Large Scale Multi-Objective Optimization for Dynamic Airspace Sectorization

Jiangjun Tang

M.S. (IT), ANU, Australia

B.A. (Economics), Shanghai University, China

A thesis submitted in partial fulfillment of the requirements

for the degree of Doctor of Philosophy at the

School of Engineering and Information Technology

University of New South Wales

Australian Defence Force Academy

Abstract

A key limitation in accommodating continuing air traffic growth is the fixed airspace structure (sector boundaries), which is largely determined by historical flight profiles that have evolved over time. The sector geometry has stayed relatively constant despite the fact that route structures and demand have changed dramatically over the past decade.

Dynamic Airspace Sectorization (DAS) is a concept where the airspace is redesigned dynamically to accommodate changing traffic demands. Various methods have been proposed to dynamically partition the airspace to accommodate traffic growth while also considering other sector constraints and efficiency metrics. However, these approaches suffer several operational drawbacks, and their computational complexity increases exponentially as the airspace size and traffic volume increase.

In this thesis, I experimentally evaluate and identify gaps in existing 3D sectorization methods, and propose an improved Agent Based Model (iABM) to address these gaps. I also propose three additional models using KD-Tree, Support Plane Bisection (SPBM) and Constrained Voronoi Diagrams (CVDM) in 3D, to partition the airspace to satisfy the convexity constraint and overcome high computational cost inherent in agent-based approaches. I then look into optimizing the airspace sectors generated by these four models (iABM, KD-Tree, SPBM, and CVDM), using a multi-objective optimisation approach with Air Traffic Controller (ATC) task load balancing, average sector flight time, and minimum distance between sector boundaries and traffic flow crossing points as the three objectives. The performance and efficiency of the proposed models are demonstrated by using sample air traffic data. Experimental results show that all the approaches have strengths and weaknesses. iABM has the best performance on task load balancing, but it can't satisfy the convexity constraint. SPBM and CVDM perform worse than iABM on task load balancing but better on average sector flight time, and they can satisfy the convexity constraint. The KD-tree based model is the most efficient, but not effective as it performed poorly on the given objectives because of its representational bias, which also limits its use in an operational environment.

To further investigate SPBM and CVDM for national airspace sectorization, a real time air traffic monitoring and advisory system, called TOP-LAT (Trajectory Optimization and Prediction of Live Air Traffic), is developed and implemented.

TOP-LAT is a real time system, synthesizing real time air traffic data to measure and analyse airspace capacity, airspace safety, air traffic flow and aviation emission, to enable ATM participants to access timely, accurate and reliable information for ATM decisions. TOP-LAT provides an ATM environment to evaluate and investigate the advanced ATM concepts, such as DAS. A number of experiments of Australian airspace sectorization by the two proposed DAS models are conducted in this thesis. In these experiments, the current and projected air traffic demands are generated based on public statistics, and some future ATM concepts (e.g. User Preferred Trajectory) are prototyped in order to investigate the performances of the proposed models. The results show that both models have advantages over the current airspace sector configurations in terms of task load balancing, longer flight sector time, larger minimum distance between sector boundaries and traffic flow crossing points, and reduced maximum task load for ATC. These experiments also show that Both models have the capability to be compatible with other advanced ATM concepts. However, no single approach can meet all complex air traffic management objectives. It is the air traffic flow pertaining to the kind of airspace and the associated traffic complexity which can determine the best approach for dynamic sectorization.

keywords

Dynamic Airspace Sectorization, Dynamic Airspace Configuration, Air Traffic Management, Task Load, Multi-agent systems, 3D Partitioning Methods, Multi-objective Optimization, Genetic Algorithms, Australian Airspace

Acknowledgement

No words of gratitude are enough to thank my supervisor Prof. Hussein A. Abbass, whose untiring support, motivation and supervision made this thesis. I also wish to acknowledge my co-supervisors, Dr. Chris Lokan and Dr. Sameer Alam, for their guidance and support throughout my thesis.

I wish to thank Viet Van Pham, Dr Vinh Quang Bui, and Wenjing Zhao from our research group, with whom I enjoyed my research collaboration.

Thanks are due to AirServices Australia for providing air traffic data and valuable discussions. I would like to extend my hearty appreciation to the anonymous reviewers for my journal and conference papers whose valuable suggestions were instrumental in improving the research work.

Finally, to my parents, my wife Yanxin and my son Ricky, for their constant encouragement, support and love during this endeavor.

Jiangjun Tang
Australia, 2011

Certificate of Originality

I hereby declare that this submission is my own work and that, to the best of my knowledge and belief, it contains no material previously published or written by another person, nor material which to a substantial extent has been accepted for the award of any other degree or diploma at UNSW or any other educational institution, except where due acknowledgement is made in the thesis. Any contribution made to the research by colleagues, with whom I have worked at UNSW or elsewhere, during my candidature, is fully acknowledged.

I also declare that the intellectual content of this thesis is the product of my own work, except to the extent that assistance from others in the project's design and conception or in style, presentation and linguistic expression is acknowledged.

Jiangjun Tang

Contents

List of Figures

List of Tables

List of Acronyms

2D	2-Dimensional
3D	3-Dimensional
4D	4-Dimensional
AAS	The advanced Airspace Scheme
ABM	Agent based Model
AIRWOLF	Automatic Identification of Risky Weather Objects in Line of Flight
ATC	Air Traffic Controller
ATCSCC	FAA ATC System Command Center
ATFM	Air Traffic Flow Management
ATM	Air Traffic Management
ATOMS	Air Traffic Operations and Management Simulator
BADA	Base of Aircraft Data
BEM2	Boeing Emission Method 2
BFS	Breadth First Search
CDA	Continuous Descent Approach
CDTI	Cockpit Display of Traffic Information
CTAS	The Center-TRACON Automation System
CVDM	Constrained Voronoi Diagram Model
DAC	Dynamic Airspace Configuration
DAS	Dynamic Airspace Sectorization
DFS	Depth First Search
DD	Dynamic Density
EC	Evolutionary Computation
ERAM	En-Route Automation Modernization
FAA	Federal Aviation Administration
FIR	Flight Information Region
GA	Genetic Algorithm
iABM	Improved Agent Based Model
ICAO	International Civil Aviation Organization
IFR	Instrument Flight Rules
ISA	International Standard Atmosphere
ITP	In-trail Procedure

JPDO	The Joint Planning and Development Office
KD-tree	k-dimensional tree
MIP	Mixed Integer Programming
MSP	Multi-Sector Planning
NextGen	The Next Generation of Air Transportation System
NCOIC	Network Centric Operations Industry Consortium
NSGA-II	Non-dominated Sorting Genetic Algorithm-II
NSWP	The National Severe Weather Playbook
OPAL DSS	Optimization Platform for Airports including Landside Decision Support System
RNP	Required Navigation Performance
SBX	Simulated Binary Crossover
SESAR	The Single European Sky ATM Research
SID	Standard instrument departure
SIGMET	Significant Meteorological Information
SPBM	Support Plane Bisection Model
ST-DBSCAN	Spatial-Temporal Density-Based Spatial Clustering of Applications with Noise
STAR	Standard Terminal Arrival Route
SUA	Special User Area
SWIM	System Wide Information Management
TF-OPTICS	Time Focus Ordering Points to Identify the Clustering Structure
TMA	Terminal Management Airspace
TOC	Top of Climb
TOD	Top of Descent
TOP-LAT	Trajectory Optimization and Prediction of Live Air Traffic
UPT	User Preferred Trajectory
URET	User Request Evaluation Tool
VFR	Visual Flight Rules

List of Publications

Journal Publications

1. Tang J., Alam S., Lokan C., and Abbass H. A. A Multi-Objective Approach for Dynamic Airspace Sectorization Using Agent Based and Geometric Models. Transportation Research Part C: Emerging Technologies, Volume 21, Issue 1, April 2012, Pages 89-121, ISSN 0968-090X, 10.1016/j.trc.2011.08.008.

2. Bui V., Pham V. V., Iorio A. W.,Tang J.,Alam S., and Abbass H. A. Bio-Inspired Robotics for Air Traffic Weather Information Management. Transactions of The Institute of Measurement and Control, 32:1–27, 2010.

3. Pham V. V., Tang J., Alam S., Lokan C., and Abbass H. A. Aviation emission inventory development and analysis. Environmental Modelling & Software, 25(12):1738–1753, 2010.

Conference Publications

1. Alam S., Zhao W., Tang J., Lokan C.,Abbass H. A., Ellejmi M., and Kirby S. Discovering delay patterns in arrival traffic with dynamic continuous descent approaches using co-evolutionary red teaming. In Proceedings of the 9th USA/Europe Air Traffic Management Research and Development Seminar, 2010. (Best Paper Award)

2. Zhao W., Tang J., Alam S., Bender A., and Abbass H. A. Evolutionary-computation based risk assessment of aircraft landing sequencing algorithms. In Distributed, Parallel and Biologically Inspired Systems, IFIP Advances in Information and Communication Technology.

3. Tang J., Alam S., Abbass H. A., and Lokan C. Modelling and evolutionary multi-objective evaluation of interdependencies and work processes in airport operations. In Proceedings of the first ACM/SIGEVO Summit on Genetic and Evolutionary Computation, GEC '09, 977–984, New York, USA, 2009.

4. Alam, S., Tang, J., Abbass, H.A., Lokan, C.J. The effect of symmetry in representation on scenario-based risk assessment for air-traffic conflict resolution

strategies. Evolutionary Computation on IEEE Congress, 2180–2187, May 2009

Chapter 1

Introduction

1.1 Overview

Air traffic continues to grow in demand and in the regions it serves; representing major challenges to the present Air Traffic Management system (ATM) (EUROCONTROL, 2008b; FAA, 2009). These challenges include congested airspace, flight delays, and environment impact. A key challenge is how to balance the limited airspace capacity in the face of growing air traffic demands.

The current airspace configuration is highly structured, fixed and is less responsive to changes causing the overall system to lack the flexibility, adaptability, and responsibility needed to handle the increasing air traffic demands in the near future. Meanwhile, the air traffic is managed by ground based air traffic controllers (ATC) who are responsible for safe and efficient air traffic management within a given airspace partition known as a sector. As a human, an ATC has cognitive limitations restricting the number of aircraft that one ATC can safely handle. In ATM, these cognitive limitations are measured by ATC workload which include the workload of monitoring, aircraft handover between sectors, conflict detection and resolution, and others. Moreover the air traffic is not evenly distributed in the airspace, which causes congestion in sectors sitting between the major airports, or the sectors around major airways. Hence, some ATCs are overloaded by the congested traffic while oth-

ers are not because of the unchangeable sector boundaries. This situation induces potential safety, efficiency, and other problems in busy sectors. Although the ATM system evolved along with the advances in technologies, the cognitive limitations of ATC and the fixed airspace configuration lag behind causing suboptimal efficient and safe airspace usage. Therefore, fundamental changes on the present airspace configuration, especially sectors, are required.

Dynamic Airspace Configuration (DAC) (Kopardekar *et al.*, 2007) is proposed in the future advanced ATM system including both the Next Generation of Air Transportation System (NextGen) (Joint Planning and Development Office, 2007) in the U.S.A. and the Single European Sky ATM Research (SESAR) (SESAR Consortium, 2007) in Europe. DAC not only aims to increase the airspace capacity to accommodate the growth of air traffic demands but also aims to be compatible with other advanced ATM concepts, such as User Preferred Trajectory (UPT), for a safe and efficient air traffic. Dynamic Airspace Sectorization (DAS) is a fundamental step of DAC, which adapts and changes sector shapes vertically and horizontally according to changes of the traffic demand for an optimised efficiency of the individual controllers as well as for the controller teams. DAS may be also triggered by other factors such as complexity and workload prediction and air traffic forecast for the day, Weather changes, and changes of ATM Policies on aviation safety. By dynamically sectorizing the airspace, ATC workload can be balanced or reduced. Hence, the limitations on the ATC and of the fixed sector boundaries can be overcome, and then the airspace capacity can be adjusted to accommodate fluctuated air traffic demands.

By its nature, DAS is a multi-objective space partitioning problem with several constraints. It aims to balance the ATC workload among sectors, the traffic flow cuts, and other operational objectives. At the same time, DAS has to satisfy the convexity of a sector, right prism in 3-Dimension (3D), computational efficiency, and other constraints. Therefore, DAS is an optimal space partitioning problem which is an NP-hard problem in 2 or higher dimensions (Khanna *et al.*, 1998).

One of the key requirements of DAS is an evaluation system where these

methodologies can be evaluated both on simulated and field test data. This system provides an environment that can evaluate DAS methodologies using both simulated and real air traffic data. Since DAS methodologies are supposed to be compatible with other activities in future ATM systems, these systems have to model and represent all major aspects of ATM, which include airspace modelling, aircraft movements, trajectory prediction, ATC workload, safety and security, weather conditions, and environment impact. It is necessary for such a system to provide a complete view of ATM activities when investigating DAS and other ATM concepts (Prevot, 2002). This air traffic monitoring system has advisory functionalities, when advanced concepts are used in an operational environment. However, such an evaluation environment doesn't exist to the present day. Some projects, such as the Network Centric operations Industry Consortium (NCOIC) (NCOIC, 2008), are still under development.

This thesis addresses the development of flexible, adaptable, and efficient multi-objective 3-Dimensional (3D) DAS models to satisfy the objectives of minimizing standard deviation of task load across sectors, maximizing average sector flight time, maximizing distance between sector boundaries and traffic flow crossing points, and minimizing the maximum task load of sectors, under different air traffic conditions. In particular, the focus of this thesis is Dynamic Airspace Sectorization methods in 3D and identifying which models are able to manage different air traffic demands and are compatible with future ATM concepts. However, Dynamic Airspace Re-sectorization methods, which continuously use DAS in an operational environment, are not included in this thesis.

To investigate DAS as well as other advanced ATM concepts, a real time air traffic monitoring and advisory system is developed. The objective of this system is to provide information on all aspects of ATM activities including air traffic common operating picture, weather conditions, safety, ATC task load, sector complexity, and environment impact and to act as an advisory system to decision makers to improve the ATM performance.

1.2 Motivation

Researchers have made many attempts to develop approaches for optimal DAS in the past decades, but most of them focus on the 2-Dimension (2D) airspace sectorization problem (Xue, 2008; Brinton and Pledgie, 2008; Trandac *et al.*, 2003; Basu *et al.*, 2009) because of the difficulties of space partitioning in higher dimensions. In fact, the traffic happens in a 3D airspace and the vertical aircraft moments have to be considered in order to achieve optimal airspace resectorization. Although a few attempts have been made to achieve optimal 3D DAS (Kicinger and Yousefi, 2009), they are not mature in practice and have some limitations in terms of not only the performance but also sector design.

Meanwhile, future ATM concepts including DAS, have to be investigated before they are implemented in the ATM system. Firstly, an environment representing both the current and future air traffic and ATM activities is needed for a successful transition to the future. Secondly, an environment being able to evaluate the ATM concepts with real or simulated air traffic is essential for concept prototyping and system testing. Finally, an automated advisory system providing decision makers with options of ATM activities is required when the advanced concepts used in practice. To satisfy these requirements, it is necessary to develop a system that is able to handle a large amount of air traffic data, model ATM activities and advanced concepts, and be able to provide air traffic situation picture and prediction, information sharing, advisory mechanism, and so on. Then, DAS methodologies and other ATM concepts can be investigated and evaluated in this system. However, no such systems have been built because of legacy constraints

My motivation of this thesis is that a 3D DAS optimization methodology is an essential step to achieve the balance between airspace capacity and air traffic demand which will lead to safety and efficiency in air travel. Furthermore, investigating and evaluating the 3D DAS methodology with advanced ATM concepts with simulated data as well as field air traffic data can pave the way to implement and use the advanced concepts in reality.

1.3 Research Questions and Hypothesis

Although the DAS promises certain advantages in better usage of airspace capacity (Zelinski, 2009), some challenges and problems exist:

- DAS deals with air traffic flows which are dynamic data in both time and space.
- DAS is highly dependant on current and future traffic demand and traffic flow, which are continuously changing and are hardly predictable in advance.
- DAS may be disruptive and could have negative consequences for ATCs situational awareness and performance
- DAS represents a radical change from current procedures that rely on published fixed sector boundaries over a long period of time. Therefore, the current ATM system is unable to adapt itself easily to accommodate DAS procedures.
- Fast evaluation tools to investigate DAS are required. Multiple evaluations and analysis on DAS are required. While this is a time consuming task, DAS is required to respond to the dynamic of traffic demands and traffic flow in a short time frame.

In this thesis, I wish to specially answer the following research question:

What is an appropriate 3D DAS model to perform dynamic airspace sectorization in order to satisfy fluctuating air traffic demands for a large airspace and under multiple objectives?

My hypothesis is that combining a well designed 3D DAS with multi-objective optimization methodologies can efficiently generate sectors in a given airspace according to predicted air traffic demands. Therefore, the primary objective of this thesis is to develop and implement feasible 3D DAS models to solve the multi-objective space partitioning problem with constraints. The objectives applied in this thesis include ATC task load balancing, long sector flight time for reducing

ATC handover workload, and large distance between sector boundaries and crossing points air traffic flows to reduce ATC conflict resolution and handover workload. At the same time, the constraints of sector design and computational efficiency are also considered. However, not all operational constraints and requirements, such as the ATC resource allocations, weather conditions and practical restrictions, are considered in this thesis. The secondary objective of the thesis is to design and implement a real time air traffic monitoring and advisory system not only for investigating DAS models and other advanced ATM concepts but also for integrating and using the advanced ATM concepts in the presence of DAS.

In order to answer the research questions, a set of sub-questions needs to be investigated:

1. **Are there any existing 3D DAS methodologies which are suitable for a real time operational environment?**

   Although many efforts in 2D DAS have been made to increase airspace capacity, 3D DAS methodologies are lacking. In DAS, the 3D sectorization is not a mere extension of 2D into a 3D model. Airspace is a 3D space and the vertical profiles of aircraft movements have to be considered when sectorizing a given airspace. Some initial attempts of 3D DAS exist in the literature, but there are critical gaps and limitations which need to be addressed before they can be used in an operational or even a simulated ATM environment. To identify the gaps of the existing models is an initial and essential step to develop feasible 3D DAS methodologies. An investigation on the state-of-art 3D DAS methodology (agent based model from Metron Aviation) based on experiments is conducted in this thesis.

2. **What are the appropriate airspace sectorization methodologies for a 3D national airspace?**

   DAS is a typical space partitioning optimization problem, which has to satisfy multiple objectives including ATC task load balancing and minimizing the air traffic flow cuts. Meanwhile, it is constrained by the principles of sec-

tor design, such as the convexity and right prism constraints. The hardness of this problem increases along with the increment of space dimensions. A heuristic method may be suitable to 3D DAS optimization. The objective of DAS is to adjust the airspace sector boundaries dynamically and responsively, but DAS has to handle a large amount of air traffic data when performing airspace sectorization tasks. Therefore, efficiency of the DAS approach is key to implement it in a real time ATM system.

3. **What is an appropriate evaluation environment for investigating DAS and other advanced ATM concepts?**
   DAS requires air traffic prediction to perform airspace sectorization and the feedback of this reconfiguration is necessary to improve the DAS methodologies. Therefore, a system, which can provide real time situation awareness of air traffic, traffic prediction, and evaluation results, is necessary. Furthermore, the objective of DAS is to provide automation aides in airspace capacity management to decision makers, therefore, an advisory mechanism is also required. The air traffic situation awareness and prediction, evaluation, and advisory functions have to be integrated together. Therefore, an efficient, accurate, and secure air traffic monitoring and advisory system is essential to implement and investigate the DAS concept in the operational environment.

1.4 Organization of the Thesis

There are eight chapters in this thesis organised as follows:

Chapter 1 presents an introduction of the thesis which includes an overview of the research field followed by the motivation and the research questions raised by the thesis. The scientific contributions of this thesis are also presented in this chapter.

Chapter 2 provides the background of the research conducted for dynamic airspace sectorization. First, different approaches and methodologies for dynamic

airspace sectorization in the literature are surveyed and the need for a real-time air traffic monitoring and advisory system to evaluate the dynamic airspace sectorization approaches is emphasised. The state-of-art multi-objective optimization methodologies are reviewed. Emergent questions raised from the literature survey are then presented.

Chapter 3 experimentally evaluates an agent based model for 3D airspace sectorization to identify existing performance and operational problems. Then, the development of an improved agent based model is presented. A comparison between the two agent based models in terms of performance and efficiency is provided and the advantages and limitations of the improved agent based model are identified.

In Chapter 4, the development and implementation of three new geometric airspace sectorization models (KD-tree, Support Plane Bisection, and Constrained Voronoi Diagram Models) are presented. A detailed comparison among all airspace sectorization models is then presented.

In Chapter 5, the design of the final 3D airspace sectorization optimization model, which combines the multi-objective optimization method (NSGA-II) and the four proposed airspace sectorization models, is provided. Three objectives: minimizing standard deviation of task load across sectors, maximizing average sector flight time, and maximizing distance between sector boundaries and traffic flow crossing points, are presented. An initial experiment to evaluate the models is presented and the results are provided. It is shown that the Constrained Voronoi Diagram model and Support Plane Bisection Model have better performance than the others.

In Chapter 6, the design and implementation of a real-time air traffic monitoring and advisory system are presented. The framework of the distributed system is described, followed by the technologies used in modeling all aspects of the ATM system including airspace configuration, flight aerodynamic, air traffic safety, weather conditions, and environment impact. A set of validation experiments is provided at last.

In Chapter 7, the integration of dynamic airspace sectorization models and

the real-time air traffic monitoring and advisory system is provided. The fourth objective of optimization, minimizing maximum task load of activated sectors, is proposed for a large scale national airspace sectorization. A set of experiments are conducted to validate the dynamic airspace sectorization models with the real-time air traffic monitoring and advisory system. The results show that the proposed airspace sectorization models are able to accommodate different traffic demands with the airspace capacity. At the end of this chapter, a large scale comparison on the proposed 3D DAS models is presented.

Chapter 8 concludes thesis and future work is discussed.

1.5 Original Contribution

A list of the scientific contributions arising from this thesis is given below:

- A state-of-the-art agent based 3D sectorization methodology from the literature is experimentally evaluated and gaps were identified. The gaps include right prism violation, embedded sectors, convexity violation, and high computation cost, and possible fixes to these gaps are worked out. Based on the results of the investigation, I developed an Improved Agent based Model (iABM), which overcomes the limitations of the original agent based model. However, some constraints in terms of sector design, such as convexity, can't be satisfied by iABM because the limitations of the grid based approach.

- Three new geometric computation based 3D airspace sectorization models based on KD-tree, Support Plane Bisection, and Constrained Voronoi Diagram are developed to overcome the shortfalls in iABM. These geometric based models show advantages over agent based models including satisfying convexity constraint and less computational cost.

- Multi-objective optimization approaches are developed and implemented for 3D airspace sectorization. NSGA-II (Deb *et al.*, 2002) is integrated with the

airspace sectorization models to find out the best solutions that trade-off the three objectives: minimizing standard deviation of task load across sectors, maximizing average sector flight time, and maximizing distance between sector boundaries and traffic flow crossing points. The experiments show that the proposed models can achieve the intended outcomes. The advantages and disadvantages of each proposed model are discussed as well.

- An Air Traffic Monitoring and Advisory system is designed and implemented for modelling and evaluating advanced ATM concepts including DAS. This system provides real-time information sharing including a common operating picture of air traffic, complexity of sectors, and environment impact to different stakeholders involved in ATM activities. It provides flight trajectory prediction to DAS models, based on which DAS models can undertake airspace resectorization to optimize the specified objectives. The optimal solutions are then provided to decision makers through the system. This system also provides feedback on the sectorization.

- A critical comparison of two proposed 3D DAS models (Constrained Voronoi and Support Plane Bisection Models) is conducted with one year of simulated air traffic data for the Australian national airspace. The performance analysis of each model is demonstrated through a comparison with the current Australian airspace sectorization. The analysis and comparison on both models performance in terms of DAS objectives and efficiency are undertaken and presented.

Chapter 2

Background

2.1 The Present Day Airspace Configuration in Air Traffic Management System

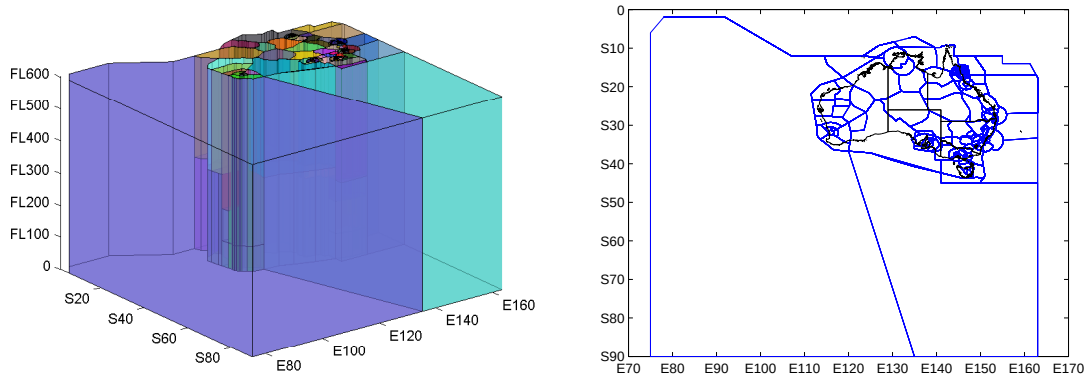
The present day Air Traffic Management (ATM) system is a highly structured and centralised system. Aircraft follow certain rules and fly in static and restrictive airspace. The primary structural elements of the current airspace are different airspace classes, sectors, airways, and waypoints. Airspace is classified into 7 Classes from Class A to Class G [1] (ICAO, 1998) based on the air traffic type, air traffic density, provided air traffic service, and other factors. These 7 airspace classes can be grouped into two types: controlled and uncontrolled airspace. Class A airspace is the most restrictive airspace in controlled classes (from Class A to F), while Class G is an uncontrolled airspace.

Two specific rules, Visual Flight Rules (VFR) and Instrument Flight Rules (IFR) (ICAO, 1999a), are associated with different airspace classes. Both rules are followed by aircraft with different equipment flying in different airspace. VFR are followed by aircraft with little or no instrumentation. The rules rely on good visibility, so pilots can see what is flying around them. Therefore, this type of flying

[1] Five of the ICAO airspace classes are used in Australia: A, C, D, E and G. Class F is not implemented in Australia.

is very weather dependent. IFR are used by jet aircraft, and most large commercial flights. Instrument flying uses sophisticated navigation equipment which allows the aircraft to fly in virtually all weather. IFR is only suitable for aircraft equipped with instrument panels for navigation. Commercial flights often fly through different airspace classes, where IFR applies and radar is activated. However, some aircraft, such as recreation aircraft, usually fly in uncontrolled airspace, where VFR is used.

Flights in controlled airspace are managed by ground based Air Traffic Controller (ATC). An ATC is responsible for managing air traffic in a partition of airspace, called a "sector". However, the human cognitive limitations of an ATC restrict the number of aircraft that can be safely managed by an ATC. The typical tasks conducted by an ATC include safe and efficient flow of air traffic, monitoring aircraft movements, conflict detection and resolution, and aircraft handover from/to his sector. Hence, the airspace is partitioned into sub-airspace as sectors where air traffic is distributed to ensure that air traffic controllers can safely and efficiently manage the air traffic without being overloaded. Figure 2.1 shows the current sectors in the Australian Flight Information Region (FIR). On the other hand, training



(a) 3D View of the Current Sectors in Australian FIR

(b) Lateral View of the Current Sectors in Australian FIR

Figure 2.1: The Present Day Sector Configurations in Australian Flight Information Region (FIR)

an ATC to be familiar with all sector characteristics, including geographic characteristics, airways, navigation waypoints, traffic flows, and procedural rules, is a long

term task. Therefore, sector boundaries in the present ATM system are fixed and remain static.

However, the structured and static airspace configuration can't satisfy the fluctuating air traffic demands, extreme weather conditions, and so on, which cause potential safety issues because of ATC workload issues. Therefore, the airspace configuration has to be flexible somehow. In the present day, the airspace configuration, such as sector boundaries, can be adjusted under certain conditions. Two types of airspace changes exist in the current ATM system: strategic and tactical airspace changes. Strategic changes to the airspace are based on operational experiences of airspace congestions, changes in the air traffic flows, and workload. Tactical changes to the airspaces are based on urgent requests. The day-to-day or hour-to-hour airspace changes are limited to combining and de-combining sectors within the same control centre. Currently, airspace changes are limited by the current configurations and most of them focus on strategic airspace configuration. Airspace change constitutes determination of airspace classification, determination of controlled airspace, designation of restricted airspace, the type of operation within a volume of airspace and/or the conditions of its use, the provision of air traffic services within any volume of airspace, temporary situation requirements to airspace architecture, and the authority managing a given volume airspace (CASA, 2008).

As mentioned above, these procedures of airspace changes are complicated tasks and they require a large amount of resources and time. A few automated tools exist to assist the strategic airspace change process (for example, MITRE (Conker *et al.*, 2007) developed a set of tools to conduct strategic large-scale airspace redesign for current operations), but automated tools are lacking for tactical based airspace sectorization. Therefore, it's difficult to change the airspace according to the air traffic demands for airspace capacity, weather conditions, or others, on a day-to-day basis.

2.1.1 A Need for Improvement

Air traffic continues to grow in capacity and in the regions it serves. AirServices Australia predicted in 2007 that air traffic in Australia would double by 2020 (Shepherd *et al.*, 2007). Predicted Air traffic growth in Europe and the US will exceed the current ATM capability in the next 20 years (EUROCONTROL, 2008b; FAA, 2009). FAA predicts that the number of passengers on U.S. airlines, domestically and internationally, will increase from 757.4 million in 2008 to 1.1 billion in 2025 (FAA, 2009). EUROCONTROL also concludes that there will be between 16.5 and 22.1 million instrument flight rule movements in Europe by 2030, which is between 1.7 and 2.2 times more than in 2007 (EUROCONTROL, 2008b).

The projected growth in air traffic brings challenges for the present day ATM system. These challenges include airport capacity, congested airways network, safety, human workload, and environmental impacts (EUROCONTROL, 2008a; Mohleji and Ostwald, 2003; Penner *et al.*, 1999). To balance airspace capacity and the growing air traffic demands is a key to meet these challenges. The structured and fixed airspace configuration and the complexity of the airspace change processes can't safely and efficiently satisfy the growing demands of air traffic. This opens up new research questions in terms of airspace capacity management as well as air traffic safety and efficiency.

In the last decades, researchers have undertaken several improvement efforts for airspace capacity management in order to satisfy growing air traffic demands. In the next section, I review some of these efforts briefly.

2.1.2 Improvement Efforts in Airspace Capacity Management

To increase the airspace capacity, various research efforts have been made including automated tools in the ATM system, air traffic flow management, and airspace reconfigurations.

Automated tools for ATM, such as the Center-TRACON Automation System (CTAS) by NASA Ames research center (Erzberger *et al.*, 1993) and the User Request Evaluation Tool (URET) developed by MITRE Corp (MITRE, 2011), aim to reduce the ground based ATC workload by introducing automated aids in order to increase airspace capacity. CTAS is a tool to plan sequencing and separation of arriving traffic in order to increase airport capacity. URET can provide real time situation awareness including automatic trajectory prediction, conflict detection and resolution to ATC, which provides the necessary aids to ATC and improves airspace capacity. Air Traffic Flow Management (ATFM) method (Masalonis *et al.*, 2003; Weigang *et al.*, 2010a), Multi-Sector Planning (MSP) (Flener *et al.*, 2007), and flight path planning (Sherali and Hill, 2009), have been proposed and developed for the growing air traffic demands. They aim to resolve airspace congestion and to balance ATC workloads by changing aircraft take-off time, remaining approach time, and enroute altitudes. However, these improvement efforts limit the future ATM system to adopt advanced concepts, such as User Preferred Trajectory (UPT) (EUROCONTROL, 2007), because the static and structured airspace is not allowed to change. Therefore, airspace reconfiguration have been investigated by researchers as well.

The advanced Airspace Scheme (AAS) Concept (EUROCONTROL, 2004a) developed by EUROCONTROL is to employ a significant change in sector design process. It is a network orientated approach to develop an airspace that is characterised with a greater flexibility and responsiveness that will permit the aircraft operates to plan and fly their preferred route. The National Severe Weather Playbook (NSWP) (Klein *et al.*, 2007) by FAA ATC System Command Center (ATCSCC) is a set of airspace configuration scenarios according to the weather conditions, which is dynamically applied in different scenarios for airspace capacity increment.

Apart from improvement efforts in the industries, researchers have made many efforts to increase airspace capacity. For example, Gianazza and Alliot (Gianazza and Alliot, 2002) proposed an approach based on a tree search method to optimize sector configurations with the capability to decide on the number of available control positions automatically. This approach regroups the sectors rather than reshapes

the sectors. However, all these efforts were undertaken under the current airspace configuration, there is no guarantee that the requirements arising from future air traffic demands can be met.

In the U.S., the Next Generation of Air Transportation System (NextGen) (Joint Planning and Development Office, 2007) is proposed by the Joint Planning and Development Office (JPDO) which aims to significantly increase capacity, safety, efficiency, and security of air traffic operations. In Europe, the Single European Sky ATM Research (SESAR) (SESAR Consortium, 2007) aims to deliver a high performance air traffic control infrastructure that will enable a safe and environmentally friendly development of air transport. Within the framework of these two future ATM systems, advanced ATM concepts, such as Dynamic Airspace Configuration (DAC) (Kopardekar *et al.*, 2007), Free Flight (Hoekstra *et al.*, 2002), User Preferred Trajectory (UPT) (EUROCONTROL, 2007), In-trail Procedure (ITP) (Careño and Muñoz, 2007), Continuous Descent Approach (CDA) (Clarke *et al.*, 2004), and aviation emission management (Baughcum *et al.*, 1996; Sandrine and Frank, 2006), have been proposed to meet these challenges for sustainable air traffic growth. Whereas, the current fixed airspace configuration consisting of sectors, waypoints, and airways is not suitable for these future ATM concepts. Furthermore, an essential element to meet the increasing demand in air traffic is to use more efficient allocation of airspace for capacity demand balancing (Ball *et al.*, 2001). This can only be achieved by managing the aircraft trajectories while the structure of the airspace as well as the Air Traffic Controller (ATC) resources are constantly adjusted to meet airspace user needs (P.U. *et al.*, 2008). Therefore, DAC concepts including Dynamic Airspace Sectorization (DAS) have been addressed to support the future ATM systems.

DAC is a new operational paradigm that proposes to migrate from the current structured, static, homogenous airspace to a dynamic, heterogeneous airspace capable of adapting to user demand while meeting changing constraints of weather, traffic congestion and complexity, as well as a highly diverse aircraft fleet (Kopardekar *et al.*, 2007). DAC in current operations has limited options in terms of how sectors and how an airspace can be reconfigured due to various technological and hu-

man factors issues (Mitchell *et al.*, 2008). Understanding the limitations of the state-of-the-art methodologies in DAC will be the necessary initial steps to designing effective airspace reconfiguration methodlogies and operational concepts in the DAC research focus area. DAC envisions the future sectors to be substantially more dynamic, changing dynamically with the changes in traffic, weather, and resource demands (Zelinski, 2009). In DAC research, Dynamic Airspace Sectorization (DAS) is an initial approach for restructuring the airspace in order to archive capacity-demand balance as well as managing the workload for air traffic controllers while ensuring an orderly flow of traffic.

In DAS, an airspace sector is dynamically reconfigured based on dynamic density measures (Kopardekar and Magyarits, 2003) (e.g. traffic flow, aircraft density, sector boundary proximity, time in sector, heading variability and speed variability), ATC workload, sector geometry constraints (e.g. convex shape, right prism constraints) and others (FAA Order 7210.3U, 2006; FAA Order 7400.2F, 2006; FAA Advisory Circular 90-99, 2003). In the next section, I review the state-of-art in the field of DAS briefly.

2.2 Dynamic Airspace Sectorization

DAS optimizes the number of sectors and adapts sector shapes vertically and horizontally according to changing traffic demand, to optimize efficiency of the individual controllers as well as the controller teams. DAS may be also triggered by other factors: complexity, workload prediction, and air traffic forecast for the day; weather changes; and changes of ATM Policies on aviation safety (Yousefi *et al.*, 2009). By dynamically sectoring airspace sectors, ATC workload can be balanced or minimized while maintaining traffic flow. Hence, limitations on sectors can be removed and then airspace capacity can be adjusted to accommodate fluctuating air traffic demands.

Zelinski (2009) investigated three different DAS algorithms with real traffic data and suggested that DAS can improve the efficiency of air traffic systems and can

balance workload over current operations in practice. Furthermore, future concepts, such as UPT and 4D trajectory operations, are compatible with the operational environment where DAS is enabled. A human factor experiment regarding DAS was conducted by Hadley and Sollenberger (Hadley and Sollenberger, 2001), which suggested that ATCs are willing to practice in the context of DAS, and DAS is enabled to improve the efficiency of ATC.

By its core nature, DAS has a number of conflicting objectives. In some recent papers Multi-Objective approaches are also suggested for DAS (Brinton and Pledgie, 2008; Xue, 2008). These conflicting objectives stem from the fact that we can't allow a large number of small sectors to be created because of the limited number of available air traffic controllers. Conversely, we can't group sectors so much that the workload of an air traffic controller within a sector increases dramatically. On the other hand, alignment between sectors and air traffic flows is necessary, which decreases the traffic flow cuts and reduces the handover workload of ATC as well.

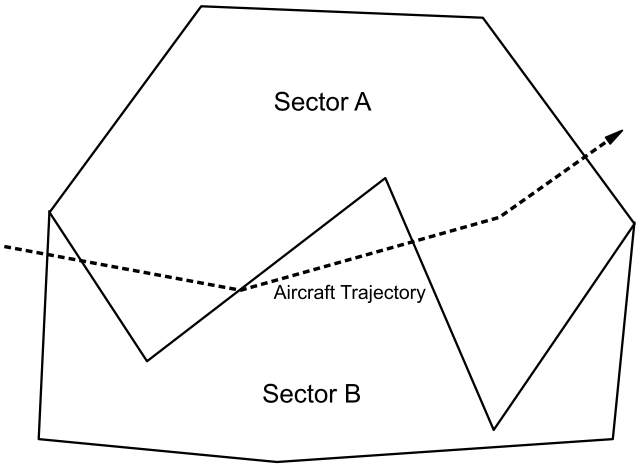


Figure 2.2: Multiple Entrances of an Aircraft to a Sector Violating the Convex Shape Constraint

In addition to the conflicting objectives, several constraints in terms of sector design exist in DAS. Firstly, a sector has to be convex in shape. The convex sector shapes prevent an aircraft entering a sector more than once, otherwise multiple entrances of an aircraft as shown in Figure 2.2 increases the handover workload of an ATC. Although the convexity may not be always necessary for some areas

(e.g. terminal management area) because of the operational, geographic or other restrictions, it is an important constraint for the en-route sector design addressed by many researchers (Delahaye *et al.*, 1998; Trandac and Duong, 2002; Xue, 2008).

Sector boundaries have to be far away from air traffic flow crossing points in order to increase the response time for resolving potential conflicts and to reduce the conflict resolution workload of ATC. Right prism is required for a 3D sectorization because the ATC has only a 2D projection on their screen with which to facilitate their comprehension of the situation. Right Prism is a prism that has two bases, one directly above the other, and that has its lateral faces as rectangles. Figure 2.3 shows an example of right prism and its 2D projection. Finally, minimizing the

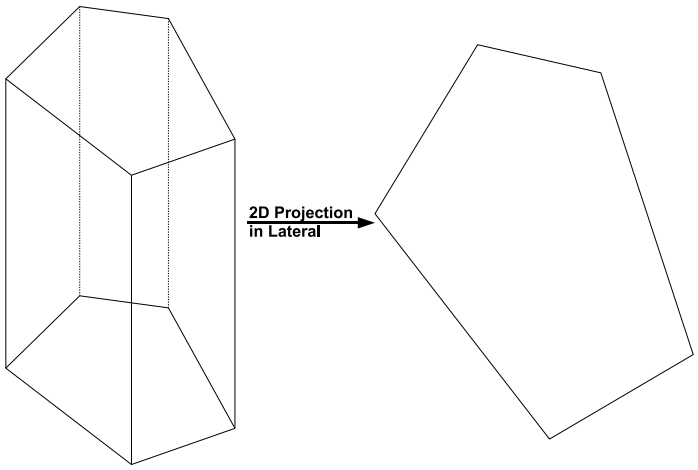


Figure 2.3: An Example of Right Prism and its 2D Projection in Lateral

computation cost of DAS is required in an operational environment because DAS has to be adaptive to the frequent traffic changes within an acceptable time window. However, DAS has to handle large volume of air traffic data, to predict flight trajectories, and to apply necessary optimization methods

Researchers have been working on the DAS problem over the last decades. An overview of the related literature is provided in the next two sections. The terminology "workload" was used in DAS approaches. However, "workload" is not measured by cognitive metrics or other factors directly from ATC but by the air

traffic data, such as aircraft counts and traffic loads. Although they reflect the ATC workload somehow, they are really measurements of "task load" rather than "workload". Therefore, "task load" is used instead of "workload" in this thesis.

2.2.1 2-Dimensional Airspace Sectorization

The 2D airspace sectorization only considers the lateral partitioning in the airspace to achieve task load balancing, maintaining traffic flows, and achieving other objectives. The vertical profiles of sectors are predefined and do not change during sectorization.

2.2.1.1 Voronoi Diagram with Genetic Algorithm

In this method, Voronoi Diagrams are used to partition the airspace and a genetic algorithm is used to optimize the partitions (Delahaye *et al.*, 1995, 1998; Xue, 2008; Songchen and Zhang, 2004). The Voronoi Diagram decomposes a space into subdivisions around given generating points. All points within a region associated with a specific generating point, are closer to that generating point than any other generating point. A Genetic Algorithm (GA) is then used to find the set of Voronoi Diagram generating points that optimize given parameters. Delahaye *et al.* (1995, 1998) addressed a GA approach to regroup the air traffic control sectors for an artificial ATC network in order to balance task loads with a specified sectors number. Xue (2008) presented another Voronoi Diagram based approach to design the airspace sectors for three objectives (Balancing task load, Minimizing Sector Crossing, and Maximizing Sector Flight Time) by GA. The available control positions (number of sectors) are manually decided at the beginning of these approaches.

2.2.1.2 Flight trajectory clustering

Another common approach is to cluster the flights based on their trajectories. Air traffic flow is composed of aircraft 4D trajectories and each trajectory is

a temporal-spatial data set. Flight trajectory clustering groups flight track positions together to sectorize airspace. Sector boundaries are then formed around the groupings of flight route segments.

Brinton and Pledgie (2008) proposed a constrained clustering algorithm for flight tracks to support airspace allocation to accommodate traffic demand dynamically. Three clustering criteria are used in the algorithm: the Euclidean distance between the current aircraft positions and the current cluster centre; the Euclidean distance between future aircraft positions and future cluster centre; and "corridor" which is a combination of the perpendicular distance from the flight position to the major axis of the group of flight positions and the difference between the flight's heading and the direction of the major axis. There is one constraint (minimum and maximum aircraft count) for each cluster in order to satisfy the work balance objective. They applied heuristic approaches, including simple swap, mid-point swap, single move and the combination of these three, to improve the performance of the constrained clustering. Their work also reveals the implied mapping between some but not all factors of Dynamic Density, such as flight resident time in sector and boundary proximity, and the clustering criteria. A weighted clustering objective (a linear combination of three clustering criteria) is adopted in the clustering algorithm. Different weights for the criteria affect the resultant airspace partition. For example, if the weight for "corridor" increases, sector boundaries are more likely to align with major traffic flows but also will decrease the flight resident time in sectors. These methodologies by exploring, clustering, and analyzing these temporal-spatial data are not straightforward for generating sector shapes and rely on additional geometric computation methods to calculate sector geometric configurations according to the clustering results.

2.2.1.3 Constraint Programming

Trandac and Duong (2002) formulated DAS as a constraint-programming problem. They identify the objectives of DAS as balancing as well as minimizing the controller task load, with three constraints: convexity, minimum distance, and min-

imum sector crossing time. The controller task load is measured by three types of task load: monitoring task load (MW), conflict task load (CW), and coordination task load (OW). In their formulation, the airspace is modelled as edges (routes) containing a set of beacons and crossing points. Each edge has two points that are directly connected. These points belong to the same or different sectors. The conflict task loads are assigned to the two vertices, the monitoring task load are divided by two and are evenly distributed for each vertex, and the coordination task load is assigned on the edge. For each sector, a predefined task load range is used to achieve the task load balancing. Three constraints are modelled in the edge and vertex as well. They extended their works by developing the sector boundaries computation by Delauney Triangulation and Voronoi Diagram (Trandac *et al.*, 2003). In both approaches, they demonstrated that their approach is able to solve a DAS problem to achieve the objectives and to satisfy the relevant constraints in a reasonable time.

2.2.1.4 Grid Based Approaches

An airspace can be treated as a discrete space consisting of a set of uniform cells. Then airspace sectorization can be undertaken as a procedure to group cells according to the air traffic data. Hence, grid based approaches are common in DAS research.

A grid based airspace partitioning approach using Equalized Traffic Mass Principle is presented in (Klein, 2005). The Equalized Traffic Mass Principle means that the total traffic counts in each Center should be equal over a selected period. A uniform sized hexagonal grid combined with rectangle grid is used in this approach. The approach starts with seed locations, such as major airports, and traffic flows, and then a seed growth algorithm is applied to partition the airspace laterally to achieve the Equalized Traffic Mass Principle. This approach shows it has the capability to balance the task load among control centers and has the elasticity to accommodate weather impacts. This algorithm highly depends on the initial seed locations, and a more objective and flexible selection method for seed locations is required. In addition, unexpected sector shapes may be generated from this grid

based approach.

The Mixed Integer Programming (MIP) algorithm discretizes the airspace into hexagonal cells, and clusters the cells according to task load and connectivity (Verlhac and Manchon, 2001; Mitchell *et al.*, 2008). The task load of a cell is the number of flight track counts within that cell. Connectivity from a cell to its neighbouring cell is the total number of flights that travel from it to a neighbour. Flow enters a cell from at least one of its neighbours and exits into exactly another one neighbouring cell. The task load of each cell is added to the flow, which is finally absorbed by a sink cell.

Ehrmanntraut and McMillan (2007) proposed a grid based sector design process and tool to balance the ATC task load for an en-route airspace. It is an iterative design process to improve airspace sectorization with the aid of simulations. It is able to balance the task load of ATC but the process is complex and time consuming. Another grid based approach proposed by Tien and Hoffman addresses the efficient utilization of controllers resource for a long term solution of DAS in the US operation context (Tien and Hoffman, 2009). It handles the variance of traffic demands.

2.2.1.5 Geometric Algorithm

Basu *et al.* (2009) presented a geometric algorithm for airspace sectorization. It is a 2D partition algorithm for enroute airspace based on historical flight trajectory data. It addresses three different aspects of DAS: minimizing the maximum task load with fixed number of sectors; minimizing the number of sectors with fixed maximum task load; minimizing the maximum task load, average task load, and number of sectors. Therefore, it has the flexibility to generate airspace sectorization according to different perspectives of users.

2.2.1.6 Approaches Driven by ATC Resources

The above approaches are proved to improve airspace capacity in theory, but they share two shortfalls. First, they don't explicitly minimize the number of sectors

which required certain amount of ATC resources. Second, many of these approaches require resectorization to be implemented frequently in short term, which is not practical in current ATM where ATC needs to be trained to be familiar with the reshaped sector boundaries. To address these issues, Bloem and Kopardekar (2008) presented a solution by combining two neighbour sectors to efficiently use the ATC resources. The sector shapes do not change and it is not necessary to train ATC to be familiar with the sectors.

2.2.2 Summary of 2D DAS Research

Table 2.1 summaries the 2D DAS approaches listed above.

These 2D DAS approaches are able to improve the airspace capacity in theory. However, one key element, the vertical configuration of sectors, is missing in these approaches. Moreover, these approaches can be applied in one or two control centres, and they may not scale up for national airspace sectorization.

2.2.3 3-Dimensional Airspace Sectorization

Altitude is not considered in 2D sectorization. However, it is a very important consideration in sectorization. Air traffic flows moving in opposite directions are always altitude separated. Also, depending on an aircraft's size, it may or may not have the ability to attain certain altitudes. Moreover, flight climb-out (during take off) and descent (during landing) make altitude consideration critical for sector designs, especially in the transition airspace area. Hence there is a recent focus on the third dimension — altitude — while looking into dynamic sectorization.

3D airspace sectorization is more complex than 2D airspace sectorization. 3D sectorization has specific geometry constraints, such as right prism constraint and 3D tessellation. A right prism requires a polytope to have bases aligned one directly above the other and has lateral faces that are rectangles. This is important because the ATC needs a 2-D projection of the sector on their screen, which is not possible

Table 2.1: Summary on DAS Approaches in Literature

Approaches	Objectives	Constraints	Methodologies	Multi-Control Centre
Delahaye *et al.* (1995) Delahaye *et al.* (1998)	task load balance		GA	One control centre
Xue (2008)	Balancing task load Minimising Sector Crossing Maximizing Sector Flight Time		GA Voronoi Diagram	One control centre
Brinton and Pledgie (2008)	Satisfy dynamic traffic demands	Minimum and Maximum Aircraft Count	Constraint clustering on flight tracks	One Control Centre
Trandac and Duong (2002) Trandac *et al.* (2003)	Task load Balance Minimizing task load	Convexity shape Minimum distance to Sector Boundaries Minimum sector crossing time	constraint-programming Delauney Triangulation and Voronoi Diagram	One control centre
Klein (2005)	Equalized Traffic Mass		Hexagonal grid based	Multi control centres
Mitchell *et al.* (2008)	Task load balance Traffic connectivity		Hexagonal grid based MIP	Multi control centres
Ehrmanntraut and McMillan (2007)	Task load Balance		Grid based Simulation aids	One control cener
Tien and Hoffman (2009)	Handling the variances of traffic demands		Grid bsed	One Control Centre
	ATC resources		MIP	
Basu *et al.* (2009)	Minimizing maximum task load Minimizing sectors number Minimizing average task load		Geometric Algorithm	One control centre
Bloem and Kopardekar (2008)	Efficient usage of ATC resource		Combine neighbouring sectors	One control centre

if the right prism constraint is violated. Sector shapes are polyhedrons in 3D space and gaps between sectors are not allowed for 3D airspace sectorization. Therefore, an appropriate 3D tessellation guaranteeing no gaps between sectors is necessary for a 3D airspace sectorization method.

Kicinger and Yousefi (2009) extended the 2D airspace sectorization method of (Mitchell *et al.*, 2008) to 3D partitioning using a heuristic method. This approach is presented as a heuristic airspace partitioning method, combining a genetic algorithm (GA) and an agent-based model. GA is used to determine the initial locations of agents, and the agent-based model is used to determine cell clustering. The airspace

is tiled with uniform grid cells in three dimensions. The sector boundaries are determined by grouping cells according to two objectives:

- Cumulative Traffic Load presents the ATC task load in a given time interval. This is calculated by accumulated traffic load (number of radar hits in each cell) in a given period, from historical or simulated data. In order to balance the task load among the sectors, the mean value of the traffic load is produced and an acceptable task load range is also given, based on the mean traffic load.
- Cumulative Commonality measures the alignment between the redesigned sectors and the main traffic flows. The commonality in each cell is quantified by the number of aircraft traveling from this cell to its neighbouring cells in a given time interval. Cumulative Commonality of a sector is the sum of the values for each cell within the sector.

Their approach uses an agent-based model (ABM) for clustering cells. Each agent represents a sector, which moves in the airspace to find and group local optimal cells and layers into its sector. The local optimal means the cells or layers are selected based on the maximum increase in a sector's Cumulative Commonality when they are added to the sector. ABM is an iterative procedure, which keeps running until all cells in the airspace are assigned to sectors or a predefined number of iterations is reached.

This approach partitions the airspace based on 3D grid cells grouping. It can't guarantee to produce sector shapes that satisfy right prism requirements. Therefore, the right prism is modelled as a constraint.

Four rules work together to achieve task load balance. The Movement Rule uses a moving-ratio (from 0.0 to 1.0), which determines the fraction of agents moving at each iteration. The Layer Growth Rule limits the task load growth when new cells or layers are added to the agent: the cumulative task load is checked to ensure that the task load doesn't exceed an upper bound (1.2 times the average task load). The Trading Rule is executed when an agent's cumulative task load exceeds the upper bound on task load variation: by the Trading rule, an agent transfers the external

boundary cells to its neighbour agent in order to reduce its own task load. The right prism constraint is also checked when grouping new cells and layers; the Repair Rule is called when cells violate the right prism constraint.

The model can partition 3D airspace according to traffic commonality and task load accumulation, and they intend to use genetic algorithms to optimize the sectorization. However, there is no optimization of airspace sectorization presented in their paper, which means that this approach cannot provide optimal DAS solutions.

2.3 Evolutionary Computation in DAS

The major objective of DAS is to achieve the ATC task load balancing among sectors as well as to reduce ATC task load by sectoring the airspace. It is a typical optimal space partitioning problem in two or more dimensions, which is proved as an NP-hard problem (Garey and Johnson, 1979; Khanna *et al.*, 1998). Thus, heuristic methods, such as Evolutionary Computation, have been used to find the optimal solutions of DAS as described in the last sections.

2.3.1 Evolutionary Computation and Genetic Algorithm

Evolutionary Computation (EC) is a subfield of artificial intelligence in computer science. Inspired by natural genetic variation and natural selection, evolutionary computation uses iterative progress to evolve a population of candidate solutions to a given problem by selection and a guided search using parallel processing.

In the 1950s and the 1960s, studies on simulations of evolution using evolutionary algorithms, artificial life, and artificial selection were conducted by researchers (Fraser, 1958; Barricelli, 1962). After Ingo Rechenberg introduced evolutionary strategies for optimising real-valued parameter engineering problems in the 1960s (Rechenberg, 1965) and early 1970s (Rechenberg, 1973), artificial evolution became a widely recognised optimization method. Genetic Algorithms (GAs) is one subfield of EC, which generates solutions to optimization problems using tech-

niques inspired by natural evolution, such as inheritance, mutation, selection, and crossover. In the 1960s, John Holland invented GAs and he and his students kept developing GAs in the 1960s and the 1970s. GAs have become popular in academic and real applications (Holland, 1975; Goldberg, 1989).

2.3.2 Multi-Objective Optimization Methods for DAS

As mentioned above, DAS is an optimization problem with multiple conflicting objectives and constraints. GA has been applied in some approaches and successfully achieved the expected results as listed in Section 2.2.1. However, these approaches model the multi-objective DAS problem as a single objective problem by a weighted sum of the objectives, and then apply a single objective GA to optimize it. The bias of weights on objectives affects the final results generated from these approaches. Therefore, a sophisticated optimization technology is needed for the multi-objective DAS problem.

DAS is a space partitioning optimization problem and can be treated as a multi-objective clustering problem, because DAS clusters flight trajectories into small partitions in a given airspace to satisfy specified objectives. A flight trajectory is a spatial-temporal data set. Several algorithms, such as TF-OPTICS (Nanni and Pedreschi, 2006) and ST-DBSCAN (Birant and Kut, 2007), have been developed for the spatial-temporal data clustering, but they are implemented for their own purposes and are not suitable for the DAS problem. Other distance based clustering algorithms, such as k-means clustering (MacQueen, 1967), are also not suitable for the DAS problem because these clustering algorithms can't generate the airspace sector boundaries straightforwardly by their results.

DAS is an optimization problem which has a huge number of possibilities for solutions to partition a given airspace. DAS has also to be able to handle the multi-objective nature of the problem. These two characteristics suggest that GA is an appropriate optimization method for this kind of problems (Mitchell, 1996). A problem with large number of possible solutions can benefit from an effective use

of parallelism. GA explores many different solutions simultaneously, which would be much faster than evaluating only one. The selection, crossover and mutation operators of GA ensure that the best solutions can survive and reproduce possibly better children in the next generation which adapt these solutions for the changes in the environment.

