation, we only considered scenarios where service suppliers have sole authority over an airspace. It may be possible for multiple service suppliers to operate in the same airspace. Even more complicated, operators may become large enough that they simultaneously are service suppliers and operators (e.g., a large corporation like Amazon may operate delivery drones and provide third-party traffic management services to others). Coordination between service suppliers becomes crucial when they share the same airspace. This is in addition to the coordination each service supplier must do among their own flights. We performed some related work in this area related to airport slot trading among airlines using a delay ledger mechanism.

- **Airspace Reservations/Corridors**: Instead of flights requesting specific trajecto-

ries, some concepts of operations allow operators to request large sections of airspace. This gives operators the flexibility to modify their trajectory within the reserved airspace or operate multiple flights in the same corridor. This concept is appealing when there is only one or a few large operators in an airspace. However, if airspace reservations are allowed to be too large in terms of size and duration, this could degrade the efficiency and fairness of the system. There are also proposals to set up airspace corridors for the exclusive use of certain classes of operations (e.g., a corridor for air taxis). Setting up the geometry of these corridors (whether static or dynamic) is an interesting problem.

- **Non-compliance**: In this dissertation, we assumed that aircraft complied with assigned instructions, either in centralized or decentralized airspace. However, flights may not always comply. AAM vehicles may be susceptible to wind, making them unable to reach assigned sectors at designated times. In addition, other vehicles may declare emergencies and need to be expedited. Other traffic may need to adjust to accommodate non-compliant aircraft. In airspace with pricing schemes, there could also be financial penalties for non-compliance if it is determined that operators are gaming the system. For example, ridesharing services could be hogging airspace to crowd out other competitors.

There are several exciting areas of research related to not just AAM traffic management but also AAM operations. The adaptation of AAM will also depend on public acceptance, particularly how bystanders react to noise and visual pollution related to AAM operations. Battery technology will also play a crucial role in many AAM vehicles. Research on AAM traffic management could also have implications for connected vehicle transportation on the ground. For example, the decentralized protocol in this dissertation could assist autonomous vehicles deconflict with each other, with traffic intersections replacing airspace sectors.

References

[1] P. Boucher, "'you wouldn't have your granny using them': Drawing boundaries between acceptable and unacceptable applications of civil drones," *Science and engineering ethics*, vol. 22, pp. 1391–1418, 2016.

[2] K. Balakrishnan, J. Polastre, J. Mooberry, R. Golding, and P. Sachs, "Blueprint for the Sky: The roadmap for the safe integration of autonomous aircraft," Airbus UTM, Tech. Rep., 2018.

[3] M. Doole, J. Ellerbroek, and J. Hoekstra, "Drone delivery: Urban airspace traffic density estimation," 2018.

[4] P. D. Vascik, R. J. Hansman, and N. S. Dunn, "Analysis of urban air mobility operational constraints," *Journal of Air Transportation*, vol. 26, no. 4, pp. 133–146, 2018.

[5] R. Sengupta, "Aviation = aircraft + airspace consumerizing airspace," in *USA/Europe Air Traffic Management R&D Seminar*, Jun. 2015.

[6] M. Hassanalian and A. Abdelkefi, "Classifications, applications, and design challenges of drones: A review," *Progress in Aerospace sciences*, vol. 91, pp. 99–131, 2017.

[7] L. A. Garrow, B. J. German, and C. E. Leonard, "Urban air mobility: A comprehensive review and comparative analysis with autonomous and electric

ground transportation for informing future research," *Transportation Research Part C: Emerging Technologies*, vol. 132, p. 103 377, 2021.

[8] N. A. Khan, N. Jhanjhi, S. N. Brohi, R. S. A. Usmani, and A. Nayyar, "Smart traffic monitoring system using unmanned aerial vehicles (uavs)," *Computer Communications*, vol. 157, pp. 434–443, 2020.

[9] J. Homola, M. Johnson, P. Kopardekar, A. Andreeva-Mori, D. Kubo, K. Kobayashi, and Y. Okuno, "Utm and d-net: Nasa and jaxa's collaborative research on integrating small uas with disaster response efforts," in *2018 Aviation Technology, Integration, and Operations Conference*, 2018, p. 3987.