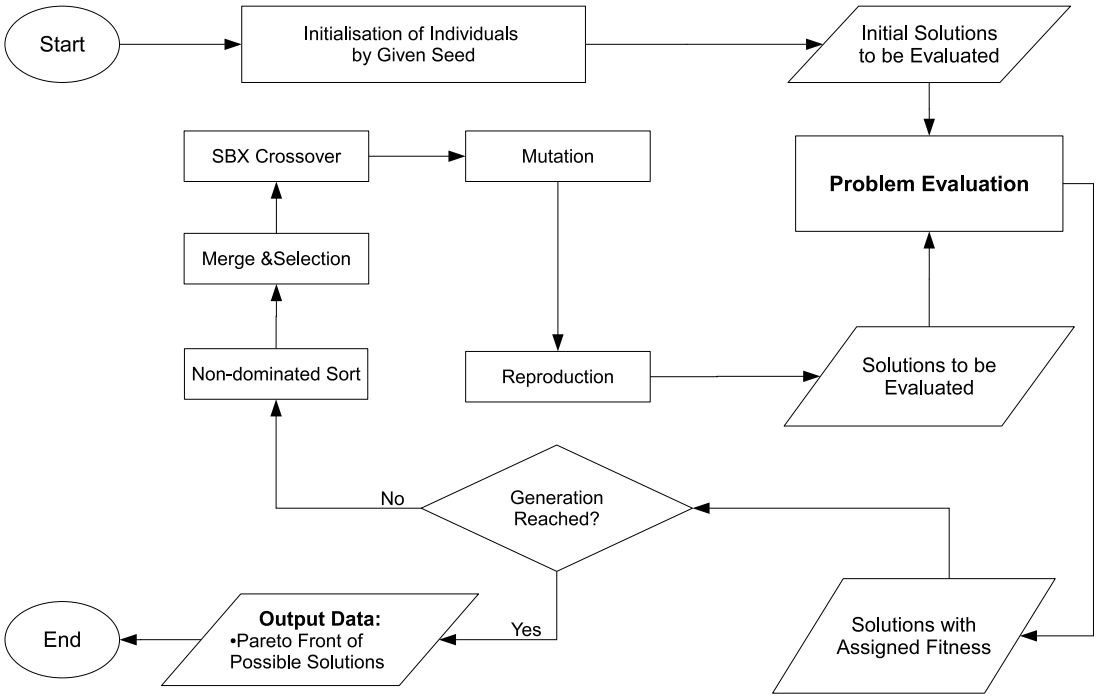


Figure 2.4: The Framework of NSGA-II

DAS is also a multi-objective optimization problem, where conflicting objectives are existing. Therefore, a suitable multi-objective optimization method is required. NSGA-II (Deb *et al.*, 2002) is an efficient multi-objective evolutionary algorithm based on the Pareto optimal concept, which has been used in many multi-objective optimization problems. NSGA-II uses elitism to preserve the best non-dominated set found so far by evolution. It starts with setting some parameters, including the archive size and population size, which are normally equal to each other. The archive is filled from the current population and with the existing solutions in the archive, if any. NSGA-II uses dominance ranking to organize a population into

a number of layers, with the first layer being the best. The archive is filled out starting from the best layer and downward. If the archive becomes full, the process terminates. Crowding distance is used to maintain diversity without specifying any additional parameters. Genetic operators are then applied including the Simulated Binary Crossover (SBX) (Deb *et al.*, 2007) and mutation operators.

As illustrated in Figure 2.4, NSGA-II has a flexible framework to import different optimization problems easily.

2.4 Air Traffic Monitoring and Advisory Systems

DAS approaches have to be integrated into ATM systems to provide advice to decision makers for airspace capacity management. An air traffic monitoring and advisory system that can monitor the air traffic and provide information is essential to integrate DAS into the ATM system. Meanwhile, future concepts, such as DAS, should not only be evaluated in a simulation environment, but also should be analyzed in a real environment.

ATM involves multiple stakeholders, including air traffic service providers, air traffic controllers, pilots, air traffic flow management centres, airlines, and airport authorities. Research into future ATM concepts needs to cover all aspects of ATM (Prevot, 2002). A distributed information system, that can provide timely, accurate, and reliable airspace information and advice to each participant, is needed (NCOIC, 2008). This is complicated, however, by the fact that different participants have different goals; some interactions among participants may be cooperative, to achieve common goals, but others may be competitive, for access to scarce resources. The system must be able to support the analysis of both cooperative and competitive behaviors.

Moreover, such a system must be able to process large volumes of data (e.g. air traffic data and airspace status) in real time, involving complex computations (e.g. computation of flight aerodynamics, aviation emission, and airspace complexity).

This is required in order to present relevant results (e.g. flight trajectories, airspace safety measurements, aviation emission distribution, etc.) to particular clients for tactical decision making timely and reliably, and to evaluate advanced ATM concepts (e.g. DAS, UPT, and emission management) safely and accurately in the real environment. To meet these requirements, advanced computation devices, high throughput communication network, a systematic integration methodology, and a distributed system architecture are essential.

The system has to be sufficiently fast, detailed and flexible to be used in an operational environment. It should enable real time construction of flight trajectories as radar data is received, and the estimation of flight aerodynamics based on the constructed trajectories.

Two factors have hindered the development of such ATM advisory systems. One is the lack of standards for information sharing and data communication; this is improving, but still an issue for the present. Information sharing and data communication are the key for this kind of advanced ATM system, as indicated in the SESAR (SESAR Consortium, 2007) and NextGen (Joint Planning and Development Office, 2007) projects. However, ATM is a complex system which involves multiple participants, each with their own aims and their own legacy systems. There are no existing standards for information sharing or data communication. Therefore, System Wide Information Management (SWIM) has been introduced in SESAR and En Route Automation Modernization (ERAM) system has been proposed in NextGen. As both SESAR and NextGen are under development, and neither will be implemented in the near future, the gap in information sharing and data communication between past and future ATM systems remains for the present.

The second factor is technology: when current ATM systems were developed over recent decades, computing and network facilities generally lacked the capabilities to efficiently handle in real time the large volumes of data involved today. As methodologies in developing and implementing distributed systems have matured (Coulouris *et al.*, 2011), some advanced ATM systems have been developed.

Some are simulation tools, that can be used for evaluating future concepts but cannot be used in real ATM operation environments to advise ATM participants for decisions. Prevot and Mercer presented an advanced simulation system that presents multiple perspectives and interactions in ATM, such as pilots, air traffic controllers, and so on (Prevot and Mercer, 2007). Donovan proposed a national data surveillance and data communication infrastructure for aircraft (Donovan, 2006), taking advantage of present computer and network technology. Both of these systems focus on the air traffic, without consideration of environmental impacts. Air Traffic Operations and Management Simulator (ATOMS) (Alam *et al.*, 2008) was developed to model multiple aspects of ATM, such as flight aerodynamics, airspace configurations, aviation emission, Cockpit Display of Traffic Information (CDTI), data communications, and weather, in order to evaluate advanced ATM concepts such as free flight, severe weather avoidance, and confliction detection and resolution. However, ATOMS is not designed to handle real-time field traffic.

Others focus on individual elements within ATM, but do not cover the whole range of airspace safety, complexity, emissions, and situation awareness for air traffic controllers. Weigang et al. developed an intelligent air traffic flow management system (Weigang *et al.*, 2010b) to control air traffic flow, taking into account airport congestion, conflict resolution, etc. Their system includes multi-agent communication among airports. OPAL DSS (Optimization Platform for Airports including Landside Decision Support System) (Zografos and Madas, 2006) was developed as an integrated decision support system for airport operations, including the consideration of airport capacity, safety, and environment (noise). AIRWOLF (Automatic identification of risky weather objects in line of flight) (Ahlstrom and Jaggard, 2010) was implemented to automatically identify severe weather objects for en-route flights. The weather polygons are visualized on an en-reroute controller workstation.

Network Centric Operations Industry Consortium (NCOIC) (NCOIC, 2008) suggests an information system for multiple ATM participants, which addresses the integration of capacity, safety, efficiency, security, agility, and attainability. This project is still under development.

There is a clear need for a real time ATM system that can facilitate air traffic management decisions in real time, and can be used to evaluate future ATM concepts, considering environmental impacts as well as airspace safety and capacity.

2.5 The Emergent Questions

From the literature survey, it is evident that most exiting DAS approaches focus on 2D airspace sectorization rather than 3D airspace. However, the vertical configuration is important for DAS approaches. A few research efforts have been made for 3D DAS but they have some limitations and are unable to generate feasible sector configurations. There is a need to develop a 3D DAS model for airspace capacity management in the future ATM system.

DAS is a multi-objective optimization problem with several constraints making up an NP-hard problem. It can't be solved by any formulas simply. To solve this problem, I divide the DAS problem into two sub-problems: The first is a 3D airspace sectorization model, which partition airspace into sectors that satisfy sector design constraints, for example convexity of sector shapes and right prism; the second is an optimization method to optimize the airspace sectorization model for several objectives, including task load balancing, flight sector time, and minimum distance between sector boundaries and traffic flow crossing points.

A good understanding of the existing 3D DAS method is an initial step towards developing a successful 3D DAS approach. In fact, a mature 3D airspace sectorization model is missing in the literature because of the difficulties of it. In Chapter 3, an agent based model (ABM) (Kicinger and Yousefi, 2009) is experimentally investigated, and several limitations of this model are identified and their causes are analysed. These challenges are addressed in Chapter 3 and 4, where four 3D airspace sectorization models are developed. On the other hand, an efficient and effective method for multi-objective DAS optimization is required to work with airspace sectorization models together in order to achieve the expected airspace sectorization results. Chapter 5 presents a multi-objective DAS problem modelling and

an optimization method based on NSGA-II to solve this problem.

DAS methods and other advanced ATM concepts need an evaluation tool, where they can be investigated with not only the simulated air traffic but also the real air traffic data before they are implemented in practice. No existing evaluation tools address the investigation in both simulation and real ATM environment at the same time. It is a challenge to develop a tool that can investigate ATM concepts with either simulated or real data. Meanwhile, DAS and other concepts are supposed to provide automated aids to decision makers for managing air traffic safely and efficiently. Therefore, a system integrated with the advanced ATM concepts and has an advisory mechanism to provide necessary and responsive advices according to different situations (traffic demands changing, weather conditions, etc.) in real time ATM environment is also required. In summary, a system modelling all aspects of ATM, sharing the information to different stakeholders, integrating advanced ATM concepts, providing advice to decision makers is required to evaluate and use the proposed DAS methods. Chapter 6 addresses this challenge by the design and implementation of a real time air traffic monitoring and advisory system.

Another question arises when this system is built: What is an appropriate 3D DAS model that can be used in a real ATM environment? Here, an appropriate 3D DAS model means that the model is efficient to handle a large amount of air traffic data (one day, one month or one year) and to find the optimal DAS solutions for decision makers, and is compatible with other future ATM concepts. To investigate it, a number of scenarios representing the current and the future traffic demands are needed at first. Secondly, a complex target airspace modelling which contains multiple control centres is required. Thirdly, some prototypes of future concepts, such as Free Flight and UPT, have to be presented in the system. Finally, a benchmark of the proposed objectives have to be established before evaluating the performance of the DAS models. All these challenges are addressed in Chapter 7, where validation and evaluation experiments are conducted with a set of scenarios representing different traffic demands in the proposed air traffic monitoring and advisory system.

Chapter 3

An Improved Agent Based Model for Airspace Sectorization

In this chapter, the agent based model (ABM) of Kicinger and Yousefi (2009) for 3D airspace sectorization is evaluated and the existing gaps of this model are identified. Based on these gaps, an improved agent model is developed and implemented.

First I must define the DAS problem, and describe the sample of simulated traffic data that is used to evaluate ABM.

3.1 Problem Definitions of DAS and Air Traffic Data

There are two objectives of this chapter. The first is to evaluate the existing ABM and identify its operational shortcomings, and then to develop an improved model to address the identified gaps. The second objective is to evaluate and to compare both models in terms of the constraints and criteria of airspace sectorization design.

3.1.1 Problem Definitions

For DAS, the inputs are a given airspace and a set of flight trajectories, and the output is a set of partitions (sectors) that satisfy specified objectives, such as balancing ATC task load.

In this thesis, two kinds of approaches are evaluated: one works on a discrete airspace (agent-based models) and the other works on a continuous airspace. Therefore, two different definitions for the airspace are used here.

The discrete airspace is denoted as $R(C(X')_{uvz})_{u=1,v=1,z=1}^{U,V,Z}$, where $C(X')_{uvz}$ are the uniform cells within R and $X' = \{x_1, x_2, x_3, ...x_c\}$ is all points inside cell C_{uvz}. $x_j = \{lat_j, lon_j, alt_j\}$ is a 3-tuple defined by latitude, longitude, and altitude. u, v , and z are the indices of a cell based on its relative location in R. They are calculated by Equation 3.1:

$$\begin{aligned} u &= \lfloor \tfrac{lat - minLat}{latSize} \rfloor + 1 \\ v &= \lfloor \tfrac{lon - minLon}{lonSize} \rfloor + 1 \\ z &= \lfloor \tfrac{alt - minAlt}{altSize} \rfloor + 1 \end{aligned} \tag{3.1}$$

where lat, lon, and alt represent the position of a point which has the minimum values of latitude, longitude, and altitude within the cell(C). The $latSize$, $lonSize$, and $altSize$ are uniformly predefined for the airspace cell dimension, for example, $latSize = 0.5°$, $latSize = 0.5°$, and $altSize = 1000ft$. $minLat$, $minLon$, and $minAlt$ represent the position of a point with the minimum values within the given R. Therefore, the relationship between cells (C) and R is

$$R = \bigcup_{u=1,v=1,z=1}^{U,V,Z} C_{uvz}$$

where U, V, and Z are the indices of a cell containing the point with the maximum values of latitude, longitude, and altitude inside R.

The continuous airspace is defined as $R(X)$, where $X = \{x_1, x_2, x_3, ...x_n\}$ is all

points included in R. $x_j = \{lat_j, lon_j, alt_j\}$ is a 3-tuple recording latitude, longitude, and altitude, with the same definition as in the discrete airspace.

The air traffic is a set of trajectories: $T = \{t_i\}_{i=1}^{N}$, where

$$t_i = (x_{ij}, spd_{ij}, time_{ij})_{j=1}^{M}$$

t_i is the trajectory of flight i, consisting of a set of ordered points $x_{ij} = \{lat_{ij}, lon_{ij}, alt_{ij}\}$ and the speed (spd_{ij}) and time stamp ($time_{ij}$) associated with each point. $(x_{ij}, spd_{ij}, time_{ij})$ is a traffic hit of flight i in the airspace.

In DAS, only the trajectories located within R are considered. Therefore,

$$\forall i \text{ and } \forall j : x_{ij} \in R$$

The airspace R is partitioned into K sectors $S_k, (k = 1, 2, 3, ..., K)$. K is the number of sectors to be generated. For discrete approaches, sectors are denoted as $S(C')_k$ where C' is a set of continuous cells building up the sector and satisfies

$$\text{IF } C' \in S_k \text{ THEN } C' \notin (R \setminus S_k)$$

The boundary cells (BC_k) of $S(C')_k$ can be found by Equation 3.2:

$$\begin{aligned} BC_k = \forall C_{u,v,z}\ IF\ & (C_{u+1,v,z} \notin C') \bigvee (C_{u-1,v,z} \notin C') \bigvee \\ & (C_{u,v+1,z} \notin C') \bigvee (C_{u,v-1,z} \notin C') \bigvee \\ & (C_{u,v,z+1} \notin C') \bigvee (C_{u,v,z-1} \notin C') \end{aligned} \tag{3.2}$$

$S(\hat{X})_k$ is used for sector notations for the continuous approach, where $\hat{X}$ is a subset of X. Additionally, for both sector definitions,

$$R = \bigcup_{k=1}^{K} S_k$$

and

$$S_i \cap S_j = \emptyset;\ i \neq j;\ i, j \in \{1, 2, 3, ...K\}$$

After the space is partitioned as S_k, all points x_{ij} in T must satisfy

$$\forall i \text{ and } \forall j : \text{IF } x_{ij} \in S_k \text{ THEN } x_{ij} \notin (R \setminus S_k)$$

For DAS, it is necessary to model the ATC task load mathematically. In this thesis, the task load of a sector is measured by the count of traffic hits within the sector during a given period. Equation 3.3 describes the task load measurement I use:

$$W_k = \sum_{i=1, x_{ij} \in S_k}^{N} |t_i(x_{ij})| \tag{3.3}$$

In the equation, W_k is the task load of a sector S_k, which is measured by the count of x_{ij} within S_k from all trajectories t_i. It is similar to the measurement used in the ABM (Kicinger and Yousefi, 2009), where the task load is measured by the cumulative number of air traffic hits in a sector.

Sector flight time is also measured in this thesis. It contributes to the alignment between sectors and traffic flows, which minimizes the traffic flow cut and therefore reduces the handover task load for ATCs (Xue, 2008). It is modelled as Equation 3.4.

$$SFT_k = \frac{\sum_{i=1}^{N} \sum_{j=1, x_{ij} \in S_k, x_{ij+1} \in S_k}^{M} (time_{ij+1} - time_{ij})}{|T(t_i)|_{x_{ij} \in S_k}} \tag{3.4}$$

The sector flight time is derived from the time stamps ($time_{ij}$) of its continuous traffic hits in t_i, which contain positions (x_{ij}) at $time_{ij}$ located within the sector (S_k). $T(t_i)$ is a set of trajectories in the space R as described above. Therefore, the count of all trajectories t_i that have some parts x_{ij} located within a sector S_k is the number of flights flying within S_k. According to these, the sector flight time for a sector can be produced by Equation 3.4.

The minimum distances (D) between traffic crossing points and sector boundaries (Delahaye *et al.*, 1998) is considered in this thesis as well. A small distance between crossing points and sector boundaries gives an ATC little time to resolve a potential conflict, so larger distance is desired. The crossing points of air traffic flows (CP) are identified based on the 4D flight trajectories ($t_i(x_{ij})$), where the two

flight footprints satisfy:

$$\begin{aligned} &Dist(x_{pu}(lat_{pu}, lon_{pu}), x_{qv}(lat_{qv}, lon_{qv})) \leq 5nm \bigvee \\ &|x_{pu}(alt_{pu}) - x_{qv}(alt_{qv})| \leq 1000ft \bigvee \\ &|time_{pu} - time_{qv}| \leq 300sec \end{aligned} \tag{3.5}$$

x_{pu} and x_{qv} are the locations of two flights p and q at $time_{pu}$ and $time_{qv}$. As shown in the Equation 3.5, I take 5nm lateral distance and 1000ft vertical separation within 300 seconds time window as a threshold to identify the crossing points between flights. When an ATC has to solve a conflict, he needs a minimum amount of time to prepare a conflict resolution. Therefore, 300 seconds is the safety time window for ATC response to a potential conflict. The two flight footprints are treated as the crossing points of traffic flows. All the crossing points are pre-calculated based on the given air traffic data as the input to the DAS models.Then the minimum distance between a traffic crossing point and the corresponding sector boundaries for a sector can be calculated as:

$$D_k = \min_{i=1}^{P}(Dist(CP_i, Boundaries(S_k))) \; IF \; CP_i \in S_k \tag{3.6}$$

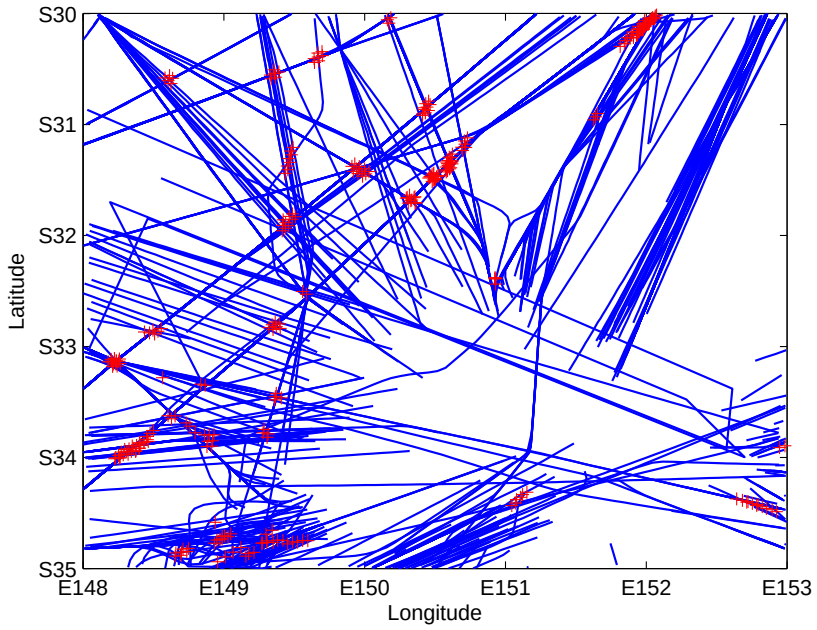
where P is the total number of crossing points identified in the given airspace R and the minimum distance (D_k) is calculated for the sector S_k only when CP_i is inside it. CP_i is a flight footprint.

3.1.2 Air Traffic Data

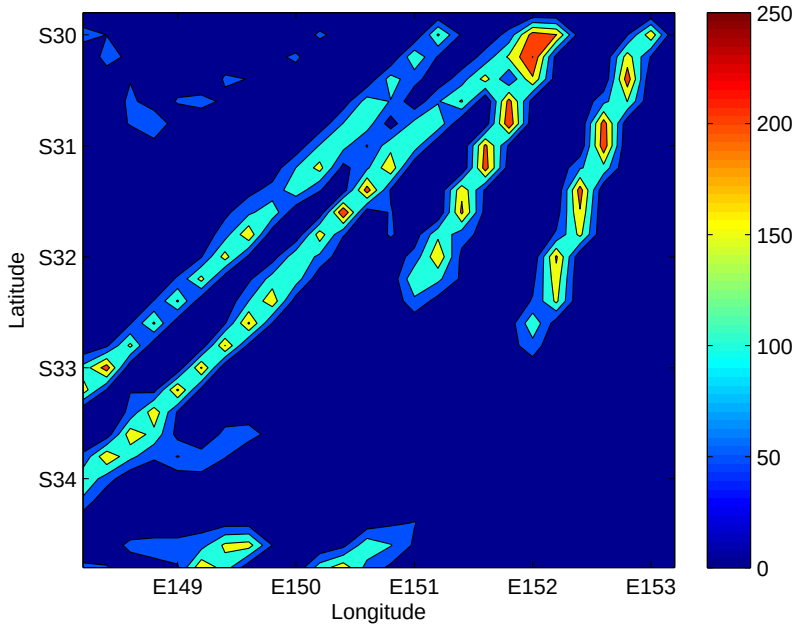
It is important to evaluate DAS approaches and algorithms with a good sample of air traffic data (high traffic feature variance like speed, heading, altitude). This may come from a historical database or a simulator.

The air traffic data used in this chapter is a set of simulated flight trajectories during a fixed time window (24 hours) in part of the Australian airspace. The airspace is a 5 degree by 5 degree en-route airspace (from FL200 to FL500), located

in the east of Australian airspace. The simulated air traffic data are sampled every 30 seconds.



(a) Lateral View of Flight Trajectories and Traffic Flow Crossing Points



(b) Lateral Heat-map of Air Traffic Hits Count

Figure 3.1: Lateral View and Heat-map of Flight Trajectories and Traffic Flow Crossing Points in the Given Area

The simulated flight plans are generated by a high-fidelity air traffic simulator (ATOMS) (Alam *et al.*, 2008). They are based on public domain statistics obtained from one month of real traffic. The average and standard deviation of hourly departures for each airport pair are calculated. From these, the number of flight departures for each airport in a given period (one hour) of a day can be generated, based on a Gaussian Distribution. A Poisson Distribution for flight departure times is also built for each airport, approximated from publically available real data, and is used to generate the departure time for each simulated flight.

A total of 691 flights, including 536 domestic, 71 outgoing, 79 incoming and 5 overflying flights, was generated from the simulator. The sample contains a total of 23619 traffic hits. The sample has 24 different aircraft, including long haul (heavy) and small (light) aircraft. A lateral view and a lateral heat-map of the flight trajectories used for evaluating the model is illustrated in Figure 3.1. Figure 3.1a also illustrates the crossing points of traffic flows as red crosses which are identified by the 4-Dimensional simulated flight trajectories. In spatial distribution, there are 249 flights from south-west to north-east and 235 flights from north-east to south-west. The major traffic flows between the top right and bottom left are highlighted in Figure 3.1b. There are also several local hot spots in the bottom and the top left corner of Figure 3.1b.

The average time of a flight staying within the area is 1045.96 seconds and the average travel distance is 129.66nm. The distribution of the flight staying time and travelling distance are plotted in the first row of Figure 3.2. The second row of the figure shows the distributions of flight directions and their maximum heading changes within 30 seconds. Most of them travel from north-east to south-west or vice versa which corresponds to the highlighted traffic flows shown in Figure 3.1b. Most flights change their headings by less than 15 degrees, a few change the heading by around 30 degrees within 30 seconds. The distributions of the flight speed changes and altitude changes are also visualized in the third row of Figure 3.2. An airport is located below the given airspace, therefore, a large number of flights change their vertical profile when flying in this area. The simulated traffic is dynamic in both

horizontal and vertical dimensions, which affects the airspace sectorization methods, especially the 3D sectorization methods.

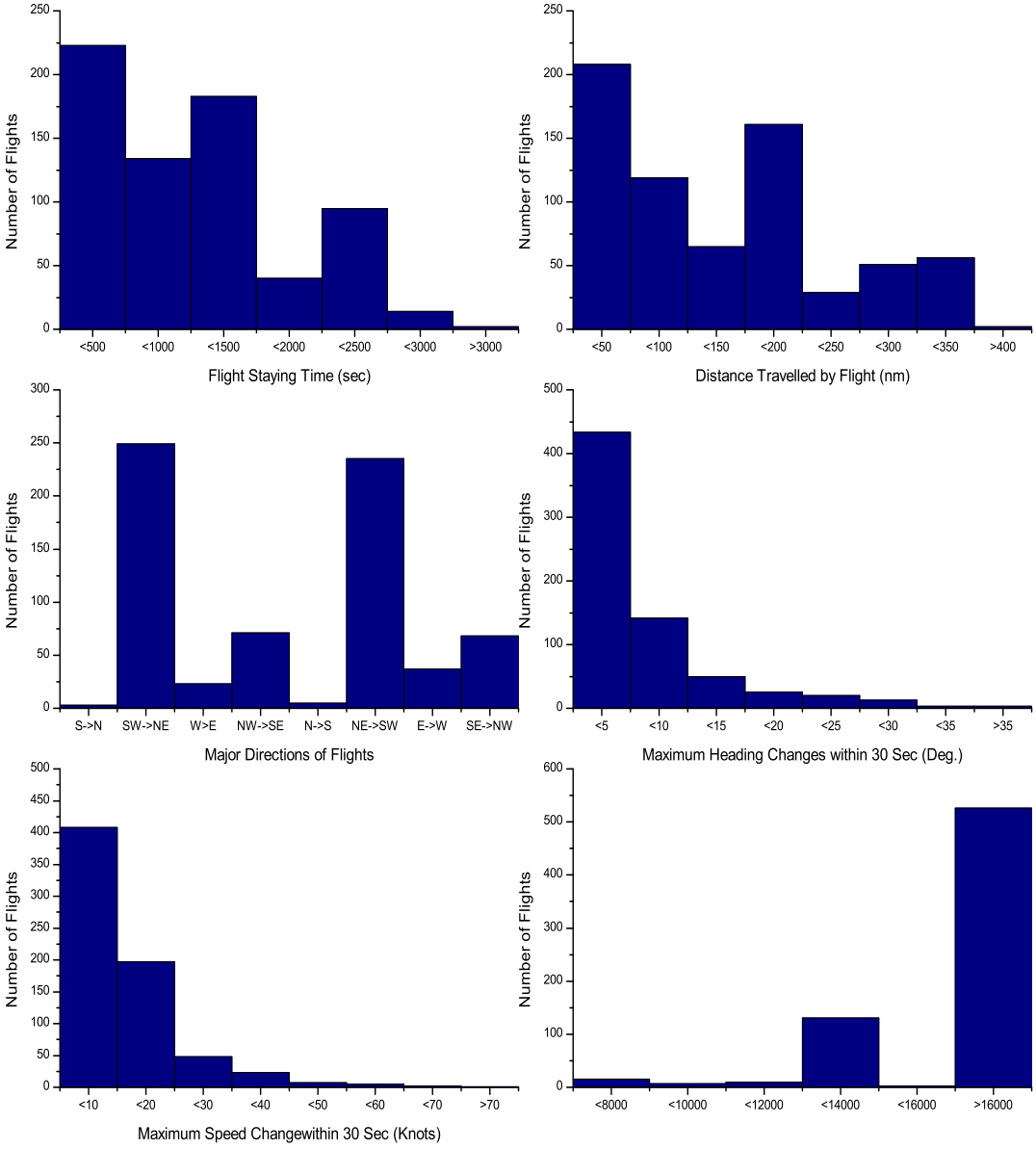


Figure 3.2: Spatial Characteristics of Simulated Flight Trajectories

3.2 Evaluation of the Agent Based Model for 3D Airspace Sectorization

3.2.1 Implementation of ABM

The ABM (Kicinger and Yousefi, 2009) has two objectives for agents (an agent represents a sector): minimizing the task load variance and maximizing the commonality of traffic. The task load of agents is calculated based on Equation 3.3 and the cumulative task load of a sector is constrained by the upper and lower bound on allowed task load variance from the mean task load W_{avg} as in Equation 3.7.

$$(1-\varepsilon) \times W_{avg} \leq W_k \leq (1+\varepsilon) \times W_{avg} \tag{3.7}$$

ε varies from 0 to 1.0. The traffic commonality of a sector is formulated as Equation 3.8.

$$CC_k = \sum_{C_{u,v,z}, C_{u',v',z'} \in S_k} TC_{C_{u,v,z}, C_{u',v',z'}} \tag{3.8}$$

Cell $C_{u,v,z}$ and Cell $C_{u',v',z'}$ both belong to sector S_k, and the $C_{u',v',z'}$ is adjacent to $C_{u,v,z}$. The commonality of cells ($TC_{C_{u,v,z}, C_{u',v',z'}}$) is defined as the total number of flight transfers from $C_{u,v,z}$ to $C_{u',v',z'}$ during a given time interval.

The ABM is implemented based on the four agent rules:

Movement Rule:

1. Find all neighbouring cells of an agent that are not assigned to other agents.
2. Find the one that satisfies the right-prism constraint (if right prism is required) and maximizes increase of agent's cumulative commonality from the list of neighboring cells.
3. Once this optimal cell is found, it is assigned to this agent.
4. The task load associated with this cell is added to this agent's cumulative

task load and the agent's cumulative commonality is updated too.

Layer Growth rule:

1. Find all neighbouring cells of an agent that are not assigned to other agents.
2. Find all vertical blocks of cells as well as horizontal layers of cells from the list of neighboring cells.
3. Find a locally optimal block or layer, which maximizes increase of agent's cumulative commonality, from the set of all valid vertical blocks and horizontal layers of cells.
4. Once the optimal block or layer is found, it is assigned to this agent.
5. The task load associated with this block or layer is added to this agent's cumulative task load and the agent's cumulative commonality is updated too.

Trading Rule: It is called when the cumulative task load of an agent exceeds the upper bound on task load variation.

1. Identify all boundary cells of the agent.
2. Remove the cells from the list that break contiguity of the sector when they are traded to other agents.
3. If right prism is required, remove the cells from those found in Step 2 that violate right prism when traded.
4. Calculate the excess task load of the agent: $\triangle W_k = W_k - (1+\varepsilon) \times W_{avg}$.
5. Set the traded task load for $\triangle TW_k = 0$.
6. For each cell ($C_{u,v,z}$) identified in Step 3:
   (a) If $\triangle TW_k >= \triangle W_k$, end.
   (b) Find all neighbouring cells $N_{u,v,z}$ of cell $C_{u,v,z}$.

(c) Find an optimal trading partner (agent) k for cell $C_{u,v,z}$, which maximizes increase of agent's cumulative commonality, from the set of neighboring cells $N_{u,v,z}$.

(d) Trade cell $C_{u,v,z}$ to agent k.

(e) Update the task load for both agents and $\triangle TW_k$

Repair Rule: It is only executed for cells violating right prism constraint. For each constraint configuration plane k defined by cell $C_{u,v,z}$ and its horizontal neighbor $C_{u',v',z}$ and vertical neighbour $C_{u,v,z'}$:

1. If configuration plane k is infeasible

   (a) Find an agent k that contains cells $C_{u',v',z}$ and $C_{u,v,z'}$.

   (b) Find an agent s that contains cell i.

   (c) Check that the cell $C_{u,v,z}$ doesn't break the contiguity of the sector s if the cell $C_{u,v,z}$ is reassigned to agent k.

   (d) Check that cell $C_{u',v',z}$ or cell $C_{u,v,z'}$ doesn't break the contiguity of the sector k if either cell $C_{u',v',z}$ or cell $C_{u,v,z'}$ is reassigned to agent s.

   (e) If true for both Step 3 and Step 4:

      i. Reassign cell from an agent with higher cumulative task load to an agent with lower cumulative task load.

   (f) Else If Step 3 is true then reassign cell $C_{u,v,z}$ to agent k.

   (g) Else If Step 4 is true then reassign either $C_{u',v',z}$ or $C_{u,v,z'}$ cell to agent s.

   (h) After reassignment, verify cell $C_{u,v,z}$ satisfies the right-prism constraint.

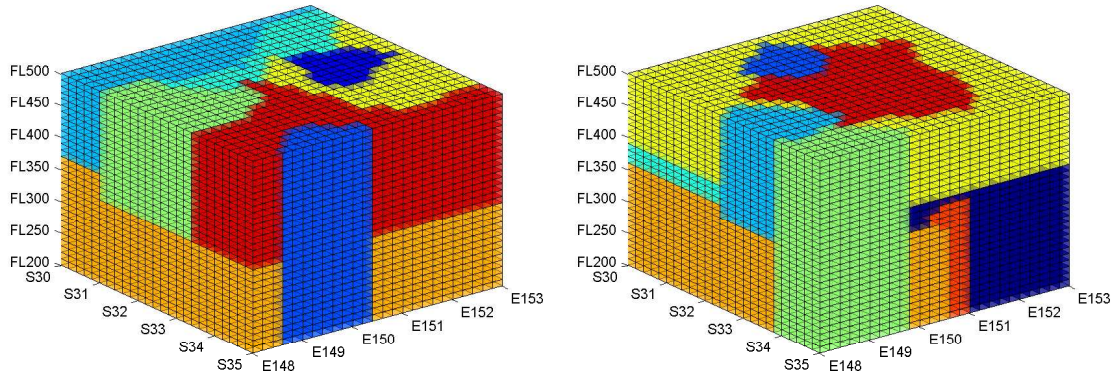
      i. If true then END

      ii. Else continue the next configuration plane $k+1$

The ABM Model is evaluated for a section in Australian airspace, using the air traffic data sample described in the previous section.

According to the Movement Rule and Layer Growth Rule, agents group the cells that increase the sector's commonality by the greatest amount without exceeding the task load upper bound (1.2 times of average task load where $\varepsilon = 0.2$). However, there is no clear movement rule definition for an agent that is not surrounded by any traffic cells. Therefore, an agent groups all empty neighbouring cells of its current sector configuration.

I initialized 10 agents with random locations, and then activated the agents to group cells iteratively according to a predefined movement ratio (0.2). Two examples of the 3D sectors produced by the ABM are visualised in Figure 3.3.



(a) Example 1 of ABM

(b) Example 2 of ABM

Figure 3.3: Two Examples of 3D Sectorization Results by ABM with Random Agent Initial Locations

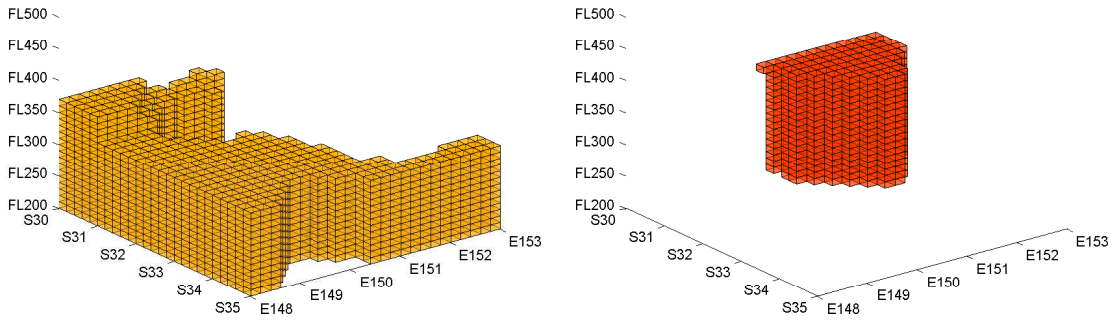
The experimental results shows that the ABM can produce a 3D sectorization solution, with the given objectives of maximizing traffic commonality and balancing sectors task load. However, right prism violation reported in (Kicinger and Yousefi, 2009) is found in the experiments as well. In addition, other problems exist in this approach in terms of sector design:

- Sector nests inside another sector.
- Sector shape doesn't satisfy the convexity constraint.
- High computational cost

The next sections examine these limitations and attempt to identify their causes.

3.2.2 Right Prism Polygon Violation

The right prism polytopes for sectors can't be guaranteed by ABM. Figure 3.4 shows two examples of the sector polytopes generated by ABM that do not meet the right prism constraint.



(a) Example of Sector 3 (b) Example of Sector 4

Figure 3.4: Two Examples of Right Prism Violation Generated from ABM

The right prism constraint requires that if one of the vertical neighbours and one of the horizontal neighbours of a cell belong to the same sector, the cell itself has to be in the same sector. For example, on the left of Figure 3.5, the cell i is assigned to sector a while its vertical and horizontal neighbours j_1 and j_4 belong to sector b, which violates the right prism constraint.

In the shown example, the Repair Rule tries to resolve the right prism violation by reassigning either the neighbour or the cell itself to other sectors, without breaking the contiguity of the sectors involved. This is achieved in one of three ways (shown on the right of Figure 3.5): to reassign the cell i to sector b, or to reassign either the cell j_1 or j_4 to sector a. The selection of the reassigned cell depends on the contiguity check of its owner sector and the task load limits. After reassignment, the cell i is checked to ensure it meets the right prism constraint. If necessary, the repair rule repeats these steps until the violation is resolved.

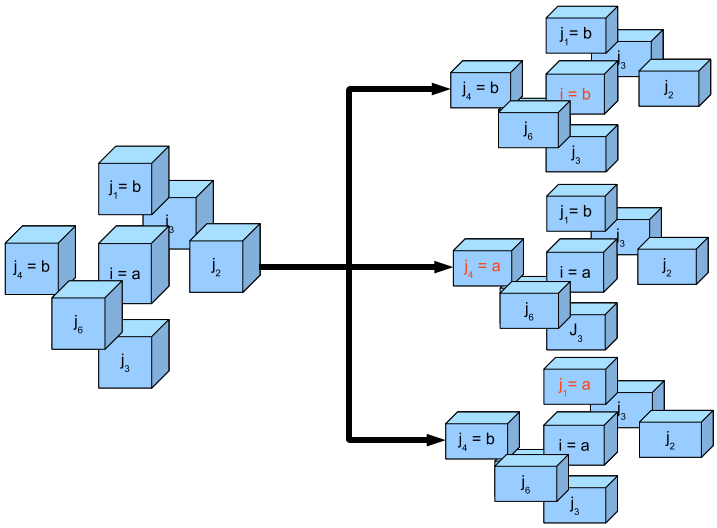


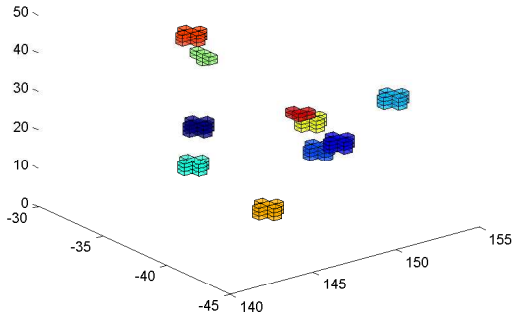
Figure 3.5: Right Prism Violation and Possible Solutions by Repair Rule in ABM

It is possible that no solution can be found. If the cells j_1 and j_4 belong to sector a and the cells j_2 and j_3 are assigned to sector b, the cell i can't be assigned to either a or b. The Repair Rule bounces cell i between these two sectors, never getting a solution satisfying right prism for both sectors.

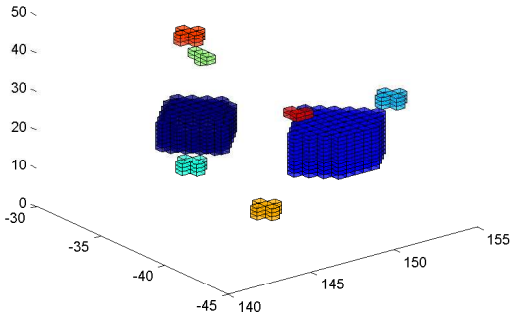
In the ABM model, the Repair Rule and right prism check are applied at the cell level, not at the geometric plane level, which implies that the sector can't satisfy the right prism constraint. Therefore, although the maximum level of right prism constraint check and Repair Rule is applied in ABM, the violation of right prism can't be eliminated.

3.2.3 Embedded Sectors

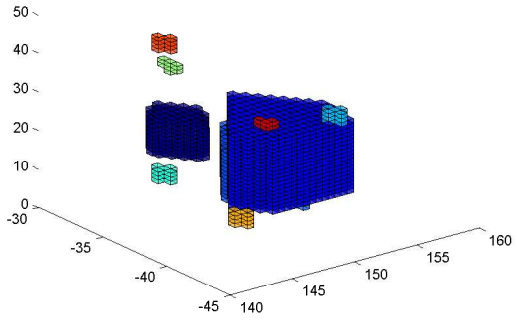
Some instances are found where the ABM method produced sectors embedded inside other sectors, as shown in Figure 3.6d. The reason is that the movement ratio for agents causes non-uniform growth of agents. Embedded sectors is not a practical outcome because of two reasons. First, the handover workload is increased because a flight may enters and exists the outer and inner sectors multiple times. Second, the 2D projection of outer sector on ATC's screen is not straightforward and it complicates ATC situation awareness and causes safety issues.



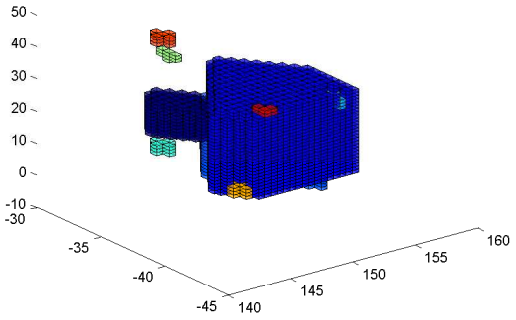
(a) Iteration 1



(b) Iteration 5



(c) Iteration 9



(d) Iteration 12

Figure 3.6: An Example of Embedded Sectors Caused by Unbalanced Growth of Agents in ABM

The Movement Ratio in this model decides which agents to be activated at each iteration. The Movement Rule and Layer Growth Rule are executed by only the activated agents that have least accumulated task loads at each iteration. Therefore, the privilege of movements for each agent is not equal at each iteration. Some agents having lower accumulated task load may grow faster than others, while some agents having higher accumulated task load may not grow at all during some iterations.

If the locations of agents are close to others, and some have greater growth rate at the beginning, the slower agents may be surrounded by the faster growing agents. There is then no space left for future movements of the slower agents. In such cases, some agents stop growing before reaching their capacity limits.

An example is shown in Figure 3.6. In Figures 3.6a and 3.6b, the blue agent is growing much faster because it has lower accumulated task load than other agents. By iteration 12 (Figure 3.6d), the red and light blue agents are surrounded by the blue agent. There are no unassigned cells around the red and light blue agents, so neither has any space to grow, even if their task loads are much lower than other agents in the future.

Randomly assigning initial positions to agents contributes to this problem. Embedded sectors are most likely to arise when the initial positions of agents are close to each other. Since there is no mechanism to limit the activities of agents whose growth rates are much higher than others in the ABM, instances of embedded sectors cannot be avoided in the ABM implementation.

3.2.4 Convexity of Sector Shape

Requiring a sector's lateral shape to be convex can avoid the situation where flights enter the same sector more than once. The convex shape is not considered in ABM: this grid based approach doesn't have any constraint check on the convexity and it can't guarantee the convexity of sectors. This is a serious limitation of the ABM approach when it comes to field testing of DAS with realistic air traffic scenarios.

3.2.5 High Computational Cost

The Movement Rule and the Layer Growth Rule have to identify the agent's boundary cells and their unassigned neighbours whenever they are executed by an agent. The worst case is to enumerate each cell of a sector to find its boundary cells. This has a computational complexity of $O(N)$, where N is the number of cells of an agent. The unassigned neighbouring cells then can be found by the boundary cells; this has a complexity of $O(N_b)$ where N_b is the number of boundary cells of an activated agent. Then, the right prism check on each neighbouring cell, and searching for a neighbouring cell with the maximum cumulative commonality, are

executed; this has a complexity of $O(N_n)$ where N_n is the number of neighbouring cells of an agent. Hence, the time complexity of the Movement Rule is $O(N+N_b+N_n)$ for an activated agent, which can be simplified as $O(N)$ because the number of cells of an agent N is normally greater than the number of its boundary and neighbouring cells.

The Layer Growth Rule has to group the neighbouring cells into vertical and horizontal layers. This can be executed at the same time as the right prism checking. It also has to find a layer with the maximum cumulative commonality, which has an additional time complexity of $O(N_v + N_h)$ where N_v is the number of cells in all vertical layers and N_h is the number of cells in all horizontal layers. $N_v + N_h$ equals N_n because all vertical and horizontal layers are made from the neighbouring cells. Therefore, the total time complexity of the Layer Growth Rule is $O(N + N_b + N_n)$ for an activated agent, which can be simplified as $O(N)$ too.

The Trading Rule is called by an agent when its task load exceeds the upper boundary. The Repair Rule is executed when some cells of a sector violate the right prism constraint. Both rules exchange an agent's cells with another agent without breaking sector shapes. Therefore, a "sector contiguity check" is necessary when the Trading and Repair rules are invoked. A grid-based sector can be treated as an undirected graph: the cells in a sector are the vertices (V) and the links between neighbouring sectors are the edges (E) of the graph. To check the contiguity of a sector is the same as to check the contiguity of a graph. Either Breadth-First-Search (BFS) or Depth-First-Search (DFS) can be used. The cost for both BFS and DFS is $O(|V| + |E|)$. In ABM, each cell has 6 edges at most; therefore the computational complexity of the sector contiguity check can be considered as $O(N)$ where N is the number of cells in a sector.

The Trading rule requires finding an exchangeable cell of a sector without breaking its contiguity. The worst case is to check each cell in a sector: this costs $O(N^2)$. The Trading Rule is used to hand over a cell from one agent to another, so the contiguity check is applied on one agent only. Therefore, the total time complexity of the Trading Rule is $O(N + N_b + N^2)$ for an agent, including finding its boundary

cells for exchanging, which is $O(N^2)$

The Repair Rule exchanges cells between two agents in order to satisfy the right prism constraint. It has to check the sector contiguity for both agents (k and s) involved, costing $O(N_k^2 + N_s^2)$, where N_k and N_s are the numbers of cells of agents k and s respectively. The right prism check is also enforced after the exchange, with cost $O(N_k+N_s)$. The total complexity of the Repair Rule is therefore $O(N_k + N_k^2 + N_s^2 + N_s)$ for two agents involved in the Repair Rule. Therefore, the time complexity of the Repair Rule is $O(N_k^2 + N_s^2)$.

Table 3.1 summarizes the computational complexity for each agent rule.

Agent Rules	Computational Complexity
Movement Rule	$O(N)$
Layer Growth Rule	$O(N)$
Trading Rule	$O(N^2)$
Repair Rule	$O(N_k^2 + N_s^2)$

Table 3.1: Computational Complexity of the Agent Rules in Agent Based Model

As shown in Table 3.1, the computational complexity of the Movement Rule and Layer Growth rules increases linearly with N as the number of cells in a sector and the number of its neighbouring cells increase, but the complexity of the Trading Rule and Repair Rule increase with the square of N as the number of cells in a sector increases. The Movement Rule and Layer Growth are executed at each iteration when the agents are activated. The other two rules are called when task load balancing or right prism constraint are violated.

The above analysis applies to each agent, in each iteration. Thus the overall complexity of ABM is $O(S \times N^2 \times I)$, where S is the number of agents and I is the number of iterations. The number of agents is a constant number when the expected number of sectors is decided, but the number of iterations and the number of times each rule is invoked vary depending on the agents' initial locations, their movements, etc.

To investigate the cost of each rule, I ran ABM with 10 different initial agent positions to group 20956 cells into 10 sectors for the given airspace and traffic data

described in Section 3.1.2. Table 3.2 lists the total time taken by each agent rule and the total number of times each rule was invoked during these 10 runs [1].

Rules	Layer Growth Rule	Movement Rule	Repair Rule	Trading Rule
Time (milliseconds)	21277.71	93.51	396107.39	15001.75
Number of times invoked	5937	143	66063	18
Time (milliseconds) per invocation	3.58	0.65	6.00	833.43

Table 3.2: Execution time and number of times invoked, of ABM Agent Rules in 10 Runs

As shown in the table, around 91% of the computation time was spent by the Repair Rule, and around 4% of the computation time was taken by the Trading Rule. The Layer Growth and Movement rules, including right prism checks after grouping cells, only accounted for around 5% of computation time. The Repair Rule was invoked most often, because the right prism is frequently violated by the Movement Rules and Trading Rules. The Trading Rule is invoked after agents stop grouping more cells. At this time, each agent reaches the maximum number of cells which it can group. The Trading Rule has to check the contiguity of two sectors at the same time, therefore, time spent by the Trading Rule on each invocation is the most expensive.

All of these show that the computation in ABM is dominated by the Repair Rule. It is the most frequently invoked rule, and its cost per invocation is higher than the other rules (except for the rarely-invoked Trading Rule). Its computational complexity is $O(N^2)$, so the computational cost of ABM rises with the square of the number of cells in a given airspace. This shows that ABM is not an efficient model for airspace sectorization, especially when the size of the airspace is large. According to the time complexity and the experimental results, if the same cell size with the same number of agents was done for the Australian airspace, the total time needed would have to be around 4000 hours.

[1] The 10 experiments are run on a PC equipped with an Intel Core2Duo 3.0GHz CPU, 4GB RAM and Windows XP

3.2.6 Summary

The limitations of the agent based model, their causes, and possible solutions are summarised in Table 3.3.

Limitations	Causes	Possible Solutions
Right Prism Polygon Violation	Limitation of Repair Rule and cell level based right prism constraint check	New Rules for both agent growth and repairing violation of right prism Constraint
Embedded Sectors	Agent Movement Ratio and Growth Rules	New Rules for both agent growth
Convexity of Sector Shape	No Convexity Constraint applied	Introducing the convexity check
High computational Cost	Sector Contiguity check by Repair Rule and Trading Rule	Remove the contiguity check of sector without breaking sector contiguity

Table 3.3: Summary of Limitations Existing in ABM and Possible Solutions for 3D Airspace Sectorization

First of all, the right prism violation prevents the ABM from being used in practice. The sectors violating right prism constraint can't be projected on a 2D ATC screen for air traffic management. Secondly, the embedded sectors and non-convex sector shapes generated by ABM allow flights to enter the same sector more than once which increases the ATC workload on handover. Finally, the computational cost of ABM is high for an operational ATM environment where an efficient DAS approach is necessary to sectorize the airspace in order to accommodate fluctuating air traffic demands. Therefore, these limitations of ABM make it unusable in practice.

3.3 An Improved Agent Based Model (iABM) for 3D Airspace Sectorization

The limitations of the ABM, their causes, and possible solutions provide a foundation for the development of an improved agent based model (iABM), with the aim of addressing the limitations of the current model. The first objectives of

the proposed model are to make the sector shapes satisfy the right prism constraint and to solve the Embedded Sector problem. Reducing the computational cost is another objective of iABM. Maximizing traffic commonality of sectors and balancing controller task load are also considered.

In the proposed iABM, instead of the Movement Rule and Layer Growth Rule, there is only one rule (Growth Rule) which is responsible for agents grouping neighbouring cells. The Growth Rule groups cells based on traffic commonality and task load; in order to satisfy the right prism constraint, cells are only grouped if they belong to the same geometric plane horizontally or vertically.

There is no Repair Rule in iABM because all sectors satisfy the right prism constraint during their growth. However, because the sectors grow by geometric planes, some gaps (unassigned cells) may be left between sectors when agents can't group any cells because of the right prism constraint. Therefore, a Gap Filling Rule is executed at the end of agents' movements. It reorganizes sector boundaries, and assigns unassigned cells at the geometric plane level to neighbouring sectors. Since the main objective of iABM is to generate sectors meeting the right prism constraint, the Trading Rule, which can lead to violations of the right prism constraint, is not used in iABM.

3.3.1 Growth Rule

The Growth Rule is executed by agents to group a set of their neighbouring cells, which belong to a same feasible geometric plane, in order to maximize their traffic commonality within the predefined task load range. It always applies at a plane level instead of at a cell level. The Growth Rule includes the following steps:

1. Find all neighbouring and not-yet-assigned cells for sector S_k.
2. Group these cells into horizontal or vertical geometric planes according to the current sector configuration.
3. From the list of geometric planes, filter out any that violate the right prism

constraint.

4. Calculate the task load for each remaining geometric plane in the list, and filter out those whose accumulated task load exceeds the upper bound of the predefined task load range.

5. From the remaining geometric planes, select the optimal geometric plane: one that most increases the sector's accumulated commonality.

6. All cells belonging to the optimal geometric plane are marked as assigned and are added into the agent's collection of cells.

7. The task load associated with all cells in the optimal geometric plane is added to this agent's accumulated task load.

Algorithm 3.1 describes the Growth Rule in detail.

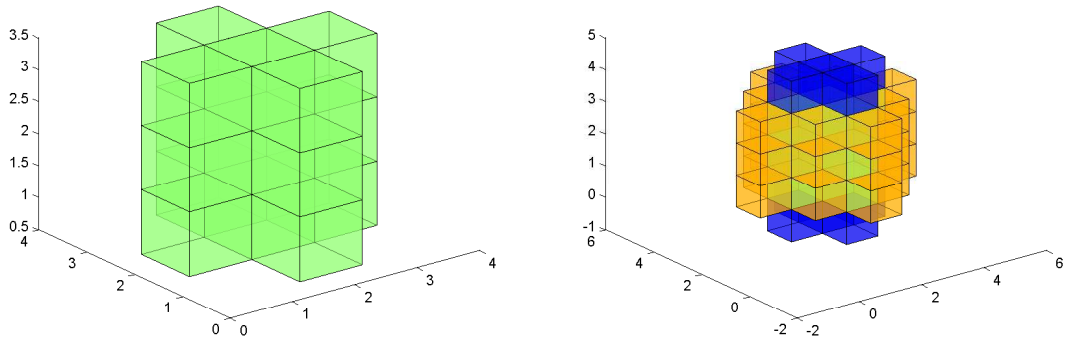
Algorithm 3.1 Agent Growth Rule of iABM

$C''_{uvz} \Leftarrow BC_k$ of S_k
$P_T \Leftarrow C''_{u,v,T}$ where $T \Leftarrow \max(z)+1$ and $C''_{u,v,T}$ is unassigned {Top horizontal layer}
$P_B \Leftarrow C''_{u,v,B}$ where $B \Leftarrow \min(z)-1$ and $C''_{u,v,B}$ is unassigned {Bottom horizontal layer}
{Vertical blocks}
$P_{v1} \Leftarrow \{C''_{u\pm1,v,z}\}$ where $C''_{u\pm1,v,z}$ is unassigned
$P_{v2} \Leftarrow \{C''_{u,v\pm1,z}\}$ where $C''_{u,v\pm1,z}$ is unassigned
$P_V \Leftarrow P_{v1} \cup (P_{v2} \setminus (P_{v1} \cap P_{v2})$
$P'_V \Leftarrow P_V$
for each P_{Vi} **in** P'_V **do**
    if P_{Vi} *violates Right Prism* **then**
        $P_V \Leftarrow P_V \setminus P_{Vi}$
    end if
end for
if $Taskload(P_T) \equiv 0$ **and** $Taskload(P_B) \equiv 0$ **and** $Taskload(P_{Vi}) = 0$ for each P_{Vi} **then**
    $S_k \Leftarrow S_k \bigcup P_T \bigcup P_B \bigcup P_V$
else
    $P' \Leftarrow (P_T \cup P_B \cup P_V)$
    for each p in $(P_T \cup P_B \cup P_V)$ **do**
        if $Taskload(p) > MAX\ task\ load$ **then**
            $P' \Leftarrow P' \setminus p$
        end if
    end for
    Select OP if $Commonality(OP) \equiv \max(Commonality(P'))$
    $S_k \Leftarrow S_k \bigcup OP$
end if
Update task load of S_k

The feasible geometric planes generated by the Growth Rule can be classified into two catalogs:

- A horizontal layer is a plane directly above or below the current sector's top or bottom layers. The lateral (latitude and longitude) shape of a valid layer is exactly same as the lateral configuration of a sector.