[10] E. Ackerman and M. Koziol, "The blood is here: Zipline's medical delivery drones are changing the game in rwanda," *IEEE spectrum*, vol. 56, no. 5, pp. 24–31, 2019.

[11] S. B. Cwerner, "Vertical flight and urban mobilities: The promise and reality of helicopter travel," *Mobilities*, vol. 1, no. 2, pp. 191–215, 2006.

[12] D.-N. Joint *et al.*, "Concepts studies for future intracity air transportation systems," [Cambridge, Mass.]: Massachusetts Institute of Technology, Dept, of ..., Tech. Rep., 1970.

[13] "Part 107 regulations," Federal Aviation Administration, Tech. Rep., 2020.

[14] B. Crumley, "Faa certifies zipline to operate us drone deliveries as an airline," *Drone DJ*, [Online]. Available: https://dronedj.com/2022/06/22/faa-certifies-zipline-to-operate-us-drone-deliveries-as-an-airline/.

[15] S. Verma, J. Keeler, T. E. Edwards, and V. Dulchinos, "Exploration of near term potential routes and procedures for urban air mobility," in *AIAA aviation 2019 forum*, 2019, p. 3624.

[16] J. Rios, I. Smith, P. Venkatesan, J. Homola, M. Johnson, and J. Jung, "UAS Service Supplier Specification: Baseline requirements for providing USS services within the UAS Traffic Management System," NASA, Tech. Rep., 2019.

[17] Federal Aviation Administration, "Urban Air Mobility: Concept of Operations v1.0," FAA, Tech. Rep., 2020.

[18] C. Barrado, M. Boyero, L. Brucculeri, G. Ferrara, A. Hately, P. Hullah, D. Martin-Marrero, E. Pastor, A. P. Rushton, and A. Volkert, "U-Space Concept of Operations: A Key Enabler for Opening Airspace to Emerging Low-Altitude Operations," en, *Aerospace*, vol. 7, no. 3, p. 24, Mar. 2020, Number: 3 Publisher: Multidisciplinary Digital Publishing Institute, ISSN: 2226-4310. DOI: 10.3390/aerospace7030024. [Online]. Available: https://www.mdpi.com/2226-4310/7/3/24 (visited on 07/14/2022).

[19] J. Rios, "Strategic deconfliction: System requirements," *NASA UAS Traffic Management (UTM) Project*, 2018.

[20] T. Prevot, J. Rios, P. Kopardekar, J. E. Robinson III, M. Johnson, and J. Jung, "UAS traffic management (UTM) concept of operations to safely enable low altitude flight operations," in *16th AIAA Aviation Technology, Integration, and Operations Conference*, 2016, p. 3292.

[21] C. Van Tilburg, *Traffic and congestion in the Roman Empire*. Routledge, 2007.

[22] A. Barnett, "Aviation safety: A whole new world?" *Transportation science*, vol. 54, no. 1, pp. 84–96, 2020.

[23] K. Yang, D. K. Jeon, N. N. M. Alsubaie, V. Sharma, and B. Wheeler, "Analyzing flight delays: Legacy and low-cost carriers in the us," 2023.

[24] M. Ball, C. Barnhart, M. Dresner, M. Hansen, K. Neels, A. Odoni, E. Peterson, L. Sherry, A. Trani, B. Zou, *et al.*, "Total delay impact study," in *NEXTOR Research Symposium, Washington DC*, 2010.

[25] M. Terrab and A. R. Odoni, "Strategic flow management for air traffic control," *Operations research*, vol. 41, no. 1, pp. 138–152, 1993.

[26] N. A. Ribeiro, A. Jacquillat, A. P. Antunes, A. R. Odoni, and J. P. Pita, "An optimization approach for airport slot allocation under iata guidelines," *Transportation Research Part B: Methodological*, vol. 112, pp. 132–156, 2018.

[27] S. J. Rassenti, V. L. Smith, and R. L. Bulfin, "A combinatorial auction mechanism for airport time slot allocation," *The Bell Journal of Economics*, pp. 402–417, 1982.

[28] J. Langan-Fox, M. J. Sankey, and J. M. Canty, "Human factors measurement for future air traffic control systems," *Human factors*, vol. 51, no. 5, pp. 595–637, 2009.