- A valid vertical block is a group of cells having the same latitude and longitude (only one pair of them) coordinates beside the current sector boundary, with all altitude levels existing in the sector.



(a) Current Sector Shape (b) Possible Geometric Planes

Figure 3.7: Possible Horizontal and Vertical Geometric Planes for A Sector Generated by Growth Rule in iABM

The horizontal layers make a sector grow vertically and the vertical block make a sector grow horizontally, without breaking the right prism constraint. For example, if the current sector shape is as shown in Figure 3.7a, the blue geometric planes in Figure 3.7b are valid horizontal layers and the yellow blocks are valid vertical blocks.

The movement ratio is still applied for this Growth Rule. Embedded sectors can't occur under the Growth Rule because the geometric plane level growth prevents an agent from surrounding others both vertically and horizontally.

3.3.2 Gap Filling Rule

Based on the Growth Rule, agents create sectors at the level of geometric planes. The sector shapes are guaranteed to satisfy the right prism constraint. However,

some cells may be unable to be assigned to any agent under this rule. Therefore, a Gap Filling Rule is needed to reorganize the sector boundary for these unassigned cells. This rule also works at the geometric plane level. It is executed only when unassigned cells exist in the airspace after all agents finish their activities. The steps involved in this rule are:

1. Find all unassigned cells (UC).
2. Find all agents directly above or below these cells.
3. If the number of agents above the unassigned cells is less than the number of agents below the unassigned cells, the agents above the unassigned cells will act to group these cells; otherwise, the agents below the unassigned cells will act. This keeps the number of affected sectors as low as possible.
4. For each activated agent(S_k):
   (a) Identify a horizontal layer(L) corresponding to the agent's sector lateral configuration that contains these unassigned cells.
   (b) Find all other agents (S_j) occupying the cells (OC) belonging to this layer(L).
   (c) Identify the horizontal layers(L') containing the cells (OC) which belong to the agents (S').
   (d) Agents(S_j) release these horizontal layers (L') and recalculate their task load and commonality.
   (e) Agent(S_k) adds the horizontal layer(L) into its collection and recalculates its task load and commonality.
5. Repeat steps 1 to 4 until there is no unassigned cell in the airspace.

Algorithm 3.2 describes the Gap Filling rule in detail.

The Gap Filling Rule may take several iterations to assign the unassigned cells. In some cases, some sectors may end up being deleted by this rule. It is complicated

Algorithm 3.2 Agent Gap Filling Rule of iABM

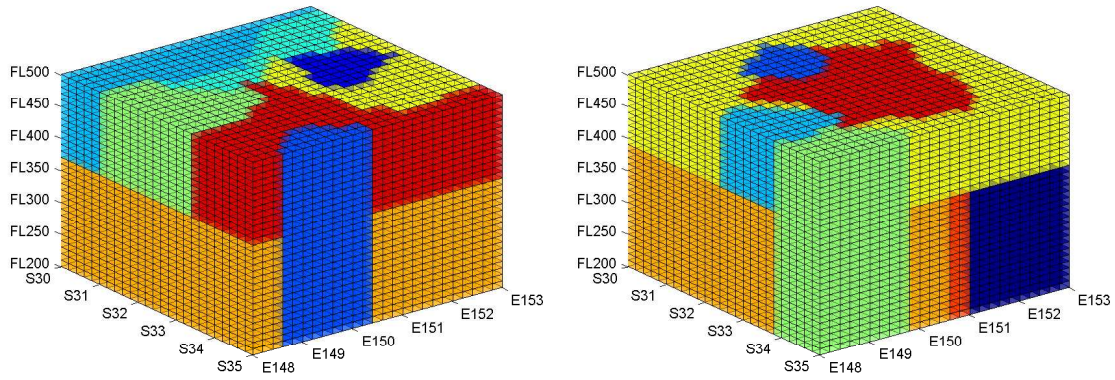
```
1: UC ⇐ all unassigned cells in R
2: while |UC| > 0 do
3:     Create two empty list of agents SU and SL
4:     for each S_k in S do
5:         if C_{u,v,z} ∈ UC and C_{u,v,z+1} ∈ S_k then
6:             put S_k in SU
7:         end if
8:         if C_{u,v,z} ∈ UC and C_{u,v,z−1} ∈ S_k then
9:             put S_k in SL
10:        end if
11:    end for
12:    if (|SU| < |SL|) then
13:        AS ⇐ SU
14:    else
15:        AS ⇐ SL
16:    end if
17:    for each S_k in AS do
18:        C''_{uvz} ⇐ BC_k of S_k
19:        if L ≡ SU then
20:            l = min(z) − 1
21:            l' = min(z)
22:        else
23:            l = max(z) + 1
24:            l' = max(z)
25:        end if
26:        L ⇐ C''_{u,v,l}
27:        for each agent S_j in (S \ S_k) do
28:            if OC ∈ L and OC ∉ S_k and OC ∈ S_j then
29:                OC''_{u'v'z'} ⇐ BC_j of S_j
30:                L' ⇐ OC''_{u',v',l'}
31:                S_j = S_j \ L'
32:                update task load of S_j
33:            end if
34:        end for
35:        S_k = S_k ⋃ L
36:        update task load of S_k
37:    end for
38:    UC ⇐ all unassigned cells in R
39: end while
```

to fill the gaps between sectors in 3D and satisfy the right prism constraint at the same time.

In summary, the Growth Rule is used for agent growth and the Gap Filling Rule is used when unassigned cells exist in the airspace. The Growth Rule groups cells to maximize a sector's commonality with consideration of task load balancing. The Gap Filling Rule fixes the problem of unassigned cells caused by the Growth Rule. Both of these rules work at a geometric plane level in order to meet the right prism constraint, which is the first objective of the iABM. Neither rule requires a sector contiguity check, saving some computational cost compared to ABM.

3.3.3 Evaluation of the Improved Agent Based Model

I evaluate iABM using the same input data and configurations as in the evaluation of ABM in section 3.2. Ten agents are initialized with the same locations used with ABM. The airspace sectorization of two Examples is visualised in Figure 3.8. Comparing Figure 3.8 with Figure 3.3 shows that problems such as right prism violation and embedded sectors that arise with ABM are solved by iABM.



(a) Example 1 of iABM

(b) Example 2 of iABM

Figure 3.8: Two Examples of 3D Sectorization Results by iABM with Random Agent Initial Locations

Beside these problems, there is another problem (high computational cost) existing in ABM model. The number of cells in an airspace affects the computational cost significantly. The Growth Rule includes finding the neighbouring cells, right prism check, and grouping cells into vertical and horizontal planes. It has the computational complexity of $O(N + N_b + N_n + N_v + N_h)$, where N is the number of cells in a sector, N_b is the number of boundary cells of a sector, N_n is the number of neighbouring cells of a sector, N_v is the number of cells of all vertical planes, and N_h is the number of cells of all horizontal planes. All vertical planes and horizontal planes are made from neighbouring cells, hence, $N_v + N_h$ equals N_n and the complexity can be considered as $O(N + N_b + N_n)$ for an activated agent, which can be simplified as $O(N)$ because the number of cells of an agent is normally greater than the number of its boundary and neighbouring cells.

The Gap Filling rule works on an activated agent (to group unassigned cells) and other affected agents (to release occupied cells) at the same time. The worst case is that all other agents are affected and the number of affected agents is $S-1$ (S is the total number of agents). In this case, the complexity of checking the boundaries and releasing occupied cells from each of these sectors is $O((S-1) \times N_j)$, where N_j is the number of cells in an affected agent. To reassign cells to the activated agent and check its boundaries is $O(N_k)$, where N_k is the number of cells in the activated agent. The complexity for the Gap Filling Rule working for one activated agent is $O(N_k + (S-1) \times N_j)$ which can be considered as $O(N_T)$ where N_T is the total number of cells in the given airspace. The Gap Filling Rule works on the agent level not on the cell level. In the worst case, the total computational complexity of the Gap Filling Rule to fill a gap is $O(S \times N_T)$ when all agents are activated by it. The computational complexity related to each agent rule of iABM is summarized in Table 3.4.

Agent Rules	Computational Complexity
Growth Rule	$O(N)$
Gap Filling Rule	$O(S \times N_T)$

Table 3.4: Computational Complexity of the Agent Rules in the Improved Agent Based Model

The Growth Rule is executed at each iteration by the activated agent, but the Gap Filling Rule is called only after all agents stoping moving and unassigned cells exist in the airspace. Both rules show the linear relationship between the computational complexity and the number of cells, which save a lot of computational cost compared with ABM. However, the Gap Filling Rule may be executed several times until all gaps are filled. This affects the total complexity of iABM dynamically, especially for the agents with randomly initial positions. I tried ten runs of iABM with the same agents initial positions, airspace configuration, and air traffic data as used with ABM to investigate the cost of each iABM agent rule. Table 3.5 shows the total time spent by each agent rule and the total times invoked by iABM in these ten runs.

Rules	Growth Rule	Gap Filling Rule
Time (milliseconds)	860.58	109.53
Number of times invoked	2921	31
Time (milliseconds) per invocation	0.29	3.53

Table 3.5: Execution time and number of times invoked, of iABM Agent Rules in 10 Runs

The Gap Filling Rule is the most expensive per invocation, but it is invoked fewer times than the Growth Rule. From the results of these ten runs, the time spent by the Gap Filling Rule is from 0% to 25% of total execution time, depending on whether gaps exist or not. Therefore, the relationship between total complexity of iABM and the number of cells may not be linear.

To compare the computational cost between the two agent based models, I evaluated the computational cost (processing time) of iABM and compared it against ABM, using different numbers of cells in the same airspace. I investigated 10 different cell sizes, giving 10 different numbers of cells in the airspace. For each number of cells, I initialized 10 sets of random positions for agents, giving 100 configurations in total. The same configurations were executed with both ABM and iABM. The other input data, such as flight trajectories and airspace boundary, were the same in each run. All runs were done on the same machine, equipped with an Intel Core2Duo 3.0GHz CPU, 4GB RAM and Windows XP.

The processing times for the 10 runs with a given number of cells was averaged, for each number of cells and for each model. The results are illustrated in Figure 3.9.

It shows that the computational cost for both models increased non-linearly as the number of cells increased, because ABM has the computational complexity $O(N^2)$ while iABM is affected by the Gap Filling Rules dynamically. The magnitudes of computational cost are very different. The minimum average time to finish sectorization (3751 cells) with iABM is 0.05 seconds while with ABM it is 3.77 seconds. The maximum average time to finish sectorization (34596 cells) with iABM is 2.71 seconds while with ABM it is 1167.3 seconds. This shows that ABM lacks scalability for large airspace. It also shows that iABM has significant computational

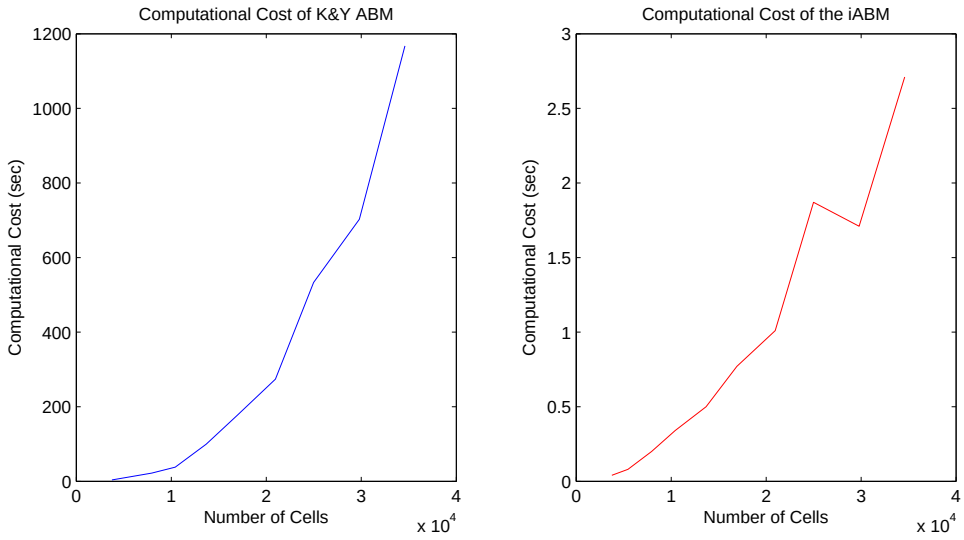


Figure 3.9: Computational Cost of Kicinger and Yousefi ABM and the Improved ABM

cost savings compared to ABM.

3.4 Chapter Summary

In this chapter, ABM is evaluated and it is found that there are several limitations, such as Right Prism Constraint violation, embedded sectors, high computational cost, etc., existing in ABM. Therefore, iABM with simplified agent rules is developed. Through the evaluations, iABM has advantages over ABM:

- Satisfying the Right Prism Constraint,
- Non-embedded sectors,
- Less computational cost.

However, iABM can't guarantee the task load balancing and other objectives without proper optimization methods although agents' movements are guided by the task load limits. The optimization of agent initial locations is addressed in Chapter 5. On the other hand, both iABM and ABM are grid based models, thus the sector shapes are all cuboid when the convexity constraint is applied. Other

approaches that are not agent based, also can produce cuboid shapes. These approaches are investigated in the next chapter.

Chapter 4

Geometric Models for Airspace Sectorization

The agent based model works on a discrete airspace which is divided into uniform cells. If the convexity constraint is a prerequisite for the agent based model, it must produce only cuboid sectors because of the unified airspace grids. To do this, agents need more rules to constrain their growth. However, there are several other efficient approaches existing in the literature, such as Octree, KD-tree, etc., to produce cuboid partitions in a space.

I investigated three more approaches, apart from agent based approaches, to sectorize 3D airspace for satisfying the requirements of convex constraint and efficiency.

Convexity is not the only constraint in 3D airspace partitioning. Another constraint addressed here is the right prism constraint, because ATC has only 2D screen to display the sector lateral boundaries and the traffic movements.

In this chapter, I start from KD-tree based model to address the convexity constraint. Then two models, based on Support Plane Bisection and Constrained Voronoi Diagram, are proposed in order to introduce a variety of sector shapes; these may better balance the task load and align sector shape and traffic flows, which can reduce the traffic flow cuts and minimize the ATC handover workload.

Because these three geometry based models don't work on the traffic directly, unlike ABM or iABM, the task load and other objectives have to be calculated based on the traffic and sector configurations. Each flight track needs to be assigned to different sectors based on its location and sector boundary when evaluating the airspace sectorization. In the worst case, each sector has to be searched for a single flight track, which is not efficient when the traffic volume and number of sectors is large. Therefore, an efficient search algorithm for each sectorization model is needed; this is described in the last section of this chapter.

4.1 Airspace Sectorization by KD-Tree

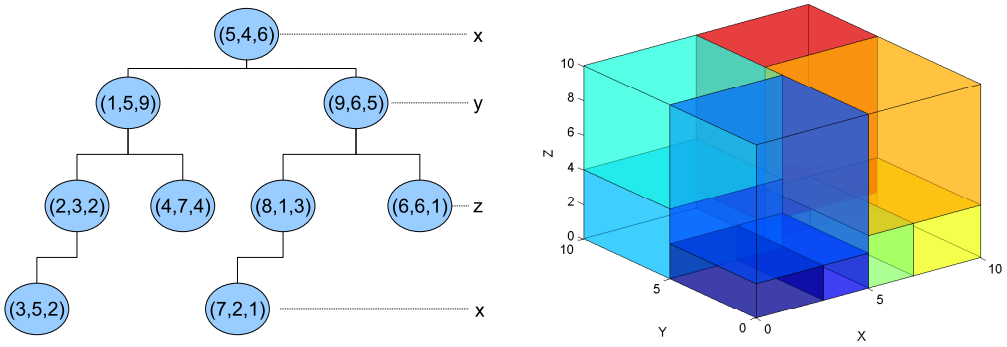
A KD-tree (k-dimensional tree) (Bentley, 1980) is a space-partitioning data structure. It is a binary tree where each node is a k-dimensional point. Every non-leaf implicitly generates a splitting hyperplane, which partitions the space into two parts. The left sub-tree of that node represents points on one side of this hyperplane, and the right sub-tree represents points on the other side of the hyperplane. The hyperplane direction is chosen by the depth of every node in the tree associated with one of the k-dimensions. The hyperplane is perpendicular to the chosen dimension's axis. The computation complexity to construct a KD-tree is $O(N \log N)$ where N is the number of given points.

I used the KD-tree based model for 3D airspace partitioning for two reasons. The first is that KD-tree can easily be constructed and partition a given airspace as mentioned above. The second is that searching a KD-tree node (sector) residing for a flight is more efficient than searching an Octree tree if they have the same depth. The computational complexity for performing a full Breadth First Search (BFS) or Depth First Search (DFS) for a tree is $O(b^d)$, where b is the branching factor and d is the depth of the tree. The time cost for KD-tree is $O(2^d)$ because it is a binary tree. However, the cost of an Octree is $O(8^d)$ because one Octree parent has 8 children. The model partitions the airspace into cuboid sectors, depending on a set of initial points. It is easy to apply and satisfies the convex shape requirement. The detail of

the model is described in Algorithm 4.1.

Algorithm 4.1 3D Airspace Sectorization by KD-trees

1: {Input: a list of points ($P\{x_1, x_2, ...x_p\}$), a given space (R), a dimension value (K), and a flag (D and initially $D \Leftarrow 1$) to determine partitioning axis}
2: **Function KDTreeAirspace**(P, R, D) {
3: **if** $|P| \equiv 0$ **then**
4: RETURN
5: **else**
6: $axis \Leftarrow D$ mod K
7: Sort P by the value of $axis$
8: Select the median x_m from P
9: Divide R by $axis$ value of x_m into S_1, S_2 where $R \equiv S_1 \cup S_2$ and $S_1 \cap S_2 \equiv \emptyset$
10: $P' \Leftarrow P \setminus x_m$
11: Create two empty list P_1 and P_2
12: **for each** x_i **in** P' **do**
13: **if** $x_i \in S_1$ **then**
14: $P_1 \Leftarrow P_1 \bigcup x_i$
15: **else**
16: $P_2 \Leftarrow P_2 \bigcup x_i$
17: **end if**
18: **end for**
19: $d \Leftarrow D + 1$
20: KDTreeAirspace(P_1, S_1, d)
21: KDTreeAirspace(P_2, S_2, d)
22: **end if**
23: }



(a) Tree Structure Constructed by KD-Tree based model

(b) Space Sectorization by KD-tree based model

Figure 4.1: An Example of 3D Space Sectorization by KD-tree

An example of KD-tree based sectorization is shown in Figure 4.1. In this example, the initial 9 points are sorted on their x-axis value first. Then the middle point (5,4,6) is selected; it divides the space along the x-axis at its x-axis value (5). All points with x less than 5 are in the left sub-tree of point (5,4,6), and the others are the right sub-tree. At the next depth of the tree, the points are sorted by their y-values, and the middle points are selected (point (1,5,9) in the left sub-tree, and

point (9,6,5) in the right sub-tree). Therefore, 4 sub-spaces are produced. Three points have been selected as splitting points; the other six are grouped into the 4 sub-spaces based on their x and y values. At the next the step, the same process is executed based on the z-values. The process is repeated until no further splitting is possible. The binary tree constructed in this example is shown in Figure 4.1a. The corresponding 10 sub-spaces are illustrated in Figure 4.1b.

As illustrated in the example, the KD-tree based model guarantees a convex shape for each sub-space, but the polytopes are limited to cuboids. This limitation may limit the KD-tree based approach's capacity to align sectors with traffic flows and to distribute task load evenly when it is used in DAS. In order to overcome these potential problems, two new models, based on Support Plane Bisection and Constrained Voronoi Diagram, which can import variety of polytope types, are developed and implemented.

4.2 3D Airspace Sectorization by Support Plane Bisection

To increase the variety of sector shapes, I developed a method called Support Plane Bisection model. It is an extension of the KD-Tree based model and relies conceptually on support vector machines (Cortes and Vapnik, 1995), still using a tree structure to store the data. Instead of using only one value along one axis to partition the space at each step, it alternates between using two values along two axes and one value on the vertical axis. First it uses a line between two points to divide a space laterally, grouping the remaining points into the sub-spaces based on their locations. The selected two points are the points which are closest to the centroid of all points. The perpendicular bisector of these two points can generate two hyperplanes where the remaining points are evenly distributed; this is similar to classifying points into groups in a given space by support vector machine. Therefore, this model is named as Support Plane Bisection Model (SPBM). The lateral partition by two points is

illustrated in 4.2.

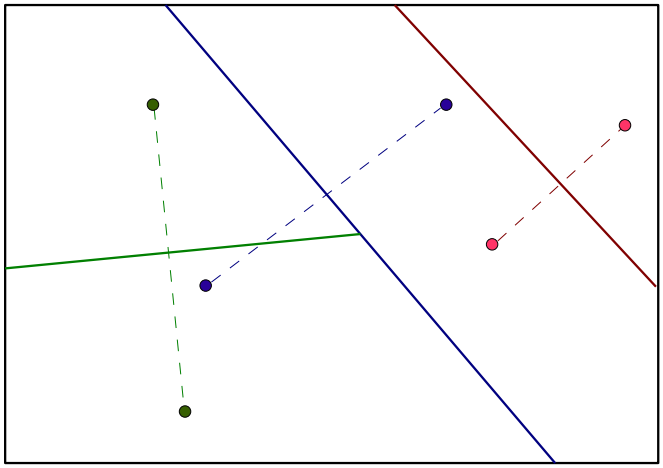
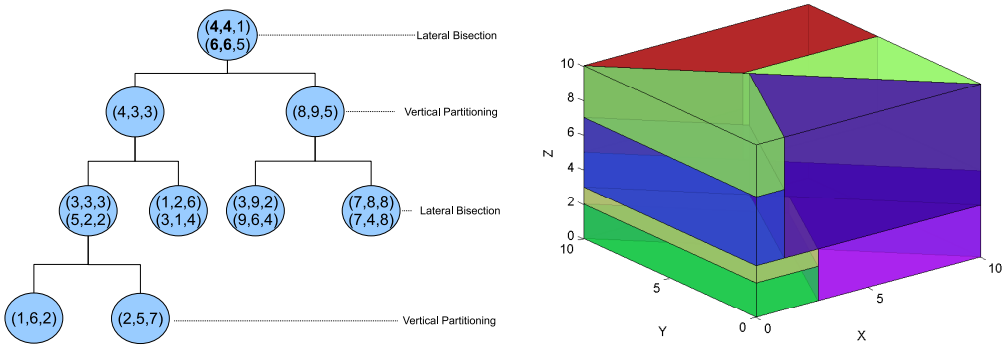


Figure 4.2: Lateral Partitioning by a Set of Points in SPBM

Then, it splits the sub-spaces vertically at the middle altitude value of the points in each sub-space. These two steps alternate until all points are processed. If it is a lateral partition's turn but the points in the sub-space only differ vertically, a vertical division is undertaken instead. The details of SPBM are described in Algorithm 4.2.



(a) Tree Structure Constructed by SPBM (b) Space Sectorization by SPBM

Figure 4.3: An Example of 3D Space Sectorization by Support Plane Bisection Model (SPBM)

A simple example of the Support Plane Bisection Model is illustrated in Figure 4.3. In this example, the space is partitioned into 10 sub-spaces by 14 points.

Algorithm 4.2 3D Airspace Sectorization by SPBM

```
1: {Input: a list of points (P{x_1, x_2, ...x_p}), a given space (R), and a flag (IsLateral) to determine partitioning
   directions}
2: Function SPBM(P, R, IsLateral) {
3: if |P| ≡ 0 then
4:     RETURN
5: end if
6: if IsLateral ≡ FALSE or |P| ≡ 1 then
7:     Sort P by the altitude value
8:     Get the middle point x_m from P
9:     Divide R vertically by altitude value of x_m into S_1, S_2 where R ≡ S_1 ∪ S_2 and S_1 ∩ S_2 ≡ ∅
10:    P' ⇐ P \ x_m
11:    Create two empty list P_1 and P_2
12:    for each x_i in P' do
13:        if x_i ∈ S_1 then
14:            P_1 ⇐ P_1 ⋃ x_i
15:        else
16:            P_2 ⇐ P_2 ⋃ x_i
17:        end if
18:    end for
19:    SPBM(P_1, S_1, TRUE)
20:    SPBM(P_2, S_2, TRUE)
21: else
22:    Get two points x_j, x_k that are closest to the centroid of all points in P
23:    Bisect the R laterally into two sub-spaceR_1, R_2 by the latitude and longitude of x_j, x_k where R ≡ S_1 ∪ S_2
       and S_1 ∩ S_2 ≡ ∅
24:    P' ⇐ (P \ x_j) \ x_k
25:    Create two empty list P_1 and P_2
26:    for each x_i in P' do
27:        if x_i ∈ S_1 then
28:            P_1 ⇐ P_1 ⋃ x_i
29:        else
30:            P_2 ⇐ P_2 ⋃ x_i
31:        end if
32:    end for
33:    SPBM(P_1, S_1, FALSE)
34:    SPBM(P_2, S_2, FALSE)
35: end if
36: }
```

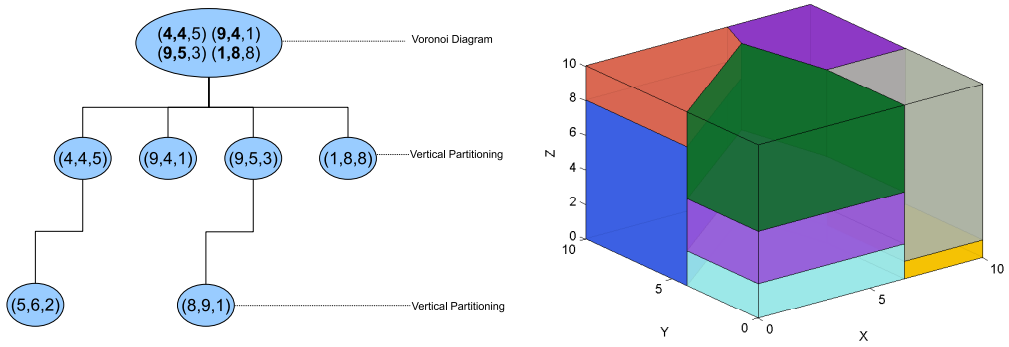
First the points (4,4,1) and (6,6,5), which are closest to the middle of the given space, are selected. The space is bisected by the (x, y) values (4,4) and (6,6) of these two points. The remaining points are grouped into these two sub-spaces. The second step divides the sub-spaces by the points having the middle z-values. These two steps are repeated alternately until all points are processed.

SPBM only generates convex shapes for hyperplanes, so the convexity constraint is always satisfied. Although the computational cost of SPBM is more than for the KD-tree based model, because of the additional geometry calculations, it is still faster than agent based models. Furthermore, SPBM introduces more polytopes for the sector shapes to capture air traffic flows, which is a big advantage over the KD-tree based model. However, the computation complexity is higher than KD-tree.

The computational complexity to get two points closest to the centroid is $O(N)$, where N is the number of given points in a space. For the whole construction of SPBM, it is $O(S \times N)$ where S is the total number of the expected sectors and N is the number of points in the space.

4.3 3D Airspace Sectorization by Constrained Voronoi Diagram

A Constrained Voronoi Diagram-based model (CVDM) for 3D Airspace Sectorization is also developed to introduce more diversity of sector shapes. A Voronoi diagram is a special kind of decomposition of a metric space, determined by distances to a specified discrete set of objects (sites) in the space. A Voronoi diagram guarantees convexity for each Voronoi cell. Although Voronoi Diagrams can partition 3D space, it can't satisfy right prism constraint. In this model, Voronoi Diagram is only used for lateral partitioning; vertical division is achieved by the vertical value of Voronoi sites to satisfy this constraint. Therefore, it is called Constrained Voronoi Diagram based Model. The optimal Voronoi Diagram Algorithm (Fortune's Sweep Line Algorithm) (Fortune, 1987) is applied in this model.



(a) Tree Structure Constructed by CVDM (b) Space Partitioned by CVDM

Figure 4.4: An Example of the 3D Space Partitioning by Constrained Voronoi Diagram based Model (CVDM)

Algorithm 4.3 3D Airspace Sectorization by CVDM

1: {Input: a list of points ($P\{x_1, x_2, ...x_p\}$), a given space (R), and N is the number of Voronoi Sites}
2: **Function CVDM**(P, R) {
3: **if** $|P| \equiv 0$ **then**
4: RETURN
5: **end if**
6: **if** $|P| \equiv 1$ **then**
7: Divide R vertically by altitude value of x_m in P into S_1, S_2 where $R \equiv S_1 \cup S_2$ and $S_1 \cap S_2 \equiv \emptyset$
8: RETURN
9: **else**
10: **if** ($N > |P|$) **then**
11: $n \Leftarrow |P|$
12: **else**
13: $n \Leftarrow N$
14: **end if**
15: Get n points from P as Voronoi sites $site_i$
16: Divide R laterally into Voronoi Cell V by the latitude and longitude of all $site_i$ where $R \equiv \cup_{i=1}^{n} V_i$ and $V_i \cap V_j \equiv \emptyset$ $(i \neq j)$
17: **for all** $V_i \in V$ **do**
18: Divide the each Voronoi Cell V_i into two sup-spaces (S_{i1}, S_{i2}) by the altitude of its site $site_i$ where $V_i \equiv S_{i1} \cup S_{i2}$ and $S_{i1} \cap S_{i2} \equiv \emptyset$
19: $S \Leftarrow S \bigcup S_{i1} \bigcup S_{i2}$
20: **end for**
21: $P' \Leftarrow P \setminus sites$
22: Create $(n \times 2)$ empty lists PL
23: **for each** x_k **in** P' **do**
24: **for each** s_{ij} **in** S **do**
25: **if** $x_k \in s_{ij}$ **then**
26: $PL_{ij} \Leftarrow PL_{ij} \bigcup x_k$
27: **end if**
28: **end for**
29: **end for**
30: **for each** S_{ij} **in** S **do**
31: CVDM(PL_{ij}, S_{ij})
32: **end for**
33: **end if**
34: }

In this model, a tree structure is again used during the processing. The inputs are a list of points (P) and a predefined number of sites N which is larger than the list size. The space is firstly decomposed into N Voronoi Cells by a sub-list (of size N) from the points list (P). Then each Voronoi Cell is divided vertically into two sub-spaces (lower and upper) by its site's altitude. Therefore, N points can produce $2 \times N$ sub-spaces. The remaining points are grouped into these sub-spaces according to their locations. The sub-spaces are further decomposed similarly. If there is only one point in the point list, only the vertical split is executed. The detail of this model is described in Algorithm 4.3.

Figure 4.4 shows an example of a given space divided into 10 subspaces, using 6 input points and with the sites number set to 4, and the corresponding tree structure.

By the nature of Voronoi diagram, the convexity of sector shape is guaranteed.

It has advantages over agent based model, such as satisfying right prism constraint, non-embedded sectors, and lower computational cost. Like the Support Plane Bisection Model, this model can generate different sector shapes compared to the KD-tree model. The computational complexity of Fortune's Sweep Line Algorithm is $O(N \log N)$ where N is the number of sites. Because the Constrained Voronoi Based Model works in a 3D space, and it divides space into sub-spaces iteratively until all given points are processed, the number of expected sub-space affects the computational complexity. As a whole, the complexity of CVDM is $O(S \times N \log N)$, where S is the number of expected sectors.

4.4 Search Algorithm for Grouping Flight Track Hits

These geometry based airspace sectorization models don't work on traffic data directly, unlike ABM or iABM, therefore it is necessary to develop a search algorithm to calculate the task load and other measurements based on flight tracks after airspace is sectorized by these models. A straightforward way is to search each sector and find each flight track hit residing in it. The computational cost is $O(S \times N_t)$, where S is the number of sectors and N_t is the total number of flight track hits. It is not an efficient method when the numbers of flight track hits and sectors are large.

The proposed geometric models partition the airspace based on a tree structure. The leaf nodes of a tree represent the final sectors in the airspace and a leaf node is the intermediate neighbouring sector of its siblings. On the other hand, a flight flies in a sector for a period and it can exit from this sector and only enter one of its neighbouring sectors. Based on these two characteristics, an efficient searching algorithm can be developed for these three DAS models.

As described in Algorithm 4.4, Breadth First Search (BFS) is used to search for a leaf node (sector) in a tree in which a given traffic hit resides. The cost to search a whole tree by BFS is $O(b^d)$, where b is the breadth of the tree and d is the depth

of the tree. But it is not necessary to traverse the whole tree for all air traffic hits of a flight unless the traffic hit is the first footprints of the flight. It starts from the tree root to search for a leaf node (sector) in which the first traffic hit (x_i1) of flight is. The second and afterward traffic hits are checked against the current sector and its siblings. If a leaf node containing the traffic hits is not found, then it searches the siblings of the node's parents and then their children until a leaf node resided by the traffic hits is found.

Therefore, the worst case of this search methods only happens once for a flight. The computational cost of this search algorithm is $O(b^d \times N_F)$, where N_F is the number of flights. Because $b^d \approx S$ and N_F is much less than N_t, this search algorithm is more efficient to handle a huge amount of air traffic data in a large airspace.

Algorithm 4.4 Sector Search Algorithm for Grouping Flight Track Hits

```
{Input: a flight trajectory i(t_i), the root of the tree (root)}
Function SearchSector(t_i, root) {
node ⇐ BFSSectorSearch(x_i1, root) {x_i1 is the first track hit in t_i}
for j = 2 → M do
    if x_ij ∈ s then
        CONTINUE
    else
        parent ⇐ node.parent
        while parent ≠ null do
            node ⇐ BFSSectorSearch(x_ij, parent)
            if node ≠ null then
                BREAK
            else
                parent ⇐ parent.parent
            end if
        end while
    end if
end for
}

Function BFSSectorSearch(x_ij, node) {
if x_ij ∈ node then
    if node is a leaf node then
        RETURN node
    else
        for all child_i of node children do
            RETURN BFSSectorSearch(x_ij, child_i)
        end for
    end if
else
    RETURN null
end if
}
```

4.5 Summary of 3D DAS Models

Table 4.1 lists the summary of all 5 models in 5 aspects: right prism, convexity, non-embedded sectors, shape diversity and algorithm efficiency. The shape diversity of sectors is a potential factor offering more flexibility to sectors for air traffic flow alignment and to keep necessary minimum distance between sector boundaries and traffic flow crossing points.

Models	Right Prism	Convexity	Non-Embedded Sector	Shape Diversity	Algorithm Efficiency
ABM				✓	
iABM	✓		✓	✓	
KD-Tree	✓	✓	✓		✓
SPBM	✓	✓	✓	✓	✓
CVDM	✓	✓	✓	✓	✓

Table 4.1: Summary of 5 DAS Models on the Sector Geometric Design and Algorithm Efficiency

As shown in the table, the ABM has more drawbacks than other models. iABM overcomes some limitations, such as right prism violation and embedded sectors, and takes into account ATC task load and sector alignment with traffic flow. However, it still lacks a mechanism to satisfy the convexity requirement. The computational cost of iABM is improved a lot compared to ABM, but is still not efficient enough.

The grid-based approach of ABM and iABM limits the sector shape to cuboid if convexity is required because they can only group unified boxes in the airspace. The KD-tree based model is more efficient, and also produces cuboid sectors that satisfy the convexity constraint and avoid problems of right prism violation and embedded sectors. On the other hand, it does not directly consider ATC task load, the alignment of sector shape with air traffic flow, or the minimum distance between sector boundaries and traffic flow crossing points, and so may produce sectorizations that are not realistic in practice.

Both SPBM and CVDM address the issue of cuboid sectors, which cause problems in alignment between traffic flow and sectors, minimum distance between sector boundaries and traffic flow crossing points, and task load distributions, while maintaining all other capabilities. SPBM is a special case of Voronoi diagrams, as it uses two points to bisect the geometric plane instead of a set of points as in Voronoi

diagrams. Hence, the CVDM may produce more variety of sector shapes.

Since there are no agent rules associated with the geometric space partitioning models, they are more efficient than both ABM and iABM. However they cannot go into finer details of sector boundaries according to traffic flows when it comes to partitioning the airspace, which can be handled effectively by agent based models. This problem may be solved by the optimization method to satisfy the objectives of DAS, as presented in the next chapter.

Chapter 5

Evolutionary Computation For DAS

In the previous two chapters, I introduce four different approaches for 3D airspace sectorization. These airspace sectorization models focus on geometric requirements and computational efficiency. I now propose multi-objective optimization approach using Genetic Algorithms to optimise the airspace sectors in terms of three objectives:

- Minimizing standard deviation of task load across sectors.
- Maximizing average sector flight time, aiming to maximize the time for which flights stay within a sector and to align the traffic flow with sectors in order to reduce the flow cuts.
- Maximizing distance between sector boundaries and traffic flow crossing points, aiming to maximize the minimum distance between the flow crossing points and sector boundaries to ensure enough responding time for ATC to resolve a potential conflict.

NSGA-II (Deb *et al.*, 2002) as a state-of-the-art multi-objective optimization algorithm is used in my approach.

ABM is excluded in the 3D airspace sectorization optimization here because of its inability to generate feasible solutions for sectors and its high computational cost as mentioned in Section 3.2.

5.1 NSGA-II and 3D DAS

NSGA-II is an efficient multi-objective evolutionary algorithm based on the Pareto optimal concept. It applies non-dominated sorting to rank individuals, and uses crowding distance to maintain diversity without specifying any additional parameters. The improved sorting approach reduces the computational complexity. It also uses an elitist strategy to expand the sample space. NSGA-II is widely used in many applications to solve multi-objective optimization problems.

NSGA-II is adopted as the optimization algorithm here, as follows:

1. Randomly initialize the first population
2. Continue the following steps until the termination condition (maximum generation reached) is satisfied:
   (a) 3D airspace sectorization by proposed models
   (b) Evaluate the fitness functions for each individual
   (c) Sort the population and produce offspring by crossover and mutation
3. Represent the optimal solutions

Figure 5.1 illustrates the architecture of the proposed system, which integrates my proposed 3D DAS models and NSGA-II to optimize airspace sectorization.

5.1.1 Chromosome Representation

Only spatial information of the airspace partitioning points is necessary to be encoded into the chromosome; temporal information is not needed. The reason is

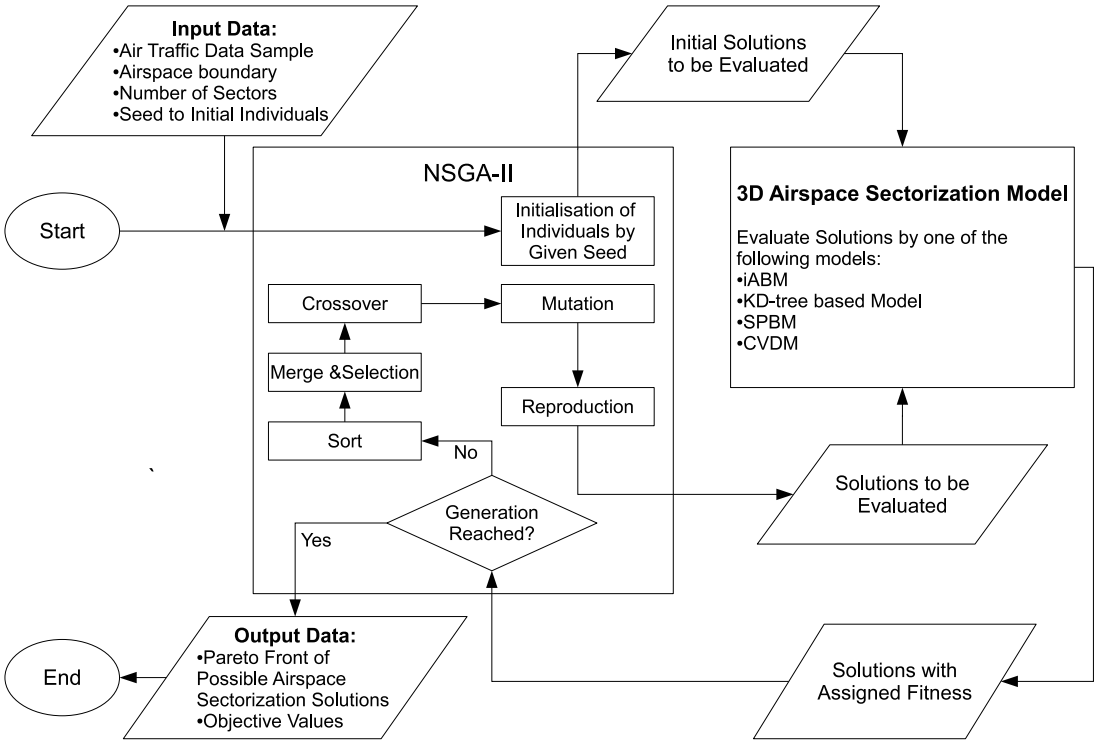


Figure 5.1: Architecture of the Integrated System of 3D DAS Models and NSGA-II

that airspace partitioning models only rely on the spatial information to produce sectors. Then, sectors wrap the 4D air traffic flows of a given period to calculate the specified DAS objectives for this period, such as task load. It is a common chromosome design method used in many 2D GA based DAS approaches (Delahaye *et al.*, 1998; Xue, 2008).

The inputs to the four 3D airspace partitioning approaches are the same: a list of points specifying latitude, longitude, and altitude in an airspace. Therefore, all four proposed models use the same basic chromosome representation in NSGA-II. However, the chromosome lengths differ. As mentioned in Chapter 4, different data structures and algorithms of these four models cause their outputs to have different number of sectors when they take the same number of control points. In order to produce 10 sectors from each model, the lengths of chromosomes are different from each other:

- iABM: 10 points in a chromosome and the length is 30.

- KD-tree: 9 points in a chromosome and the length is 27.
- SPBM: 14 points in a chromosome and the length is 42.
- CVDM : 6 points in a chromosome and the length is 18.

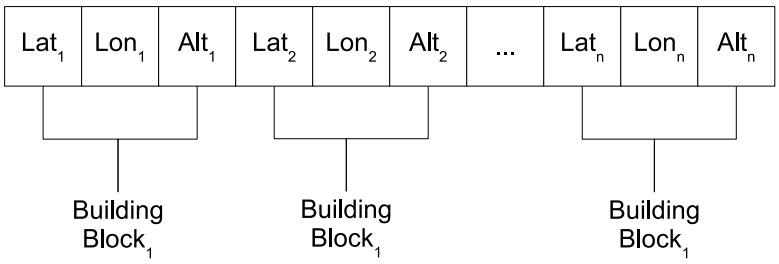


Figure 5.2: Chromosome Representation in NSGA-II for 3D Airspace Sectorization

Each chromosome contains a list of triplets. Each triplet has values of latitude, longitude, and altitude for a point, as shown in Figure 5.2. These triplets represent different information for different models: agent locations for iABM; partitioning points for KD-tree and SPBM; and Voronoi sites or vertical partitioning points for CVDM. The triplets in the chromosomes of all models are initialized randomly.

5.1.2 GA Operators

Some special operators manipulating the relationship between the chromosome (airspace partitioning points) and the flight trajectories may help GA to get convergence quicker. However, it has an impact on the computational cost. As found in Section 3.2.5, the computation cost of air traffic flow commonality searching is O(N). If it is introduced as a special operator in GA, the computational cost will be increased for each individual at each generation. Meanwhile, NSGA-II searches the better solutions according to their fitness function. If the relationship between chromosome and flight trajectories can be modelled into the fitness function, it is not necessary to introduce other operators with increased computational cost. Therefore, only the standard NSGA-II operators are used.

5.1.2.1 SBX Crossover

For recombination, I use the standard SBX operator (Deb *et al.*, 2007) with cross-over probability $p_c = 0.9$ and a distribution index of $\eta_c = 15$. The SBX operator stands for simulated binary crossover, whose search power is similar to that of the single-point crossover used in binary-coded GAs. SBX is found to be particularly useful in problems like the one at hand, in which multiple optimal solutions exist with a narrow global basin, or in problems where the lower and upper bounds of the global optimum are not known a priori.

5.1.2.2 Mutation

Eq. 5.1 details the mutation operator employed by the genetic algorithm (see Deb and Goyal (1996)).

$$y_i = x_i + (x_i^U - x_i^L)\delta_i \tag{5.1}$$

$$\delta_i = \begin{cases} (2r_i)^{1/(\eta_m+1)} - 1 & \text{if } r_i < 0.5 \\ 1 - |2r_i|^{1/(\eta_m+1)} & \text{otherwise} \end{cases} \tag{5.2}$$

where x_i is the value of the i^{th} parameter selected for mutation; y_i is the result of the mutation; x_i^L and x_i^U are the lower bound and the upper bound of x_i respectively, and r_i is a random number in [0,1]; η_m is a control parameter ($\eta_m = 20$ in my study). A boundary check mechanism is embedded inside NSGA-II, which ensures that the mutated triplets in a chromosome are always located within the given airspace.

5.1.3 Fitness Function

The first objective is modelled as the standard deviation of the sectors' task loads, as shown in Equation 5.3.

$$\begin{aligned} F(W) &= \sqrt{\tfrac{1}{K}\textstyle\sum_{k=1}^{K}(W_k - W_{avg})^2} \\ W_k &= \textstyle\sum_{i=1, x_{ij}\in S_k}^{N} |t_i(x_{ij})| \end{aligned} \tag{5.3}$$

W_k is the task load of sector S_k, which is derived from Equation 3.3. W_{avg} is the average task load of all sectors in the given airspace. K is the number of desired sectors, which is predefined as an input to the optimization. W_{avg} is calculated by Equation 5.4:

$$W_{avg} = \frac{\sum_{i=1,x_{ij}\in R}^{K} |t_i(x_{ij})|}{K} \tag{5.4}$$

The second objective is modelled as the average sector flight time of all sectors, as shown in Equation 5.5. SFT_k is the sector flight time of sector S_k, produced by Equation 3.4.

$$\begin{aligned} F(SFT) &= \frac{\sum_{k=1}^{K} SFT_k}{K} \\ SFT_k &= \frac{\sum_{i=1}^{N}\sum_{j=1,x_{ij}\in S_k,x_{ij+1}\in S_k}^{M} (time_{ij+1} - time_{ij})}{|T(t_i)|_{x_{ij}\in S_k}} \end{aligned} \tag{5.5}$$

The third objective is modelled as the minimum distance between traffic flow crossing points and sector boundaries among all sectors, as shown in Equation 5.6.

$$\begin{aligned} F(D) &= \min_{k=0}^{K} D_k \\ D_k &= \min_{i=1}^{P}(Dist(CP_i, Boundaries(S_k)))\ IF\ CP_i \in S_k \end{aligned} \tag{5.6}$$

Where D_i is calculated by Equation 3.6. $F(D)$ is the minimum distance among all sectors which excludes the distances between the crossing points and the airspace boundaries.

The first two objectives do not conflict directly, but there is implicit conflict between them. Equation 5.5 shows that to maximize the flight time within a sector, the sector should have more traffic hits from the same flights. That means a sector can achieve longer sector flight time by aligning its shape with the major traffic flows, to minimize the flow cut. This may cause sectors to encapsulate as much traffic flow as possible, but this will violate the objective of balancing ATC task load. Although chromosome design only considers the spatial information for airspace partitioning, the second objective works directly on the temporal information of flight trajectories, which intends to increase the flight sector time and then to align the air traffic flows with sector boundaries. Therefore, the relationship between the chromosome and flight trajectories are modelled in this fitness function. The third objective ensures

that the distances between crossing points of traffic flows and sector boundaries are as far as possible. Therefore it is a multi-objective problem, and there is no straightforward way apart from multi-objective optimization to generate a trade-off set for the given objectives.

As mentioned in Section 3.3, the Gap Filling rule can eliminate some sectors in some cases. This problem is solved by the optimization algorithm as it attempts to minimize the task load standard deviation, which is calculated with a fixed K. If a sector is eliminated, the task load standard deviation increases; therefore, in subsequent generations such solutions are eliminated by the Genetic Algorithm.

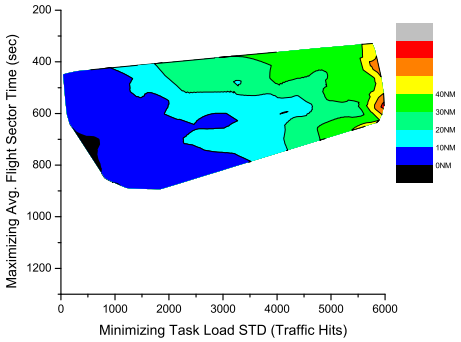
5.1.4 Experiment Design and Results

I evaluate the four 3D airspace sectorization models in a given airspace with the same traffic sample which was used to evaluate the agent based model. As mentioned in section 3.1.2, the airspace is a 5 degree by 5 degree en-route airspace with a 30000 ft vertical range (from 200FL to 500FL).

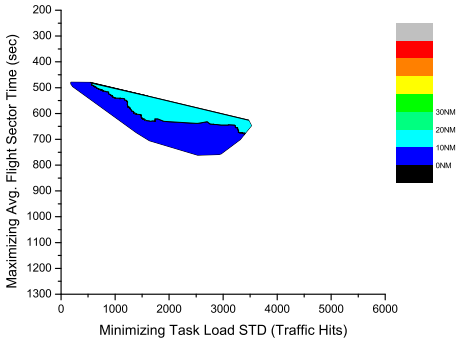
The objective is to divide the airspace into 10 sectors optimally according to the 3 objectives: minimizing standard deviation of task load across sectors, maximizing average sector flight time, and maximizing distance between sector boundaries and traffic flow crossing points. The input traffic contains a total of 23619 traffic hits, so the average task load is 2361.9 hits for 10 sectors.

NSGA-II was used to optimize the resulting airspace sectorizations, with all four models. For each model, I create 500 random individuals as the initial population and evolve them for 500 generations.

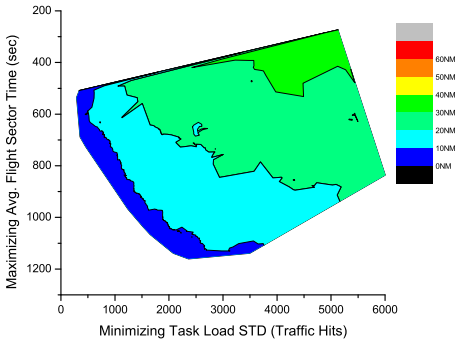
Each model was run 10 times with the same 10 seeds. In the experiments, 10 pre-determined seeds were used by each model. Each seed was used to initialise NSGA-II for each model in each run. Because the three objectives specified in this paper conflict, a Pareto Front (a set of non-dominated solutions) is generated instead of a single solution. All populations from the 10 runs generated by the same model were combined; the overall Pareto Front of the standard deviation of task load and



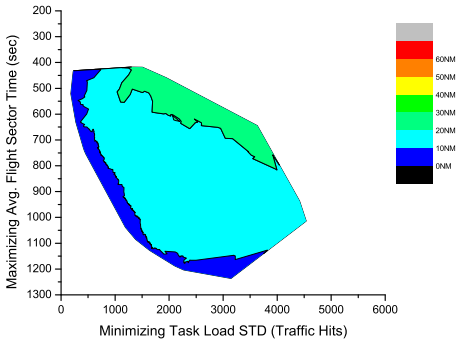
(a) Improved Agent Based Model



(b) KD-tree based Model



(c) Support Plane Bisection Model



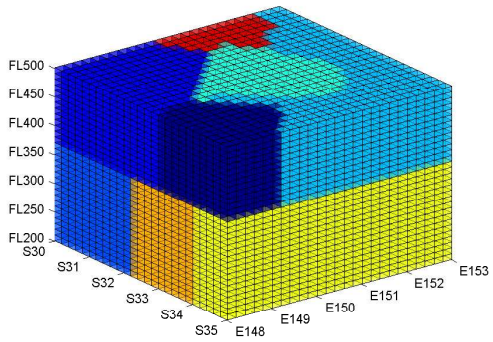
(d) Constrained Voronoi Diagram based Model

Figure 5.3: Pareto Fronts of 4 Airspace Sectorization Models

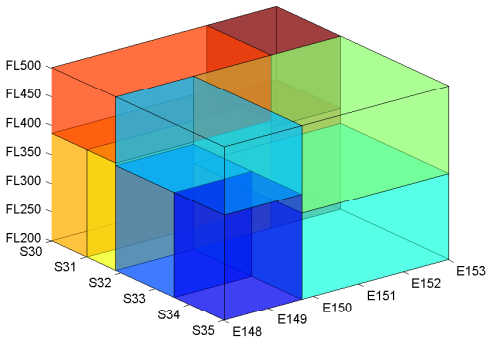
average flight sector time for each model is shown in Figure 5.3. The color is mapped by the values of the third objective (minimum distance between traffic flow crossing points and sector boundaries).

The plotted figures are the best airspace sectorization solutions, which are not dominated by other solutions in terms of the three objectives, found with each model. The values of the first two objectives (task load balance and sector flight time) are plotted along with the X and Y axes respectively, and the values of the third objective (distance between crossing points and sector boundary) are mapped into color codes. As illustrated in the figure, a solution is better in terms of the first two objectives when it is closer to the bottom-left corner. The red color of a solution indicates the farther distance between crossing points and sector boundaries.

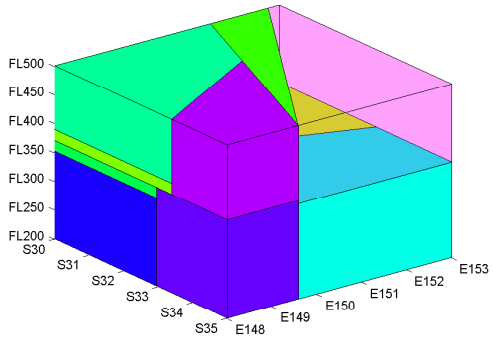
Figure 5.4 shows the examples with the minimum standard deviation of task



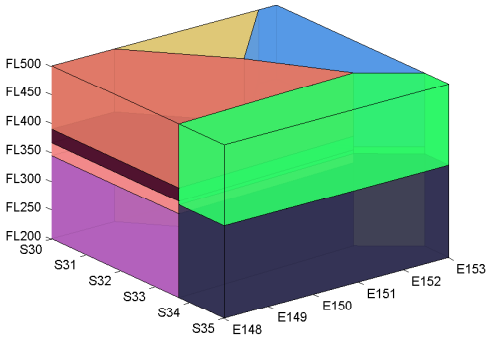
(a) iABM: F(W) = 40.22, F(SFT) = 426.65, F(D) = 0.08



(b) KD-tree based Model: F(W) = 182.78, F(SFT) = 478.81, F(D) = 0.17



(c) SPBM: F(W) = 290.79, F(SFT) = 536.03, F(D) = 0.25



(d) CVDM: F(W) = 181.72, F(SFT) = 520.75, F(D) = 0.30

Figure 5.4: Examples of Airspace Sectors (Minimum Standard Deviation of task load) Generated by 4 Airspace Sectorization Models

load among the possible solutions from each model. Although the air traffic data and airspace are the same for these four models, they result in strikingly different sectorizations. The next section discusses the results for each model in detail.

5.2 Analysis of Results

As illustrated above, all models are able to generate airspace sectors in order to optimize three objectives, but their outputs and performance are quite different from each other. In this section, I analyze the results for each model in terms of performance and computational cost.

5.2.1 Performance Analysis

The task load balancing is the most important criterion to evaluate a DAS approach. Therefore, a sensitivity analysis of all Pareto fronts from the proposed four models on the task load standard deviation is conducted. In order to provide a comparison on the task load standard deviation against the average task load, a coefficient of average task load (ε) is used in Equation 5.7 to analyse the sensitivity.

$$W_e = W_{avg} \times \varepsilon \tag{5.7}$$

The distributions of the Pareto solutions are calculated as illustrated in Figure 5.5.

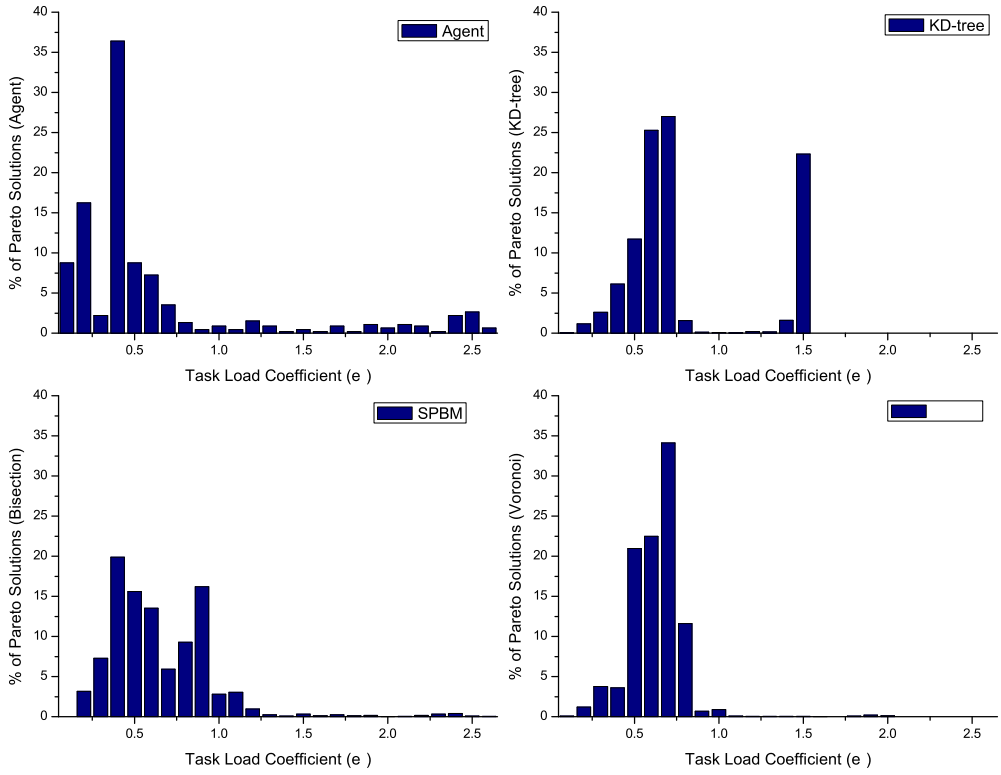
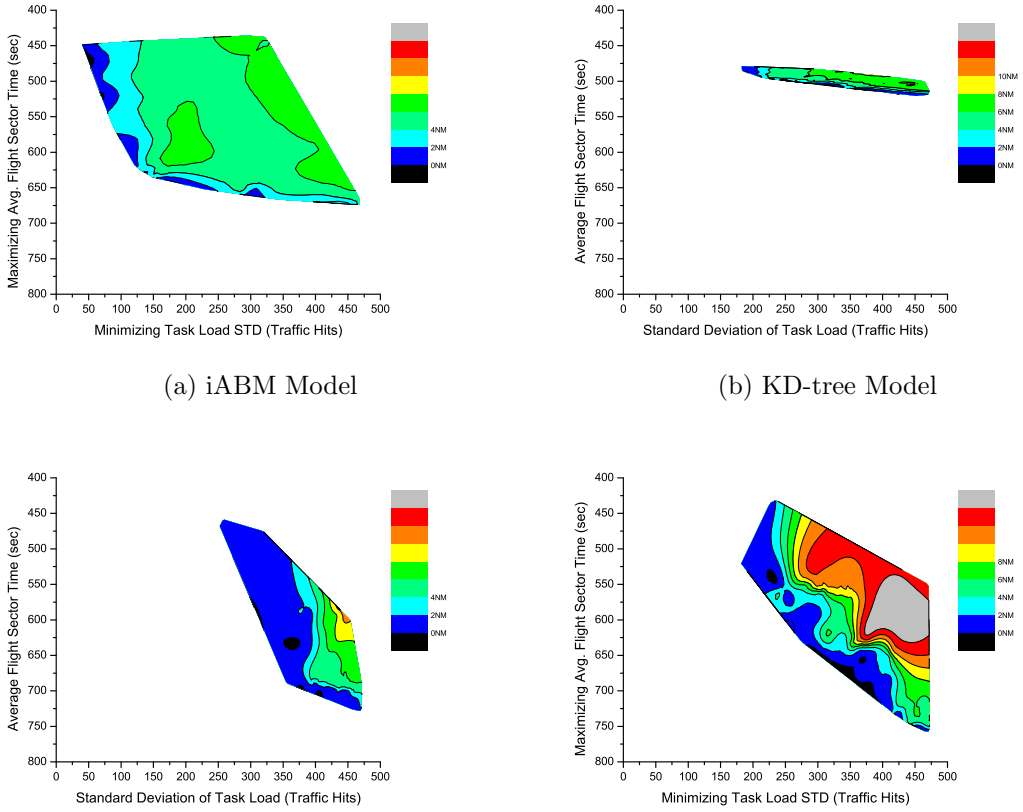


Figure 5.5: Sensitivity analysis of Pareto fronts for 4 airspace sectorization models based on the distributions of achieved task load standard deviation

More than half of the solutions from all models fall in the range where ε is less than 1.0. However, around 25% solutions of KD-tree based model with the task load



(a) iABM Model

(b) KD-tree Model

(c) Support Plane Bisection Model

(d) Constrained Voronoi Diagram based Model

Figure 5.6: The Subsets of Pareto Fronts by 4 3D Airspace Sectorization Methods

standard deviations are above average task load. The Agent, KD-tree, and CDVM have solutions on the Pareto fronts, whose task load standard deviations are below one-tenth of the average task load. But the solutions from SPBM with the lowest task load standard deviation is below two-tenth of the average task load. Therefore, only the subsets of the Pareto fronts, which have a task load standard deviation below two-tenths of the average task load, are considered in the performance analysis and comparisons from now on. As the average task load is 2361.9 and the coefficient (ε) is 0.2, the subsets of the Pareto front ($F(W) \leq 472.38$) from all models are plotted in Figure 5.6.

As shown in Figure 5.6a, the iABM can achieve the best task load balancing but has no good results for average sector flight time when being compared with the SPBM and CVDM. Both SPBM and CVDM have better performance on the sector

flight time, although CVDM is better than SPBM in terms of task load balancing. The KD-tree has some solutions with lower task load standard deviation but all solutions have a shorter average sector flight time, because the cuboid sectors are unable to tune the sectors' shapes according to the traffic flows; this resulted in the lower average sector flight time.

On the other hand, neither iABM nor KD-tree model can keep sector boundaries far away from the traffic flow crossing points. The minimum distances between crossing points and sector boundaries of their solutions are all below 8nm. The grid based approach adopted by iABM generates the fine boundaries of sectors limiting the capability to keep boundaries away from the traffic flow crossing points. The cuboid sectors produced by KD-tree Model have less flexibility to tune the boundaries in order to satisfy all three objectives at the same time. Both SPBM and CVDM are able to produce solutions where the distances between sector boundaries and traffic flow crossing points are more than 10nm, although the CVDM has the largest minimum distance (13.50nm) between the sector boundaries and traffic flow crossing points among all proposed models.

The lowest task load standard deviation (40.22) was achieved by iABM. There are two reasons for this. The first is that iABM only allows agents to group the cells within a predefined task load range, unless the Gap Filling Rule takes actions. The second is that agents are working on a gridded airspace, and small grids give the agent more flexibility to partition airspace. However, the average sector flight time of this model is lower than SPBM and CVDM because the Growth Rule prevents the agent from grouping more traffic flows when the task load limits are exceeded.

Table 5.1 lists the percentage of acceptable solutions with each model. iABM has the highest percentage of acceptable solutions, because the Growth Rule limits the task load variance; the other models have no specific guidance to limit the task load variance when sectorizing airspace, and so produce fewer acceptable solutions than iABM. Table 5.1 also shows the objective values of the best-performed examples, in terms of minimum task load variance, maximum average sector flight time, or maximum smallest distance between sector boundaries and traffic flow crossing

Models		iABM	KD-Tree	SPBM	CVDM
% of Acceptable Solutions		**25.00%**	1.21%	3.16%	1.29%
Example of Min(F(W))	**F(W)**	**40.22**	182.78	290.79	181.72
	F(SFT)	448.65	478.81	536.03	520.75
	F(D)	0.08	0.17	0.25	0.30
Example of Max(F(SFT))	F(W)	469.99	450.97	469.10	472.27
	F(SFT)	673.98	520.73	728.34	**757.81**
	F(D)	4.95	0.94	0.78	0.07
Example of Max(F(D))	F(W)	386.70	414.23	454.14	453.12
	F(SFT)	557.53	494.73	600.66	544.55
	F(D)	7.35	7.96	10.02	**13.50**

Table 5.1: Summary of Acceptable Solutions: the percentages of acceptable solutions based on task load standard deviation of 4 airspace sectorization models, and their best solutions for three objectives

points, generated by each model. iABM has the best solution for task load balancing, the CVDM has the longest sector flight time and the longest minimum distance between sector boundaries and traffic flow crossing points. The SPBM has better results on the maximum average sector flight time and larger minimum distance between sector boundaries and traffic flow crossing points than both iABM and the KD-tree model, however the minimum task load standard deviation achieved by it is worse than them.

Figure 5.7 shows the distributions of the acceptable solutions with each model. The left column shows the distributions according to task load balancing, the middle column shows distributions according to the average sector flight time, and the right column shows the distributions according to the minimum distance between sector boundaries and traffic flow crossing points.

Higher bars in the figures mean more solutions falling in the nominated objective range. Again it is seen that iABM performs best on task load balancing (larger percentage of solutions with lower task load variance) among the models. Both SPBM and CVDM based models have better performance when maximizing average sector flight time. They also both have some solutions falling into portion of the larger minimum distance between sector boundaries and crossing points compared to the other models. The KD-tree based model performs poorly on all objectives.

Table 5.2 shows the performance of the examples (shown in Figure 5.4) that achieve the best task load balance in each sector, with each model. The minimum

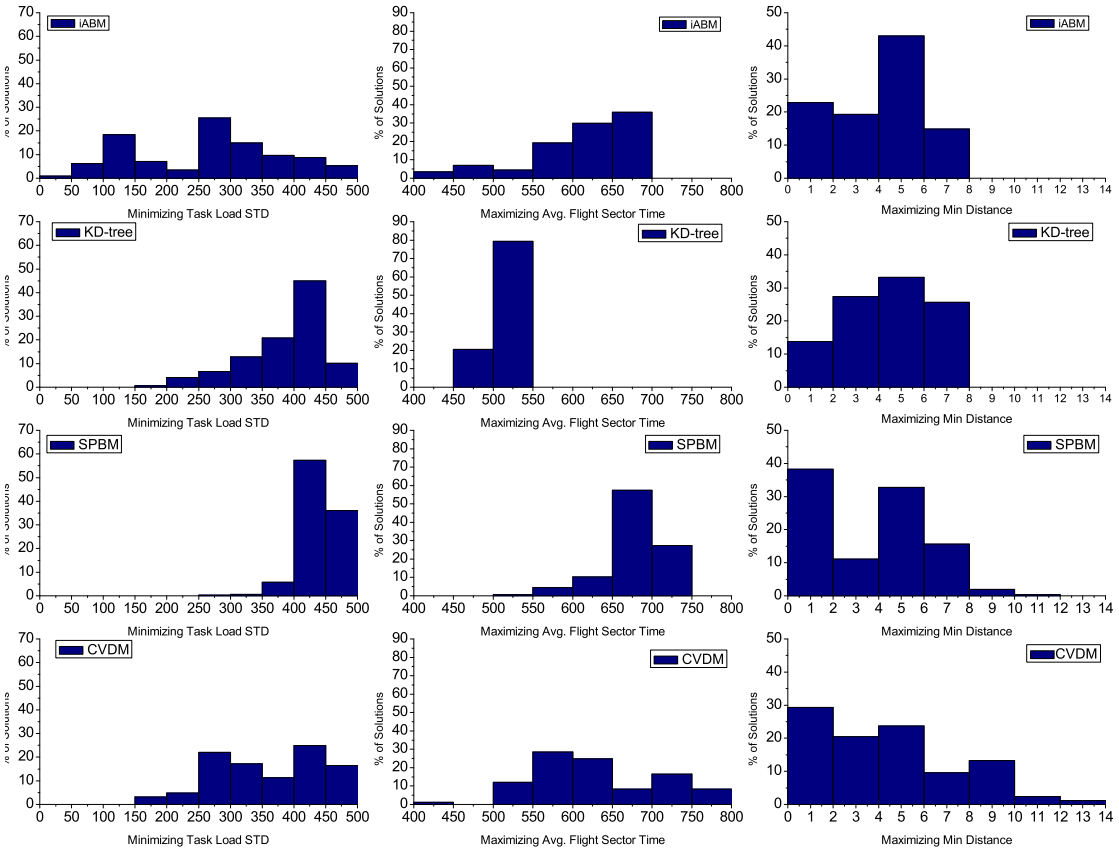


Figure 5.7: Distributions of all acceptable solutions based on task load standard deviation, average sector flight time, and the minimum distance between crossing points and sector boundaries for 4 airspace sectorization models

distance between sector boundaries and traffic flow crossing points does not exist for some sectors when there are no crossing points inside the sectors, therefore the value of the minimum distance is marked as N/A in the table. The task load variances of the sectors produced by iABM are smaller than with other models, and therefore the standard deviation of the task load with this model is the lowest. The other models are capable of achieving acceptable task load balance, as shown in the examples, but the variation is greater. However, the average sector flight time and the minimum distance between sector boundaries and traffic flow crossing points are not that high for all models.

Conversely, Table 5.3 shows the performance of the examples achieving the best average sector flight time, among the acceptable solutions produced with each

Sectors	3D DAS Models					
	iABM			KD-tree		
	Var. W	SFT_k	D_k	Var. W	SFT_k	D_k
S_1	-53.90	238.76	3.36	-205.90	333.40	7.82
S_2	-50.90	592.56	11.33	57.10	549.77	1.43
S_3	-39.90	627.57	38.79	195.10	525.41	48.78
S_4	-13.90	437.52	1.83	-26.90	188.31	31.12
S_5	-5.90	234.04	7.42	296.10	275.92	N/A
S_6	8.10	430.91	31.57	43.10	385.83	0.17
S_7	12.10	494.58	3.26	28.10	539.10	23.43
S_8	17.10	499.09	1.89	-284.90	759.88	9.22
S_9	52.10	332.20	8.42	-241.90	594.39	10.51
S_{10}	75.10	599.26	0.08	140.10	636.10	27.49
Objectives	**40.22**	448.65	0.08	**182.78**	478.81	0.17
	F(W)	**F(SFT)**	**F(D)**	**F(W)**	**F(SFT)**	**F(D)**

Sectors	3D DAS Models					
	SPBM			CVDM		
	Var. W	SFT_k	D_k	Var. W	SFT_k	D_k
S_1	163.10	224.78	1.12	-9.90	235.99	6.77
S_2	300.10	273.49	3.65	-117.90	288.93	1.92
S_3	-290.90	267.80	30.74	-249.90	581.28	0.30
S_4	-311.90	272.12	0.65	275.10	627.86	0.73
S_5	33.10	565.75	0.78	-157.90	418.48	16.26
S_6	-236.90	827.92	N/A	240.10	804.74	N/A
S_7	-270.90	825.39	0.25	74.10	395.03	0.92
S_8	557.10	557.77	0.47	-34.90	450.39	0.48
S_9	257.10	618.66	0.88	-219.90	531.07	1.37
S_{10}	-199.90	926.57	10.89	201.10	873.75	0.56
Objectives	**290.79**	536.03	0.25	**181.72**	520.75	0.30
	F(W)	**F(SFT)**	**F(D)**	**F(W)**	**F(SFT)**	**F(D)**

Table 5.2: Task load variance ($Var.W = W_k - W_{avg}$ and $W_{avg} = 2361.9$) , average sector flight time (SFT_k) and minimum distance between sector boundaries and traffic flow crossing points (D_k) of each sector of the examples with best task load balance produced by 4 airspace sectorization models

model. All models can achieve longer flight times, but with higher task load variance and smaller minimum between sector boundaries and traffic flow crossing points.

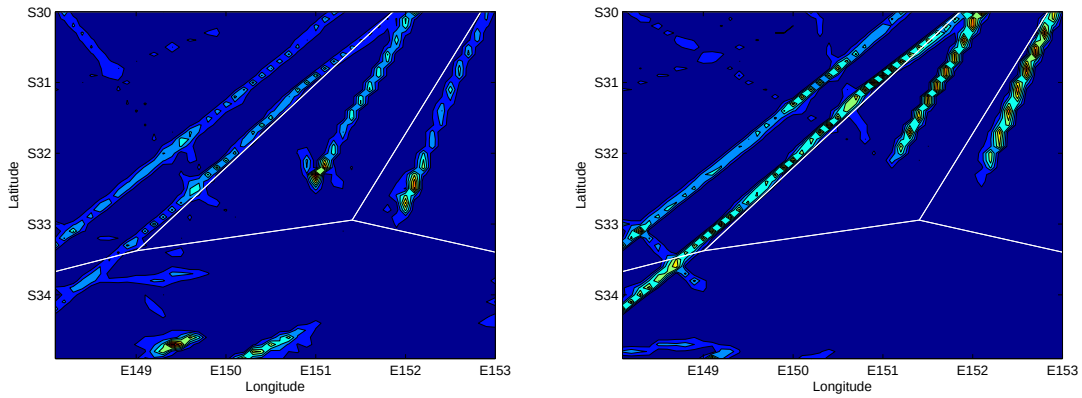
The longer sector flight time is achieved by alignment between sectors and traffic flows and minimizing the traffic flow cuts by sectors. Figure 5.8 shows the example of some sectors produced by the CVDM (Task load Standard Deviation = 472.27 and Average Sector Flight Time = 757.81, and Minimum Distance between Sector Boundaries and Traffic flow Crossing Points = 0.07) as listed in Table 5.3. It demonstrates that sectors encapsulate and align with the major air traffic flows in both the low and high airspace. Only one major air traffic flow is segmented by sectors in order to balance the task load. Therefore, the number of traffic flow cuts is minimized.

Finally, the examples with the largest minimum distance between sector bound-

Sectors	3D DAS Models					
	iABM			KD-tree		
	Var. W	SFT_k	D_k	Var. W	SFT_k	D_k
S_1	-631.90	320.37	31.89	25.10	356.27	8.81
S_2	-522.90	227.98	7.42	892.10	682.66	0.94
S_3	-281.90	1006.45	6.00	-136.90	606.82	75.64
S_4	-179.90	770.12	10.80	299.10	220.52	35.45
S_5	-127.90	736.48	18.61	-343.90	254.37	N/A
S_6	-93.90	756.00	13.16	-515.90	407.21	4.81
S_7	36.10	1057.94	4.95	-180.90	491.95	22.28
S_8	370.10	1037.47	N/A	-637.90	783.64	4.88
S_9	373.10	416.50	6.65	49.10	669.72	23.78
S_{10}	1059.10	410.52	6.74	550.10	734.12	40.75
Objectives	469.99	**673.98**	4.95	450.97	**520.73**	0.94
	F(W)	**F(SFT)**	**F(D)**	**F(W)**	**F(SFT)**	**F(D)**

Sectors	3D DAS Models					
	SPBM			CVDM		
	Var. W	SFT_k	D_k	Var. W	SFT_k	D_k
S_1	223.10	994.23	9.29	452.10	456.32	1.04
S_2	-447.90	941.31	13.62	163.10	714.62	0.11
S_3	-342.90	484.56	4.18	310.10	229.68	3.78
S_4	162.10	1009.60	13.65	355.10	335.43	0.24
S_5	-626.90	1269.51	14.46	-401.90	753.85	N/A
S_6	-412.90	512.89	2.77	-1044.90	1067.84	N/A
S_7	734.10	488.84	12.06	-491.90	935.00	0.28
S_8	518.10	230.40	1.74	33.10	970.95	0.64
S_9	-359.90	984.59	N/A	106.10	1073.04	0.07
S_{10}	553.10	367.44	0.78	519.10	1041.33	0.32
Objectives	469.10	**728.34**	0.78	472.27	**757.81**	0.07
	F(W)	**F(SFT)**	**F(D)**	**F(W)**	**F(SFT)**	**F(D)**

Table 5.3: Task load variance ($Var.W = W_k - W_{avg}$ and $W_{avg} = 2361.9$) , average sector flight time (SFT_k) and minimum distance between sector boundaries and traffic flow crossing points (D_k) of each sector of the examples with longest average sector flight time produced by 4 airspace sectorization models



(a) Low Level Sectors and Major Traffic Flows (FL200-FL370)

(b) High Level Sectors and Major Traffic Flows (FL370-FL500)

Figure 5.8: Alignment between sectors and traffic flows: an example with the longest average sector flight time among the acceptable solutions produced by CVDM.

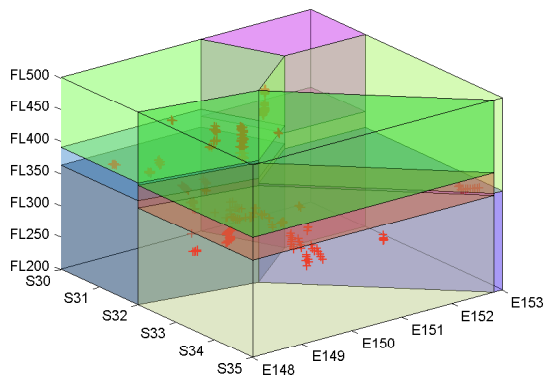
Sectors	3D DAS Models					
	iABM			KD-tree		
	Var. W	SFT_k	D_k	Var. W	SFT_k	D_k
S_1	-605.90	675.38	16.40	-332.90	320.37	7.96
S_2	-402.90	489.75	15.55	-402.90	511.04	7.96
S_3	-371.90	320.97	18.25	325.10	541.01	49.36
S_4	-185.90	615.85	9.82	-332.90	166.77	14.49
S_5	-48.90	913.03	300.00	-520.90	210.00	134.12
S_6	66.10	543.58	18.26	497.10	422.51	8.85
S_7	230.10	283.80	13.20	-22.90	527.59	25.65
S_8	256.10	347.52	7.35	-238.90	786.30	25.85
S_9	305.10	571.50	12.26	269.10	674.62	23.20
S_{10}	758.10	813.91	11.11	760.10	787.06	40.18
Objectives	386.70	557.53	**7.35**	414.23	494.73	**7.96**
	F(W)	**F(SFT)**	**F(D)**	**F(W)**	**F(SFT)**	**F(D)**

Sectors	3D DAS Models					
	SPBM			CVDM		
	Var. W	SFT_k	D_k	Var. W	SFT_k	D_k
S_1	-449.90	807.89	10.02	-294.90	543.95	62.72
S_2	-260.90	851.76	12.97	268.10	830.53	17.45
S_3	117.10	336.52	11.08	-495.90	227.56	19.49
S_4	-377.90	901.82	16.70	-441.90	411.43	N/A
S_5	-72.90	995.22	10.08	-130.90	527.01	22.83
S_6	610.10	422.56	28.99	256.10	561.00	56.02
S_7	883.10	479.56	22.47	541.10	268.80	13.50
S_8	-654.90	226.59	65.28	933.10	349.29	62.80
S_9	-14.90	442.83	N/A	-264.90	806.54	23.30
S_{10}	221.10	541.89	36.95	-369.90	919.38	28.52
Objectives	454.14	600.66	**10.02**	453.12	544.55	**13.50**
	F(W)	**F(SFT)**	**F(D)**	**F(W)**	**F(SFT)**	**F(D)**

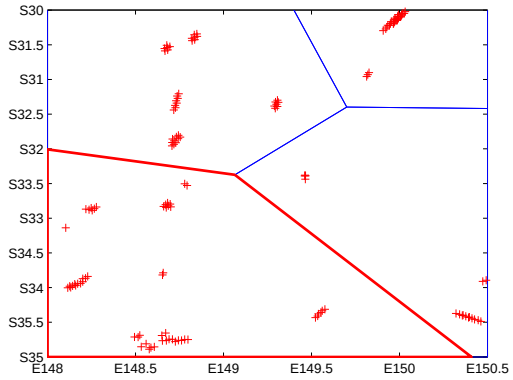
Table 5.4: Task load variance ($Var.W = W_k - W_{avg}$ and $W_{avg} = 2361.9$) , average sector flight time (SFT_k) and minimum distance between sector boundaries and traffic flow crossing points (D_k) of each sector of the examples with the largest minimum distance between sector boundaries and traffic flow crossing points produced by 4 airspace Ssectorization models

aries and traffic flow crossing points, among the acceptable solutions are listed in the Table 5.4. The largest minimum distance (13.5nm) is achieved by the CVDM. iABM and KD-tree models can't generate sectorizations which have the minimum distance larger than 8nm.

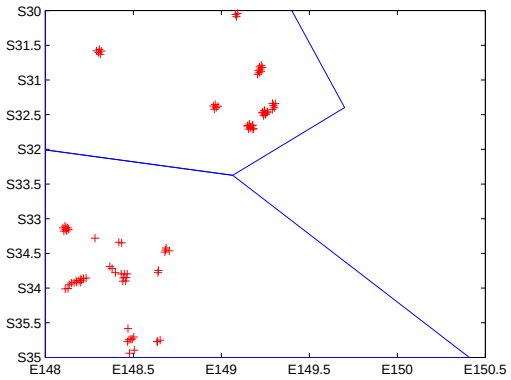
Figure 5.9 shows an example with the largest minimum distance between sector boundaries and traffic flow crossing points generated by the CVDM. In this case, the minimum distance is 13.50nm among all sectors, which happens in a lower sector as highlighted in Figure 5.9b. The task load standard deviation of the shown example is 453.12. However, its average sector flight time (544.55) is not as high as the previous example. For this example, the traffic flows are segmented by sectors in order to keep their boundaries away from the crossing points.



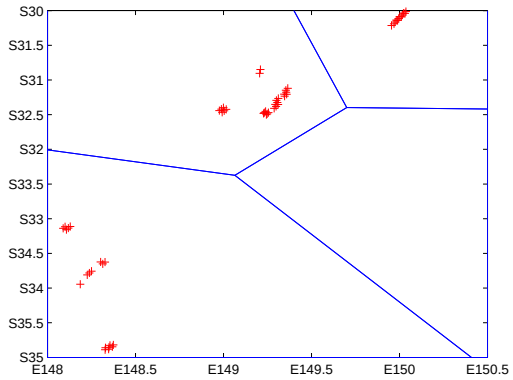
(a) Sectorization with Largest F(D)



(b) Lower Sectors



(c) Middle Sectors



(d) Upper Sectors

Figure 5.9: Distance between sector boundaries and traffic flow crossing points: an example with the largest minimum distance between sector boundaries and traffic flow crossing points produced by CVDM.

In summary, iABM performs best on task load balancing, but cannot satisfy the convexity constraint on sector shape. Both SPBM and CVDM have better performance on average sector flight time, but the CVDM has better performance on maximizing the minimum distance between sector boundaries and traffic flow crossing points. Although they can't achieve the lower task load variance of iABM, both can achieve acceptable task load balance. The KD-tree based model performed poorly, because of its geometry limitation which also limits its value in a real operational environment. The results also show that the three objectives adopted in this 3D DAS optimization conflict with each other. The trade-off between the three objectives needs to be considered by decision makers depending on the operational

conditions.

5.2.2 Computational Cost Comparison

DAS is intended to be used in a real time operational environment, to adjust the airspace sectorization quickly in order to accommodate fluctuating air traffic demand. An efficient DAS method is necessary to satisfy this requirement. In this section, I compare the computational cost of the four proposed models based on the execution time.

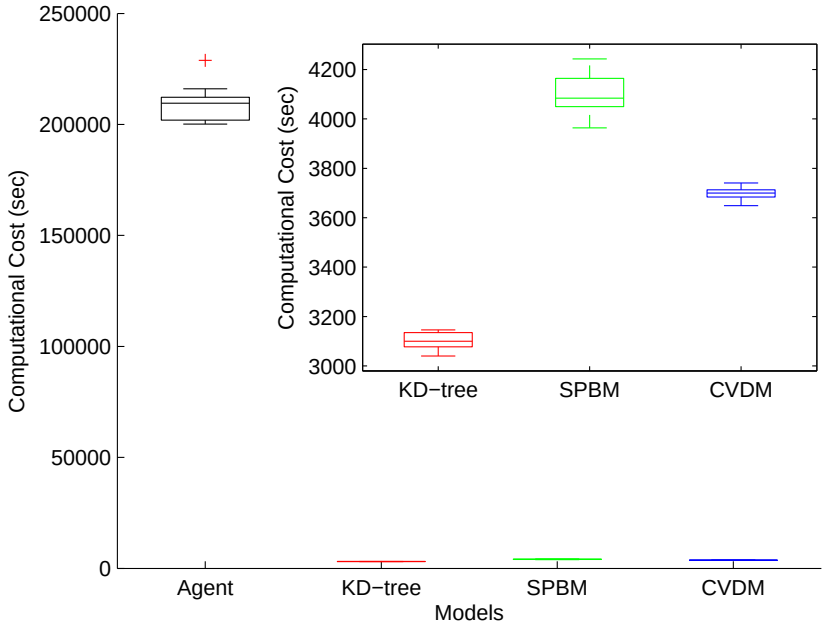


Figure 5.10: Box Chart of the Computation Costs by 4 Models: the KD-tree, SPBM, and CVDM The figure depicts the averages, the medians, the interquartile range (between 25th and 75th percentiles), the 5th and 95th percentiles, and the minimum and maximum costs of 10 runs for each model.

The computational cost of the optimizations was measured 10 times with all four models, and the average cost as well as the minimum and maximum cost were calculated for each model. The computational cost measured here is the total cost to finish a run by the proposed DAS models and NSGA-II, which includes the airspace partitioning, fitness calculations, and GA operators. The computation costs of the

proposed models are shown in Figure 5.10 as box charts. The average time consumed by iABM is 209321 seconds, which is the highest among all models. The KD-tree based model had the lowest average computational cost of 3097.40 seconds. The SPBM (average computational cost of 4096.78 seconds) took around 10% longer time than the CVDM (average computational cost is 3696.60 seconds) because it needs to find the two points closest to the middle of a space when bisecting the space, which requires some extra computational cost. The computational cost of iABM is not at the same magnitude as other models (almost 50 times of the SPBM) because of the complex agent rules associated with the model. Therefore, the comparison on the computational costs between models excluding iABM is zoomed in in Figure 5.10. The difference between the minimum and maximum computational cost of SPBM is also larger than the other two models.

5.3 Chapter Summary

In this chapter, I applied multi-objective optimization using NSGA-II with the proposed four DAS models to optimize the airspace sectorization in terms of ATC task load balancing, the maximum average sector flight time, and the minimum distance between sector boundaries and traffic flow crossing points; and used the Pareto concept of optimality for performance analysis. iABM has the best performance on task load balancing, but it can't satisfy the convexity constraint on sector shape. Both SPBM and CVDM have better performance on the average sector flight time. The CVDM is better on the minimum distance between sector boundaries and traffic flow crossing points than the SPBM . Although they can't achieve the as low a task load variance as iABM, both of them achieved expected task load balance. KD-tree based model shows poor performance because of its geometric limitation, which also limits its use in an operational environment. This chapter shows that there might not be a silver bullet for the dynamic airspace sectorization problem. Different approaches each have advantages and limitations. What is required is to adapt them to specific needs based on airspace constraints and user requirements.

This chapter shows that the SPBM and CVDM both have the better overall performance and efficiency to satisfy the specific objectives for an airspace section with one air traffic scenario. It is necessary to investigate both models further for a national airspace with different air traffic from both simulation and reality in an ATM environment. Therefore, an evaluation ATM environment for the proposed models is presented in the next chapter.

Chapter 6

Air Traffic Monitoring and Advisory System

As the needs for evaluating the proposed DAS models for a national airspace with simulated or field data arising, an air traffic monitoring and advisory system, called TOP-LAT (Trajectory Optimization and Prediction of Live Air Traffic)[1,2], is developed and presented in this chapter. In addition to evaluating the advanced ATM concepts, TOP-LAT provides a real time full situation awareness of airspace (control zones, sectors, terminal airspace, special use airspace). This includes air traffic flow, spatial aviation emission, airspace complexity metrics, airspace safety indicators, and weather information. Based on these, it is also able to provide advices on ATM activity to different users.

TOP-LAT is a distributed system, based on a client-server architecture. The server synthesizes and integrates all this information, and distributes timely, ac-

[1]TOP-LAT is developed by the ATM research team at UNSW@ADFA. My contributions include system architecture design and implementation, Flight Aerodynamic Module (M3), Fuel and Emission Module(M4), Airspace Complexity Module (M6), Airspace Safety Module (M7), Database Design, Graphical User Interfaces and visualizations. I was also involved in the works of other modules in the TOP-LAT.

[2]The TOP-LAT system is protected by New South Innovation (NSi), which is UNSW Commercial arm. The data received from AirServices Australia are under confidentiality agreement with AirServices Australia and therefore are not presented in this thesis. BADA is offered by EUROCONTROL under a license to UNSW. The business rules are protected by the confidentiality of the data.

curate, and reliable information (traffic flows, complexity, safety, and emissions) to relevant clients that suit the operational and research needs of different users (air traffic controllers, airline operation centers, air traffic flow management centers, etc.). TOP-LAT is built using a modular approach, enabling various advanced ATM concepts to be incorporated easily for evaluation and analysis; it also enables users to define their own modules for specific requirements. Figure 6.1 shows the distributed client-server architecture of TOP-LAT and the information sharing and data communication between elements in the system.

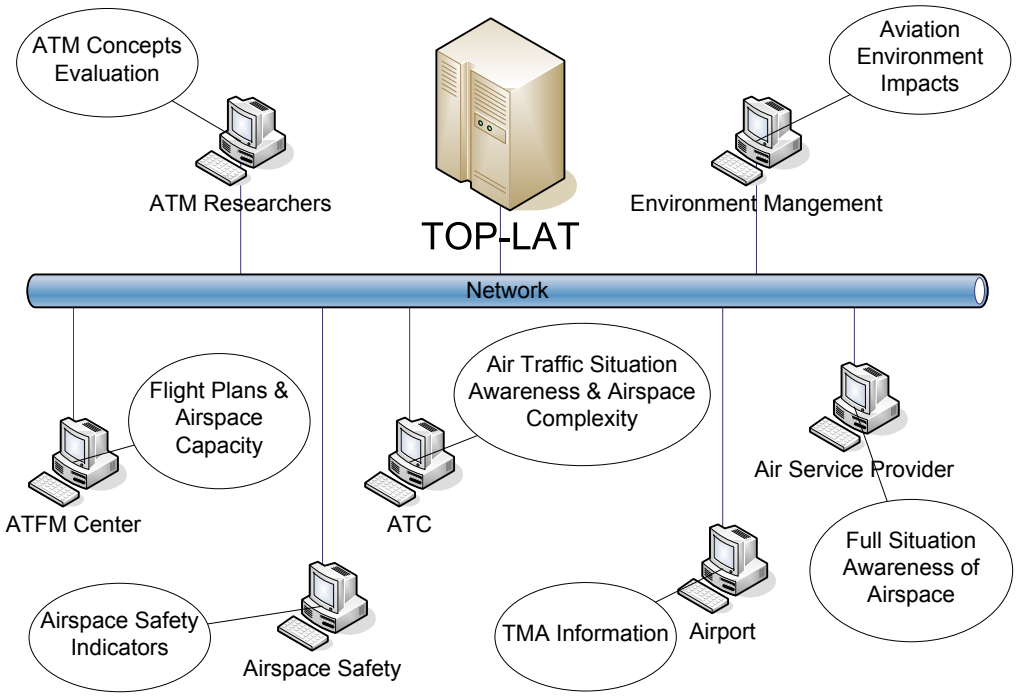


Figure 6.1: Distributed and Server-Client Architecture of TOP-LAT

The structure of the chapter is as follows. The system architecture and the methodologies adopted by TOP-LAT are described in the next section: this section depicts the details of the modular approach and modules, the function of models in each module, the database, and the user interfaces. The software implementation and validation are presented in Section 6.2.

6.1 Design and Development

In this section, I describe the system level architecture and models for the system.

I use a modular approach, in which each module and the models it employs are independent of other modules and models. This approach makes the system flexible, and able to be configured by users who can deploy their own models in each module to meet their special requirements. For example, users may use the Boeing Emission Model (Baughcum *et al.*, 1996) rather than the DLR Model (Schmitt and Brunner, 1997) to estimate aviation emission.

6.1.1 Architecture of the system

Figure 6.2 illustrates the top level architecture of the system.

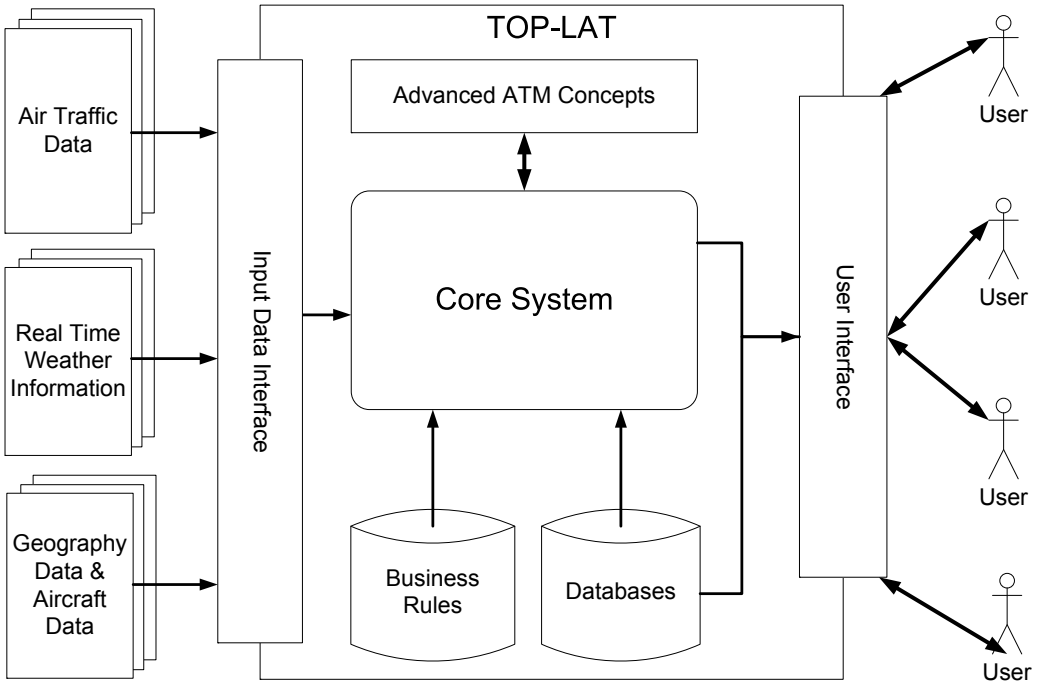


Figure 6.2: The Architecture of An Integrated Real Time Air Traffic Flow Information System

An Input Data Interface receives the incoming real time air traffic data, deal-

ing with time synchronization and data packet management. It also receives real time weather information, including the wind grids and the Significant Meteorological Information (SIGMET). This interface also gets data from airspace and geographic databases, as well as aircraft aerodynamic databases and parameters set from BADA (Nuic, 2004). This ensures the consistency of air traffic data (their geographic positions and verification of aerodynamic parameters). Simulated air traffic data can also be fed into the system instead of the field data.

These data packets are then forwarded to the Core System. This system outputs information such as aviation emission, air traffic flow, airspace safety, and complexity measurements, which are stored in Databases and also directed to the User Interface.

As shown in the figure, the Core System has a link with the modeled advanced ATM concepts, such as DAS. It provides the processed air traffic data to ATM concepts and gets the outputs. This provides an evaluation environment for the advanced ATM concepts with the field data or the simulated data.

6.1.2 Core System

The components of the core system are illustrated in Figure 6.3.

The Core System consists of a kernel, and seven sub-modules with specific functional models. The kernel assigns different tasks to the different modules and coordinates the work between them, ensuring the reliability, consistency, atomicity and integrity of data. It also interacts with Databases for storing and retrieval of required information. The kernel also interfaces with a knowledge base of Business rules, for validation and verification of air traffic and flight data.

Figure 6.3 shows the seven functional modules in the core system. The kernel manages the interaction and data management among these modules, and forwards intermediate results to target modules for further processing, or to the databases for storing, or to the user interface. We now describe each of the modules in the following sections, according to the sequence of the process flow.

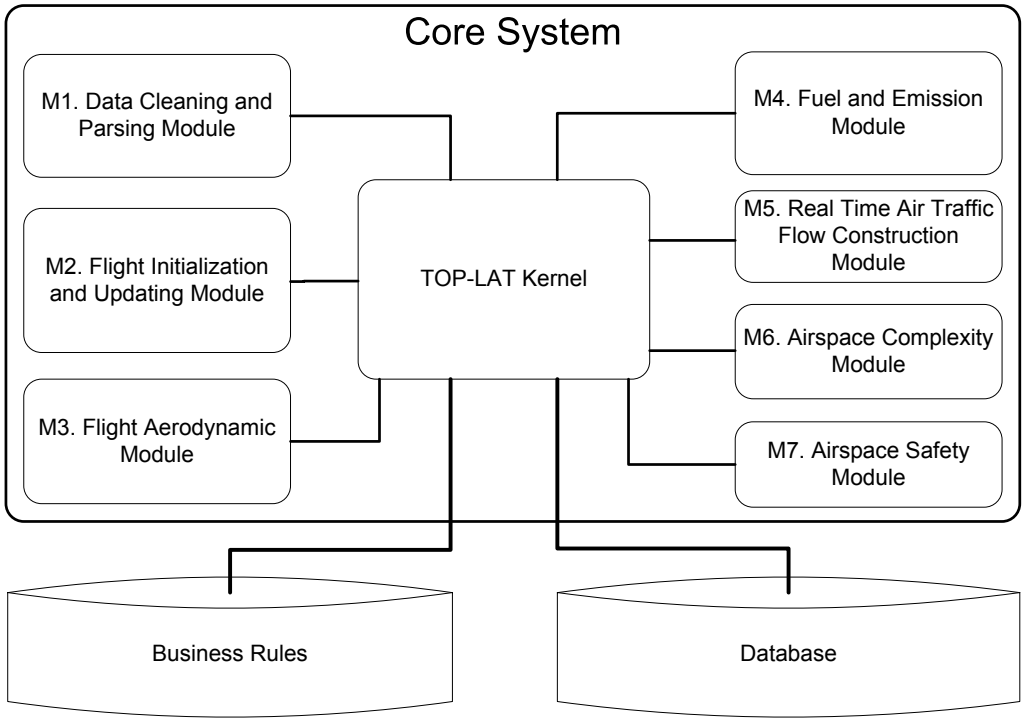


Figure 6.3: The Modules in the Core System

6.1.2.1 Data Cleaning and Parser Module

TOP-LAT processes live air traffic data, received from AirServices Australia. It is based on a generic format; any data compatible to this format provided by any air navigation service provider can be used in TOP-LAT.

The real time air traffic data come from several sources (ground based radar, satellite based navigation, in-flight GPS etc.) and may contain inconsistencies or errors due to network transmission, noise or attenuation. Thus it is necessary to pre-process the data before further activities. This is the function of the Data Cleaning and Parser Module (M1). Figure 6.4 illustrates the architecture of the module.

When messages containing real time air traffic data arrive, the kernel sends the data to the Data Cleaning and Parser module M1. The Data Cleaning Model (M1.1) cleans the inconsistency and other transmission errors from the incoming data. The Data Parser Model (M1.2) decomposes the cleaned message into small pieces, each corresponding to partial information about a flight.

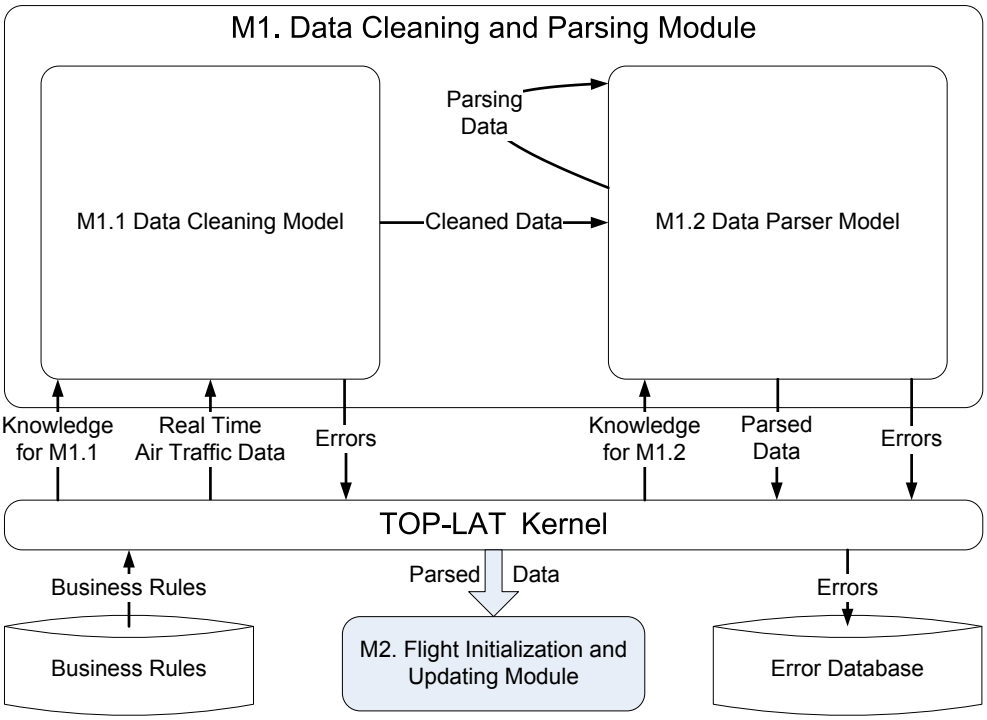


Figure 6.4: The Data Cleaning and Parser Module (M1)

If incoming data is successfully processed, the parsed information is returned to the kernel. The kernel forwards this information to the Flight Initialization and Updating Module (M2).

If the data can't be cleaned or parsed, the module ignores the data and records it in an Error Database. The errors that are recorded in this database help us to evolve M1.1 and M1.2 to handle more kinds of data noise and message formats.

6.1.2.2 Flight initialization and updating module

A typical flight trajectory contains 9 phases: taxi-out, take-off, climbing out, climbing, cruise, descent, approach, landing and taxi-in. This is illustrated in Figure 6.5.

The Flight Initialization and Updating Module (M2) uses the real time flight data that is output from M1 to construct and update flight trajectories. This involves constructing a trajectory for a new flight as it takes off within the airspace

or enters the airspace from elsewhere, updating the trajectory as air traffic data is received for the flight, and finalizing the trajectory as the flight lands or leaves the airspace. It may also involve estimating some critical aspects of the trajectory if they are not provided in the data: Standard Instrument Departure route (SID) and Standard Terminal Arrival Route (STAR) (the routes that a flight normally takes below the altitude of 10,000 feet); and Top of Climb (TOC) and Top of Descent (TOD) (these play a crucial role in deciding the flight status and cataloguing aviation emission according to the flight phases).

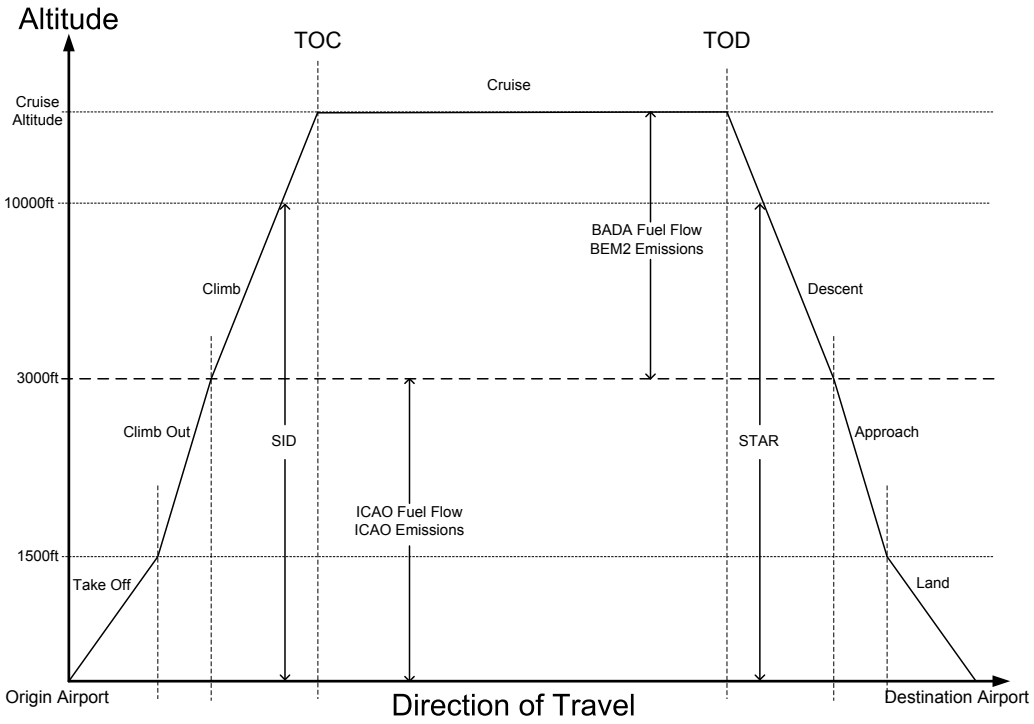


Figure 6.5: A Typical Flight Trajectory

There are six models undertaking different tasks in this module:

- Flight Initialization Model (M2.1)
- Flight Update Model (M2.2)
- Flight Completion Model (M2.3)
- Flight TOC and TOD Estimation Model (M2.4)

- Flight SID and STAR Estimation Model (M2.5)
- Flight Runway Estimation Model (M2.6)

The first three models are triggered by the content of the incoming data received from the kernel. The last three models start working when some conditions are satisfied. Figure 6.6 shows the interactions and relationships among the six models.

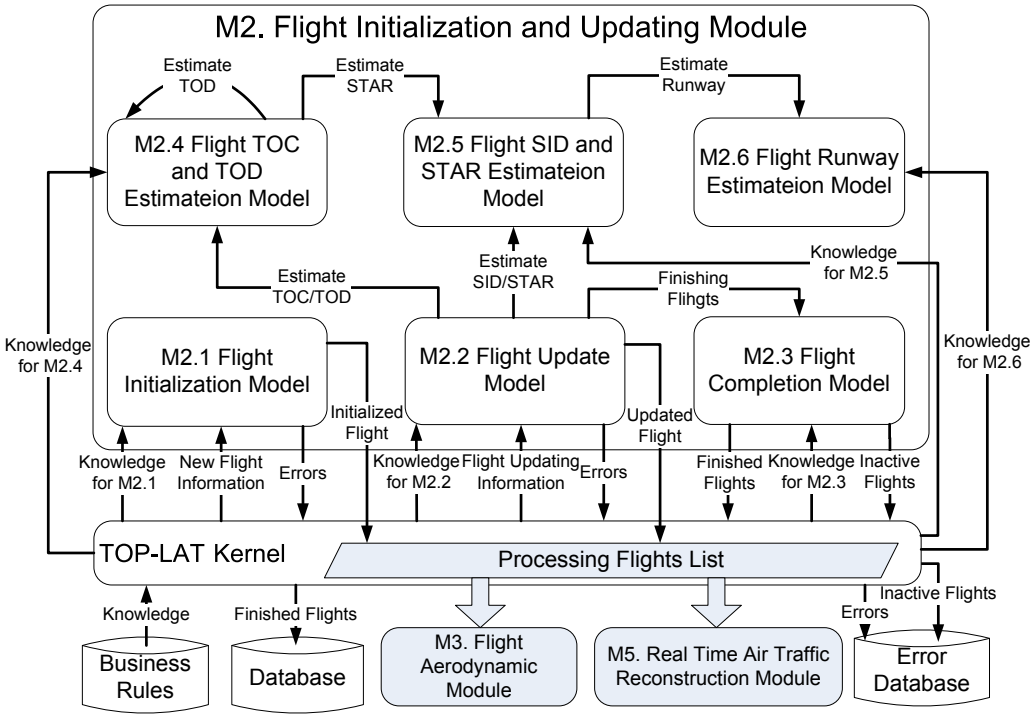


Figure 6.6: The Real Time Flight Trajectory Constructor Module (M2)

For a new flight, the Flight Initialization Model (M2.1) initializes this flight according to a set of business rules based on the message received. When a flight is created, basic information such as flight call sign, registration, origin, destination, departure plan time, and aircraft type. are assigned to the flight. The initialized flights are stored in a Processing Flights List in the kernel.

The system uses BADA model (Nuic, 2004) to model flight aerodynamics as well as to estimate thrusts and fuel flows. The BADA model includes the performance parameters of 89 aircraft types, which are then mapped to another 275 aircraft using

a mapping table as per Sutkus *et al.* (2001). Some flights, such as small recreation flights and helicopters, are not processed by the system as they are not in the BADA database.

If the air traffic data contain information for an existing flight in the processing flights list, the Flight Update Model (M2.2) uses the new flight information to update the flight's departure time, destination, trajectory and other parameters.

The Flight Completion Model (M2.3) determines when a flight is completed. There are several conditions which indicate that the flight is terminated: the flight has a radar signature which has a wheel-on-runway indicator (we know it has landed); information for the flight hasn't been received for a period which is longer than the time spent by the flight to travel from the last received radar signature to its destination at its minimum speed plus a threshold (we assume it has landed); information for the flight hasn't been received for more than 2 hours.

The initialized, updated and terminated flights are returned by M2 to the kernel. The completed flights are stored into the database by the kernel. Active flights (initialized and updated flights) are stored in a Processing Flights List, which is sent by the kernel to other modules, such as Flight Aerodynamic Model (M3) and Real Time Air Traffic Flow Construction Module (M5), for further processing.

The Flight Update Model (M2.2) triggers other models to estimate TOC and TOD (M2.4), SID and STAR (M2.5), and runways (M2.6), when conditions associated with business rules are satisfied.

When the first radar signature of a flight arrives, Flight TOC and TOD Estimation Model (M2.4) starts estimating TOC. If TOC is fixed, it then starts estimating TOD. The method of estimating a flight's TOC is to simulate the flight climbing from its origin to its cruising altitude along its route, including SID estimated from M2.5, fixing the TOC position when it reaches its cruising altitude. Similarly, TOD is estimated by simulating the flight climbing from its destination to its cruising altitude backward along its route, including STAR estimated from M2.5, fixing the TOD position when it reaches the cruising altitude. The condition to determine

TOC or TOD is that there is a radar signature after the estimated TOC or TOD. Sometimes, if a flight's destination is changed because of the weather, airport capacity or any other reason, the TOD of the flight is re-calculated.

The Flight SID and STAR Estimation Model (M2.5) is implemented to estimate both SID and STAR. A flight's SID is only available if its origin airport is within Australia, and STAR is only available when the destination airport is within Australia.

This model uses frequency matrices of SID and STAR that we have built, for each Australian airport for each month, using historical data. The SID data comprises frequency matrices of SID way points, destinations, operators, and aircraft types. The frequency matrix of way points has the way points around a local airport area, which is a circle centered on the airport with a radius of 100nm. STAR data comprises frequency matrices of STAR way points, origins, operators, and aircraft types. The values in each of these matrices are the frequencies of a way point, an origin, a destination, an operator or an aircraft type associated with a SID or a STAR in historical flights. SID and STAR frequency matrices for each airport, in each month, for each sub-matrix, are stored in the database.

To illustrate how the matrices are built, suppose a way point W appears N times in historical flight trajectories in a month. If P flights took SID SID_1 and Q flights took SID SID_2, the way point W's frequency of SID_1 is P and its frequency of SID_2 is Q in the SID way points matrix for this month. Another example is that an operator O appears M times in historical data in a month. If J of this operator's flights took STAR $STAR_1$ and K of this operator's flights took STAR $STAR_2$, the operator O's frequency of $STAR_1$ is J and its frequency of $STAR_2$ is K in the STAR operators matrix for this month. The other frequency matrices are produced in the same way.

From this SID and STAR frequency matrices a set of summary frequency matrices of SID and STAR are calculated by Equation 6.1 for way points, origin/destination, operators, and aircraft types respectively during the system run-

ning.

$$F_{s,j} = \sum_{m=1}^{M} F_{s,j,m}(1 - \frac{1}{e^m}) \tag{6.1}$$

In Equation 6.1, $F_{s,m,j}$ is the frequency of a way point, origin, destination, aircraft, operator associated with a SID(s) or a STAR(s) in a month m. j is the index of an object (waypoint, origin, destination, aircraft, and operator) whose frequency is being calculated. m is an index of the past months which starts from 1 for the first month to M for the latest month. $(1 - \frac{1}{e^m})$ is a weight factor for each month. e is an irrational constant. The latest month has the highest weight, while the earlier month has a lower weight. After the summary matrices are produced, we normalize the values in these matrices to get a set of probability matrices by Equation 6.2.

$$P_{s,j} = \frac{F_{s,j}}{\sum_{i=1}^{N} F_{i,j}} \quad s \in [1, N] \tag{6.2}$$

In Equation 6.2, s and i are the indices of all SIDs or STARs at an airport and N is the total number of the SIDs or STARs at the airport. j is the index of an object. These SID and STAR probability matrices are used by M2.5 to estimate SID and STAR.