[29] W.-C. Li, P. Kearney, G. Braithwaite, and J. J. Lin, "How much is too much on monitoring tasks? visual scan patterns of single air traffic controller performing multiple remote tower operations," *International journal of industrial ergonomics*, vol. 67, pp. 135–144, 2018.

[30] V. Lopezbarrena Arenas, "Study for the implementation of remote control towers in european airports," B.S. thesis, Universitat Politècnica de Catalunya, 2021.

[31] D. Gianazza, "Forecasting workload and airspace configuration with neural networks and tree search methods," *Artificial intelligence*, vol. 174, no. 7-8, pp. 530–549, 2010.

[32] M. Bloem, P. Kopardekar, and P. Gupta, "Algorithms for combining airspace sectors," *Air Traffic Control Quarterly*, vol. 17, no. 3, pp. 245–268, 2009.

[33] A. Basu, J. S. Mitchell, and G. K. Sabhnani, "Geometric algorithms for optimal airspace design and air traffic controller workload balancing," *Journal of Experimental Algorithmics (JEA)*, vol. 14, pp. 2–3, 2010.

[34] M. Xue, "Airspace sector redesign based on voronoi diagrams," *Journal of Aerospace Computing, Information, and Communication*, vol. 6, no. 12, pp. 624–634, 2009.

[35] J. Li, T. Wang, M. Savai, and I. Hwang, "Graph-based algorithm for dynamic airspace configuration," *Journal of guidance, control, and dynamics*, vol. 33, no. 4, pp. 1082–1094, 2010.

[36] O. Richetta and A. R. Odoni, "Dynamic solution to the ground-holding problem in air traffic control," *Transportation research part A: Policy and practice*, vol. 28, no. 3, pp. 167–185, 1994.

[37] O. Richetta and A. R. Odoni, "Solving optimally the static ground-holding policy problem in air traffic control," *Transportation science*, vol. 27, no. 3, pp. 228–238, 1993.

[38] P. B. Vranas, D. J. Bertsimas, and A. R. Odoni, "The multi-airport ground-holding problem in air traffic control," *Operations Research*, vol. 42, no. 2, pp. 249–261, 1994.

[39] L. Brunetta, G. Guastalla, and L. Navazio, "Solving the multi-airport ground holding problem," *Annals of operations research*, vol. 81, no. 0, pp. 271–288, 1998.

[40] A. R. Odoni, "The flow management problem in air traffic control," in *Flow control of congested networks*, Springer, 1987, pp. 269–288.

[41] D. Bertsimas and S. S. Patterson, "The air traffic flow management problem with enroute capacities," *Operations research*, vol. 46, no. 3, pp. 406–422, 1998.

[42] D. Bertsimas, G. Lulli, and A. Odoni, "An integer optimization approach to large-scale air traffic flow management," *Operations research*, vol. 59, no. 1, pp. 211–227, 2011.

[43] G. Lulli and A. Odoni, "The European Air Traffic Flow Management Problem," *Transportation Science*, vol. 41, no. 4, pp. 431–443, Nov. 2007.

[44] D. Bertsimas, G. Lulli, and A. Odoni, "The air traffic flow management problem: An integer optimization approach," in *Integer Programming and Combinatorial Optimization: 13th International Conference, IPCO 2008 Bertinoro, Italy, May 26-28, 2008 Proceedings 13*, Springer, 2008, pp. 34–46.

[45] J. Krozel, R. Jakobovits, and S. Penny, "An algorithmic approach for airspace flow programs," *Air Traffic Control Quarterly*, vol. 14, no. 3, pp. 203–229, 2006.

[46] T. Vossen, M. Ball, R. Hoffman, and M. Wambsganss, "A general approach to equity in traffic flow management and its application to mitigating exemption bias in ground delay programs," *Air Traffic Control Quarterly*, vol. 11, no. 4, pp. 277–292, 2003.

[47] D. Bertsimas and S. Gupta, "Fairness and collaboration in network air traffic flow management: An optimization approach," *Transportation Science*, vol. 50, no. 1, pp. 57–76, 2015.