The idea behind this approach is that flights with same flight route, origin, destination, aircraft and operator are most likely to take the same SID or STAR at the same airport. We introduce a deviation check method to improve the accuracy of the estimation. The deviation check is based on the distance between a radar signature and a SID or a STAR. A longer distance means a bigger deviation. We calculate distances between the flight's real trajectory (radar signatures) and each SID or STAR every time a new radar signature arrives. Then the maximum distance (D_{max}) between the radar signature and all SID or STAR is calculated. The distance matrices are translated to probability matrices.

$$PD_s = \frac{D_{max} - D_s}{\sum_{i=1}^{N} D_{max} - D_i} s \in [1, N] \tag{6.3}$$

In Equation 6.3, PD_s is the probability of a SID or a STAR obtained from distance.

The SIDs or STARs closer to a radar signature have higher probability. After these two probability matrices are produced, we select a SID and STAR from the joint distribution of these two probability matrices. The SID or STAR with higher probability has more chance to be selected in this method.

In the system, there are two steps to choose a SID for a flight. The first step is to check if it is necessary to update the SID route or not, mostly according to the deviation between the estimated SID and real time flight trajectory, as shown in Algorithm 6.1. If a flight SID needs to be re-estimated according to Algorithm 6.1, the system will go to the second step to estimate SID based on SID summary frequency matrices.

Algorithm 6.1 Check Prerequisite to Update a Flight's SID

```
1: F is a flight
2: wpList is F's current constructed trajectory
3: M is the message about F in real time air traffic data Data
4: if wpList contains a radar signature then
5:     if F.isTOC_Fixed ≡ false then
6:         F.startEstimatingSID ⇐ false
7:         if F.isSID_Estimated ≡ true then
8:             if wpList is changed or F.Destination is changed then
9:                 F.startEstimatingSID ⇐ true
10:            else if F.SID ≠ Null and F has a new radar signature RS in M then
11:                MinDist ⇐ Max_Vaule
12:                for each p in F.SID do
13:                    d ⇐ distance between RS and p
14:                    if d < MinDist then
15:                        MinDist ⇐ d
16:                    end if
17:                end for
18:                if d > 2nm then
19:                    F.startEstimatingSID ⇐ true
20:                    F.SID ⇐ Null
21:                end if
22:            end if
23:        else
24:            F.startEstimatingSID ⇐ true
25:        end if
26:    end if
27: end if
```

These two procedures for SID estimation are undertaken repeatedly until flight's TOC is determined, and then the flight's SID is fixed. The STAR estimation only happens after flight's TOC is determined. STAR estimation takes a similar procedure as for SID.

Finally, the Flight Runway Estimation Model (M2.6) estimates the flight's departure runway and taxi-out time after the flight's SID is fixed, and the arrival

runway and taxi-in time are estimated immediately after its STAR is determined.

The model uses two procedures to estimate departure and arrival runway. In the first procedure, if a SID is associated with a single runway, this runway will be the departure runway. Similarly, if a STAR is associated with a single runway, this runway will be the arrival runway. If a SID or STAR has a number of runways associated with it, the second procedure is used.

The second procedure applies a method similar to that used in Flight SID and STAR Estimation Model (M2.5). Probability matrices of way points, destinations, and aircraft types associated with departure runways, and probability matrices of way points, origin, and aircraft types associated with arrival runways, are built from historical data. Then we apply the same algorithm as used in M2.5 to calculate the probabilities of runway selections to estimate the departure or arrival runway: runways are selected from the joint distribution of these probability matrices.

6.1.2.3 Flight aerodynamic module

Given the nature of real time air traffic data, trajectories of flights constructed by M2 may not be always complete. Only the partial trajectories of flights are fed into the Flight Aerodynamic Module (M3) by the kernel. This module starts to process a flight once its TOC is fixed, to reduce the computational cost of processing a partial trajectory.

The module utilizes BADA to model the aerodynamics for each flight based on its aircraft type. Figure 6.7 shows the components in this module, which involves decomposing flight trajectory into segments, determining the flight phase, correcting flight speed profile if required, and generating flight aerodynamic parameters.

The Flight Trajectory Decomposition Model (M3.1) decomposes the partial trajectory into small segments, where each start and end of a segment has position, speed, altitude, and time stamp. The start point of each segment is either an origin airport, a radar signature, TOC, or TOD. A way point can be the first point in a segment if the origin airport is outside Australia. The end point of a segment either

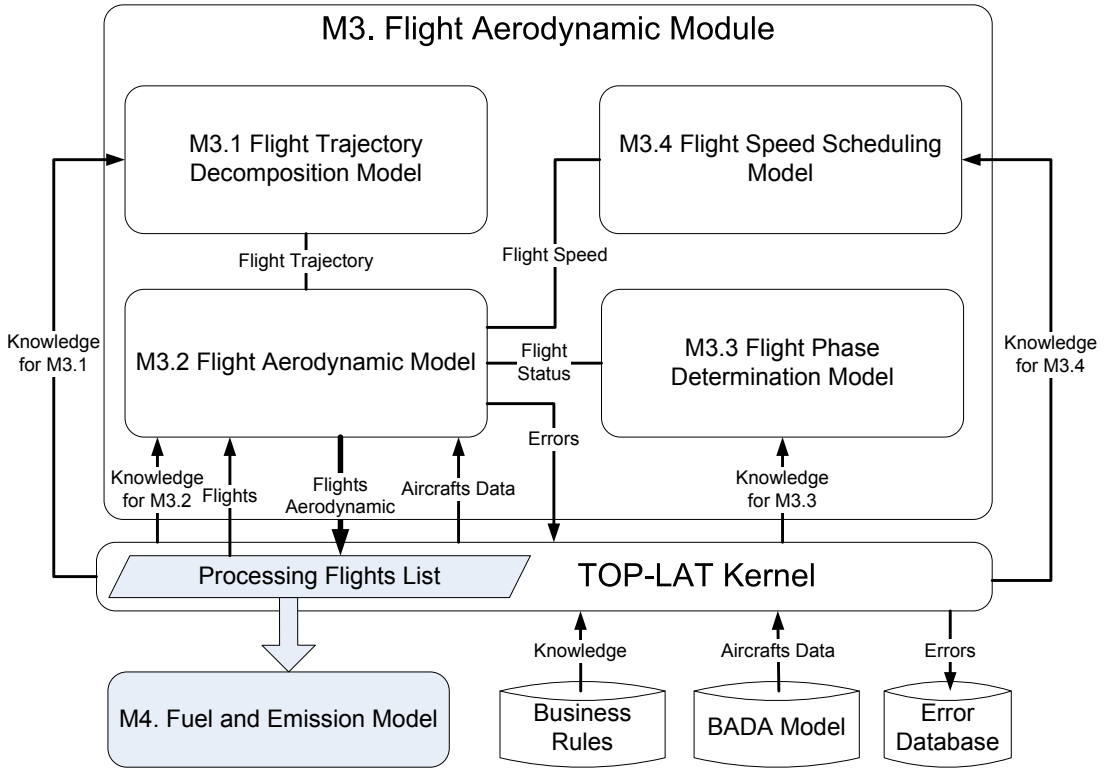


Figure 6.7: The Flight Aerodynamic Module (M3)

is a radar signature, TOC, or TOD. When a flight finishes inside Australian airspace the end point in the segment is either the destination airport or (for outbound or overflying flights) the last way point.

Decomposing the flight trajectory into segments means our system is able to model the aircraft movements continuously, even if that the flight trajectory is incomplete.

M3.1 generates different types of segments, depending on the types of the start and end points. The possible segments are shown in Table 6.1.

After segmenting the flight trajectory, Flight Aerodynamic Model (M3.2) uses the same aircraft performance modelling in ATOMS (Alam *et al.*, 2008) to process the aircraft aerodynamics at every time step within a segment based on the BADA model. The parameters and their descriptions in BADA model are listed in Table B.1 in the Appendix.

Table 6.1: The Possible Flight Trajectory Segments

Start	End	Description	Happens
ORIGIN	TOC	There is no radar signature between them and flight is in climbing phase.	Yes
ORIGIN	RADAR SIGNATURE	Radar signature is before TOC.	Yes
TOC	TOD	There is no radar signature between them and flight is in cruising phase.	Yes
TOC	WAY POINT	There is no radar signature between them and way point is before TOD. Flight is in cruising phase.	Yes
TOC	RADAR SIGNATURE	A flight is in cruising phase.	Yes
TOD	RADAR SIGNATURE	A flight is in descending or approach phase.	Yes
TOD	WAY POINT	There is no radar signature between them and flight is in descending or approaching phase.	Yes
TOD	DESTINATION	There is no radar signature between them and flight is in descending or approaching phase.	Yes
RADAR SIGNATURE	RADAR SIGNATURE	Flight phase depends on the radar signatures are in which phases: the origin to TOC, TOC to TOD, or TOD to destination segment.	Yes
RADAR SIGNATURE	TOC	A flight is in climbing phase.	Yes
RADAR SIGNATURE	TOD	A flight is in cruising phase.	Yes
RADAR SIGNATURE	DESTINATION	A flight is in descending phase.	Yes
RADAR SIGNATURE	WAY POINT	The case happens only when the way point is the last point in the flight route. Flight goes to overseas and it is in cruising phase.	Yes
WAY POINT	RADAR SIGNATURE	The case happens only when the way point is the first point in the flight whole route and the flight comes from overseas. Flight is in cruising phase.	Yes
WAY POINT	TOD	The case happens only when the way point is the first point in the flight's whole route and the flight comes from overseas. There is no radar signature between them. Flight is in cruising phase.	Yes
WAY POINT	WAY POINT	A flight is in cruising phase.	Yes
WAY POINT	TOC	The way point is the first point in the flight whole route and the flight is come from overseas. It will not happen because the TOC can't be calculated by a way point without knowing origin airport.	No
WAY POINT	DESTINATION	It will not happen because the TOD is always available when the destination is known.	No

The BADA data in combination with the underlying performance model can be used to calculate lift and drag as well as thrust and fuel flow in all flight phases. The aircraft model in BADA is a point-mass model which balances the rate of work done by forces acting on the aircraft and the rate of increase in potential and kinetic energy. This approach, referred to as a Total Energy Model (TEM), is represented by the following equation (EUROCONTROL, 2004b):

$$(T - D) \times v_{TAS} = M \times g \times \frac{dh}{dt} + M \times v_{TAS} \times \frac{dv_{TAS}}{dt} \tag{6.4}$$

where : T = thrust[N] D = aerodynamic drag[N]

M = aircraft mass [kg] g = gravitational acceleration$[m/s^2]$
v_{TAS} = true air speed[m/s] h = altitude[m]
Equation 6.4 includes three independent variables which represent typical aircraft control inputs: thrust T, true airspeed v_{TAS} and rate-of-climb (or descent) dh/dt.

The International Standard Atmosphere (ISA) (ICAO, 2004) is typically assumed for BADA calculations, although a temperature deviation from ISA could be specified. Air temperature and density vary with altitude and can be calculated from ISA assumptions. Mach numbers from the BADA speed schedule can be converted to true airspeeds by the following equation:

$$v_{TAS} = M \times a = M \times \sqrt{\gamma \times R \times T^*} \tag{6.5}$$

where γ = isentropic expansion coefficient for air a = local speed of sound$[m/s^2]$
R = universal gas constant for air $[m^2/Ks^2]T^*$ = local temperature[K]

Since the aerodynamic drag is required in equation 6.4, lift and drag coefficients C_L and C_D as well as the respective forces are calculated using the following equations:

$$C_L = \frac{2 \times m \times g}{\rho \times V_{TAS}^2 \times S \times cos\phi} \tag{6.6}$$

$$C_D = C_{D0,CR} + C_{D2,CR} \times C_L^2 \tag{6.7}$$

$$L = \frac{1}{2} \times C_L \times \rho \times V_{TAS}^2 \times S \tag{6.8}$$

$$D = \frac{1}{2} \times C_D \times \rho \times V_{TAS}^2 \times S \tag{6.9}$$

where: ρ = air density$[kg/m^3]$ $C_{D0,CR}$ parasitic drag coefficient[-]
ϕ = bank angle[degrees] $C_{D2,CR}$ = induced drag coefficient[-]
S = reference wing surface area$[m^2]$ Wing area and drag coefficients are given by the BADA.

Performance characteristics of aircraft three phases are modeled in BADA as well:

Cruise Phase Performance characteristics:

During cruise phase (level flight), the flight path angle ψ is assumed to be zero. The engine thrust is set equal to aerodynamic drag (T = D), and lift equals aircraft weight (L=W). Combining the lift equation above with the level flight condition of L = W = mg, the lift coefficient, C_L can be determined as (EUROCONTROL, 2004b)

$$C_L = \frac{2mg}{\rho V_{TAS}^2 S} \tag{6.10}$$

where m (kg) is aircraft mass, g is the gravitational acceleration ($9.81 m^2/s$), ρ is the air density, V_{TAS} is the true airspeed and S is the wing reference area.

For a jet aircraft, the drag coefficient C_D is expressed as a parabolic function of the lift coefficient C_L, also called the parabolic drag polar, as follows:

$$C_D = C_{D0,CR} + C_{D2,CR} \times (C_L)^2 \tag{6.11}$$

where $C_{D0,CR}$ is the parasitic drag coefficient, i.e. the drag coefficient when the lift coefficient is zero, and $C_{D2,CR}$ is the induced drag coefficient, which presents the portion of drag due to lift.

Climb Phase Performance characteristics:

In the ISA condition, BADA estimates the maximum climb thrust T_{climb} as a quadratic function of altitude h:

$$T_{climb,ISA} = C_{Tc1} \times (1 - \frac{h}{C_{Tc2}} + C_{Tc3} \times h^2) \tag{6.12}$$

where C_{Tc1} (Newton), C_{Tc2} (feet), C_{Tc3} ($1/feet^2$), are the max climb thrust coefficients. In other atmospheric condition, where temperature deviations from the standard atmosphere is ΔT_{ISA}, the corrected climb thrust is defined as:

$$T_{climb} = T_{climb,ISA} \times (1 - C_{Tc5} \times (\Delta T_{ISA})_{eff}) \tag{6.13}$$

where $(\Delta T_{ISA})_{eff} = \Delta T_{ISA} - C_{Tc4}$, C_{Tc4} (deg. Celsius), C_{Tc5} (1/deg. C) are the thrust temperature coefficients and $0 \leq (\Delta T_{ISA})_{eff} \times C_{Tc5} \leq 0.4$

Descent Phase Performance characteristics:

Descent Thrust is calculated using different correction factors for high and low altitude, based on the transition altitude h_{des} for calculation of descent thrust, and approach and landing configurations.

- High altitude descent, i.e. top of descent (TOD) altitude is higher than the transition altitude h_{des} ($h > h_{des}$)

$$T_{des,high} = C_{Tdes,high} \times T_{climb} \tag{6.14}$$

- Low altitude descent, i.e. TOD altitude is lower than the transition altitude h_{des} ($h < h_{des}$)

$$T_{des,low} = C_{Tdes,low} \times T_{climb} \tag{6.15}$$

- Approach: once the aircraft has descended below 8000 ft (h < 8000ft) and the airspeed falls below a certain threshold ($V_{TAS} < V_{min,cruise}$ + 10kts), the approach flap setting and thrust setting are used.

$$T_{des,app} = C_{Tdes,app} \times T_{climb} \tag{6.16}$$

- Landing: once the aircraft has descended below 3000 ft (h < 3000ft) and the airspeed falls below a threshold ($V_{TAS} < V_{min,approach}$ + 10kts), the landing flap setting and thrust setting are used.

$$T_{des,ld} = C_{Tdes,ld} \times T_{climb} \tag{6.17}$$

where $C_{Tdes,high}$ and $C_{Tdes,low}$ are the high and low altitude descent thrust coefficients respectively. $C_{Tdes,app}$ and $C_{Tdes,ld}$ are the approach and landing thrust coefficients respectively. T_{climb} is the maximum climb thrust.

The minimum speeds for the aircraft is specified as follows:

$$V_{min,cruise} = C_{V_{min}} \times V_{stall,CR} V_{min,approach} = C_{V_{min}} \times V_{stall,AP} \tag{6.18}$$

where C_{Vmin} is the minimum speed coefficient and set to 1.3 for all aircraft. $V_{stall,CR}$ and $V_{stall,AP}$ are the cruise and approach stall speeds.

Flight Aerodynamic Model (M3.2) estimates and constructs the 4D aircraft trajectories from one point to another and models the aircraft performance characteristics based on the received air traffic data.

There are two other models in this module to support the correct flight aerodynamic modeling: one is a Flight Phase Determination Model (M3.3) to determine the flight phases, and another is a Flight Speed Scheduling Model (M3.4) to correct the speed of a flight if required.

M3.3 determines the flight phase based on the altitude of the flight, which is divided into six legs: take off (climbing and below 1500 feet), climb out (climbing and below 3000 feet), climb (climbing from 3000 feet until reaching TOC), cruise (between TOC and TOD), descent (descending from TOD to 3000 feet), and approach (descending and below 3000 feet).

M3.4 checks the speed retrieved from a radar signature and corrects it if it is not within the performance envelope of the aircraft. We get a minimum speed and a maximum speed from BADA according to the aircraft's altitude and its flying phase. If the radar speed is in the range between the minimum speed and the sum of maximum aircraft speed and wind speed (a constant value equal to 115 knots), we will accept the radar speed. Otherwise, there are two solutions to correct the radar speed:

1. If a flight is in cruising phase we check the cruising speed or altitude in its plan.

   (a) If cruising speed is available, we use the cruising speed for this radar signature.

(b) If cruising speed is unavailable but altitude is available, we use the reference speed from BADA for that cruising altitude.

(c) If neither cruising speed or altitude are available in the flight plan, we use the reference speed from BADA for this radar signature.

2. If the flight isn't in cruising phase, we use the reference speed from BADA for this radar signature.

The algorithm is also used to assign speed to a flight when it is at way points.

The computed aerodynamic values (Thrust, Mach Number etc.) are returned to the kernel, which then sends them to the Fuel and Emission Module (M4) for fuel flow and aviation emission calculations.

6.1.2.4 Fuel and emission module

The aircraft's aerodynamic parameters computed by M3 are used to calculate the fuel flow and emission by the Fuel and Emission Module (M4). The models in this module are shown in Figure 6.8.

The BADA model is used to calculate lift and drag in all flight phases. For fuel flow computation, assuming nominal aircraft mass, the thrust specific fuel consumption η in kg/s/kN is specified as a function of true airspeed, V_{TAS} (knots) for the jet engines. The nominal fuel flow, $\int_{nom}$(kg/s), can then be calculated for jet engine aircraft using the thrust, T_h as:

$$\int_{nom} = \eta * T_h \;\; Where \;\; \eta = C_{f1}(1 + \frac{V_{TAS}}{C_{f2}}) \tag{6.19}$$

C_{f1} (kg/s/kN) is first thrust specific fuel consumption coefficient, and C_{f2} (knots) is a second thrust specific fuel consumption obtained from BADA aircraft performance tables. The absolute amount of fuel burned in a flight segment is calculated by multiplying fuel flow with time.

The emission computation process is divided into two phases, below 3000ft and

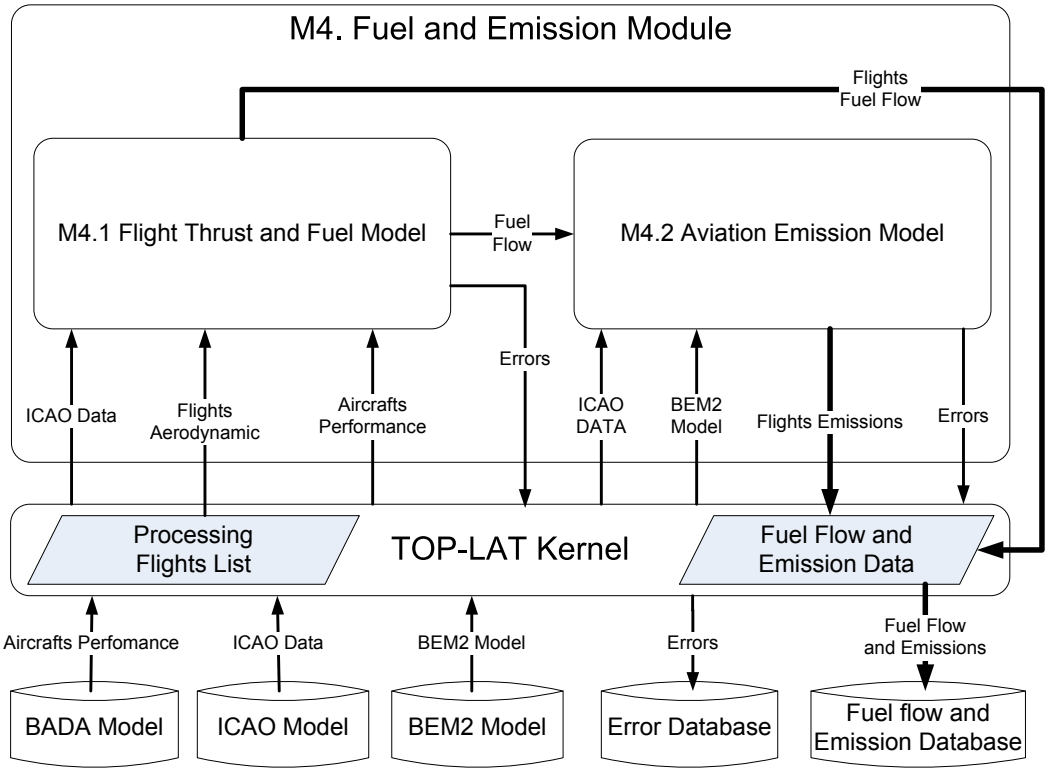


Figure 6.8: The Fuel and Emission Module (M4)

above 3000ft, as shown in Figure 6.5. Below 3000ft, the emission calculation is based on the ICAO Engine Exhaust Emissions Data Bank (ICAO, 1995). The fuel burn calculation is based on the Landing and Take-Off Cycle (LTO) defined by the ICAO Engine Certification specifications. ICAO LTO covers four engine operation modes, which are used to model the six phases of aircraft operations as shown in Table 6.2.

Table 6.2: ICAO Landing Take off Cycle and time in mode

Segment	Throttle Settings	Time in Mode (min)
Taxi-Out	7% take-off thrust	Actual time in taxi-out
Take-Off	100% take-off thrust	Actual time in take-off
Climb-Out	85% take-off thrust	Actual time in climb-out
Approach	30% take-off thrust	Actual time in approach
Landing	30% take-off thrust	Actual time in landing
Taxi-In	7% take-off thrust	Actual time in taxi-in

Above 3000 feet, BEM2 model (Baughcum *et al.*, 1996; Brian *et al.*, 2005) is used for emission calculation. BEM2 is an accepted and widely used methodology

for calculating flight emissions. ICAO's Committee on Aviation Environmental Protection (CAEP) has recommended the adoption of BEM2 as a standard method for calculating emissions.

BEM2 allows for the estimation of emissions for pollutants such as NO_x, HC, CO. The emission for CO_2 (with CO corrections) and SO_x is directly proportional to fuel burn. BEM2 computes flight emissions using, as a base, the measured fuel flow and the engine ICAO data sheets. It accounts for ambient pressure, temperature and humidity as well as Mach number.

In our system the Time in Mode is derived from actual flight trajectory data, and runway timings are specific for each airport and aircraft type. This gives the emission computation a high level of fidelity, compared to using generic ICAO LTO cycle timings.

6.1.2.5 Real time air traffic flow construction module

Modules M2 and M3 construct and maintain a list of trajectories for individual active flights. These need to be aggregated to give a real time picture of the overall air traffic situation, in order to visualize and analyze air traffic flow, airspace safety, and airspace complexity. This is the function of the Real Time Air Traffic Flow Construction Module (M5).

As shown in Figure 6.9, this module includes four models to achieve the following objectives:

- Time synchronization of active flights in Australian airspace (M5.1);
- Estimating the status, including 3D position, speed, and heading of active flights at the synchronized time (M5.2);
- Predicting the future trajectories of active flights based on flight plans and flight 4D position (M5.3);
- Calculating the historical tracks of active flights (M5.4).

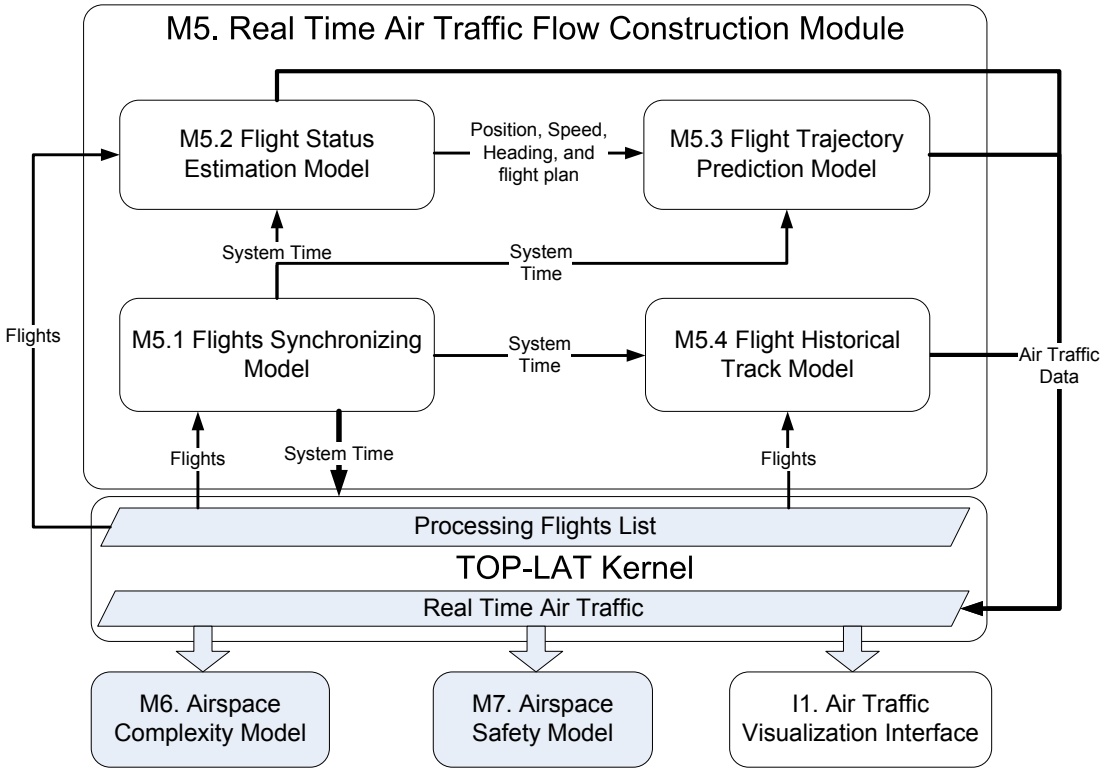


Figure 6.9: The Real Time Air Traffic Flow Construction Module (M5)

Each air traffic data file contains several flight signatures, with time stamps that may not be necessarily the same. The first task of this module is to synchronize the time of each flight signature with the system time (UTC). To set the system time, the module scans the first radar data file from M2 (Flight Initialization and Updating Module), and sets the system time to the time of the latest radar signature in the file; this process is done the first time when the system is initialized.

A flight's position at a given time is estimated from the two consecutive radar signatures whose time stamps are closest to the system time. Based on the interval between the system time and the time stamps of the radar signatures, a function is applied to estimate the flight position by extrapolation.

Using the flight's position, speeds, and headings estimated by the above algorithm and the flight plan, the future trajectories are predicted by the same extrapolation function. The historical tracks are obtained from the flight trajectories constructed by the Flight Initialization and Updating Module (M2).

The results from this module are returned to the kernel. From there, they are directly used in the air traffic visualization interface (described in Section 6.1.4). They are also fed into other modules such as Airspace Complexity Module (M6) and Airspace Safety Module (M7), which are explained in the following sections.

6.1.2.6 Airspace complexity module

The Airspace Complexity Module (M6) calculates a variety of measures of airspace complexity.

Airspace complexity is defined as the interactions between the airspace characteristics and air traffic (Sridhar *et al.*, 1998). The airspace characteristics are fixed for a sector by the spatial and physical attributes of the sector, such as terrain, number of airways, airway crossings and navigation aids. The air traffic characteristics vary as a function of time, and depend on features like number of aircraft, mix of aircraft, weather, separation between aircraft, closing rates, aircraft speeds and flow restrictions. The combination of these structural and flow parameters influences an air traffic controller's workload, which is an important indicator of airspace capacity.

The aircraft count in a sector is taken as a common measurement for airspace complexity. However, aircraft count has significant shortcomings in its ability to accurately measure and predict sector level complexity (Chatterji and Sridhar, 2001).

Dynamic Density (DD) is defined as factors, or variables, that contribute to the sector level air traffic control complexity or difficulty at any given time (Federal Aviation Administration, 2001). DD metrics perform better than aircraft count in practice (Kopardekar and Magyarits, 2003). In this model, we have adopted four DD metrics from the literature as complexity measurements:

- FAA WJTHC/Titan System Metrics (Kopardekar, 2000)
- NASA Metric 1 (Chatterji and Sridhar, 2001)
- NASA Metric 2 (Laudeman *et al.*, 1999; Sridhar *et al.*, 1998)
- Metron Aviation Metric (Wyndemere Inc., 1996)

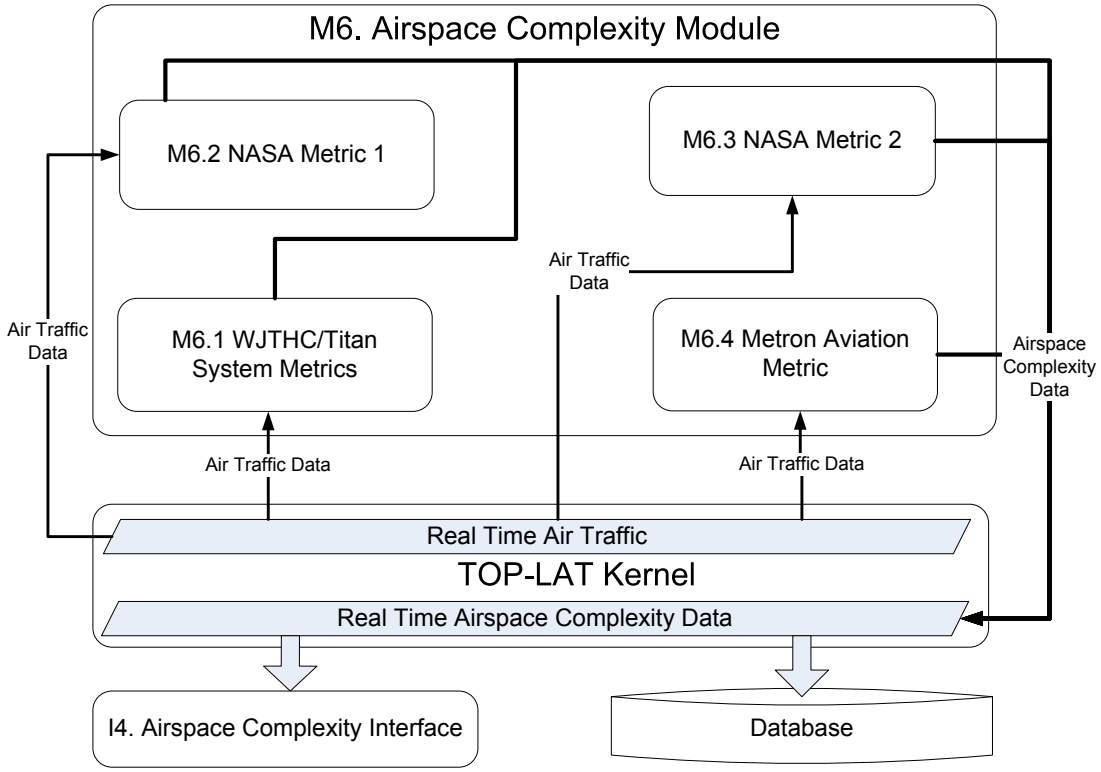


Figure 6.10: The Airspace Complexity Module (M6)

These DD metrics include a wide range of elementary measures of complexity, such as number of aircraft, aircraft density, climbing and descending aircraft, aircraft separation, and speed variance, and are widely used.

These four metrics are calculated every time step, based on the output from M5, and stored in spatial and temporal dimension along with the complexity measures in the database. The models in this module are shown in Figure 6.10.

The outputs from this module are stored in the database and directed to the user interfaces by the kernel.

6.1.2.7 Airspace safety module

Maintaining safe separation between aircraft and ensuring a smooth flow of air traffic is fundamental to ATM.

The Airspace Safety Module (M7) evaluates five different aspects of airspace

safety, based on the output from M5 and with reference to safety guidelines in EUROCONTROL Safety Regulatory Requirement (ESARR2) document (EUROCONTROL, 2009). These five aspects of airspace safety are way point congestion, conflicting aircraft, conflicting flight plans, flights crossing Special Use Areas (SUA), and Required Navigation Performance (RNP) measurements. Each is handled by a separate model, as shown in Figure 6.11.

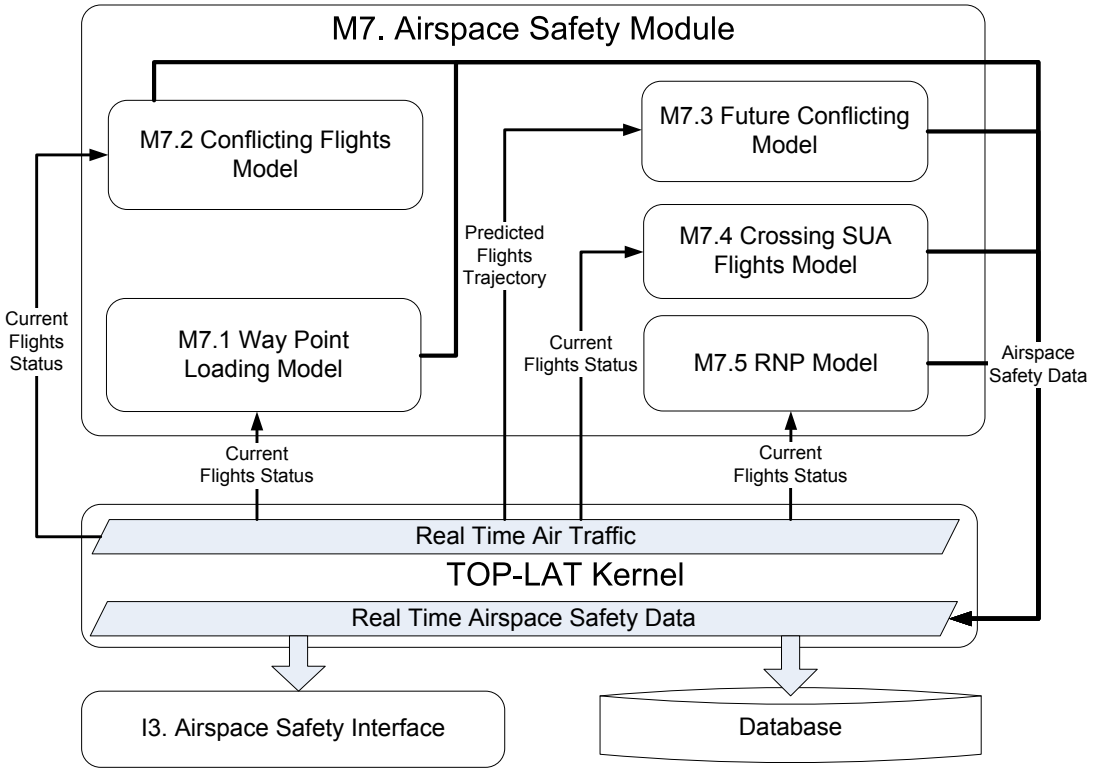


Figure 6.11: The Airspace Safety Module (M7)

M7.1 measures the congestion level at each way point for a given period. The congestion of each way point is indicated by the count of flights that are crossing a way point at the same time interval. A user-defined threshold of congestion alerting is applied in the system. If the count of flights at a way point exceeds the threshold, the way point is identified as a congested way point.

Safety module M7.2 checks for conflicting flights at all times. The separation standards, including horizontal and vertical separations, are based on ICAO standards (5nm horizontal and 1000ft vertical) or can be defined by the user. To check

for future conflicts, M7.3 provides a method for identifying future conflicts based on the predicted flight trajectories and flight plans.

Civilian flights at all times must avoid active SUA's during their flight. More than 500 SUAs exist in Australian airspace; they are used by different organizations such as the Department of Defence and AirServices Australia. M7.4 identifies flights that are crossing or may cross an active SUA, using Point in Polygon algorithm.

RNP (implemented in M7.5) allows enhancement of airspace capacity and efficiency, while at the same time maintaining or improving safety. RNP places limits on the lateral and longitudinal (cross-track and along-track: see Figure 6.12) deviations between the actual and and planned flight path (ICAO, 1999b). There are five levels of RNP: RNP 1, RNP 4, RNP 10, RNP 12.6, and RNP 20. RNP 1 requires that the deviation between desired and true flight path is less than 1nm; the others means the deviations are limited to 4nm, 10nm, 12.6nm and 20nm respectively.

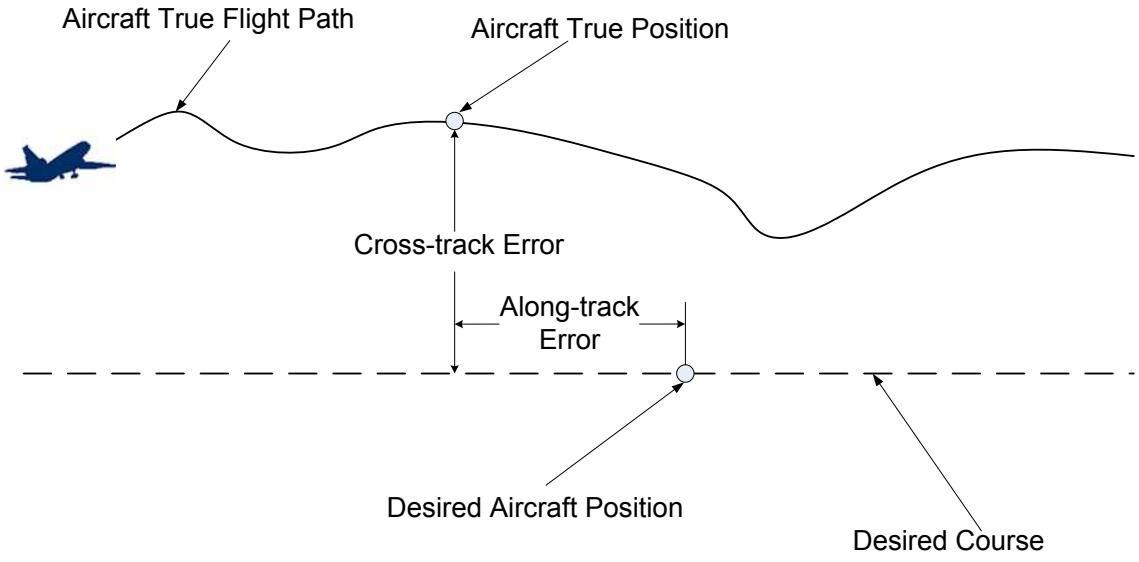


Figure 6.12: Cross-track Error and Along-track Error in RNP

Although RNP 4 and RNP 10 are required and operated in Australia RNP airspace (CASA, 2005a,b), all five RNP types are implemented in this safety module for analysis.

All outputs from the Airspace Safety Module are forwarded to the kernel, which directs them to the user interfaces.

6.1.3 Databases

A relational database system stores the information processed by the system. It consists of a Flight Information Database, a Flight Emission Database, a 3D Grid Emission Database, Complexity Database, Safety Database, and an Error Database.

The Error Database records the errors that are generated by the system during the processing. The types of errors are cataloged, as real time air traffic data error, unknown aircraft type error, flight aerodynamic processing error, and fuel flow and emission calculation error. Real time air traffic data error happens when the data contains unexpected transmitting errors, improper format and others. If a flight's aircraft type is unavailable in our aircraft database, a unknown aircraft type error will be reported to the Error Database. The flight aerodynamic processing error is generated when the aerodynamic processing for a flight can't be completed, and fuel flow and emission calculation errors occurs in the Fuel Flow and Emission Model.

The Flight Information Database stores the information of individual flights, such as the flight's call sign, operator, origin, destination, 4D trajectory, departure time, and fuel flow.

The Flight Emission Database contains the emission inventory of different phases of each individual flight. There are 8 phases of flight for which the data is recorded i.e. taxi out, take off, climb out, climb, cruise, descent, approach, and taxi in. For each flight, the Flight Emission Database also records the emission values for HC, CO, CO_2, NO_x and SO_x.

The 3D Grid Emission Database stores the spatial and temporal emission inventory for Australian airspace. Each grid cell is of dimension 1 degree of latitude by 1 degree of longitude by 1000 feet of altitude. Each record contains the latitude, longitude, altitude and the values of fuel flow, HC, CO, CO_2, NO_x and SO_x. The values of fuel flow and emissions in each grid cell are accumulated over a period of time.

The Complexity Database records the complexity measures for the four DD metrics computed by the system. The data can be used directly for analysis and

visualization. In the long term it can also support the development of innovative airspace configurations, such as dynamic airspace sectorization.

The Safety Database records all the instances (in spatial and temporal dimension) of safety violations, such as loss of separation between two aircraft, crossing of SUA's, Cross Track Errors, and Way point congestion saturation. The data is useful for evaluating the current airspace safety. It also can be used in future flight research.

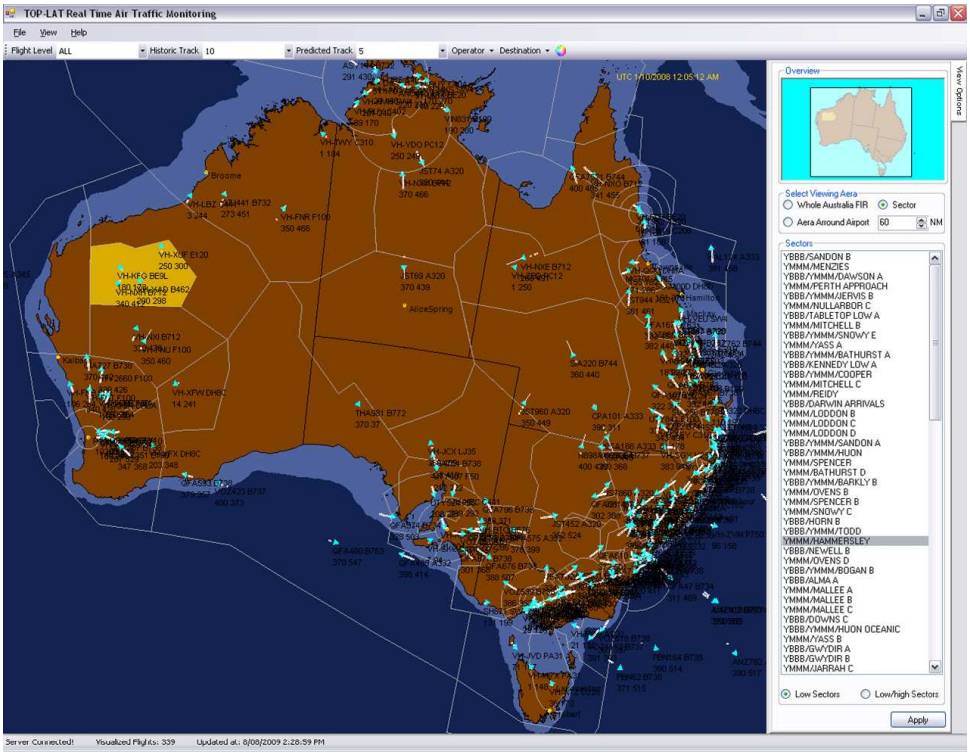
All the databases interact with the kernel of the system.

6.1.4 User interfaces

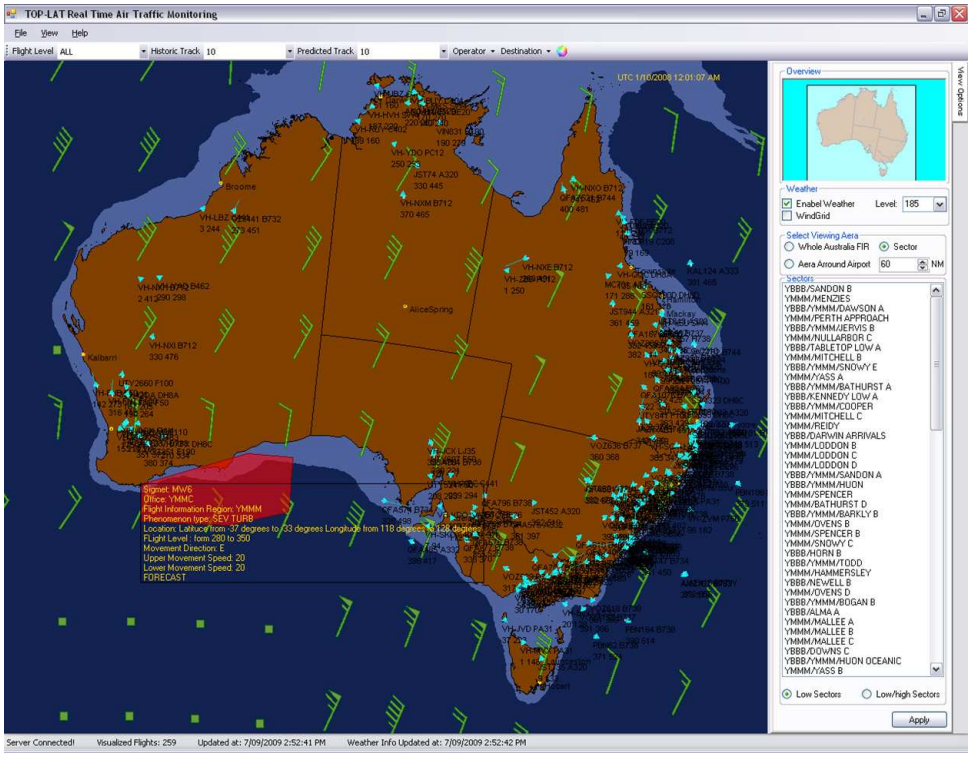
As recommended by the ICAO for air traffic visualization, TOP-LAT utilizes the Lambert conformal conical (LCC) (Bugayevskiy and Snyder, 1995) projection for displaying airspace features and air traffic movements on the graphical user interface. The LCC projection offers the advantage that the shapes of small areas are maintained (no shearing), tearing only occurs around the edges, amount of distortion of areas is minimal near lines of tangency (compression) and distances are correct along the lines of tangency.

Four main user interfaces are provided by this system:

- Air traffic situation awareness
- Airspace safety monitoring
- Airspace complexity visualization
- Aviation emission visualization

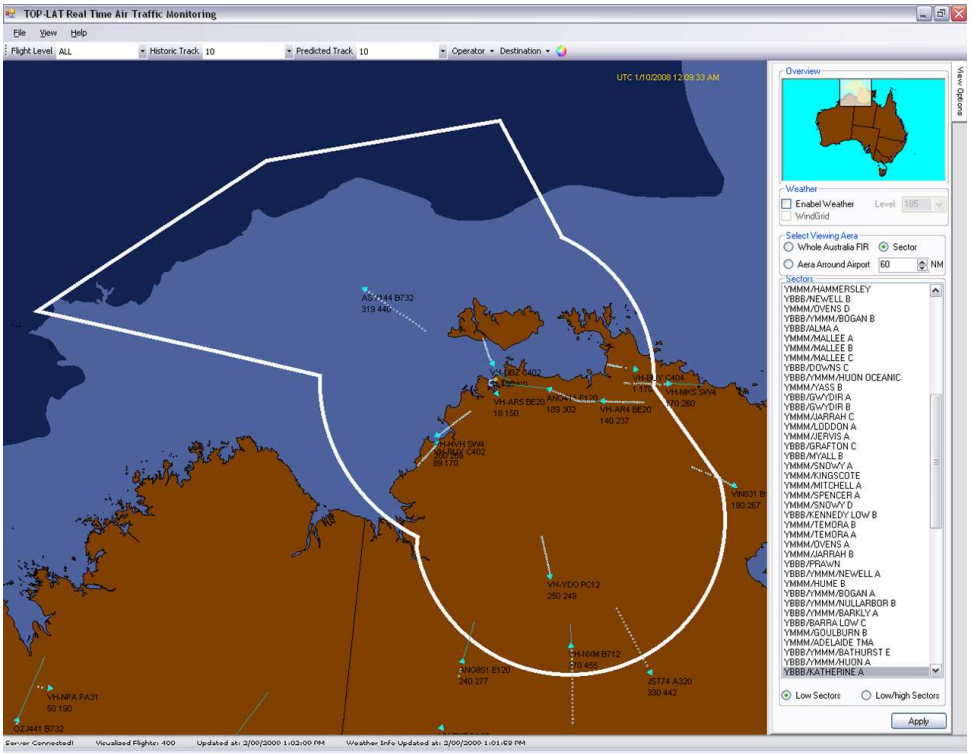


(a) Whole Australian Airspace

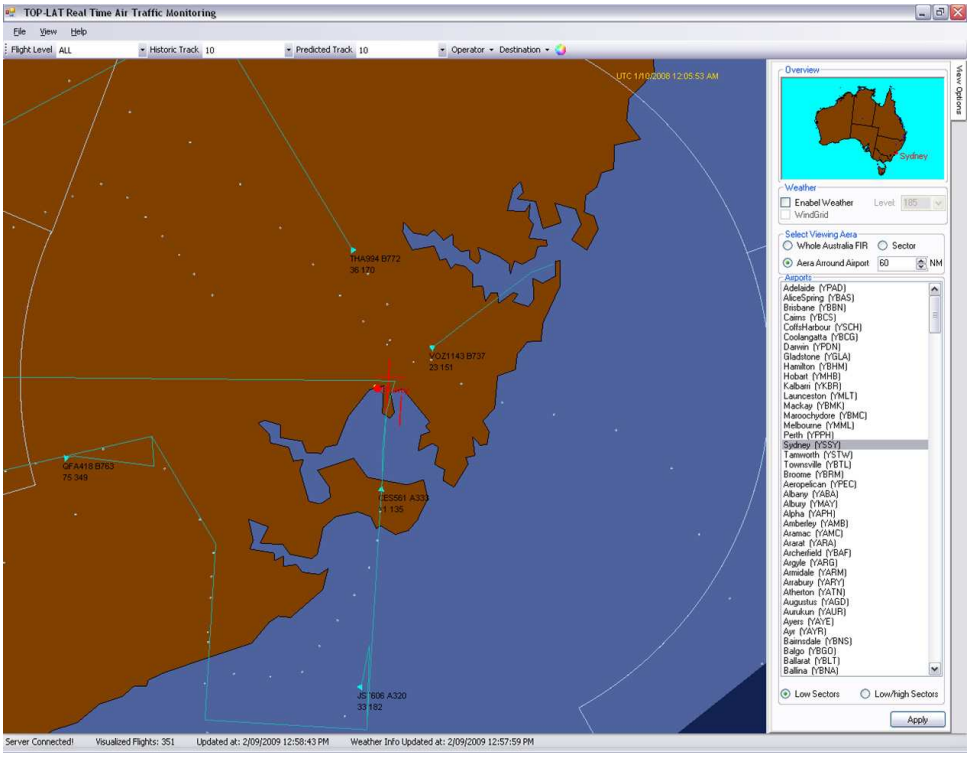


(b) Wind and Sever Weather Information in Australian Airspace

Figure 6.13: Air Traffic Situation Awareness Interface: Whole Australian Airspace



(a) Sector of YBBB/KATHERINE A



(b) Terminal Management Area of Sydney Airport

Figure 6.14: Air Traffic Situation Awareness Interface: Sections of Australian Airspace

The air traffic situation awareness interface provides the situation awareness of the current air traffic, at three levels: the whole Australian airspace, at the sector level, or at the airport level. The interface is shown in Figure 6.14, 6.14a, and 6.14b. The main part of the screen illustrates the visualized information, including airspace configurations, airport locations, 4D flights' positions, flights' historic and future trajectories, flights SID and STAR routes, weather information. The right hand of the screen contains an options panel, which enables users to change their view point to the whole airspace, a specific sector, or a specific airport.

The airspace safety monitoring interface presents the data from M7 module, as shown in Figure 6.15, 6.16, 6.17, and 6.18. As illustrated in the figure, the main part of the screen highlights the congested way points, SUAs that are violated, flights violating RNP, or (especially importantly) flights that have a current or future loss of separation. The interface also gives users the options to specify the threshold for way point congestion, the standards for current and future separation, as well as the RNP specifications.