[48] C. Barnhart, D. Bertsimas, C. Caramanis, and D. Fearing, "Equitable and efficient coordination in traffic flow management," *Transportation science*, vol. 46, no. 2, pp. 262–280, 2012.

[49] D. Bertsimas and S. Gupta, "Fairness and collaboration in network air traffic flow management: An optimization approach," *Transportation Science*, vol. 50, no. 1, pp. 57–76, 2016. DOI: 10.1287/trsc.2014.0567.

[50] D. Bertsimas, V. F. Farias, and N. Trichakis, "On the Efficiency-Fairness Trade-off," en, *Management Science*, vol. 58, no. 12, pp. 2234–2250, Dec. 2012, ISSN: 0025-1909, 1526-5501. DOI: 10.1287/mnsc.1120.1549. (visited on 07/14/2022).

[51] D. Bertsimas, V. F. Farias, and N. Trichakis, "The Price of Fairness," en, *Operations Research*, vol. 59, no. 1, pp. 17–31, Feb. 2011, ISSN: 0030-364X, 1526-5463. DOI: 10.1287/opre.1100.0865.

[52] O. Rodionova, H. Arneson, B. Sridhar, and A. Evans, "Efficient trajectory options allocation for the collaborative trajectory options program," in *2017 IEEE/AIAA 36th Digital Avionics Systems Conference*, IEEE, 2017.

[53] A. Nilim and L. El Ghaoui, "Algorithms for air traffic flow management under stochastic environments," in *Proceedings of the 2004 American Control Conference*, IEEE, vol. 4, 2004, pp. 3429–3434.

[54] G. Andreatta, P. Dell'Olmo, and G. Lulli, "An aggregate stochastic programming model for air traffic flow management," *European Journal of Operational Research*, vol. 215, no. 3, pp. 697–704, 2011.

[55] A. Agustı, A. Alonso-Ayuso, L. Escudero, and C. Pizarro, "On air traffic flow management with rerouting. part ii: Stochastic case," *European Journal of Operational Research*, vol. 219, no. 1, pp. 167–177, 2012.

[56] K. Ng, C. Lee, F. Chan, and Y. Qin, "Robust aircraft sequencing and scheduling problem with arrival/departure delay using the min-max regret approach," *Transportation Research Part E: Logistics and Transportation Review*, vol. 106, pp. 115–136, 2017.

[57] K. Ng, C. Chen, and C. Lee, "Mathematical programming formulations for robust airside terminal traffic flow optimisation problem," *Computers & Industrial Engineering*, vol. 154, p. 107 119, 2021.

[58] J. Chen, L. Chen, and D. Sun, "Air traffic flow management under uncertainty using chance-constrained optimization," *Transportation Research Part B: Methodological*, vol. 102, pp. 124–141, 2017.

[59] M. C. R. Murca, "Collaborative air traffic flow management: Incorporating airline preferences in rerouting decisions," *Journal of Air Transport Management*, vol. 71, pp. 97–107, 2018.

[60] Y. Xu, X. Prats, and D. Delahaye, "Synchronised demand-capacity balancing in collaborative air traffic flow management," *Transportation Research Part C: Emerging Technologies*, vol. 114, pp. 359–376, 2020.

[61] M. O. Ball, C.-Y. Chen, R. Hoffman, and T. Vossen, *Collaborative decision making in air traffic management: Current and future research directions*. Springer, 2001.

[62] A. Bicchi, A. Marigo, G. Pappas, M. Pardini, G. Parlangeli, C. Tomlin, and S. Sastry, "Decentralized air traffic management systems: Performance and fault tolerance," *IFAC Proceedings Volumes*, vol. 31, no. 27, pp. 259–264, 1998.

[63] G. Pappas, C. Tomlin, J. Lygeros, D. Godbole, and S. Sastry, "A next generation architecture for air traffic management systems," in *Proceedings of the 36th IEEE Conference on Decision and Control*, IEEE, vol. 3, 1997, pp. 2405–2410.

[64] M. Tlig, O. Buffet, and O. Simonin, "Decentralized traffic management: A synchronization-based intersection control," in *2014 International Conference on Advanced Logistics and Transport (ICALT)*, IEEE, 2014, pp. 109–114.

[65] A. F. Santamaria, C. Sottile, A. Lupia, and P. Raimondo, "An efficient traffic management protocol based on ieee802. 11p standard," in *International Symposium on Performance Evaluation of Computer and Telecommunication Systems (SPECTS 2014)*, IEEE, 2014, pp. 634–641.

[66] S. I. Guler, M. Menendez, and L. Meier, "Using connected vehicle technology to improve the efficiency of intersections," *Transportation Research Part C: Emerging Technologies*, vol. 46, pp. 121–131, 2014.

[67] S. H. Low, F. Paganini, and J. C. Doyle, "Internet congestion control," *IEEE control systems magazine*, vol. 22, no. 1, pp. 28–43, 2002.

[68] A. Atta, S. Abbas, M. A. Khan, G. Ahmed, and U. Farooq, "An adaptive approach: Smart traffic congestion control system," *Journal of King Saud University-Computer and Information Sciences*, 2018.

[69] H. Khadilkar and H. Balakrishnan, "Network congestion control of airport surface operations," *Journal of Guidance, Control, and Dynamics*, vol. 37, no. 3, pp. 933–940, 2014.

[70] A. Eryilmaz and R. Srikant, "Fair resource allocation in wireless networks using queue-length-based scheduling and congestion control," *IEEE/ACM transactions on networking*, vol. 15, no. 6, pp. 1333–1344, 2007.

[71] S. Badrinath and H. Balakrishnan, "Robust control of arrivals into a queuing network," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 5, pp. 4474–4489, 2022. DOI: 10.1109/TITS.2020.3045030.

[72] J. Gregoire, X. Qian, E. Frazzoli, A. De La Fortelle, and T. Wongpiromsarn, "Capacity-aware backpressure traffic signal control," *IEEE Transactions on Control of Network Systems*, vol. 2, no. 2, pp. 164–173, 2014.

[73] X. Sun and Y. Yin, "A simulation study on max pressure control of signalized intersections," *Transportation research record*, vol. 2672, no. 18, pp. 117–127, 2018.

[74] M. W. Levin and D. Rey, "Conflict-point formulation of intersection control for autonomous vehicles," *Transportation Research Part C: Emerging Technologies*, vol. 85, pp. 528–547, 2017.

[75] M. Ribeiro, J. Ellerbroek, and J. Hoekstra, "Review of conflict resolution methods for manned and unmanned aviation," *Aerospace*, vol. 7, no. 6, p. 79, 2020.

[76] E. D'Amato, M. Mattei, and I. Notaro, "Distributed reactive model predictive control for collision avoidance of unmanned aerial vehicles in civil airspace," *Journal of Intelligent & Robotic Systems*, vol. 97, no. 1, pp. 185–203, 2020.

[77] I. Hwang, J. Kim, and C. Tomlin, "Protocol-based conflict resolution for air traffic control," *Air Traffic Control Quarterly*, vol. 15, no. 1, pp. 1–34, 2007.

[78] H. Y. Ong and M. J. Kochenderfer, "Markov decision process-based distributed conflict resolution for drone air traffic management," *Journal of Guidance, Control, and Dynamics*, vol. 40, no. 1, pp. 69–80, 2017.

[79] E. Cruck and J. Lygeros, "A mathematical framework for subliminal air traffic control," in *AIAA Guidance, Navigation and Control Conference and Exhibit*, 2007, p. 6693.

[80] S. Lu, V. Bharghavan, and R. Srikant, "Fair scheduling in wireless packet networks," *IEEE/ACM Transactions on networking*, vol. 7, no. 4, pp. 473–489, 1999.

[81] A. D. Evans, M. Egorov, and S. Munn, "Fairness in Decentralized Strategic Deconfliction in UTM," in *AIAA Scitech 2020 Forum*, 2020, p. 2203.

[82] C. Chin, K. Gopalakrishnan, M. Egorov, A. Evans, and H. Balakrishnan, "Efficiency and fairness in unmanned air traffic flow management," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 9, pp. 5939–5951, 2021. DOI: DOI:10.1109/TITS.2020.3048356.