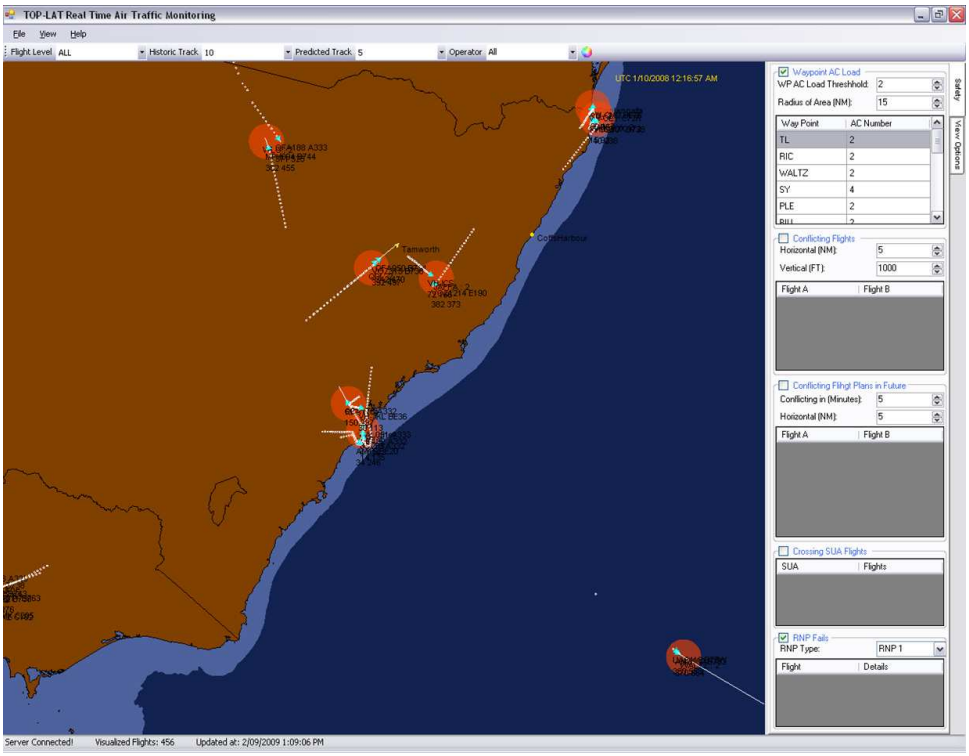


Figure 6.15: Airspace Safety Interface: Congested Way Points

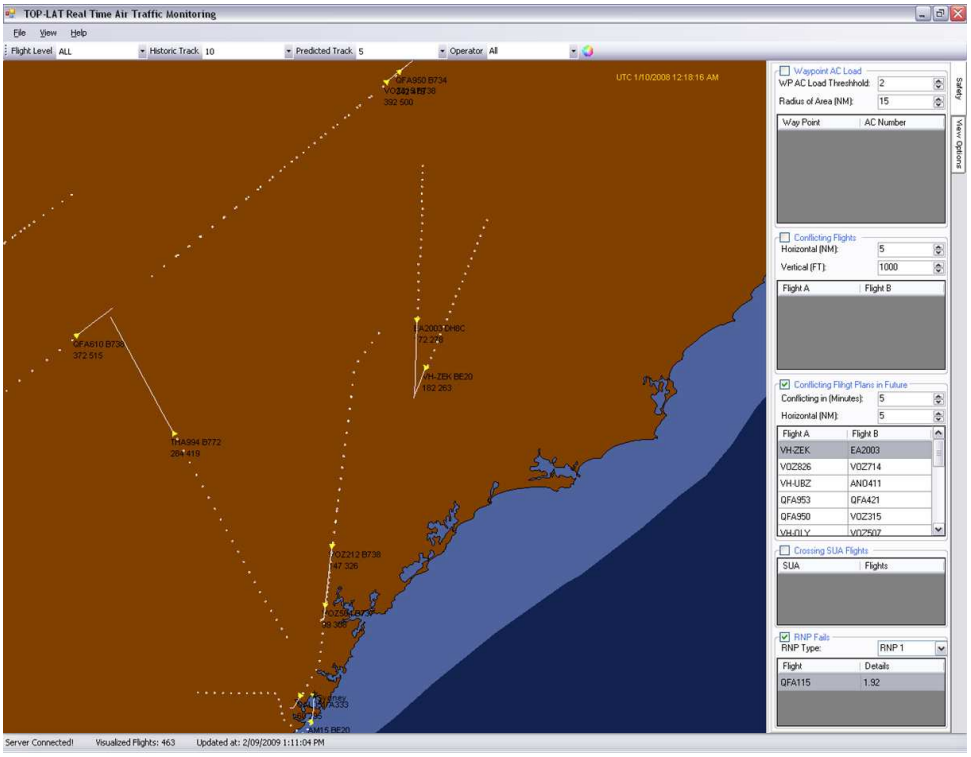


Figure 6.16: Airspace Safety Interface: Flight Plans in Conflict

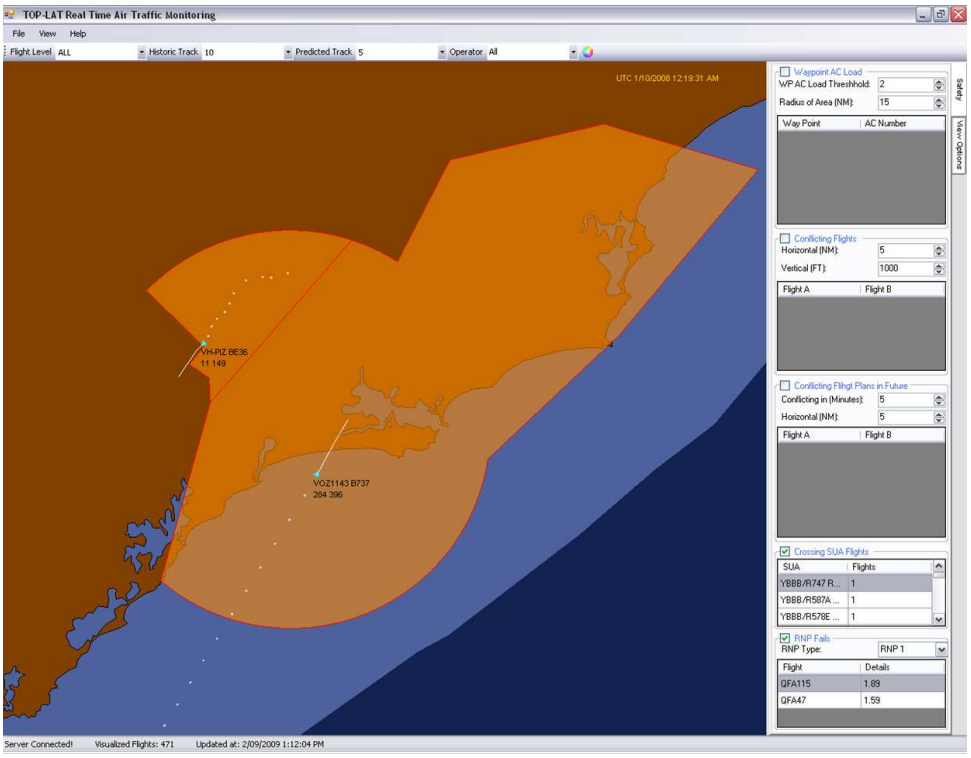


Figure 6.17: Airspace Safety Interface: Flights Crossing SUA

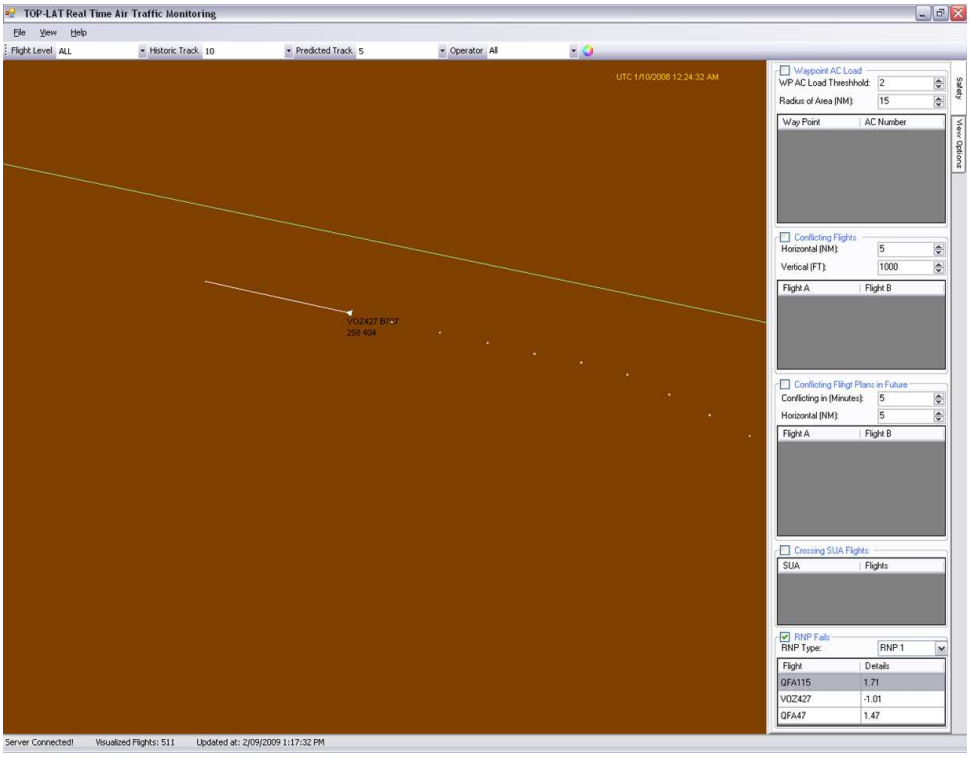
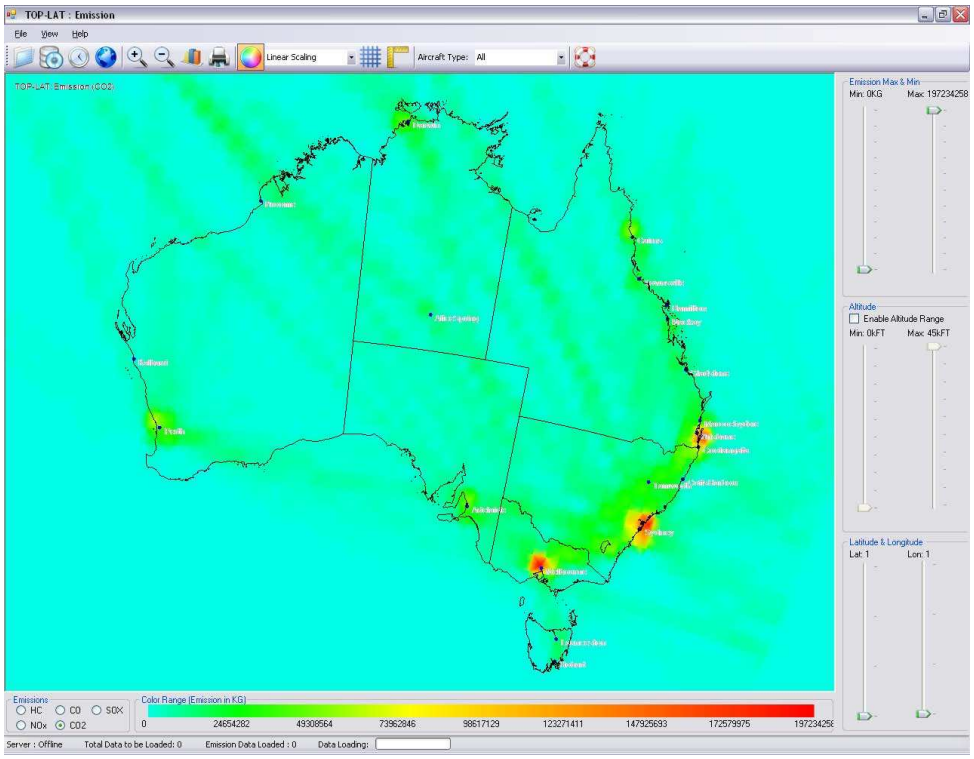


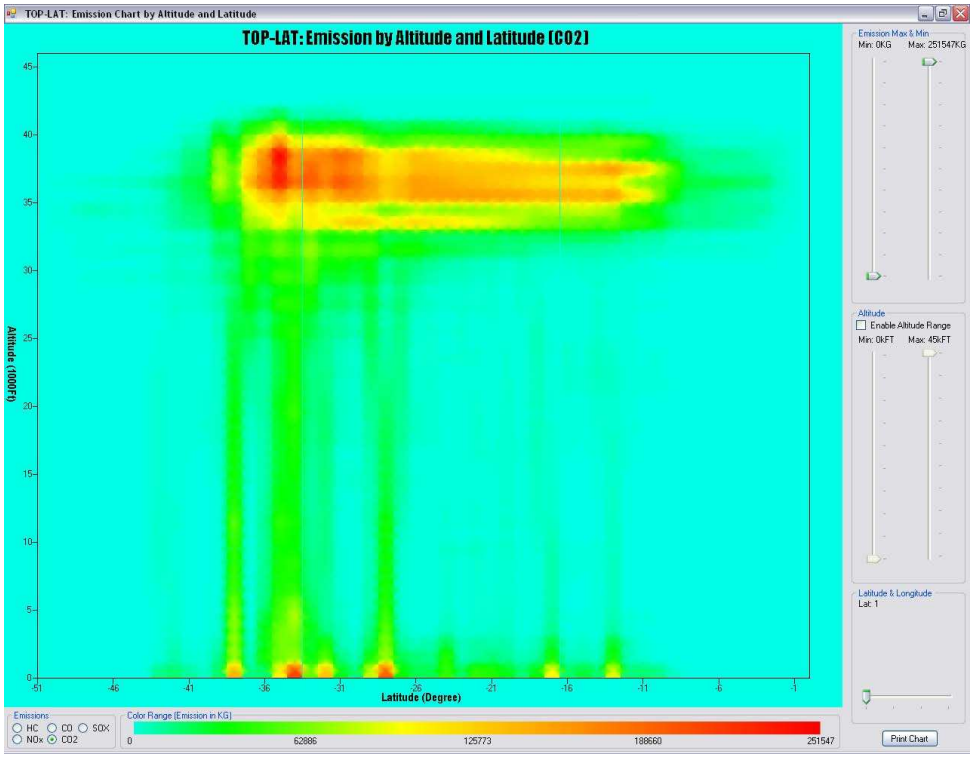
Figure 6.18: Airspace Safety Interface: Flights Violating RNP

The aviation emission interfaces provides a color mapping for the additive values of aviation emission that are stored in the database. Figure 6.19 shows the emission of CO_2 distributed in Australian airspace. Users can choose any one of the emissions (HC, CO, NO_x, $C0_2$, and SO_x) to be visualized. Figure 6.19a shows that CO_2 emission is mainly concentrated around major airports, such as Sydney, Melbourne, Perth, and Brisbane, as shown by the red color that is used to depict the highest emission. The CO_2 emission is also concentrated along the major air routes as shown in the figure. The interface also can show emissions by altitude, as illustrated in Figure 6.19b. It shows that CO_2 emission are distributed at high altitude level because CO_2 is emitted mostly in cruise phase of an aircraft.

This client can be used in aviation environment management to provide an instinctive view of the aviation emission impacts in Australian airspace. The emission database behind this visualization can be used for flight planning, for an even distribution of aviation emission in airspace, or perhaps to assign flights to cruising altitudes where emissions are less concentrated.



(a) Emission of CO_2 Distribution by Latitude and Longitude in Australia



(b) Emission of CO_2 Distribution by Altitude in Australia

Figure 6.19: Emission of CO_2 in Australia Airspace

The airspace complexity visualization uses a similar approach to the emission interface, with color codings, but it uses the color mapping to represent the complexity values for sectors. Since the sectors are divided into low and high according to altitude ranges, some of them overlap in a 2D view. Users can choose which sector category to view, and which dynamic density complexity metrics (or individual metric values) to view. Users can also select an individual sector in the airspace for monitoring and analysis. This information may enable users to reroute a flight for better airspace capacity balancing or sector overload management, or to merge or split sectors.

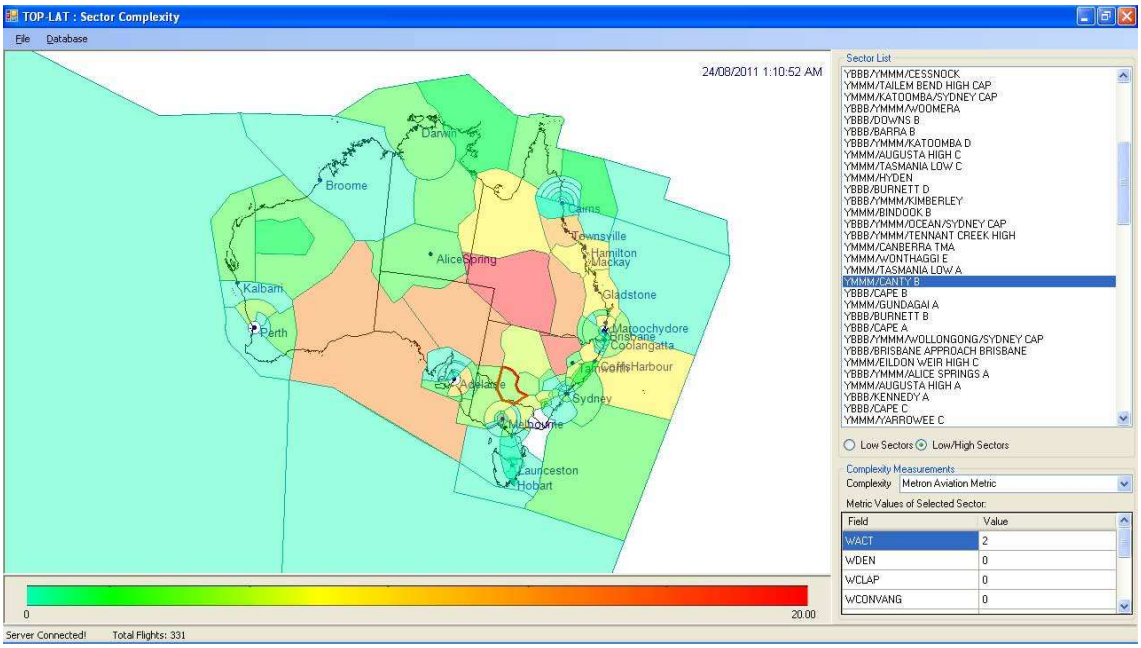


Figure 6.20: Sector Complexity (Human Factor) of Australia Airspace

6.2 System Implementation and Validation

6.2.1 System implementation

According to the architecture described in Section 6.1, the whole system has been implemented as a client-server model in Microsoft Visual C# in the Microsoft .Net^{TM} framework.

A data server receives real time incoming air traffic data from AirServices Australia through an HTTP secure port. The server then forwards the real time radar data to a data server which is located in a private network. The main models are resident on the server to process the incoming data. The processed results, such as air traffic and emissions, are sent to clients through a local area network and stored in MySql databases. The air traffic monitor client displays the real time air traffic flows; the emission client visualizes the up-to-date aviation emission distribution; the airspace safety client evaluates and visualizes the safety issues; at the same time the airspace complexity client shows the current color-mapped complexity metrics for all sectors. Between them these four clients provide real time situation awareness to the users.

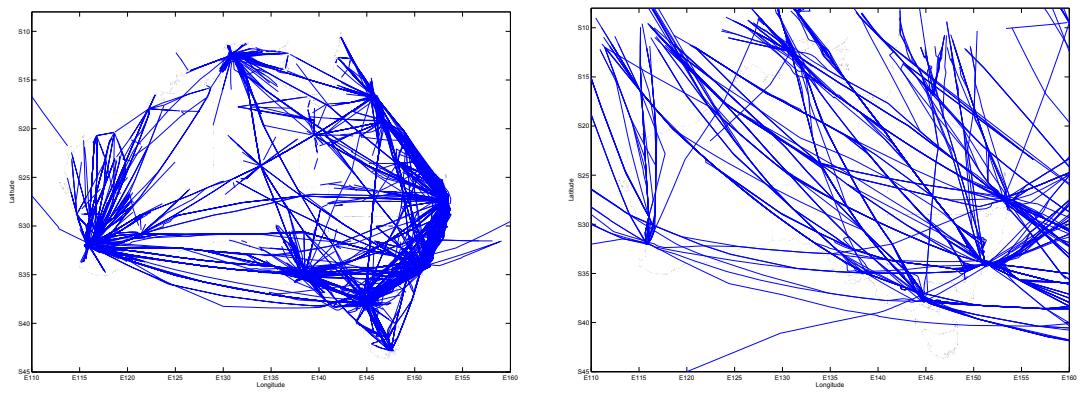
6.2.2 System validation

To validate the system, we must confirm that the input part of the system (real time data processing and flight trajectory construction) works correctly, and that the output parts of the system (fuel and emissions, safety, etc) work correctly.

The foundation is the real time air traffic data processing and flight trajectory construction. If these are implemented correctly, and fuel burn is modeled and implemented correctly, the core is there: we should be able to have confidence in the results.

Validation of the trajectory construction is done by checking all radar signatures in a flight trajectory in a time series, and checking all locations of way points corresponding to all radar signature positions: are they all consistent, and does the final calculated airborne time agree with actual data from the flight data recorder? Validation of the fuel and emissions processing is done by computing fuel and emissions from the constructed trajectory, and comparing them with actual data.

To illustrate the validation process, we choose a flight from Sydney to Perth as an example. The flight information and plan produced by M1 and M2 based on the received data is shown in Table 6.3.



(a) Domestic Flight Trajectories (b) International Flight Trajectories

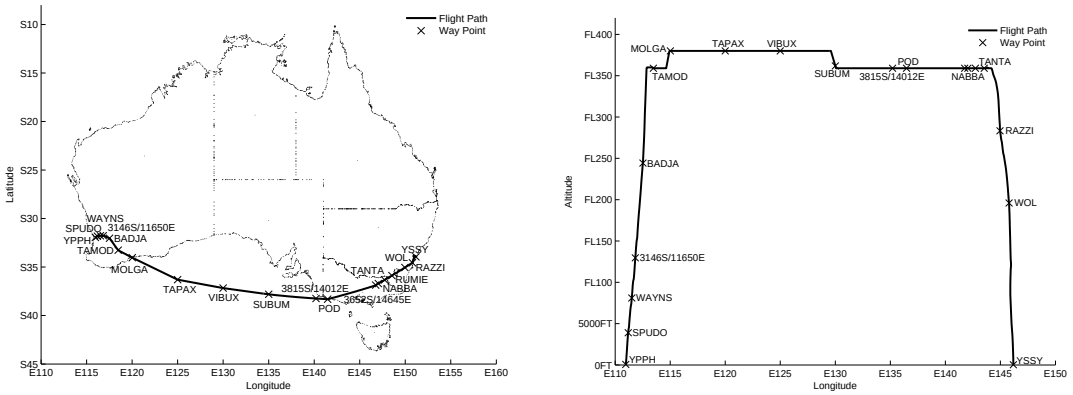
Figure 6.21: All Flight Trajectories for One Day in March 2009

Table 6.3: Flight plan parameters for the example flight

Parameter	Value
Origin	Sydney
Destination	Perth
Aircraft	B743
Engine type & code	CF6-80C2B1
No of engines	4
Way point Route	YSSY; WOL; RAZZI; TANTA; RUMIE; NABBA; 3652S/14645E; POD; 3815S/14012E; SUBUM; VIBUX; TAPAX; MOLGA; TAMOD; BADJA; 3146S/11650E; WAYNS; SPUDO; YPPH
ETD	6/11/2008 10:27 AM
ETA	6/11/2008 16:20 PM
Cruise level speed	445kts
Cruise level altitude	36000ft
Time en-route	4 hrs 53 mins

The flight trajectory constructed by M2 is shown in Figure 6.22a (planar view) and Figure 6.22b (vertical view). It can be seen that the flight track follows the way points listed in the above table. Figure 6.22b illustrates the altitude changes of the flight along the longitude. It shows that the TOC of the flight is around at way point TANTA. The flight climbed from Sydney airport to TANTA and then started cruising. During the beginning of its cruising phase, it maintained its altitude at 35,900 feet which is at the planned level. It climbed to 38,000 feet when it was around the way point SUBUM and maintained this flight level until it reached way point MOLGA which is considered as its TOD. The altitude changes may be due to

the weather conditions or other airspace management issues. It started descending from MOLGA, and approached and landed at Perth airport. This illustrates how M1 and M2 are capable of constructing the flight trajectory piece by piece in real time.



(a) Latitude and Longitude profile of a Flight from Sydney to Perth

(b) Altitude profile of a Flight from Sydney to Perth

Figure 6.22: Trajectory of a Flight from Sydney to Perth

Once the flight trajectories are constructed by M1 and M2, the aerodynamic, fuel flow and emission computations are done. Based on the flight's aerodynamics and fuel flow generated by M3 and M4, Table 6.4 presents the emission results from M4 for the example flight.

Table 6.4: Fuel and Emission Validation for a Flight

Phase	Fuel (kg)	HC (kg)	CO (kg)	NO_x (kg)	CO_2 (kg)	SO_x (kg)
TAXI OUT	70.56	0.67	3.05	0.26	222.41	0.06
TAKE OFF	109.68	0.01	0.06	3.08	345.71	0.09
CLIMB OUT	93.65	0.01	0.05	1.99	295.18	0.08
CLIMB	9277.91	1.09	7.25	200.96	29243.98	7.79
CRUISE	42310.73	19.85	73.20	520.72	133363.41	35.54
DESCENT	651.13	36.76	106.08	3.06	2052.35	0.55
APPROACH	120.74	0.03	0.29	1.07	380.56	0.10
TAXI IN	58.80	0.56	2.54	0.22	185.34	0.05
Sum	52693.19	58.97	192.52	731.37	166088.93	44.26

We then validated the flight trajectory and fuel burn by taking the data from the system and comparing the results for the fuel burned and flight time with the actual data from the aircraft's flight data recorder (provided to us for this study by the airline concerned). Table 6.5 illustrates the calculated and actual results for several flights. The results produced by the system are consistent with the data

Table 6.5: Fuel and Airborne Time Comparison with Flight Data Recorder

Flight	Fuel Burn (kg)	Airborne Time (Mins)	FDR Fuel Burn (kg)	FDR Airborne Time (Mins)	Variance Fuel Burn (%)	Variance Airborne Time (%)
Flight 1	4785	63	4800	63	0.32%	0.00%
Flight 2	5566	65	5697	66	2.30%	1.52%
Flight 3	8380	204	9036	201	7.26%	1.49%
Flight 4	20472	228	19178	229	6.75%	0.44%
Flight 5	5723	71	6000	69	4.62%	2.90%
Flight 6	5100	62	5243	63	2.74%	1.59%
Summary	50026	693	49954	691	0.14%	0.29%

from the airline, in terms of the fuel burn and airborne time, that was recorded by flight management system. This comparison shows the consistence between the outputs from TOP-LAT and the actual data.

6.3 Chapter Summary

In this chapter, we presented a real time air traffic monitoring and advisory system, which integrates the calculation and presentation of airspace safety, capacity complexity, situation awareness, and emissions. This is the first attempt to develop a real time monitoring and advisory system for both ATM objectives and the environment. The system can provide users with real time information to make ATM decisions from the perspective of ATM, the environment, or both.

Real time air traffic data from AirSevices Australia is used to construct the flight trajectories; these in turn are used to compute fuel flows and emissions, to summarize air traffic flow, and to measure the airspace safety and capacity. Radar and Flight plan data is processed and synthesized in real time to compute aerodynamic and flight parameters for fuel flow computation and emission estimation. For emission computation, BEM2 model along with ICAO engine emission data bank is used. Fuel flow is computed by aircraft specific aerodynamic and thrust models, with weight correction. The system takes account of the way point congestion, current conflicts, future conflicts, SUA, and RNP as the issues of airspace safety and capacity. Airspace sector complexity is computed in real time for four dynamic density metrics from the literature, which can support the evaluation of human fac-

tors in ATM. The system provides four different user interfaces, which present real time information about the current air traffic situation, aviation emission, airspace safety, and airspace complexity. All these modules are integrated to provide users with necessary information.

The airspace safety, complexity, and emission data generated by the system can also be used for the further investigation of advanced ATM procedures, such as User Preferred Trajectories (UPT), Continuous Descent Approach (CDA), In-Trail Procedures (ITP), and DAS. This may help to improve the airspace capacity and safety, and to reduce the emission footprint, to support sustainable air traffic growth in the future. In the next chapter, the proposed DAS models are evaluated by TOP-LAT for the Australian airspace.

Chapter 7

3D DAS Models for National Airspace

In this chapter, we investigate two proposed 3D DAS models, Support Plane Bisection Model (SPBM) and Constrained Voronoi Diagram Model (CVDM), in TOP-LAT. The investigation includes a set of experiments which partition Australian airspace based on daily air traffic to achieve optimal airspace sectorization results. The experiments consider all flight movements within two control centres, Brisbane and Melbourne, in the whole Australian FIR. The DAS models have to handle a large amount of traffic data and finish the optimization in a given time window (no more than 24 hours). Therefore, iABM is excluded from these experiments because of its high computational cost. KD-tree Based Model has poor performance in the preliminary experiments as shown in Chapter 5, so it is also excluded.

7.1 Evaluation Overview

Figure 7.1 shows the framework of my experiments. Both SPBM and CVDM perform the airspace sectorization based on the processed air traffic data from TOP-LAT for evaluation. The current airspace sectorization from TOP-LAT system is also evaluated against the solutions from DAS models. Then the airspace sector-

ization suggestions can be provided to TOP-LAT for decision makers, but this is outside the scope of this thesis. The air traffic data can be either the real data or the simulated data.

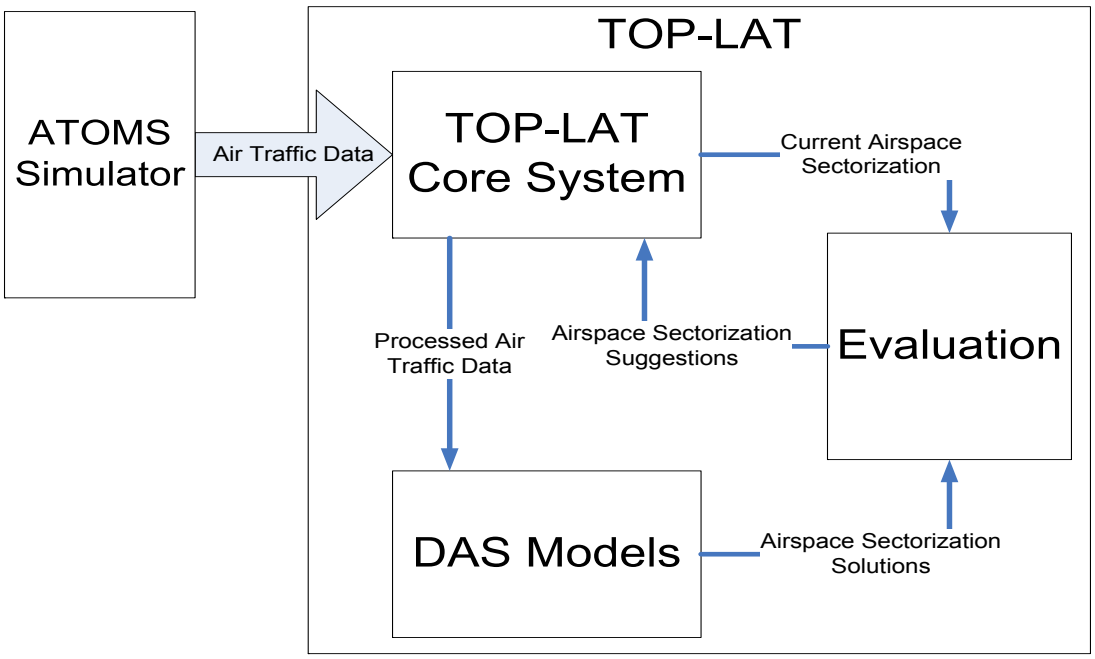


Figure 7.1: The Framework of Experiment and the Interactions among Three Systems: 3D DAS Models, ATOMS, and the Air Traffic Monitoring and Advisory System

As illustrated in the figure, TOP-LAT processes the air traffic data feedings from ATOMS and then processes and forwards them to DAS models for evaluation.

Both SPBM and CVDM in the experiments are used to partition the whole Australian airspace into sectors on a daily basis according to the air traffic demands in a year for the specified objectives same as described in Chapter 5. Meanwhile, several aspects are addressed in my experiments:

- Australian airspace modelling, where the boundaries of two control centres are considered when sectorizing the airspace.

- Air traffic scenarios: the present day and projected air traffic demands should be represented in the scenarios.

- Future ATM Concepts, such as UPT, have to be imported into the system when evaluating the compatibility of the proposed DAS models.
- A suitable task load formulation is needed when the DAS models take account the whole airspace including the TMA (Terminal Management Area) because the airports are the most congested area in the airspace and they have special procedures to handle more aircraft than en-route sectors.
- The comparison needs to take place between DAS models and against the present day sector configurations.

Each of them is addressed in the following sections.

7.1.1 Australian Airspace Modelling

The Australian Flight Information Region (FIR) has two control centres, Brisbane and Melbourne Control Centres, as shown in Figure A.1 in the Appendix, and both the FIR boundary and control centre boundaries have to be considered when the two DAS models partitioning the airspace. The reason is that my DAS models are not intending to modify the infrastructure of the Australian airspace which involves a large amount of additional efforts such as control centre location changes, international cooperations, and so on. All these are beyond the scope of this thesis.

Therefore, modifying both models in order to make sector boundaries to be aligned with their corresponding control centres is necessary. Weiler-Atherton Clipping algorithm (Weiler and Atherton, 1977) is used in each model when a portioning shape overlapping the boundaries of Australian FIR or two control centres.

As illustrated in Figure 7.2, the final sector shape is defined by the output of the Weiler-Atherton Clipping algorithm. The ownership of sectors is decided by the location of the point which is used to partition the airspace. As shown in the example, both sector A and B belong to Melbourne Control Centre because point (P) is located inside of Melbourne Control Centre. Algorithm 7.1 and 7.2 explain the details of how the clipping algorithm works in my DAS models. One or more

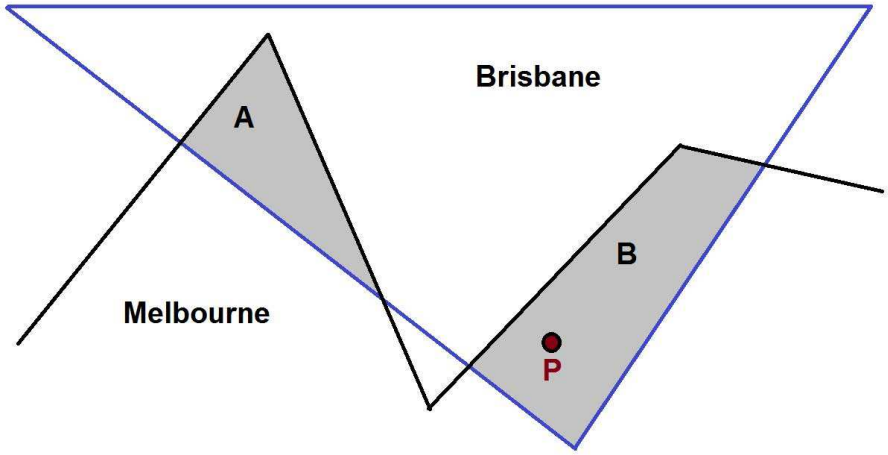


Figure 7.2: An Exapmle of Sectors Produced by Weiler-Atherton Clipping Algorithm when Partitioning Shape Overlapping on the Control Centre Boundary

sectors can be generated from this algorithm depending on the intersections between the partitioning shape and control centre boundary. It introduces uncertainty in the number of sectors in a given airspace. However, the air traffic distribution is not even and not all sectors are activated at the same time under the current airspace sector configurations. Meanwhile, my model is supposed to be flexible according to traffic demands in terms of the objectives I proposed. Therefore, the uncertainty of the sector number is not a problem at all. As the example shows in Figure 7.2, this algorithm gives the flexibility to the proposed models for adjusting the sector boundaries according to airspace constraints (e.g. FIR boundaries), geographic restrictions and other operational requirements (e.g. Terminal Management Area) when the convexity is not necessary.

Algorithm 7.1 Sector Lateral Clipping Algorithm

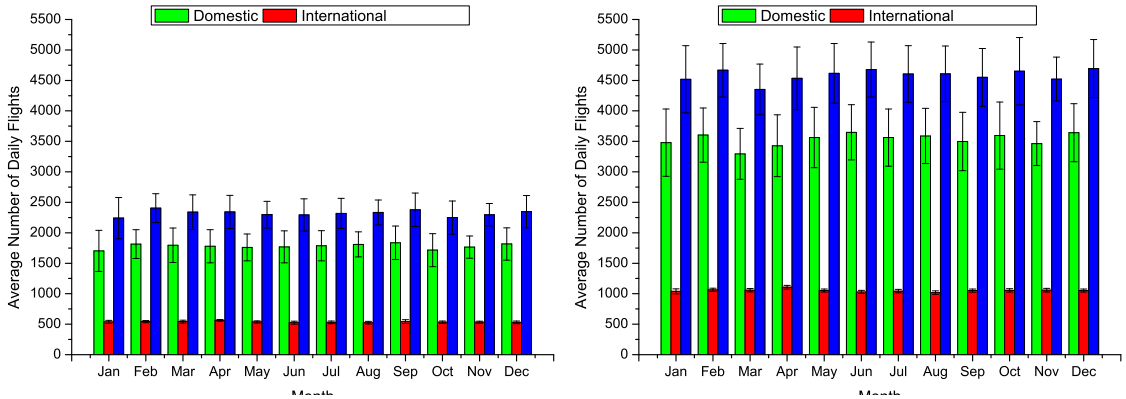
```
{Input: a sector(s) boundary (B_s), a point P generates sector s, and list of control centres (CC}
for all CC_i ∈ CC do
    if P ∈ CC_i then
        if B_s and CC_i are intersected then
            Get intersected sub-spaces (IS) by Algorithm 7.2
            for all IS_i ∈ IS do
                control centre of IS_i is CC_i
                Add IS_i into list of sectors
                RETURN
            end for
        end if
    end if
end for
```

Algorithm 7.2 Weiler-Atherton Clipping algorithm

1: {Input: a sector(s) boundary (B_s), a control centre boundary (CC_i)}
2: {Output: a list of intersected sub-spaces (IS)}
3: **Function WAPolygonClipping**(B_s, CC_i) {
4: Get all inbound intersected points (IP) of B_s and CC_i
5: Create an empty list IS for intersected sub-spaces
6: **if** IP is empty **then**
7: Add B_s into IS
8: **else**
9: Insert all points in IP and B_s into a list B' in a clockwise order
10: $n \Leftarrow |B'|$
11: Insert all points in IP and CC_i into a list C' in a clockwise order
12: **for** $i = 1 \rightarrow n$ **do**
13: **if** p_i is not visited **then**
14: **if** p_i is an intersection **then**
15: Create an empty list for a clipped polygon (CP)
16: Add p_i into CP
17: $flag \Leftarrow OUT$ {OUT means from the subject polygon to clip polygon. IN means from the clip polygon to subject polygon.}
18: **SearchVertices**(B', C', p_i, CP, $flag$)
19: Add CP to IS
20: **end if**
21: **end if**
22: Mark p_i as visited
23: **end for**
24: **end if**
25: RETURN IS
26: }
27: **Function SearchVertices**(B', C', p_i, CP, $flag$) {
28: **if** $flag \equiv OUT$ **then**
29: $flag \Leftarrow IN$, $PG \Leftarrow C'$ and $m = |C'|$
30: **else**
31: $flag \Leftarrow OUT$, $PG \Leftarrow B'$ and $m = |B'|$
32: **end if**
33: $j \Leftarrow$ index of p_i in PG
34: **for** $k = j + 1 \rightarrow m$ **do**
35: **if** p_k in PG is not visited **then**
36: Add p_k into CP and mark p_k as visited
37: **if** p_k is an intersection **then**
38: **SearchVertices**(B', C', p_k, CP, $flag$)
39: **end if**
40: **else**
41: RETURN
42: **end if**
43: **end for**
44: }

7.1.2 Artificial Flight Plans and Traffic Demands

Although TOP-LAT receives the air traffic data from AirServices Australia, the data can't be used in my experiments because of the confidentiality agreement in place. Another reason is that the projected air traffic demands and future ATM concepts don't exist in the field data. Therefore, artificial air traffic data simulated by ATOMS (Alam *et al.*, 2008) are fed into TOP-LAT for evaluating the proposed DAS models.



(a) Current Air Traffic Demands (Total 845176 Flights)

(b) Projected Air Traffic Demands (Total 1672562 Flights)

Figure 7.3: Average and Standard Deviation of Daily Flights in each Month of Artificial Flight Plans Representing Current and Projected Air Traffic Demands.

Scenarios presenting the current and projected air traffic demands are essential for my experiments. The present day air traffic demands within Australia can be estimated by public domain statistics. I have implemented a flight plan generator based on statistics obtained from one year of real traffic. The average and standard deviation of hourly departures for each airport pair are calculated. From these, the number of flight departures for each airport in a given period (one hour) of a day can be generated, based on a Gaussian Distribution. A Poisson Distribution for flight departure times is also built for each airport, based on real data, and is used to generate the departure time for each simulated flight. The average and standard deviation of daily flight movements within Australia FIR are also calculated based on the available statistics, and then the flight plans of one year can be generated.

The frequencies of route, cruising altitude, and cruising speed of each airport pair are built up by the historical air traffic data. Based on these distributions, flight route, cruising altitude, and cruising speed are assigned to each flight. Only commercial aircraft are considered in my experiments because the VFR flights are not managed by ATC. Therefore, the flight plans for the current air traffic demands contains around 85000 commercial flights for one year. Figure 7.3a shows the flight plans of one year which including the domestic and international flights.

Air traffic within Australia FIR will be doubled by 2020 (Shepherd *et al.*, 2007). The projected air traffic demands are produced by the same flight plan generator with different parameters: the number of hourly departure flights between each city pairs is doubled and the time interval between two flights departing from the same origin is reduced by half of the original time interval. As a result of this, the doubled air traffic demands are generated for my experiments as shown in Figure 7.3b

As illustrated in both figures, the flight plans generated from my flight plan generator not only represent the current and future air traffic demands but also reflect the variety of daily air traffic demands.

7.1.3 Future ATM Concepts Modelling

As mentioned before, DAS works in the future ATM system where other advanced ATM concepts are also enabled. It is necessary for a DAS approach to be compatible with other concepts. The prototype of the UPT is implemented by my flight plan generator. UPT means that an aircraft operator has freedom to operate his aircraft in a way meeting his individual targets such as safety, cost saving, on time performance, environment impact, and others (EUROCONTROL, 2007). In this UPT prototype, instead of assigning a fixed airway to a flight, a great circle route between two airports is assigned to it. In addition, half of the flights (selected randomly based on a uniform distribution) are allowed to deviate from their great circle route laterally. The deviation from the great circle route is varied from 0 to 10nm, and it is randomly produced based on a uniform distribution as well. This

prototype of UPT represents the UPT concept.

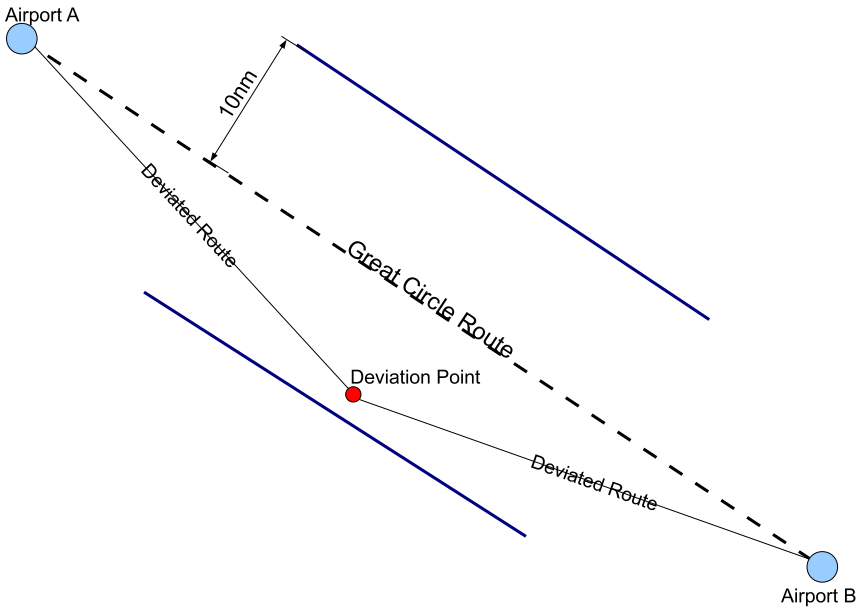


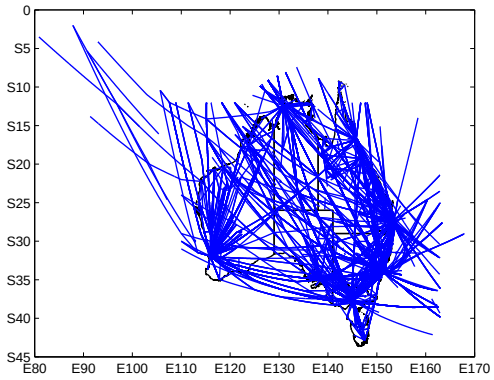
Figure 7.4: A Deviated Route Generated form the Great Circle Route between Two Airports by a Deviation Point

Figure 7.4 illustrates how a deviation route is produced by the great circle route and a deviation point. The location of the deviation point is randomly decided and it always sits outside the airport TMA area. The great circle routes and deviation routes together mimic the impact of UPT concept.

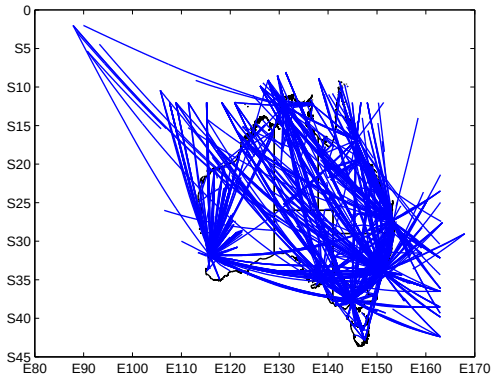
Air traffic within Australia FIR will be doubled by 2020 (Shepherd *et al.*, 2007). The projected air traffic demands are produced by the same flight plan generator with different parameters: the number of hourly departure flights between each city pairs is doubled and the time interval between two flights departing from the same origin is reduced by half of the original time interval. As a result of this, the doubled air traffic demands are generated for my experiments as shown in Figure 7.3b

As illustrated in both figures, the flight plans generated from my flight plan generator not only represent the current and future air traffic demands but also reflect the variety of daily air traffic demands.

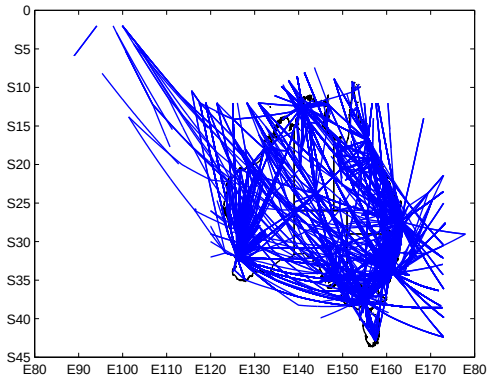
7.1.4 Scenarios



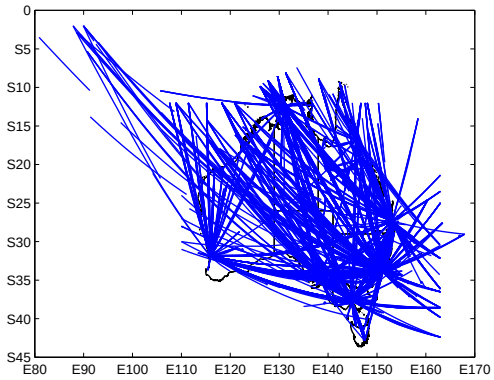
(a) One Day Traffic of Scenario 1 (Total 1966 Flights)



(b) One Day Traffic of Scenario 2 (Total 1949 Flights)



(c) One Day Traffic of Scenario 3 (Total 4494 Flights)



(d) One Day Traffic of Scenario 4 (Total 4496 Flights)

Figure 7.5: Examples of the Air Traffic in One Day from the 4 Scenarios

Based on air traffic demands estimation and future ATM concepts modelling, 4 different scenarios are built up to investigate the proposed 3D DAS models:

- Scenario 1: All flights fly along the classic routes assigned by the flight plan generator and the traffic volume is based on current air traffic demands.
- Scenario 2: All flights fly along the great circle or deviated routes assigned by the flight plan generator and the traffic volume is based on current air traffic demands.

- Scenario 3: All flights fly along the classic routes assigned by the flight plan generator and the traffic volume is based on projected air traffic demands.
- Scenario 4: All flights fly along the great circle or deviated routes assigned by the flight plan generator and the traffic volume is based on projected air traffic demands.

Table 7.1: Summary of 3D DAS Model Investigation Scenarios

	Classic Route	Great Circle and Deviated Routes
Current Air traffic Demands	Scenario 1	Scenario 2
Projected Air traffic Demands	Scenario 3	Scenario 4

These four scenarios are summarised in Table 7.1.

Examples of the air traffic in one day from these scenarios are shown in Figure 7.5. As the trajectories illustrate in the figures, Scenario 1 and Scenario 3 have the similar flight foot prints in the space, while Scenario 2 and Scenario 4 have different tracks of flight movements. This different trajectory patterns with the different air traffic volume help me to investigate the proposed 3D DAS models.

7.1.5 Relative Task Load

The whole Australian airspace including TMA is considered in my experiments. The airports are the most congested area in the airspace, the task load measured by the absolute air traffic hits used in the last chapter is no longer suitable in this context. Therefore, a relative measurement of the task load by air traffic hits is applied in my experiments. The task load is scaled based on the distance between the flight and its origin and destination as shown in Equation 7.1.

$$W = \begin{cases} W & \text{if } d_o > D \text{ and } d_d > D \\ W \times \frac{d_o}{D} & \text{if } d_o \leq D \\ W \times \frac{d_d}{D} & \text{if } d_d \leq D \end{cases} \tag{7.1}$$

In the equation, D is a constant and is defined by 50nm from the center of an airport based on the TMA definition of FAA/EUROCONTROL. d_o and d_d are the

distances between the flight and its origin and destination respectively. The task load by the flights flying over airports is not scaled according to this equation. Using this equation, the benchmarks from current sector configurations are built.

7.1.6 Benchmark of the Current Sectorization

According to the previous experiments, three objectives are used in my 3D DAS models: minimizing standard deviation of task load across sectors, maximizing average sector flight time, and maximizing distance between sector boundaries and traffic flow crossing points. They are calculated based on current Australian airspace sector configuration by the 4 scenarios as described in Section 7.1.4. These results are the benchmarks for my evaluation.

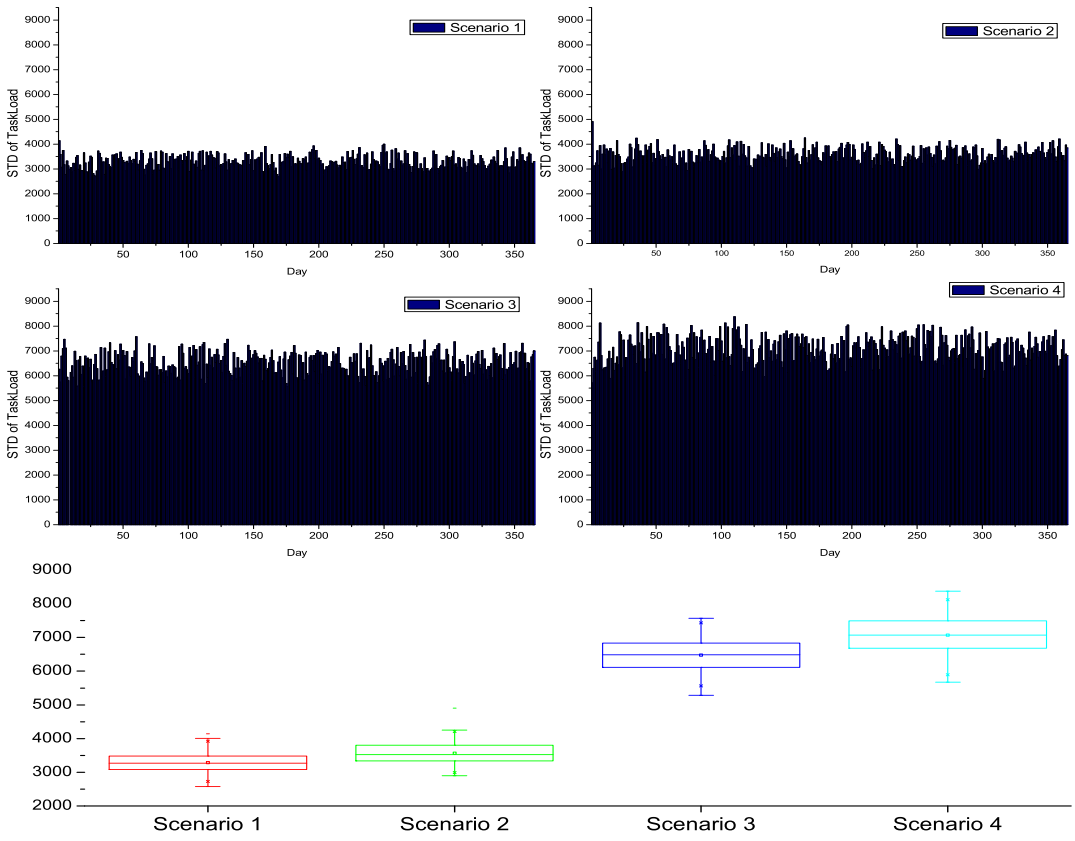


Figure 7.6: Benchmark: The Task Load Standard Deviation of Sectors based on the 4 Scenarios

Figure 7.6 shows the task load standard deviation in each day of 4 scenarios based on the current airspace sectorization. It illustrates the large variance of task load when the air traffic increased and it also shows that the task load standard deviation is a bit higher when the great circle, and deviated routes are enabled in the current airspace configuration.

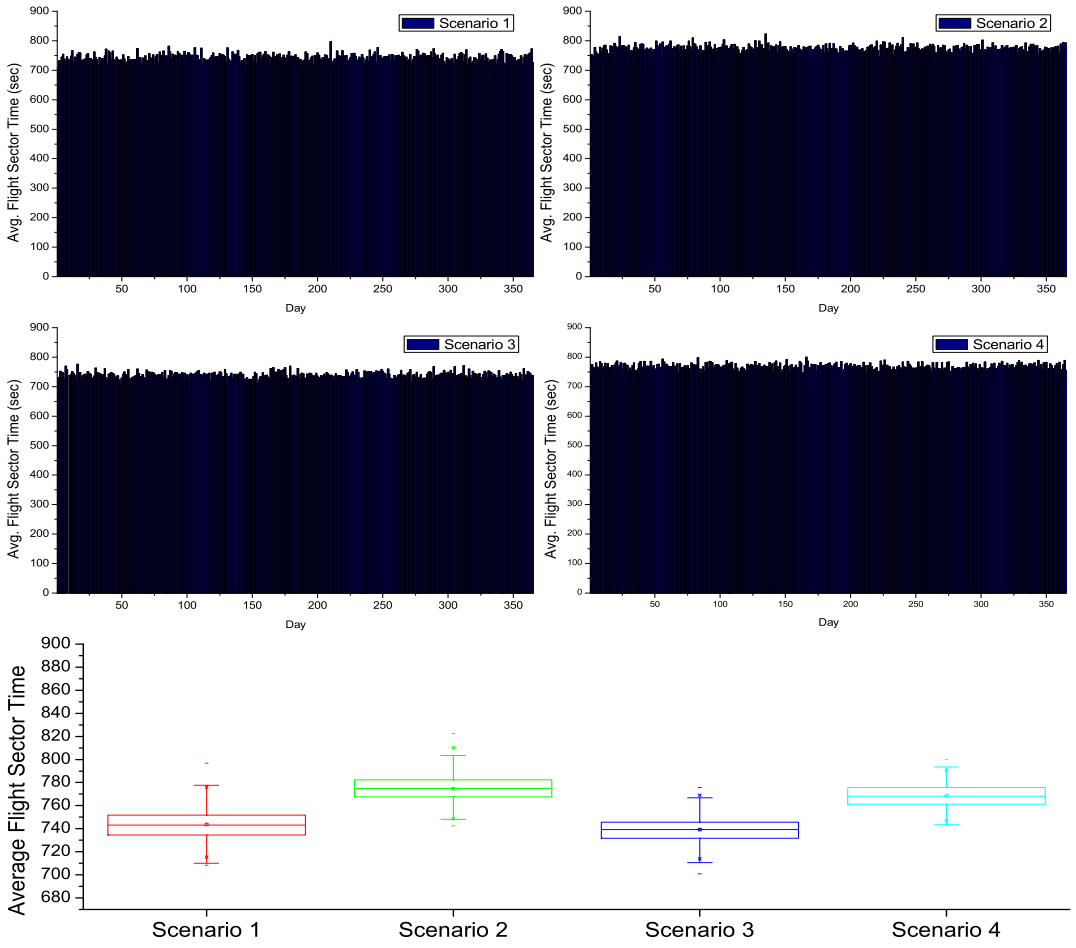


Figure 7.7: Benchmark: The Average Flight Sector Time based on the 4 Scenarios

The statistics on the average flight sector time of one day for the yearly traffic of 4 scenarios is shown in Figure 7.7. The traffic volume doesn't affect the average flight sector time but the average flight sector time increases when the great circle and deviated routes are used by flights. However, this effect has some impacts on the task load standard deviation as shown in Figure 7.6 because it implicitly increases

a flight stay longer in some sectors which increases the ATC task load.

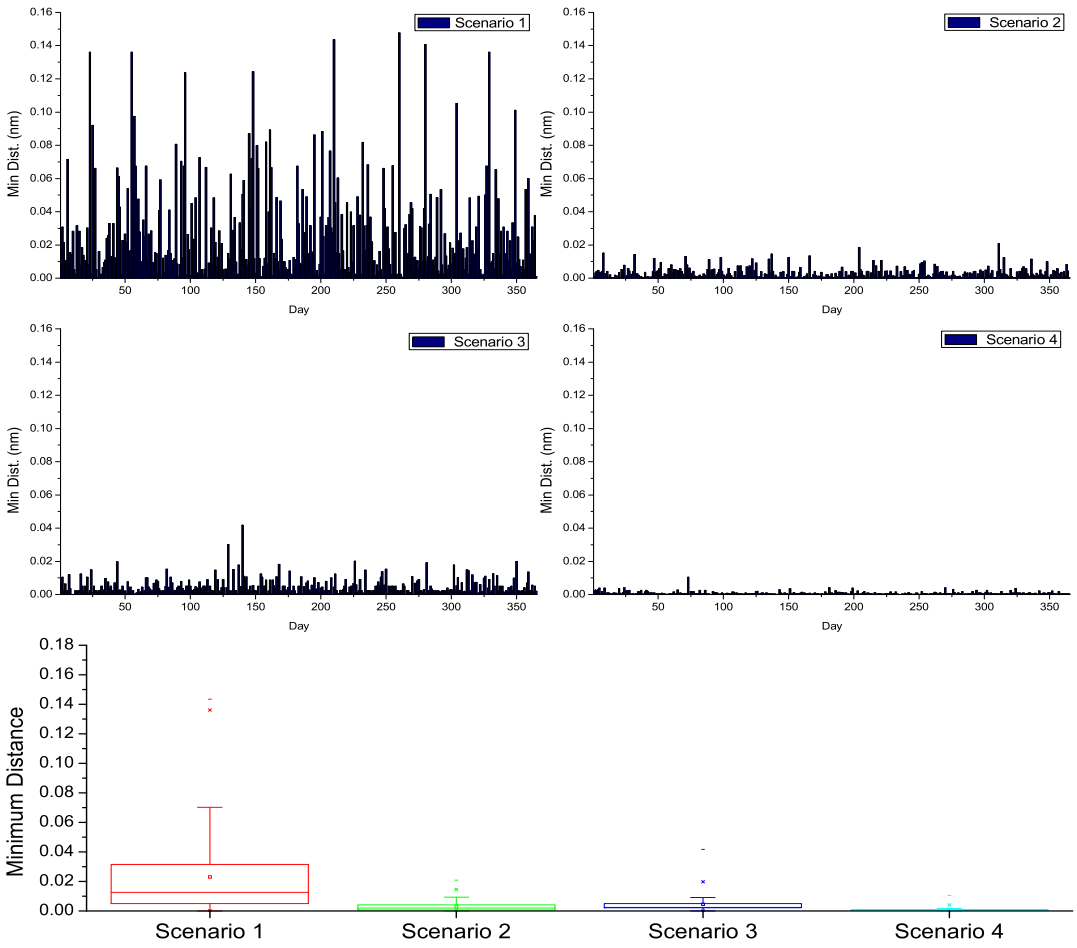


Figure 7.8: Benchmark: The Minimum Distance between Sector Boundaries and Air Traffic Flow Crossing Points based on the 4 Scenarios

When the traffic volume doubled and the flexibility of flight routes introduced, the minimum distance between sector boundaries and traffic flow crossing points are difficult to be maintained at some level as shown in Figure 7.8. More flights flying in the airspace may have more traffic flow crossing points, and more flexible flight routes may cause more potential conflicts. Therefore, the current sectorization works better under the current traffic situation (scenario 1) than others.