[83] H. Balakrishnan and B. G. Chandran, "A distributed framework for traffic flow management in the presence of unmanned aircraft," in *USA/Europe Air Traffic Management R&D Seminar*, Jun. 2017.

[84] M. Xue, "Urban air mobility conflict resolution: Centralized or decentralized?" In *Aiaa aviation 2020 forum*, 2020, p. 3192.

[85] L. Sedov, V. Polishchuk, and V. Bulusu, "Decentralized self-propagating ground delay for UTM: Capitalizing on domino effect," in *2017 Integrated Communications, Navigation and Surveillance Conference (ICNS)*, IEEE, 2017, pp. 6C1–1.

[86] L. Sedov and V. Polishchuk, "Centralized and distributed UTM in layered airspace," in *8th ICRAT*, 2018.

[87] Í. R. de Oliveira, E. C. P. Neto, T. T. Matsumoto, and H. Yu, "Decentralized air traffic management for advanced air mobility," in *2021 Integrated Communications Navigation and Surveillance Conference (ICNS)*, IEEE, 2021, pp. 1–8.

[88] S. Bharadwaj, S. Carr, N. Neogi, and U. Topcu, "Decentralized control synthesis for air traffic management in urban air mobility," *IEEE Transactions on Control of Network Systems*, vol. 8, no. 2, pp. 598–608, 2021.

[89] K. Yang and S. Sukkarieh, "3d smooth path planning for a uav in cluttered natural environments," in *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, IEEE, 2008, pp. 794–800.

[90] Y. Wu, K. H. Low, B. Pang, and Q. Tan, "Swarm-based 4d path planning for drone operations in urban environments," *IEEE transactions on vehicular technology*, vol. 70, no. 8, pp. 7464–7479, 2021.

[91] Y. Zhao, Z. Zheng, and Y. Liu, "Survey on computational-intelligence-based uav path planning," *Knowledge-Based Systems*, vol. 158, pp. 54–64, 2018.

[92] H. Tang, Y. Zhang, V. Mohmoodian, and H. Charkhgard, "Automated flight planning of high-density urban air mobility," *Transportation Research Part C: Emerging Technologies*, vol. 131, p. 103 324, 2021.

[93] X. Yang and P. Wei, "Scalable multi-agent computational guidance with separation assurance for autonomous urban air mobility," *Journal of Guidance, Control, and Dynamics*, vol. 43, no. 8, pp. 1473–1486, 2020.

[94] X. He, F. He, L. Li, L. Zhang, and G. Xiao, "A route network planning method for urban air delivery," *Transportation Research Part E: Logistics and Transportation Review*, vol. 166, p. 102 872, 2022.

[95] M. O. Ball, G. L. Donohue, and K. Hoffman, "Auctions for the safe, efficient, and equitable allocation of airspace system resources," in *Combinatorial Auctions*, P. Cramton, Y. Shoham, and R. Steinberg, Eds., The MIT Press, Dec. 9, 2005, pp. 507–538, ISBN: 978-0-262-03342-8. DOI: 10.7551/mitpress/9780262033428.003.0021. [Online]. Available:

http://mitpress.universitypressscholarship.com/view/10.7551/mitpress/9780262033428.001.0001/upso-9780262033428-chapter-21 (visited on 10/06/2021).

[96] B. Skorup, "Auctioning airspace," *Mercatus Research Paper*, 2018.

[97] T. W. M. Vossen and M. O. Ball, "Slot trading opportunities in collaborative ground delay programs," *Transportation Science*, vol. 40, no. 1, pp. 29–43, Feb. 2006, ISSN: 0041-1655, 1526-5447. DOI: 10.1287/trsc.1050.0121. [Online]. Available: http://pubsonline.informs.org/doi/abs/10.1287/trsc.1050.0121 (visited on 10/13/2021).

[98] L. Corolli, T. Bolić, L. Castelli, and D. Rigonat, "Tradable mobility permits for the strategic allocation of air traffic," presented at the 6th International Conference on Research in Air Transportation, 2014, p. 8.

[99] B. Skorup, "Auctioning airspace," *North Carolina Journal of Law & Technology*, vol. 21, pp. 79–109, 2019.

[100] S. Seuken, P. Friedrich, and L. Dierks, "Market design for drone traffic management," *arXiv:2110.13784 [cs]*, Oct. 26, 2021. arXiv: 2110.13784. [Online]. Available: http://arxiv.org/abs/2110.13784 (visited on 10/27/2021).

[101] David M. Grether, R. M. Isaac, and Charles R. Plott, *The Allocation of Scarce Resources: Experimental Economics and the Problem of Allocating Airport Slots* (Underground Classics in Economics). Westview Press, 1989.

[102] Federal Aviation Administration, "Urban Air Mobility (UAM) Concept of Operations Version 1.0," Federal Aviation Administration, Washington, DC, Tech. Rep., Jun. 2020.

[103] D. Teodorović, K. Triantis, P. Edara, Y. Zhao, and S. Mladenović, "Auction-based congestion pricing," *Transportation Planning and Technology*, vol. 31, no. 4, pp. 399–416, Aug. 2008, ISSN: 0308-1060, 1029-0354. DOI: 10.1080/03081060802335042. [Online]. Available:

http://www.tandfonline.com/doi/abs/10.1080/03081060802335042 (visited on 01/05/2022).

[104] M. Vasirani and S. Ossowski, "A market-inspired approach for intersection management in urban road traffic networks," *Journal of Artificial Intelligence Research*, vol. 43, pp. 621–659, Apr. 24, 2012, ISSN: 1076-9757. DOI: 10.1613/jair.3560. [Online]. Available: https://jair.org/index.php/jair/article/view/10759 (visited on 01/13/2022).

[105] D. Carlino, S. D. Boyles, and P. Stone, "Auction-based autonomous intersection management," in *16th International IEEE Conference on Intelligent Transportation Systems (ITSC 2013)*, The Hague, Netherlands: IEEE, Oct. 2013, pp. 529–534, ISBN: 978-1-4799-2914-6. DOI: 10.1109/ITSC.2013.6728285. [Online]. Available: http://ieeexplore.ieee.org/document/6728285/ (visited on 01/05/2022).

[106] C. Chin, V. Qin, K. Gopalakrishnan, and H. Balakrishnan, "Traffic management protocols for advanced air mobility," *Frontiers in Aerospace Engineering*, vol. 2, p. 1 176 969, 2023.

[107] M. Egorov, V. Kuroda, and P. Sachs, "Encounter aware flight planning in the unmanned airspace," in *2019 Integrated Communications, Navigation and Surveillance Conference (ICNS)*, 2019.

[108] Y. Liu and M. Hansen, "Evaluation of the Performance of Ground Delay Programs," *Transportation Research Record*, vol. 2400, no. 1, pp. 54–64, Jan. 2014.

[109] K. Balakrishnan, J. Polastre, J. Mooberry, R. Golding, and P. Sachs, "Blueprint for the Sky: The roadmap for the safe integration of autonomous aircraft," Airbus UTM, Tech. Rep., 2018.

[110] Booz Allen Hamilton, "Urban Air Mobility (UAM) Market Survey," NASA, Tech. Rep., Nov. 2018, https://ntrs.nasa.gov/citations/20190001472.

[111] Crown Consulting, "Urban Air Mobility (UAM) Market Survey," NASA, Tech. Rep., Nov. 2018, https://ntrs.nasa.gov/citations/20190002046.

[112] Drone Industry Insights, https://droneii.com/global-companies-invest-in-drones-despite-recession, 2021.

[113] A. Karp, *Gambling on advanced air mobility*, Aerospace America, Apr. 2022.

[114] H. Idris, C. Chin, and A. D. Evans, "Accrued delay application in trajectory-based operations," in *USA/Europe Air Traffic Management R&D Seminar*, 2019.

[115] A. Ghodsi, M. Zaharia, B. Hindman, A. Konwinski, S. Shenker, and I. Stoica, "Dominant resource fairness: Fair allocation of multiple resource types.," in *Nsdi*, vol. 11, 2011, pp. 24–24.

[116] R. Rocha and B. Thatte, "Distributed cycle detection in large-scale sparse graphs," in *Proceedings of the Simposio Brasileiro de Pesquisa Operacional Pernambuco, Brazil*, Sobrapo, 2015, pp. 25–28.