Under an air traffic distribution, some sectors are not activated in some days, but some congested sectors potentially can become the bottlenecks of air traffic management. Therefore, I need to identify the maximum task load among sectors

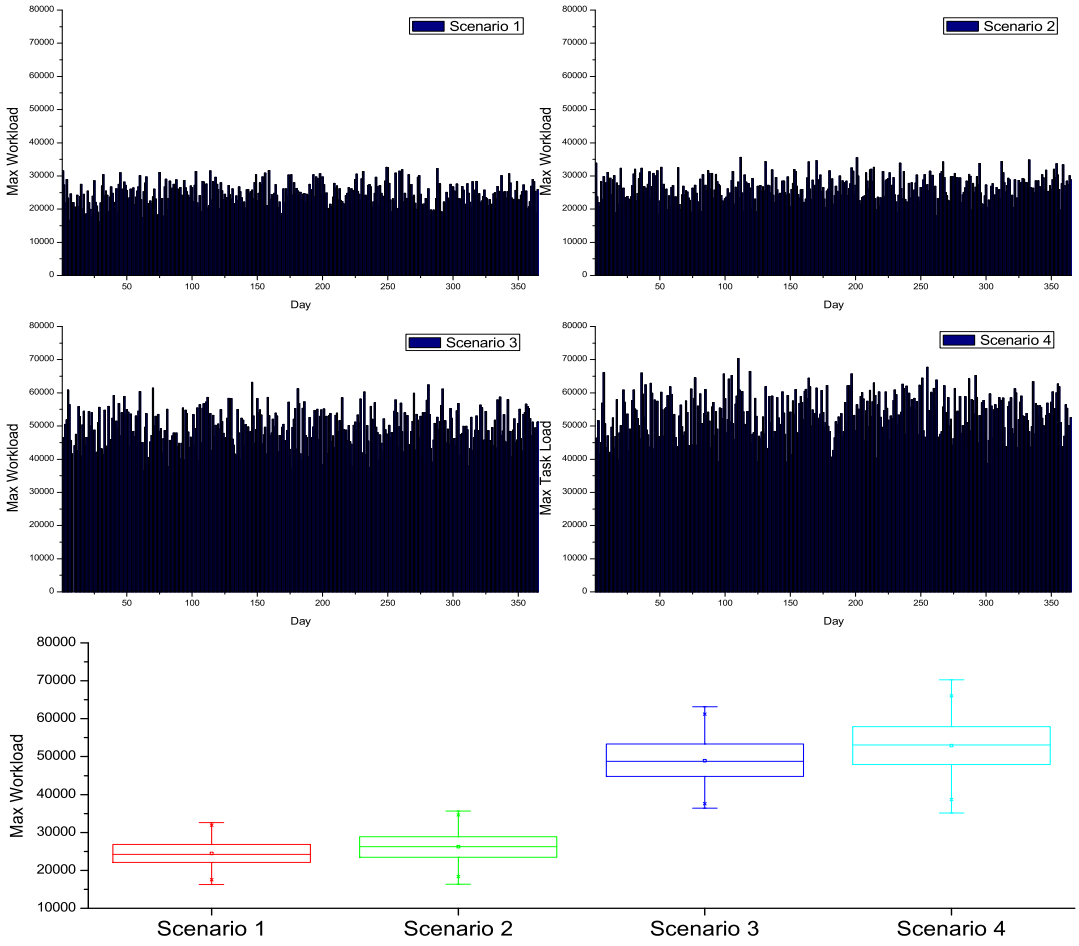


Figure 7.9: Benchmark: The Maximum Task Load among Sectors based on the 4 Scenarios

in each day too. Figure 7.9 shows the statistics on the maximum task load of each day for the yearly traffic of 4 scenarios. The maximum task load of activated sectors in the scenarios with projected traffic volume is almost twice of the scenarios with the present day traffic volume as shown in the figure.

Although my first objective (minimizing standard deviation of task load) works on task load balancing among the activated sectors, it doesn't intend to reduce the maximum task load of activated sectors. Hence, I also introduce a forth objective for reducing the maximum task load of activated sectors (Basu *et al.*, 2009) in my 3D DAS Models. The fitness function of it is formulated as Equation 7.2. During

the evolution, this objective is to be minimized.

$$\begin{aligned} &\min(F(W_{max})) = \min(\max_{k=1}^{K}(W_k)) \\ &W_k = \sum_{i=1, x_{ij} \in S_k}^{N} |t_i(x_{ij})| \end{aligned} \tag{7.2}$$

7.2 Evaluation Experiments and Results

Both Support Plane Bisection (SPBM) and Constrained Voronoi Diagram (CVDM) Models are investigated in my experiments. According to the benchmarks based on current Australian sectors, each model is used to sectorize the whole Australian airspace daily based on the traffic demands. Two models, four scenarios, and 365 days require total 2920 runs.

The current Australian airspace contains 229 sectors in total and they are assumed all open for civilian aircraft in the experiments. The exact sector number can't be guaranteed when considering the Australian FIR boundaries as mentioned in Section 7.1.1. But the number of sectors in my models is targeted around 230 to be compatible with the current configuration. Hence, the chromosome length of SPBM is set as 600 (200 points) and the chromosome length of CVDM is set as 480 (160 points) because of the different data structures and algorithms as described in Chapter 4. Four objectives are addressed in my models in my experiments:

- Minimizing standard deviation of task load across sectors
- Maximizing average sector flight time
- Maximizing distance between sector boundaries and traffic flow crossing points
- Minimizing maximum task load of sectors

The random seed initializing NSGA-II is generated by a uniform distribution and within the range from 0 to 1. All other parameters are as same as the parameters used in Chapter 5.

7.2.1 Experiments Results

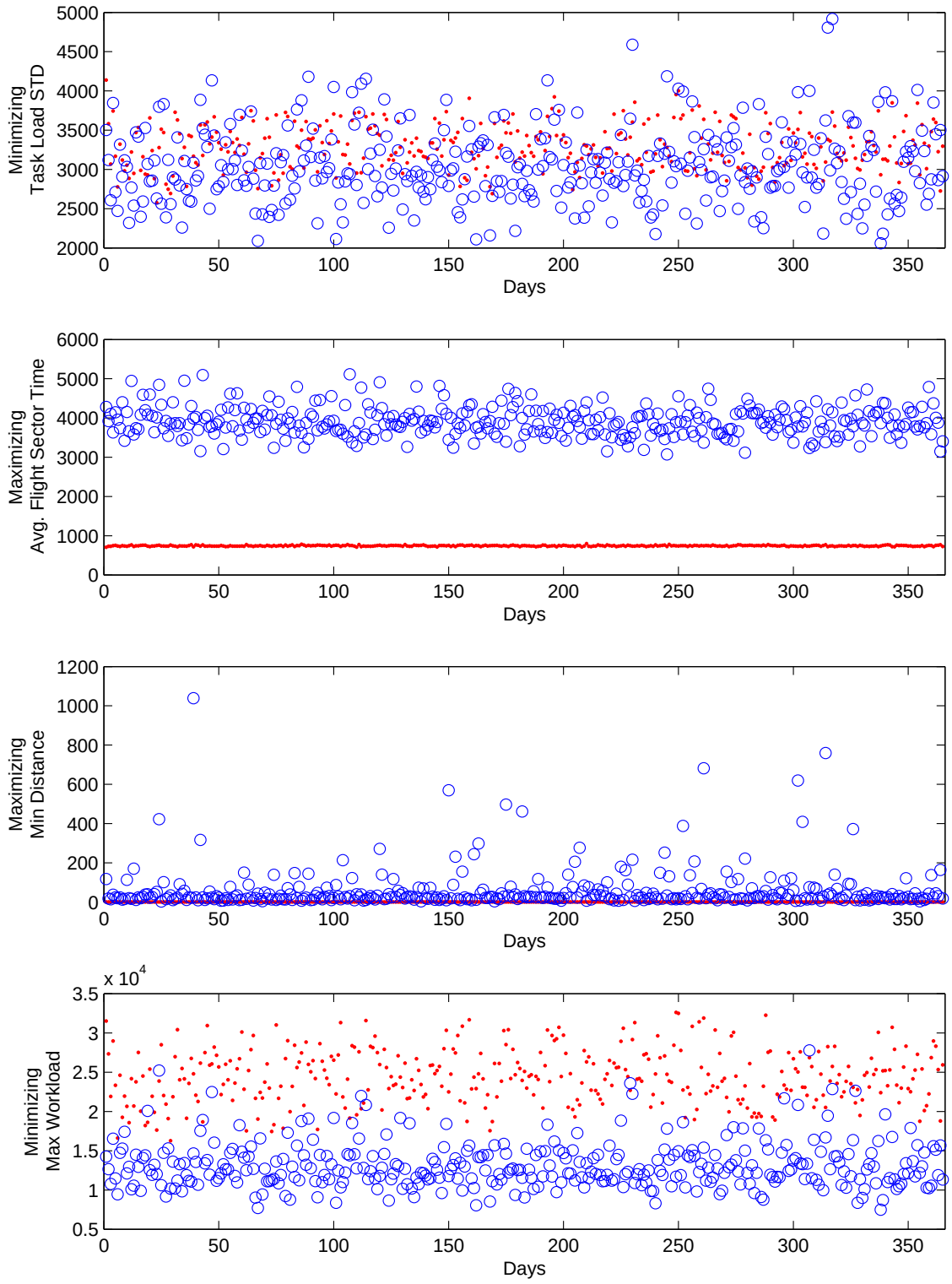


Figure 7.10: Results of Daily Airspace Sectorization by SPBM in Scenario 1: Average values of each objective in the Pareto sets of the daily airspace sectorization solutions generated by SPBM.

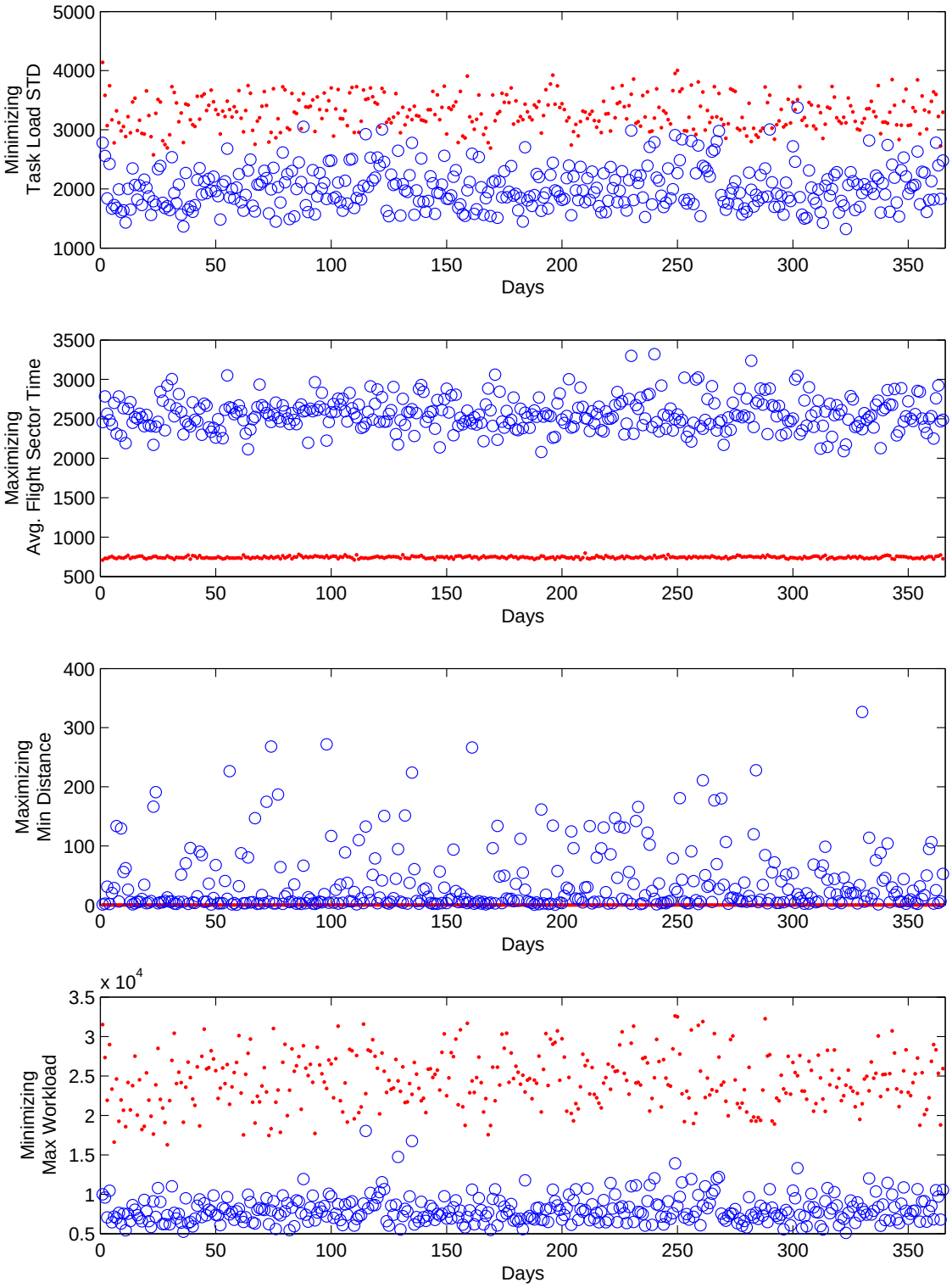


Figure 7.11: Results of Daily Airspace Sectorization by CVDM in Scenario 1: Average values of each objective in the Pareto sets of the daily airspace sectorization solutions generated by CVDM.

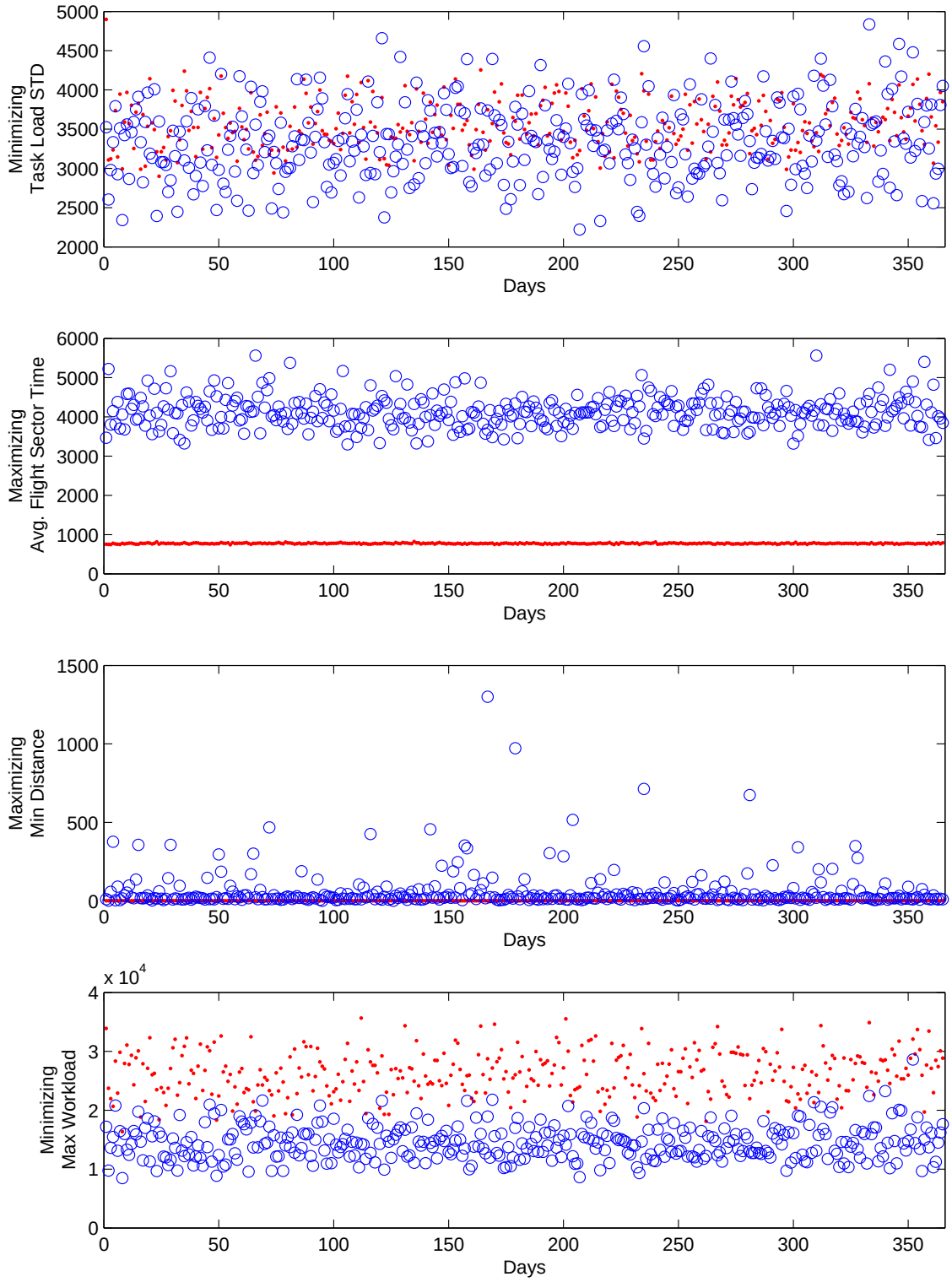


Figure 7.12: Results of Daily Airspace Sectorization by SPBM in Scenario 2: Average values of each objective in the Pareto sets of the daily airspace sectorization solutions generated by SPBM.

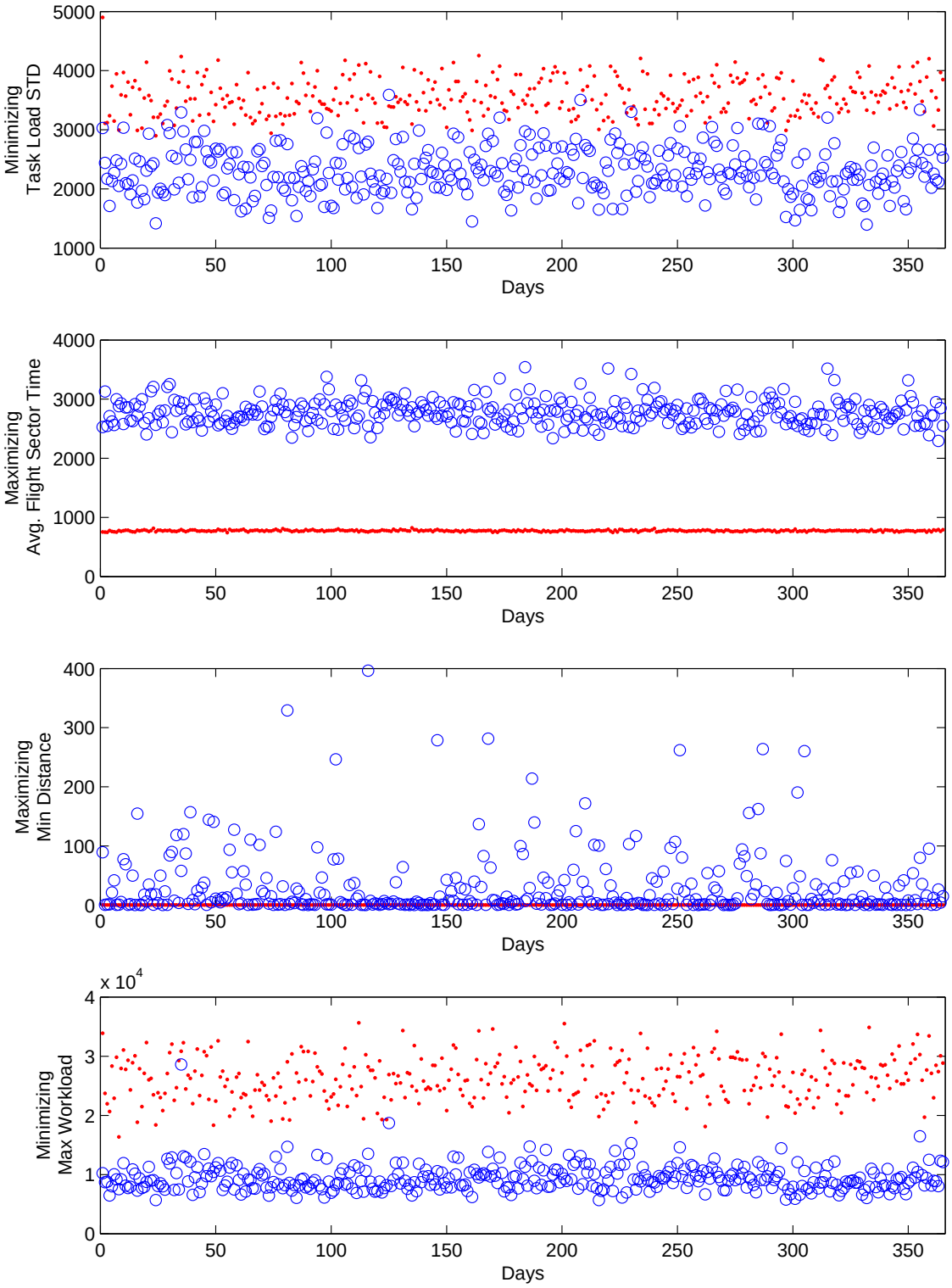


Figure 7.13: Results of Daily Airspace Sectorization by CVDM in Scenario 2: Average values of each objective in the Pareto sets of the daily airspace sectorization solutions generated by CVDM.

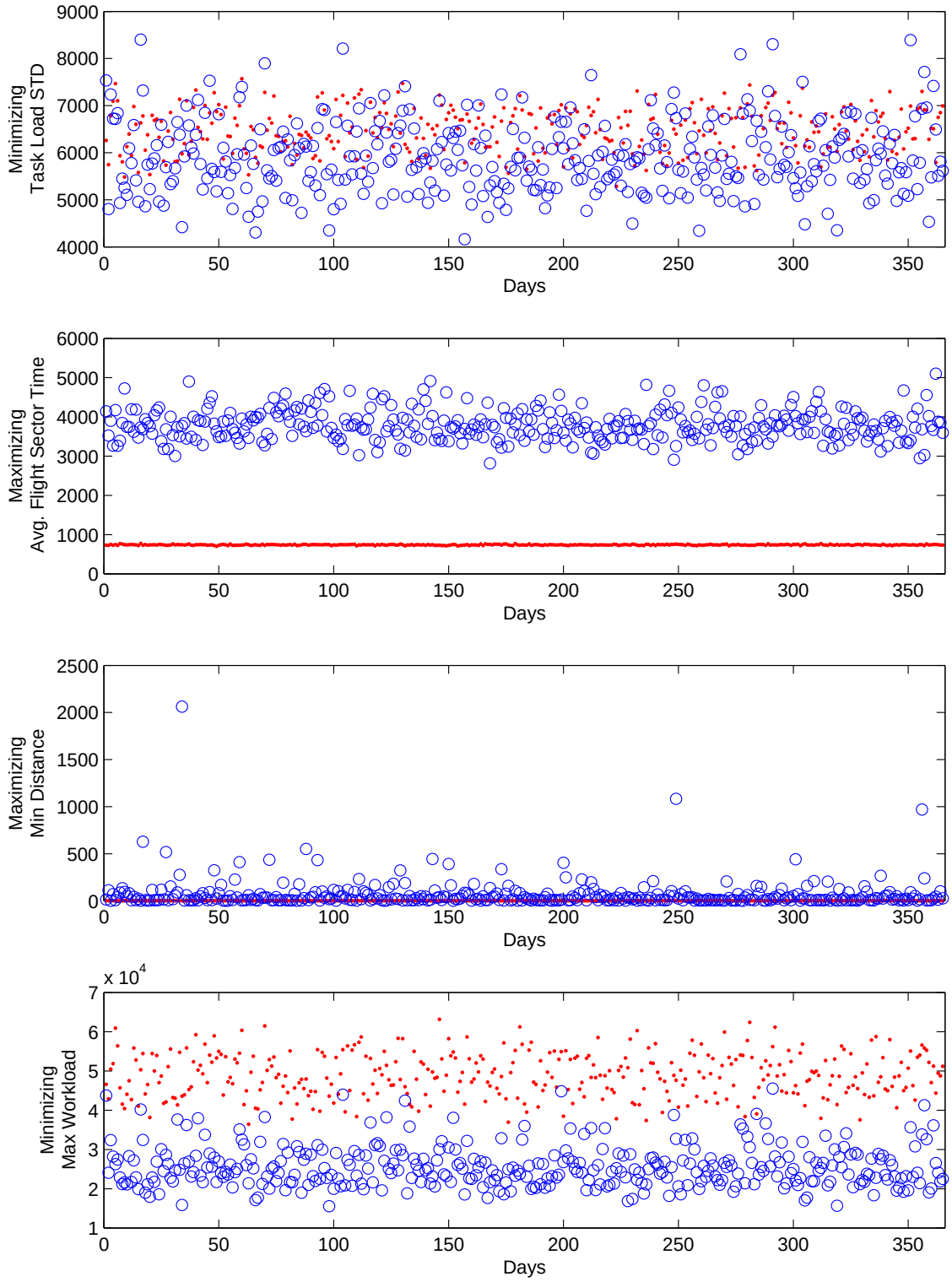


Figure 7.14: Results of Daily Airspace Sectorization by SPBM in Scenario 3: Average values of each objective in the Pareto sets of the daily airspace sectorization solutions generated by SPBM.

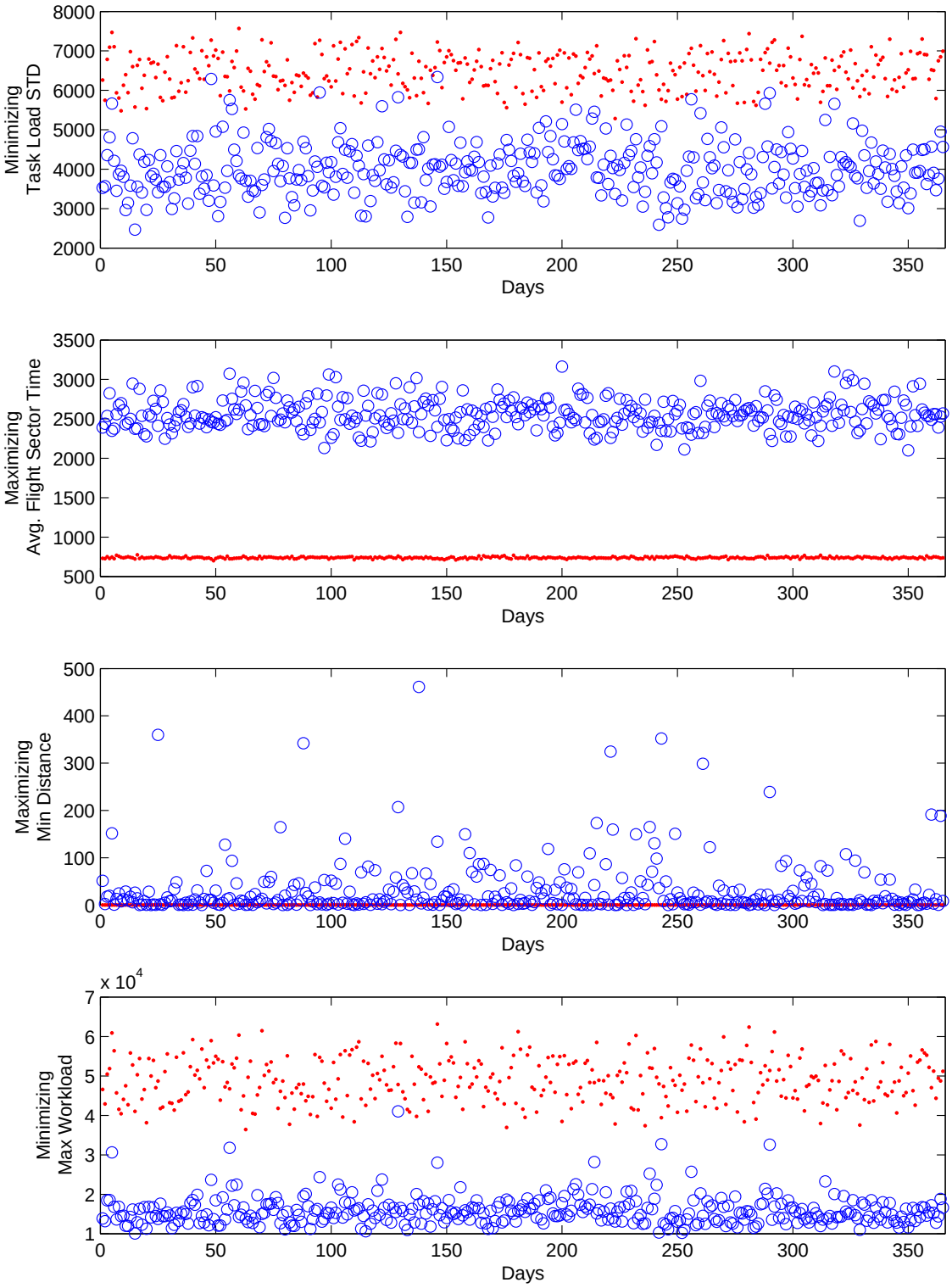


Figure 7.15: Results of Daily Airspace Sectorization by CVDM in Scenario 3: Average values of each objective in the Pareto sets of the daily airspace sectorization solutions generated by CVDM.

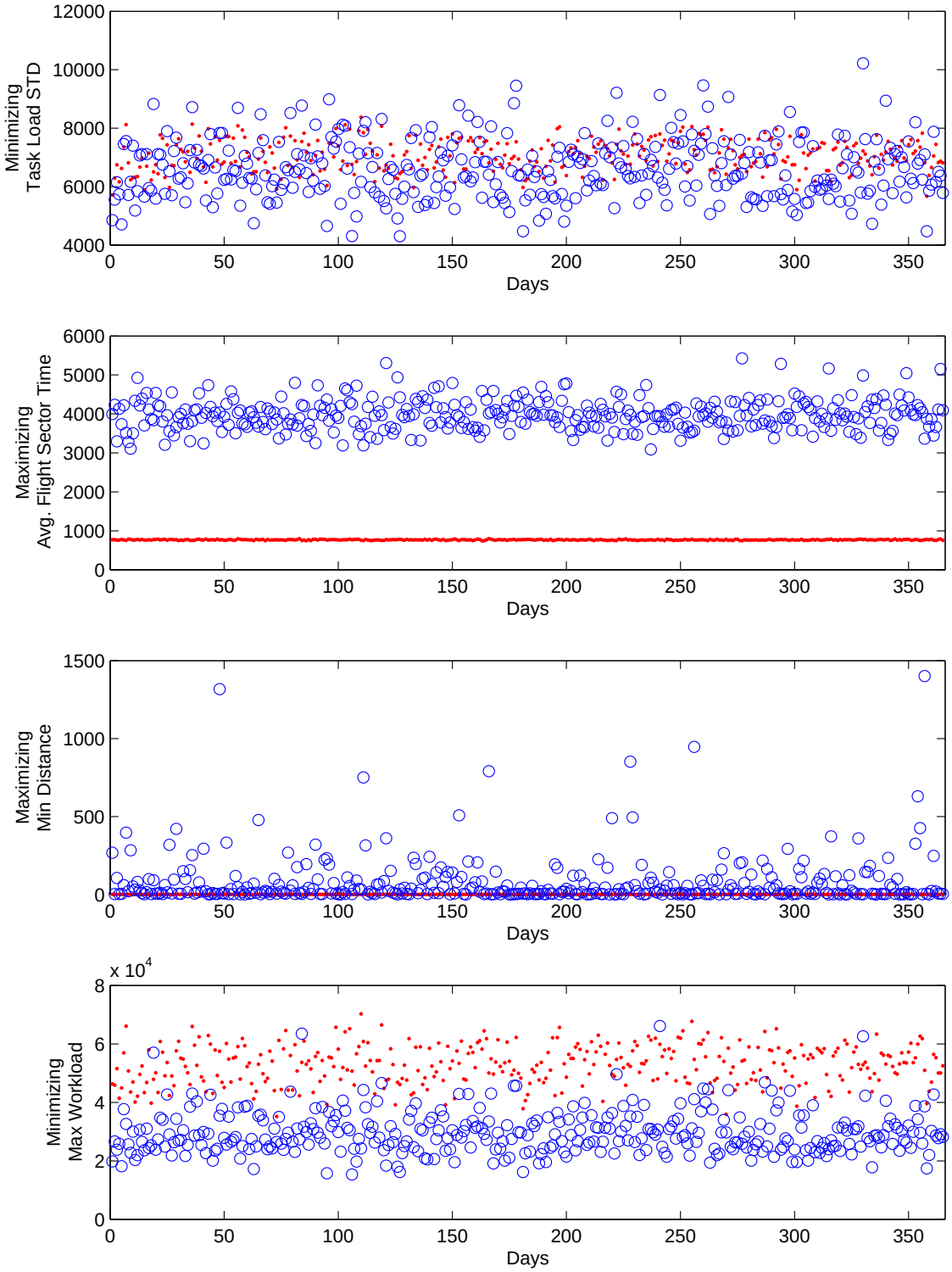


Figure 7.16: Results of Daily Airspace Sectorization by SPBM in Scenario 4: Average values of each objective in the Pareto sets of the daily airspace sectorization solutions generated by SPBM.

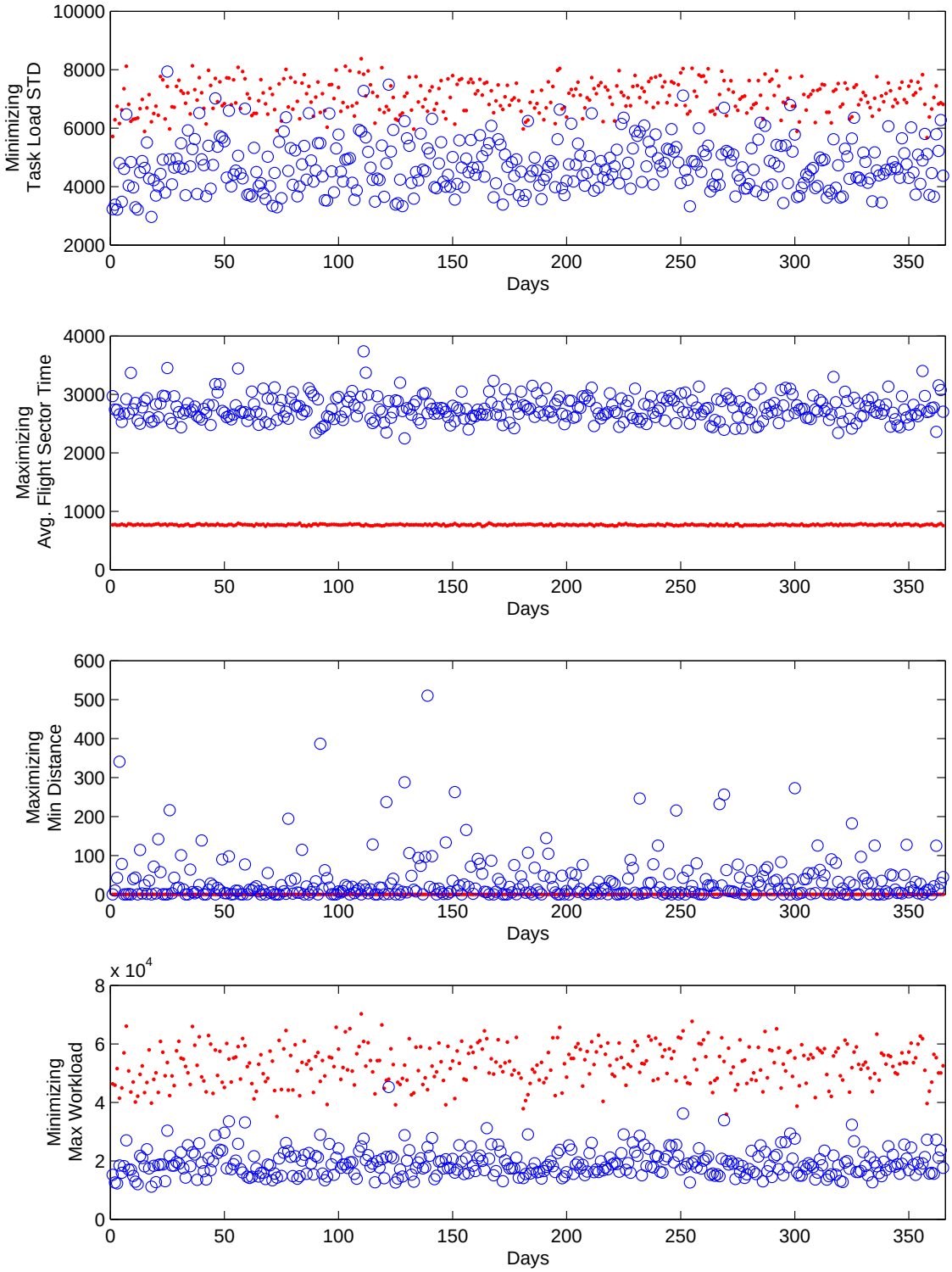


Figure 7.17: Results of Daily Airspace Sectorization by CVDM in Scenario 4: Average values of each objective in the Pareto sets in the daily airspace sectorization solutions generated by CVDM.

The average values of each objective in the Pareto sets of the daily airspace sectorization solutions for 4 scenarios from both models are visualized from Figure 7.10 to Figure 7.17. They show the overall performance of Pareto sets compared against the benchmarks. The red dots in each figure are the benchmarks of each objective calculated based on current airspace sectorization.

As shown in the figures, both SPBM and CVDM can generate better airspace sectorization for most days in terms of the 4 specified objectives, especially for the average flight sector time. Although the minimum distance between sector boundaries and traffic flow crossing points is still lower than I expected in some days, they are much better than the benchmarks as shown in the figures. However, the task load standard deviation and maximum task load from SPBM are not as good as expected on a few days.

On the other hand, the diversity of traffic demands (number of flights) and distributions (flight trajectories) of different days may have some effects on the results of task load standard deviation and maximum task load among days. The number of flights are sorted from low to high and plotted as the blue lines in the following figures (from Figure 7.18 to 7.25). The achieved average values of two objectives, minimizing task load standard deviation and minimizing maximum task load, are also plotted along with the corresponding number of flights for every day. Then, A linear fit based on least squares, as shown in Equation 7.3, is used to approximate the trends of both objectives based on the number of flights. These trends are plotted as green lines in the figures. In Equation 7.3, x is the number of flights in a day, and y is the task load standard deviation or the maximum task load of sectors.

$$\begin{aligned} p(y) &= p_2 \times x + p_1 \\ p_1 &= \frac{\sum_{i=1}^{n}(x_i-\bar{x})\times(y_i-\bar{y})}{\sum_{i=1}^{n}(x_i-\bar{x})^2} \\ p_2 &= \bar{y} - p_1 \times \bar{x} \\ \bar{x} &= \frac{\sum_{i=1}^{n} x_i}{n} \\ \bar{y} &= \frac{\sum_{i=1}^{n} y_i}{n} \end{aligned} \tag{7.3}$$

These figures show that both objectives become worse when the number of flights increase. However, the slope of the objective trends are less than the slope of the number of flights growth as illustrated in the figures. It means that both objectives become worse more slowly than the increments of flights number. In other words, both models are adaptive to the air traffic demands changes. Comparing with the Scenario 1 and 2, the values of the both objectives are almost doubled in the Scenario 3 and 4 because the density of traffic is doubled but the number of sectors are not doubled in my experiments. On the other hand, both objectives in Scenario 2 are a little worse than the results in Scenario 3, meanwhile Scenario 3 is also a bit better than Scenario 4. The reason is that the air traffic are more congested along the great circle routs between airports when the scenarios enable the great circle and deviated routes. It is more difficult to achieve the task load balancing and to push down the maximum task load where the air traffic is highly congested along some routes instead of being distributed to different airways.

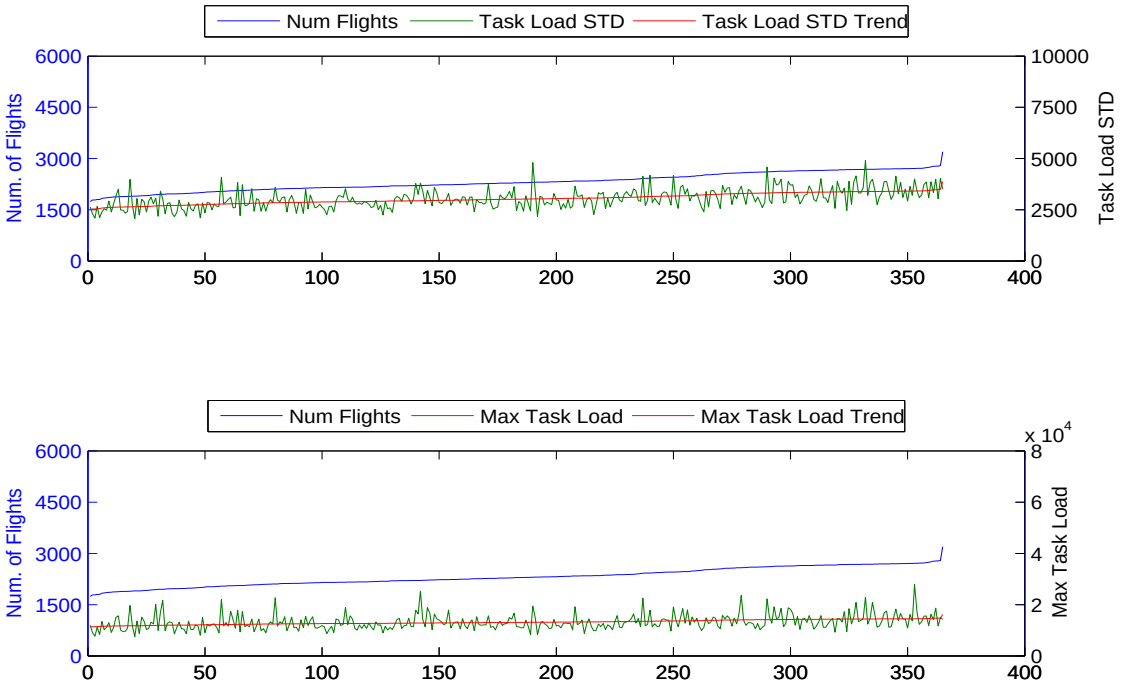


Figure 7.18: The trends of average Task Load Standard Deviation and Maximum Task Load of daily airspace sectorization by SPBM along with the number of flights increasing in Scenario 1

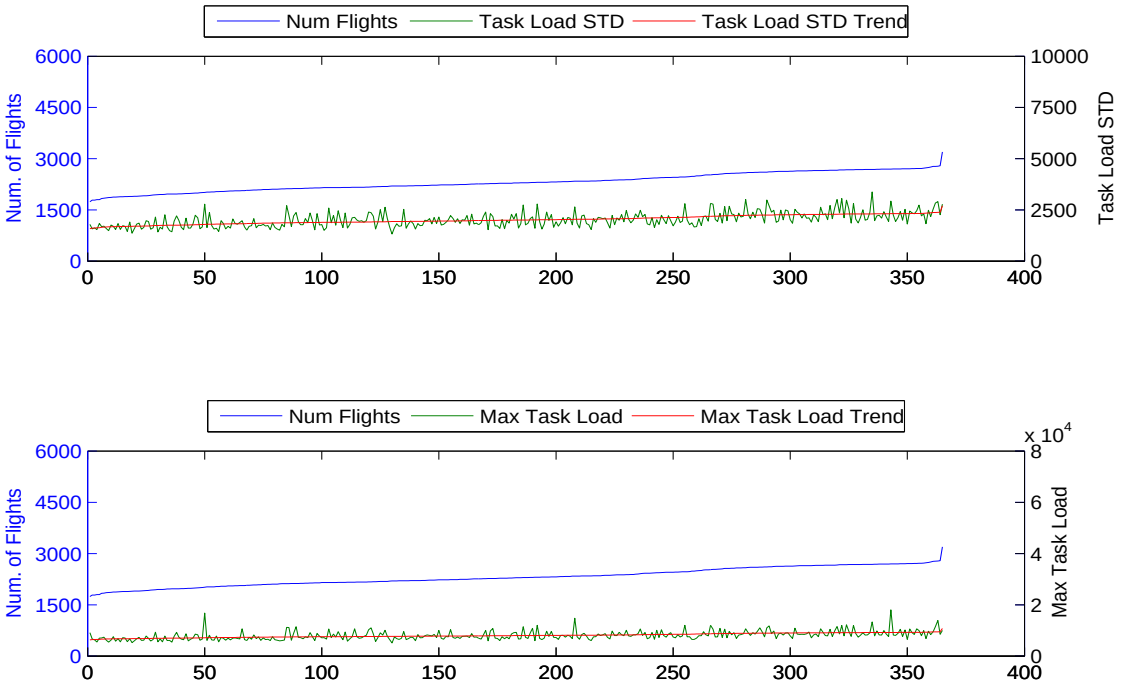


Figure 7.19: The trends of average Task Load Standard Deviation and Maximum Task Load of daily airspace sectorization by CVDM along with the number of flights increasing in Scenario 1

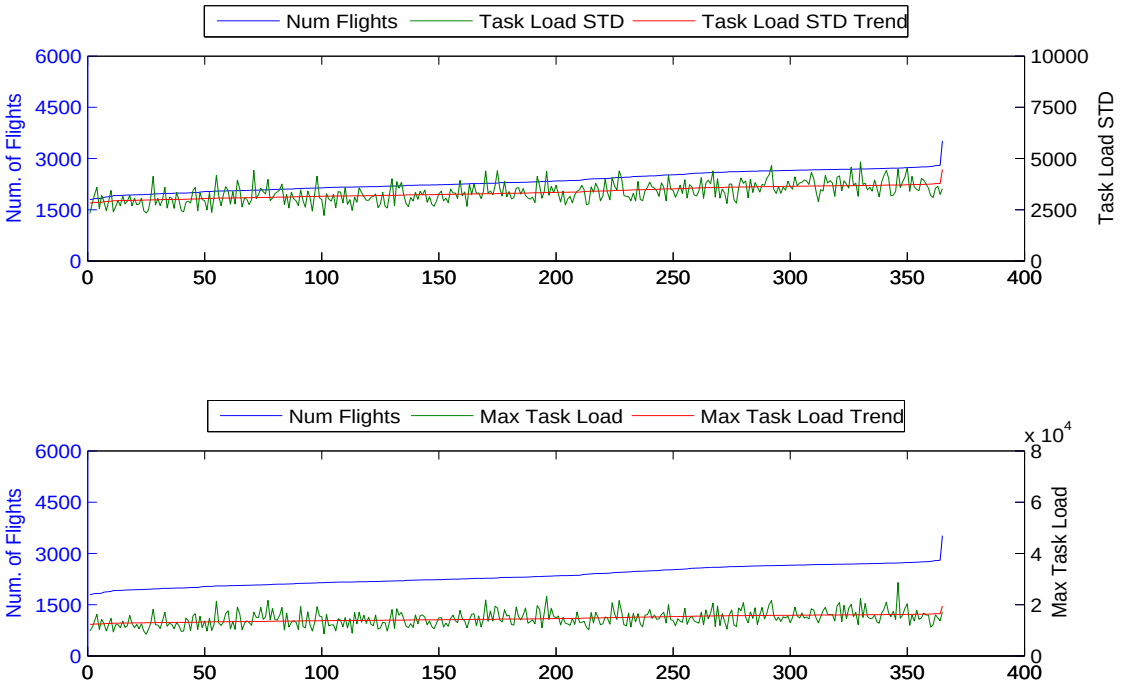


Figure 7.20: The trends of average Task Load Standard Deviation and Maximum Task Load of daily airspace sectorization by SPBM along with the number of flights increasing in Scenario 2

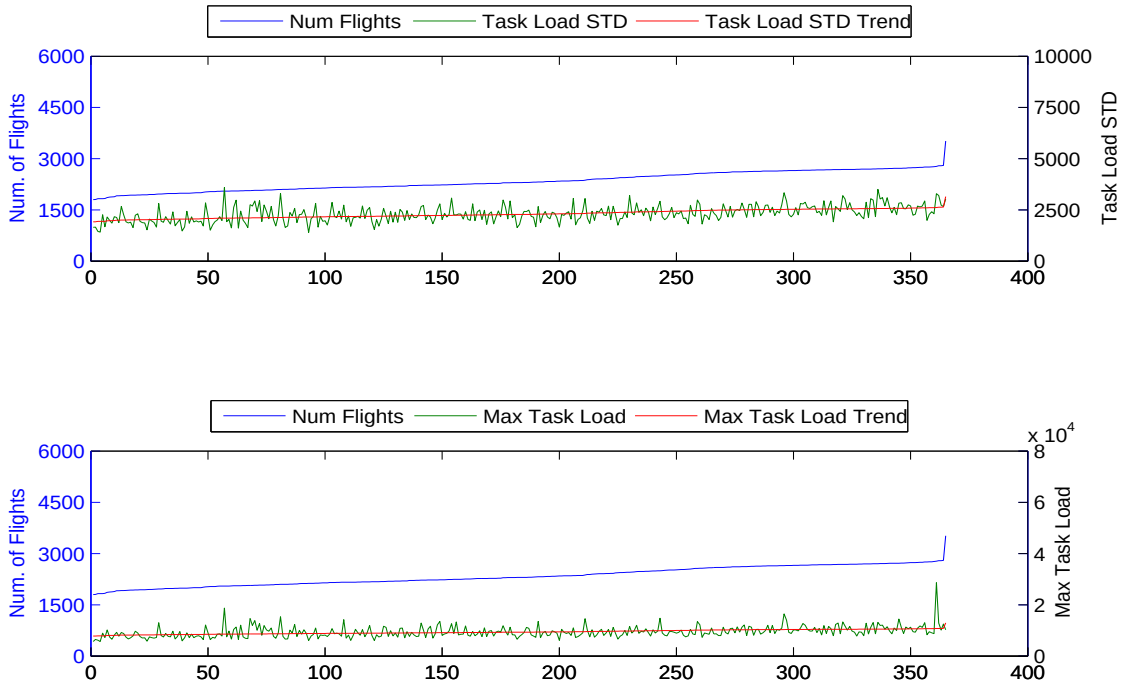


Figure 7.21: The trends of average Task Load Standard Deviation and Maximum Task Load of daily airspace sectorization by CVDM along with the number of flights increasing in Scenario 2

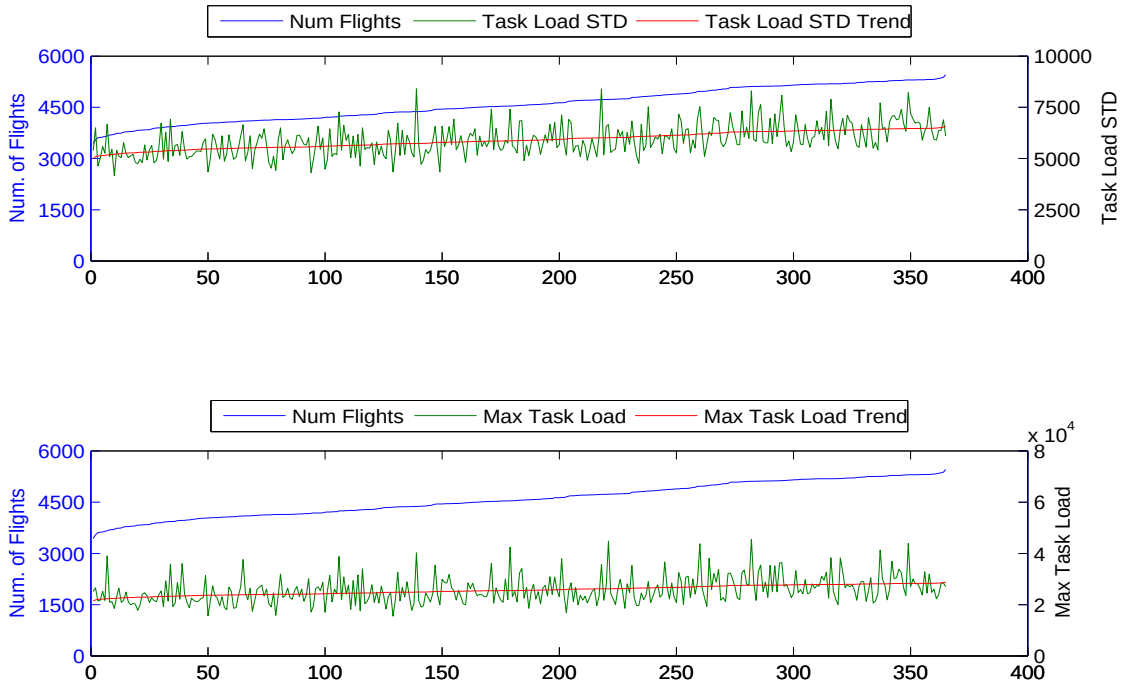


Figure 7.22: The trends of average Task Load Standard Deviation and Maximum Task Load of daily airspace sectorization by SPBM along with the number of flights increasing in Scenario 3

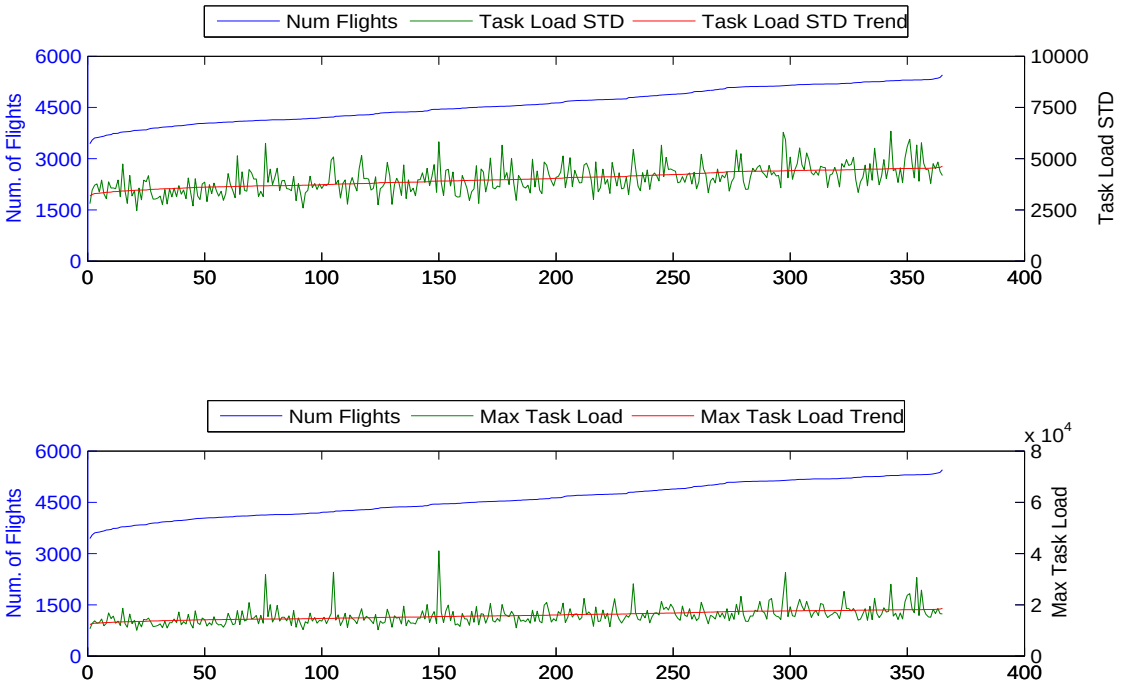


Figure 7.23: The trends of average Task Load Standard Deviation and Maximum Task Load of daily airspace sectorization by CVDM along with the number of flights increasing in Scenario 3

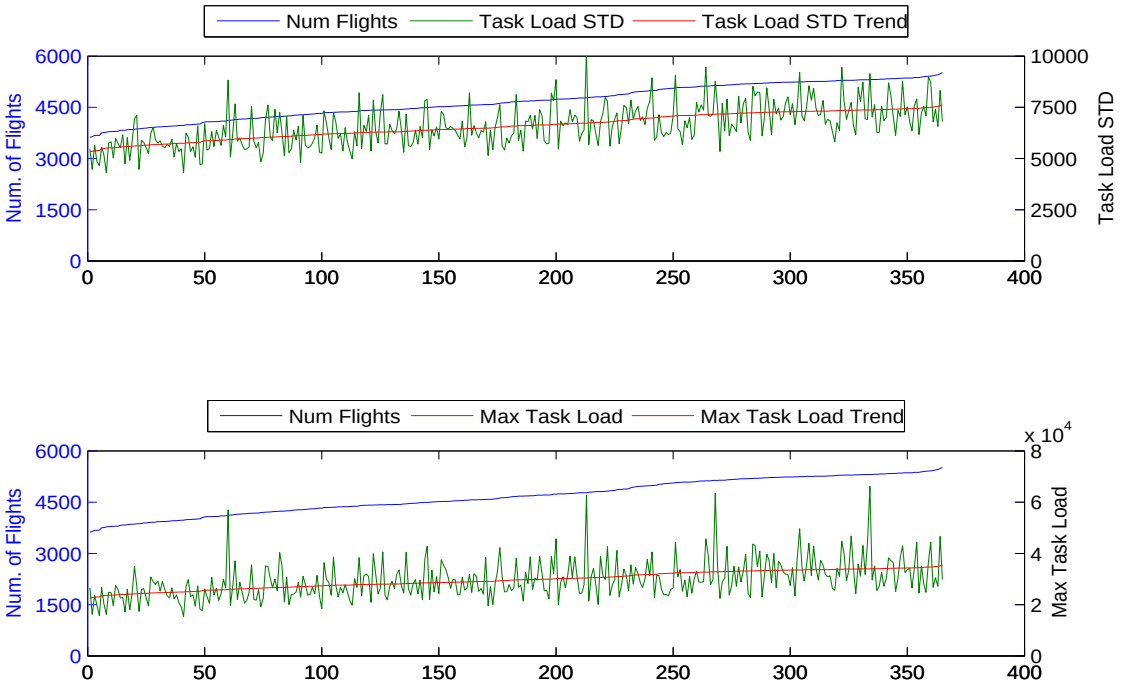


Figure 7.24: The trends of average Task Load Standard Deviation and Maximum Task Load of daily airspace sectorization by SPBM along with the number of flights increasing in Scenario 4

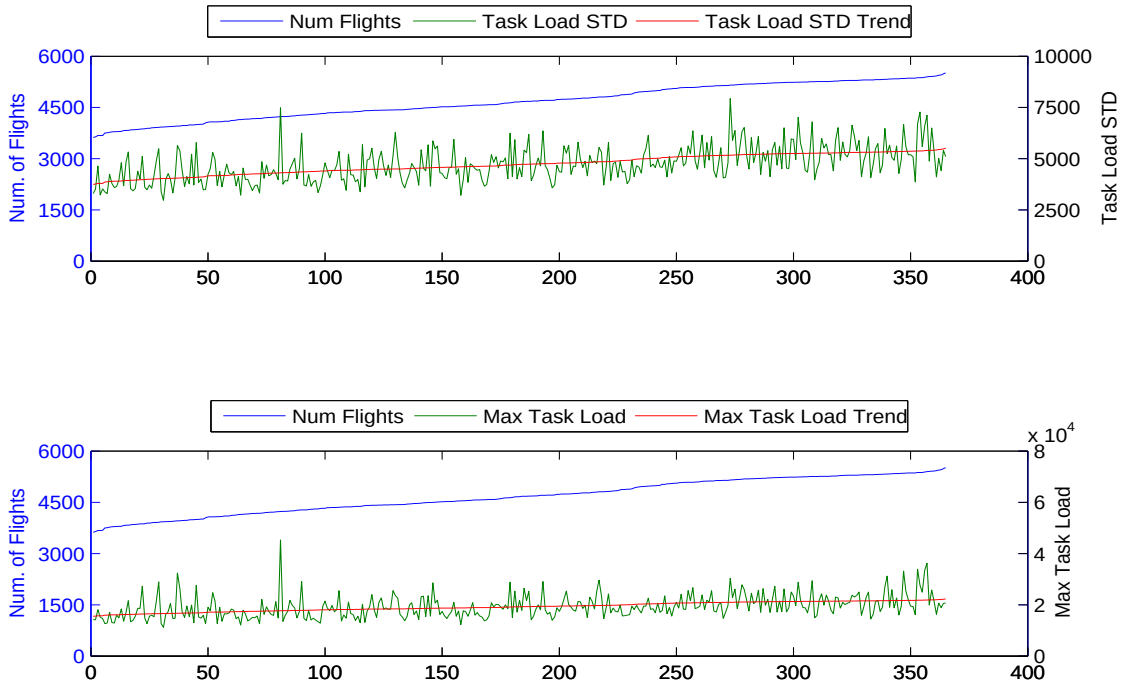


Figure 7.25: The trends of average Task Load Standard Deviation and Maximum Task Load of daily airspace sectorization by CVDM along with the number of flights increasing in Scenario 4

7.2.2 Comparison against the Present Day Sectorization

In order to compare the results with the current Australian airspace sectorization, five typical solutions are picked out from each solutions set of each day. The first four are the solutions with the best achieved objective value on each objective independently. The last one is the solution with a balanced performances on all objectives. A balanced solution is a solution with the highest weighted sum of all objectives. The weights of each objective are equal to each other. Here, the minimization objectives are converted into maximization before addition of the objectives takes place. These five types of selected solutions are listed as following:

- Solutions with Best Task load Standard Deviation (S_W)
- Solutions with Longest Average Flight Sector Time (A_T)
- Solutions with Largest Minimum Distance (M_D)

- Solutions with Lowest Maximum Task load (M_W)
- Solutions with Balanced Performance (K)

I first present an overall performance comparison between the two models and the current sectorization of the Australian Airspace using a year-worth of data.

Figure 7.26, 7.27, 7.28, and 7.29 show the box plots of the results from SPBM compared with the current airspace sectorization. The red boxes represent the benchmarks (BM) and the blue boxes are the results from SPBM. The overall performance of the 5 selected solution types produced by this model are better as shown in the figures, but some days performed worse than the current benchmark. The number of days with the better or worse performance on each objective are listed in Table 7.2, 7.3, 7.4, and 7.5, which show my SPBM has better objective achieved in most days.

Figure 7.30, 7.31, 7.32, and 7.33 show the box plots of the results from CVDM compared with the current airspace sectorization. The overall performance of the 5 selected solution types produced by this model are better as shown in the figures, but a few of them are not as good as the current benchmark. The number of days with the better or worse performance on each objective are listed in Table 7.6, 7.7, 7.8, and 7.9. CVDM has better results in the most days.

From the comparisons, I also find conflicts between the specified objectives. When the solutions want to achieve higher flight sector time, the more days have bad performances on task load standard deviation and maximum task load. Meanwhile, the larger minimum distance between sector boundaries and traffic flow crossing points achieved by solutions result bad performance on task load standard deviation too. To push sector boundaries far away from the crossing points limits the flexibility to configure the sector for task load balancing.

The monthly performance is also investigated by the averaged objective values for the 5 selected solutions, which is shown in the following figures (from Figure 7.34 to Figure 7.41). All the monthly averaged results from my models are better than the benchmarks except the standard deviation generated by SPBM in some months

when the model is trying to achieve the longer average flight sector time.

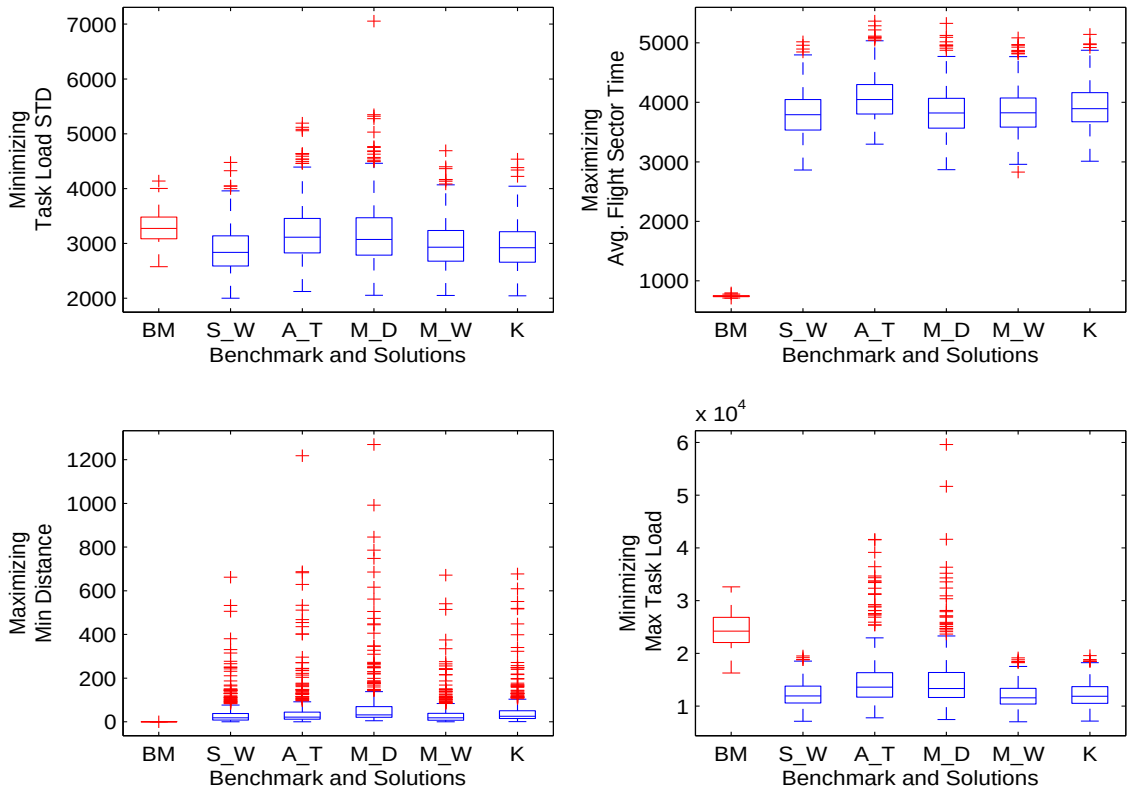


Figure 7.26: Overall Comparison between the Current Australian Sector Configurations and Airspace Sectorization by SPBM in Scenario 1. BM: Benchmark of current airspace sectorization; S_W: Solutions with Best Task load Standard Deviation; A_T: Solutions with Longest Average Flight Sector Time; M_D: Solutions with Largest Minimum Distance; M_W: Solutions with Lowest Maximum Task load; K: Solutions with Balanced Performance.

Solutions	Minimizing Task load Standard Deviation		Maximizing Avg. Flight Sector Time		Maximizing Min Distance		Minimizing Max Task load	
	Better	Worse	Better	Worse	Better	Worse	Better	Worse
Solutions (S_W)	88.22%	11.78%	100.00%	0.00%	99.45%	0.55%	100.00%	0.00%
Solutions (A_T)	66.58%	33.42%	100.00%	0.00%	100.00%	0.00%	93.42%	6.58%
Solutions (M_D)	67.12%	32.88%	100.00%	0.00%	100.00%	0.00%	94.25%	5.75%
Solutions (M_W)	80.27%	19.73%	100.00%	0.00%	99.45%	0.55%	100.00%	0.00%
Solutions (K)	83.01%	16.99%	100.00%	0.00%	100.00%	0.00%	100.00%	0.00%

Table 7.2: Percentage of Days with Better and Worse Results by SPBM for Scenario 1. S_W: Solutions with Best Task load Standard Deviation; A_T: Solutions with Longest Average Flight Sector Time; M_D: Solutions with Largest Minimum Distance; M_W: Solutions with Lowest Maximum Task load; K: Solutions with Balanced Performance.

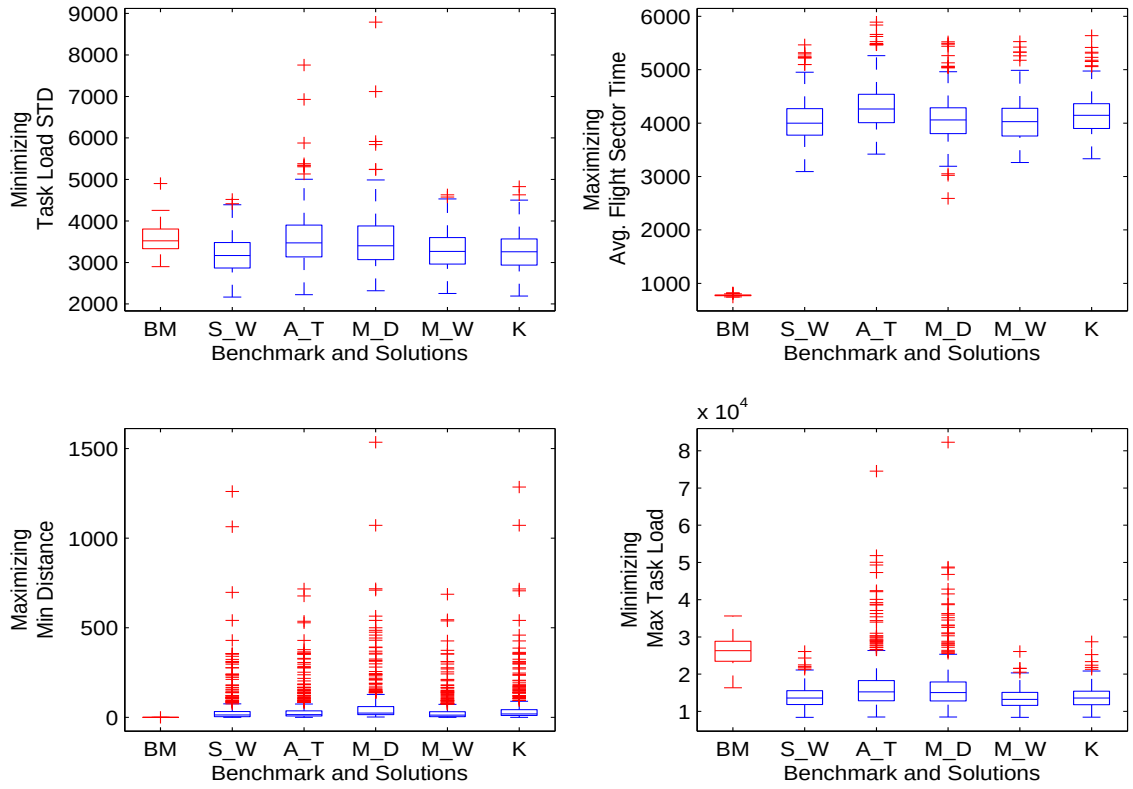


Figure 7.27: Overall Comparison between the Current Australian Sector Configurations and Airspace Sectorization by SPBM in Scenario 2. BM: Benchmark of current airspace sectorization; S_W: Solutions with Best Task load Standard Deviation; A_T: Solutions with Longest Average Flight Sector Time; M_D: Solutions with Largest Minimum Distance; M_W: Solutions with Lowest Maximum Task load; K: Solutions with Balanced Performance.

Solutions	Minimizing Task load Standard Deviation		Maximizing Avg. Flight Sector Time		Maximizing Min Distance		Minimizing Max Task load	
	Better	Worse	Better	Worse	Better	Worse	Better	Worse
Solutions (S_W)	84.66%	15.34%	100.00%	0.00%	100.00%	0.00%	100.00%	0.00%
Solutions (A_T)	57.53%	42.47%	100.00%	0.00%	99.73%	0.27%	89.86%	10.14%
Solutions (M_D)	64.11%	35.89%	100.00%	0.00%	100.00%	0.00%	92.05%	7.95%
Solutions (M_W)	74.79%	25.21%	100.00%	0.00%	100.00%	0.00%	100.00%	0.00%
Solutions (K)	77.26%	22.74%	100.00%	0.00%	100.00%	0.00%	100.00%	0.00%

Table 7.3: Percentage of Days with Better and Worse Results by SPBM for Scenario 2. S_W: Solutions with Best Task load Standard Deviation; A_T: Solutions with Longest Average Flight Sector Time; M_D: Solutions with Largest Minimum Distance; M_W: Solutions with Lowest Maximum Task load; K: Solutions with Balanced Performance.

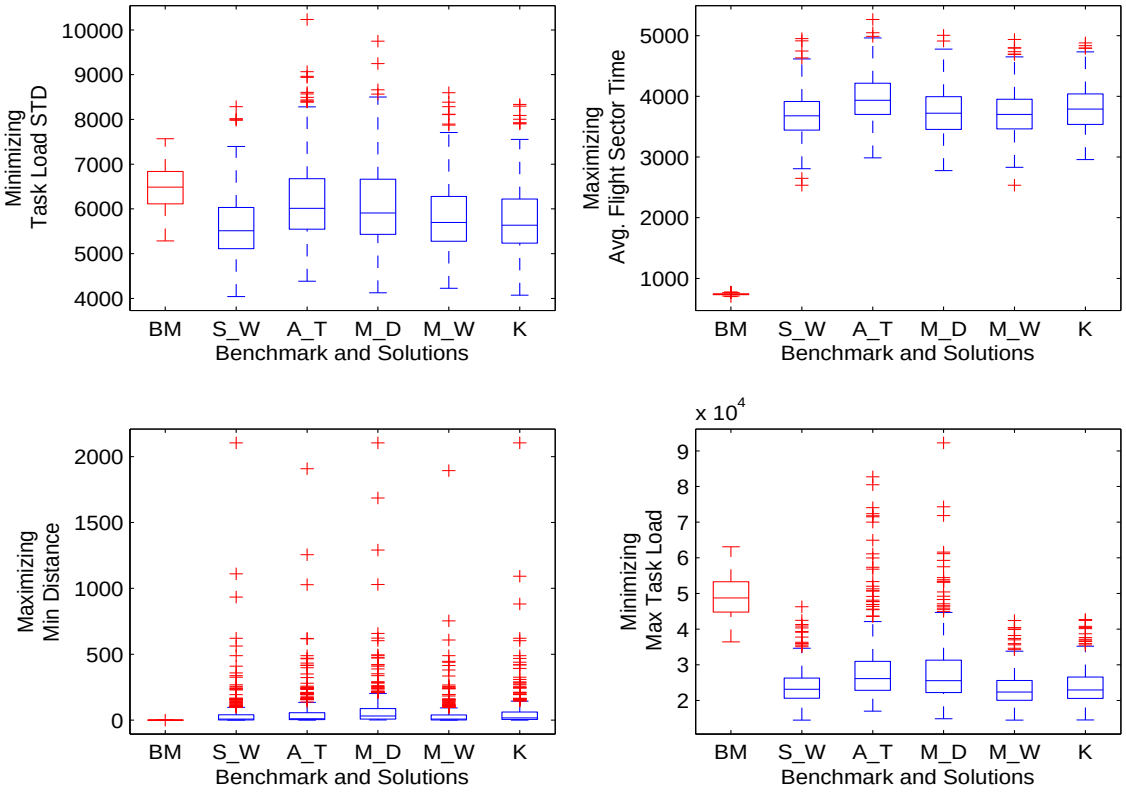


Figure 7.28: Overall Comparison between the Current Australian Sector Configurations and Airspace Sectorization by SPBM in Scenario 3. BM: Benchmark of current airspace sectorization; S_W: Solutions with Best Task load Standard Deviation; A_T: Solutions with Longest Average Flight Sector Time; M_D: Solutions with Largest Minimum Distance; M_W: Solutions with Lowest Maximum Task load; K: Solutions with Balanced Performance.

Solutions	Minimizing Task load Standard Deviation		Maximizing Avg. Flight Sector Time		Maximizing Min Distance		Minimizing Max Task load	
	Better	Worse	Better	Worse	Better	Worse	Better	Worse
Solutions (S_W)	92.05 %	7.95 %	100.00 %	0.00 %	99.73 %	0.27 %	100.00 %	0.00 %
Solutions (A_T)	72.05 %	27.95 %	100.00 %	0.00 %	99.45 %	0.55 %	95.62 %	4.38 %
Solutions (M_D)	70.41 %	29.59 %	100.00 %	0.00 %	100.00 %	0.00 %	96.16 %	3.84 %
Solutions (M_W)	84.93 %	15.07 %	100.00 %	0.00 %	99.73 %	0.27 %	100.00 %	0.00 %
Solutions (K)	85.48 %	14.52 %	100.00 %	0.00 %	99.73 %	0.27 %	99.73 %	0.27 %

Table 7.4: Percentage of Days with Better and Worse Results by SPBM for Scenario 3. S_W: Solutions with Best Task load Standard Deviation; A_T: Solutions with Longest Average Flight Sector Time; M_D: Solutions with Largest Minimum Distance; M_W: Solutions with Lowest Maximum Task load; K: Solutions with Balanced Performance.

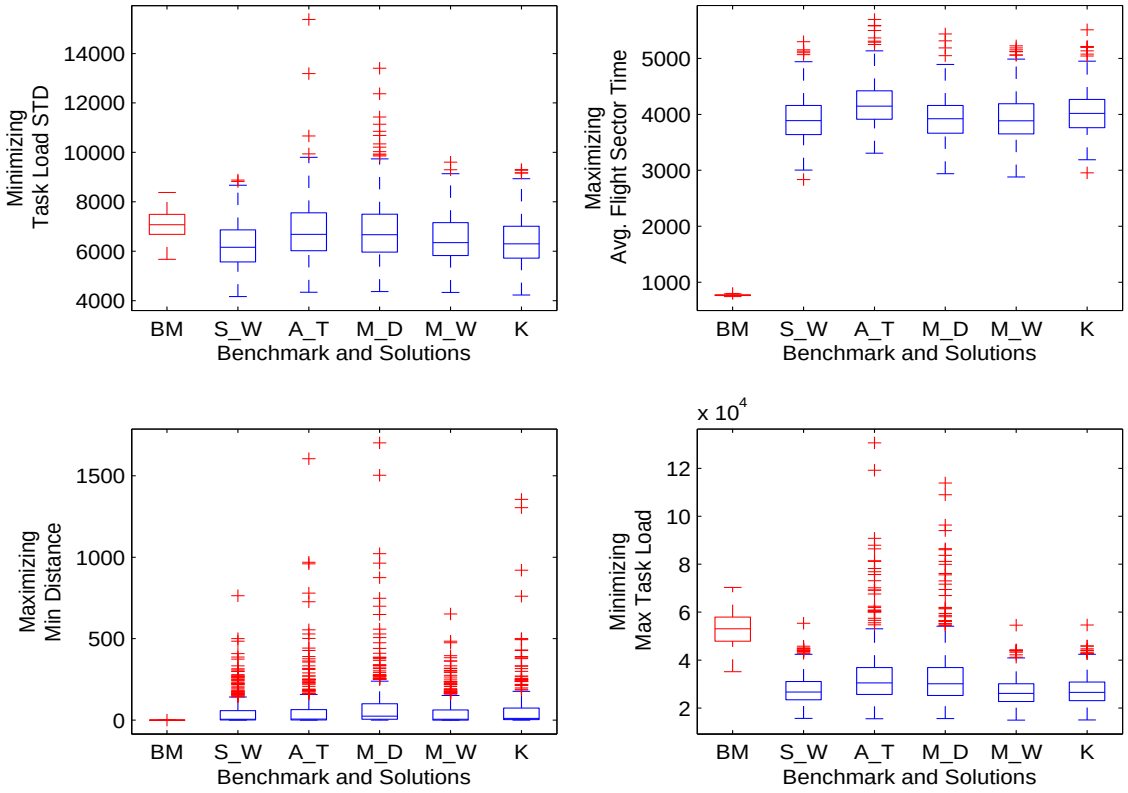


Figure 7.29: Overall Comparison between the Current Australian Sector Configurations and Airspace Sectorization by SPBM in Scenario 4. BM: Benchmark of current airspace sectorization; S_W: Solutions with Best Task load Standard Deviation; A_T: Solutions with Longest Average Flight Sector Time; M_D: Solutions with Largest Minimum Distance; M_W: Solutions with Lowest Maximum Task load; K: Solutions with Balanced Performance.

Solutions	Minimizing Task load Standard Deviation		Maximizing Avg. Flight Sector Time		Maximizing Min Distance		Minimizing Max Task load	
	Better	Worse	Better	Worse	Better	Worse	Better	Worse
Solutions (S_W)	86.58%	13.42%	100.00%	0.00%	99.73%	0.27%	99.73%	0.27%
Solutions (A_T)	66.30%	33.70%	100.00%	0.00%	100.00%	0.00%	91.78%	8.22%
Solutions (M_D)	66.03%	33.97%	100.00%	0.00%	100.00%	0.00%	92.60%	7.40%
Solutions (M_W)	81.10%	18.90%	100.00%	0.00%	100.00%	0.00%	99.73%	0.27%
Solutions (K)	81.37%	18.63%	100.00%	0.00%	100.00%	0.00%	99.73%	0.27%

Table 7.5: Percentage of Days with Better and Worse Results by SPBM for Scenario 4. S_W: Solutions with Best Task load Standard Deviation; A_T: Solutions with Longest Average Flight Sector Time; M_D: Solutions with Largest Minimum Distance; M_W: Solutions with Lowest Maximum Task load; K: Solutions with Balanced Performance.

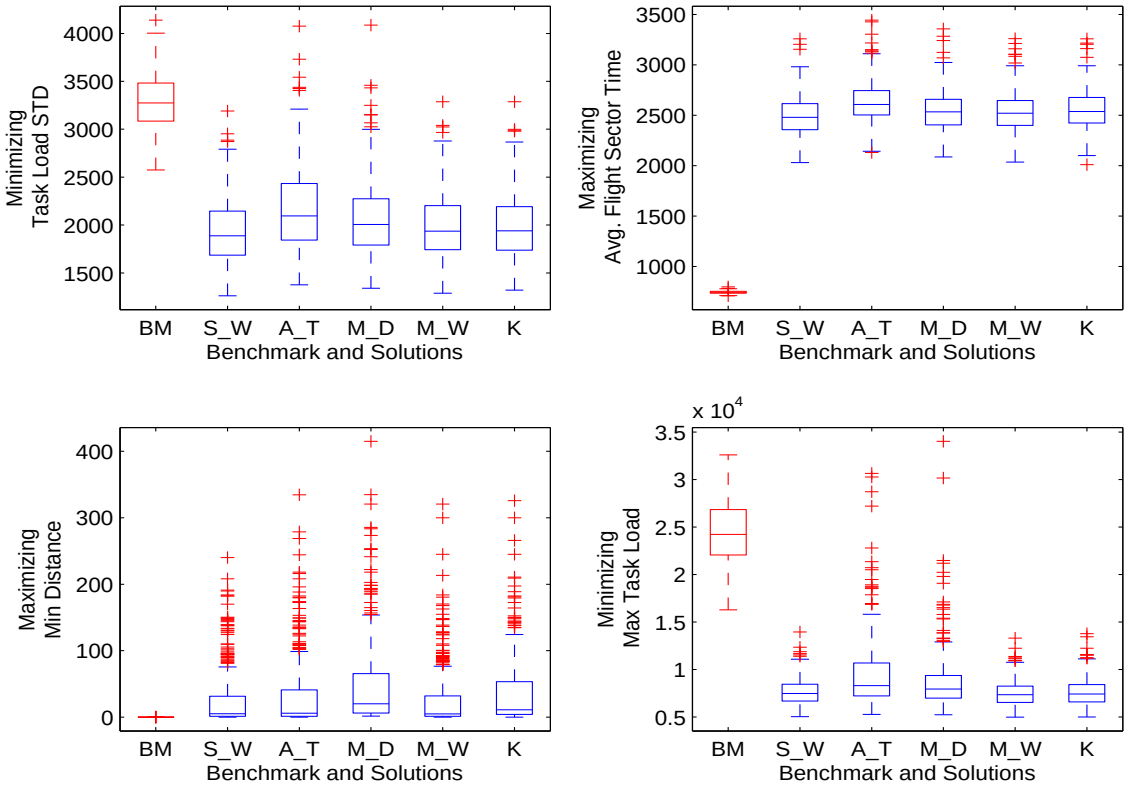


Figure 7.30: Overall Comparison between the Current Australian Sector Configurations and Airspace Sectorization by CVDM in Scenario 1. BM: Benchmark of current airspace sectorization; S_W: Solutions with Best Task load Standard Deviation; A_T: Solutions with Longest Average Flight Sector Time; M_D: Solutions with Largest Minimum Distance; M_W: Solutions with Lowest Maximum Task load; K: Solutions with Balanced Performance.

Solutions	Minimizing Task load Standard Deviation		Maximizing Avg. Flight Sector Time		Maximizing Min Distance		Minimizing Max Task load	
	Better	Worse	Better	Worse	Better	Worse	Better	Worse
Solutions (S_W)	100.00%	0.00%	100.00%	0.00%	98.90%	1.10%	100.00%	0.00%
Solutions (A_T)	98.90%	1.10%	100.00%	0.00%	99.73%	0.27%	98.90%	1.10%
Solutions (M_D)	99.73%	0.27%	100.00%	0.00%	100.00%	0.00%	99.45%	0.55%
Solutions (M_W)	100.00%	0.00%	100.00%	0.00%	98.63%	1.37%	100.00%	0.00%
Solutions (K)	100.00%	0.00%	100.00%	0.00%	99.73%	0.27%	100.00%	0.00%

Table 7.6: Percentage of Days with Better and Worse Results by CVDM for Scenario 1. S_W: Solutions with Best Task load Standard Deviation; A_T: Solutions with Longest Average Flight Sector Time; M_D: Solutions with Largest Minimum Distance; M_W: Solutions with Lowest Maximum Task load; K: Solutions with Balanced Performance.

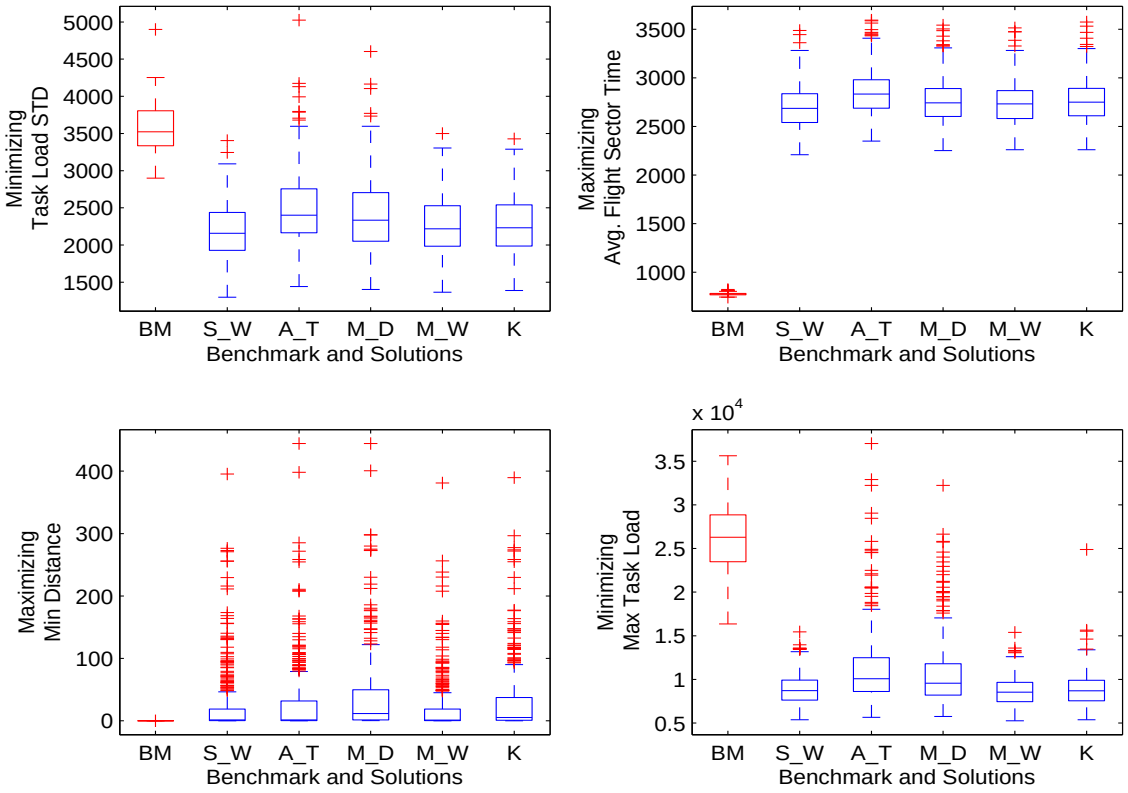


Figure 7.31: Overall Comparison between the Current Australian Sector Configurations and Airspace Sectorization by CVDM in Scenario 2. BM: Benchmark of current airspace sectorization; S_W: Solutions with Best Task load Standard Deviation; A_T: Solutions with Longest Average Flight Sector Time; M_D: Solutions with Largest Minimum Distance; M_W: Solutions with Lowest Maximum Task load; K: Solutions with Balanced Performance.

Solutions	Minimizing Task load Standard Deviation		Maximizing Avg. Flight Sector Time		Maximizing Min Distance		Minimizing Max Task load	
	Better	Worse	Better	Worse	Better	Worse	Better	Worse
Solutions (S_W)	100.00%	0.00%	100.00%	0.00%	100.00%	0.00%	100.00%	0.00%
Solutions (A_T)	98.08%	1.92%	100.00%	0.00%	99.73%	0.27%	98.63%	1.37%
Solutions (M_D)	98.63%	1.37%	100.00%	0.00%	100.00%	0.00%	99.18%	0.82%
Solutions (M_W)	100.00%	0.00%	100.00%	0.00%	99.18%	0.82%	100.00%	0.00%
Solutions (K)	100.00%	0.00%	100.00%	0.00%	100.00%	0.00%	100.00%	0.00%

Table 7.7: Percentage of Days with Better and Worse Results by CVDM for Scenario 2. S_W: Solutions with Best Task load Standard Deviation; A_T: Solutions with Longest Average Flight Sector Time; M_D: Solutions with Largest Minimum Distance; M_W: Solutions with Lowest Maximum Task load; K: Solutions with Balanced Performance.

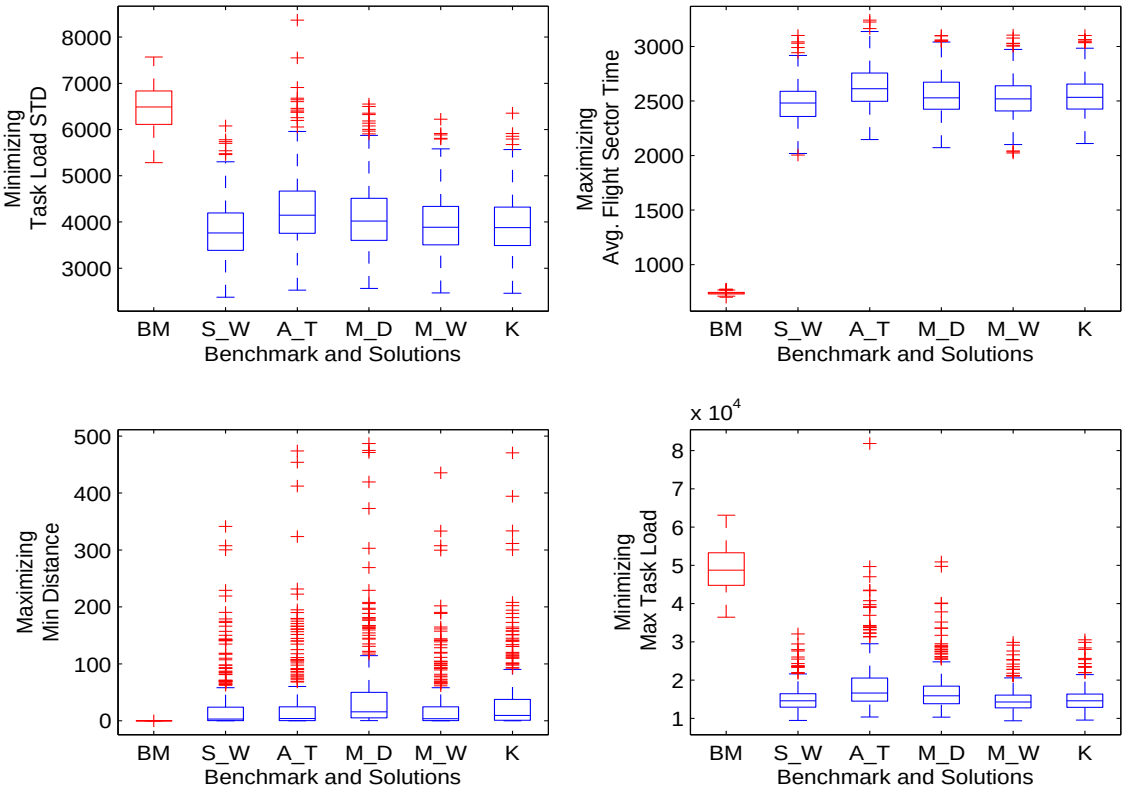


Figure 7.32: Overall Comparison between the Current Australian Sector Configurations and Airspace Sectorization by CVDM in Scenario 3. BM: Benchmark of current airspace sectorization; S_W: Solutions with Best Task load Standard Deviation; A_T: Solutions with Longest Average Flight Sector Time; M_D: Solutions with Largest Minimum Distance; M_W: Solutions with Lowest Maximum Task load; K: Solutions with Balanced Performance.

Solutions	Minimizing Task load Standard Deviation		Maximizing Avg. Flight Sector Time		Maximizing Min Distance		Minimizing Max Task load	
	Better	Worse	Better	Worse	Better	Worse	Better	Worse
Solutions (S_W)	100.00%	0.00%	100.00%	0.00%	99.45%	0.55%	100.00%	0.00%
Solutions (A_T)	98.63%	1.37%	100.00%	0.00%	99.18%	0.82%	99.18%	0.82%
Solutions (M_D)	99.45%	0.55%	100.00%	0.00%	100.00%	0.00%	99.45%	0.55%
Solutions (M_W)	100.00%	0.00%	100.00%	0.00%	98.90%	1.10%	100.00%	0.00%
Solutions (K)	100.00%	0.00%	100.00%	0.00%	99.73%	0.27%	100.00%	0.00%

Table 7.8: Percentage of Days with Better and Worse Results by CVDM for Scenario 3. S_W: Solutions with Best Task load Standard Deviation; A_T: Solutions with Longest Average Flight Sector Time; M_D: Solutions with Largest Minimum Distance; M_W: Solutions with Lowest Maximum Task load; K: Solutions with Balanced Performance.

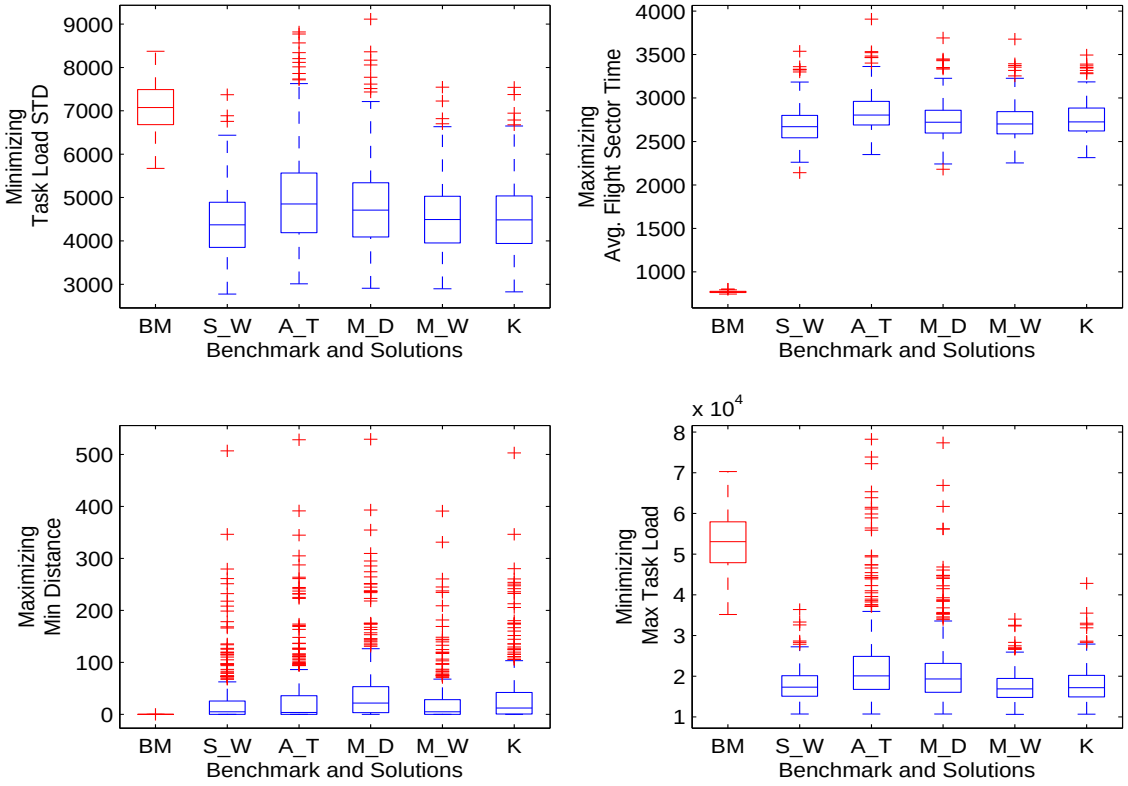


Figure 7.33: Overall Comparison between the Current Australian Sector Configurations and Airspace Sectorization by CVDM in Scenario 4. BM: Benchmark of current airspace sectorization; S_W: Solutions with Best Task load Standard Deviation; A_T: Solutions with Longest Average Flight Sector Time; M_D: Solutions with Largest Minimum Distance; M_W: Solutions with Lowest Maximum Task load; K: Solutions with Balanced Performance.

Solutions	Minimizing Task load Standard Deviation		Maximizing Avg. Flight Sector Time		Maximizing Min Distance		Minimizing Max Task load	
	Better	Worse	Better	Worse	Better	Worse	Better	Worse
Solutions (S_W)	100.00%	0.00%	100.00%	0.00%	99.45%	0.55%	100.00%	0.00%
Solutions (A_T)	96.16%	3.84%	100.00%	0.00%	98.63%	1.37%	96.16%	3.84%
Solutions (M_D)	97.53%	2.47%	100.00%	0.00%	100.00%	0.00%	98.90%	1.10%
Solutions (M_W)	99.73%	0.27%	100.00%	0.00%	99.73%	0.27%	100.00%	0.00%
Solutions (K)	99.45%	0.55%	100.00%	0.00%	100.00%	0.00%	100.00%	0.00%

Table 7.9: Percentage of Days with Better and Worse Results by CVDM for Scenario 4. S_W: Solutions with Best Task load Standard Deviation; A_T: Solutions with Longest Average Flight Sector Time; M_D: Solutions with Largest Minimum Distance; M_W: Solutions with Lowest Maximum Task load; K: Solutions with Balanced Performance.

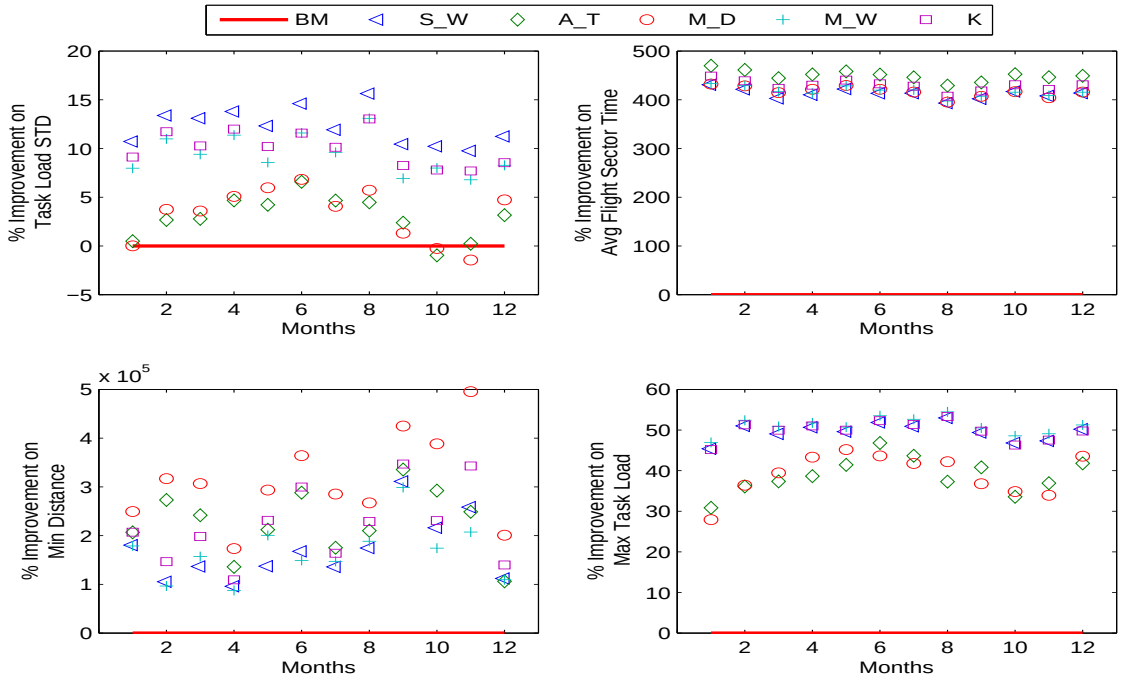


Figure 7.34: Comparison on the Monthly Averaged Objectives between the Current Australian Sector Configurations and Airspace Sectorization by SPBM in Scenario 1

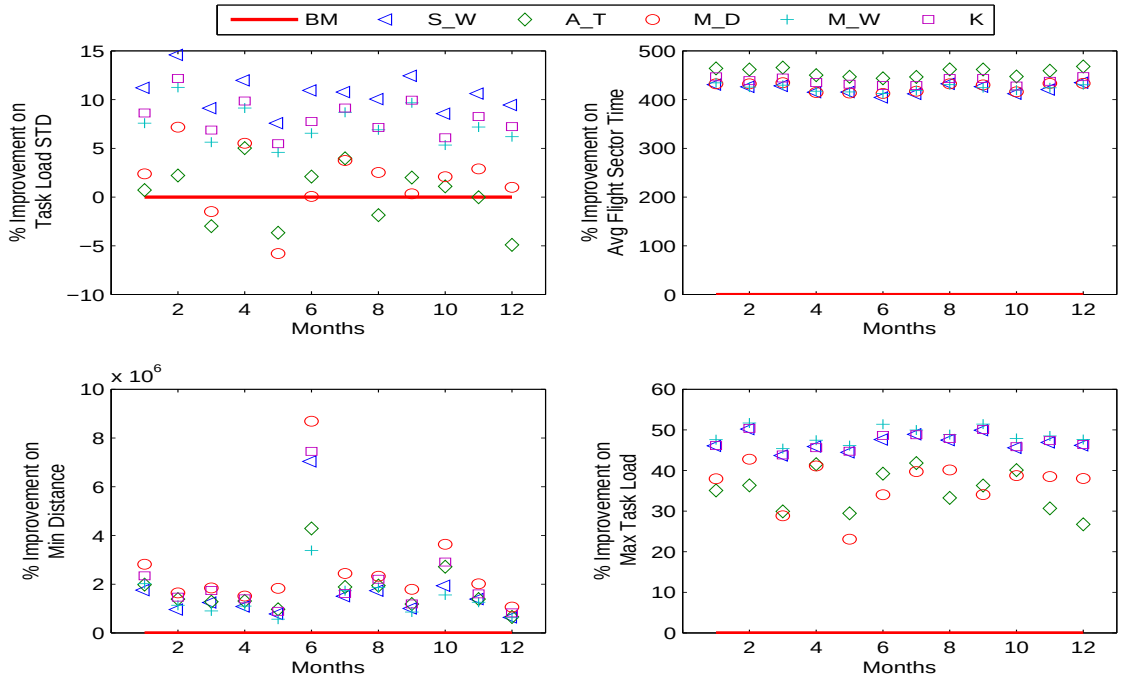


Figure 7.35: Comparison on the Monthly Averaged Objectives between the Current Australian Sector Configurations and Airspace Sectorization by SPBM in Scenario 2

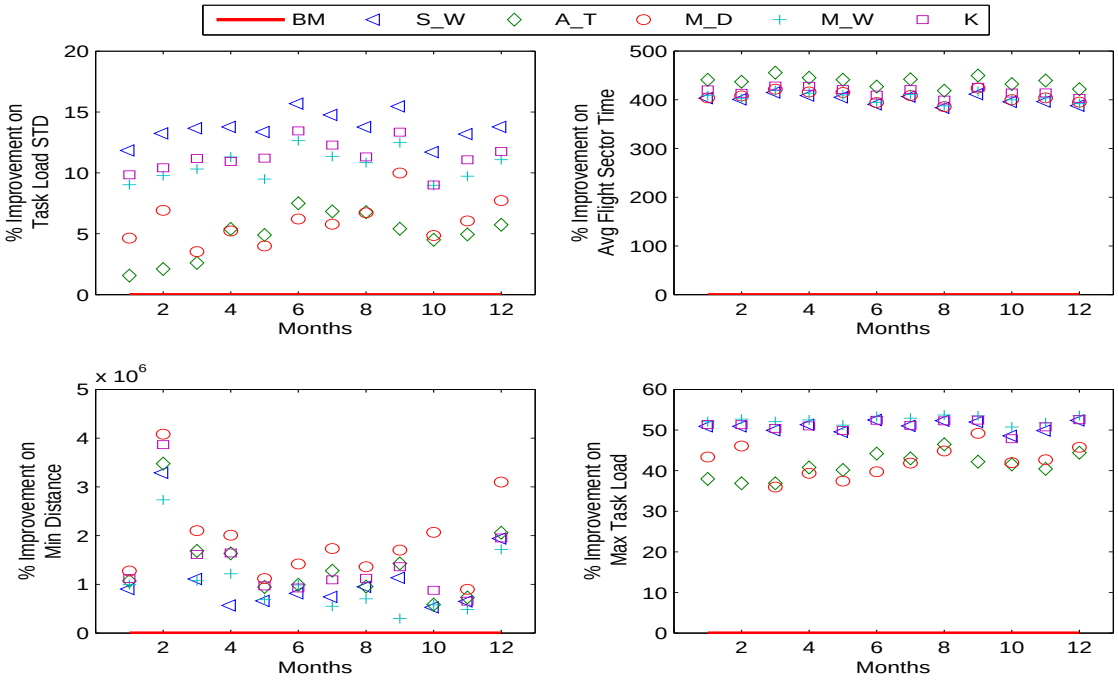


Figure 7.36: Comparison on the Monthly Averaged Objectives between the Current Australian Sector Configurations and Airspace Sectorization by SPBM in Scenario 3

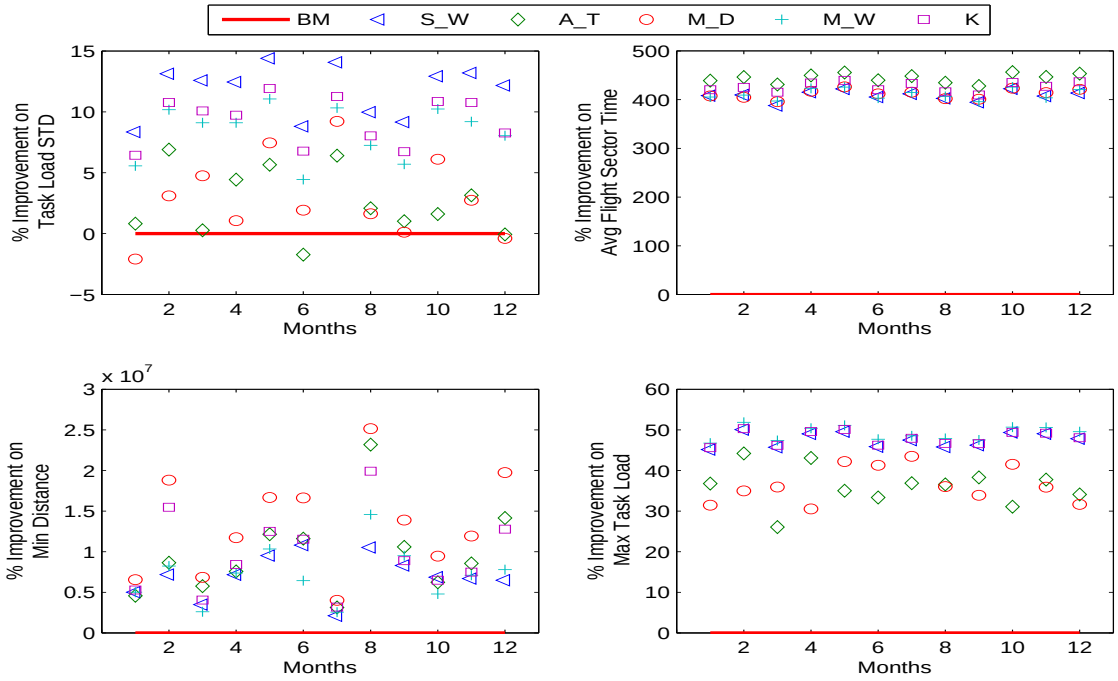


Figure 7.37: Comparison on the Monthly Averaged Objectives between the Current Australian Sector Configurations and Airspace Sectorization by SPBM in Scenario 4

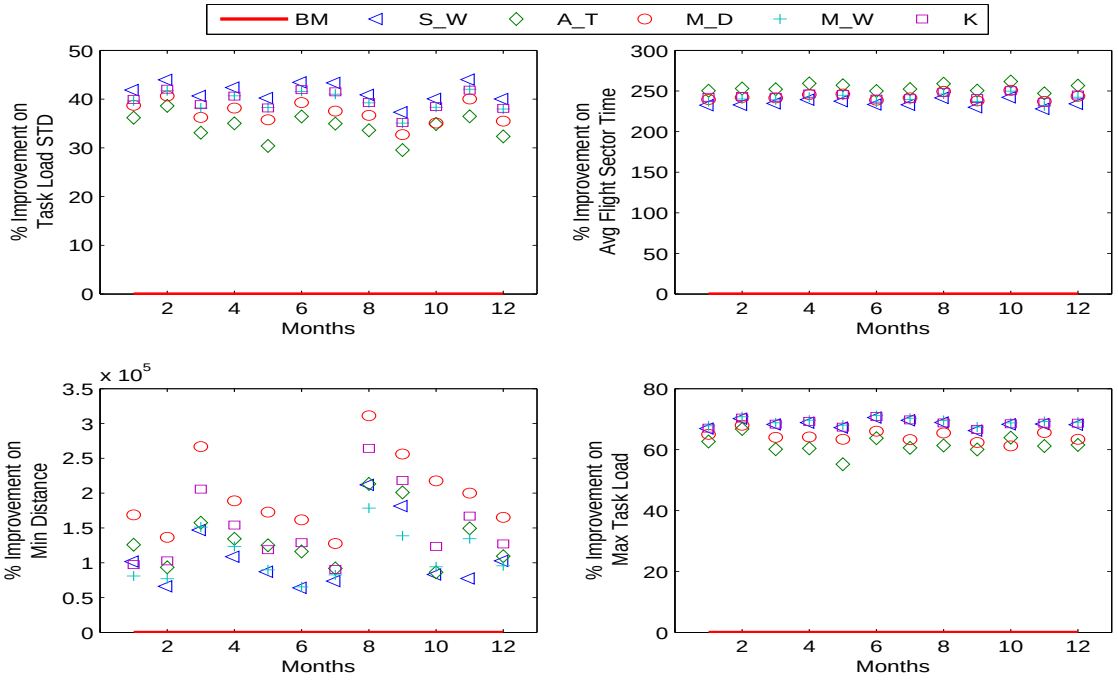


Figure 7.38: Comparison on the Monthly Averaged Objectives between the Current Australian Sector Configurations and Airspace Sectorization by CVDM in Scenario 1

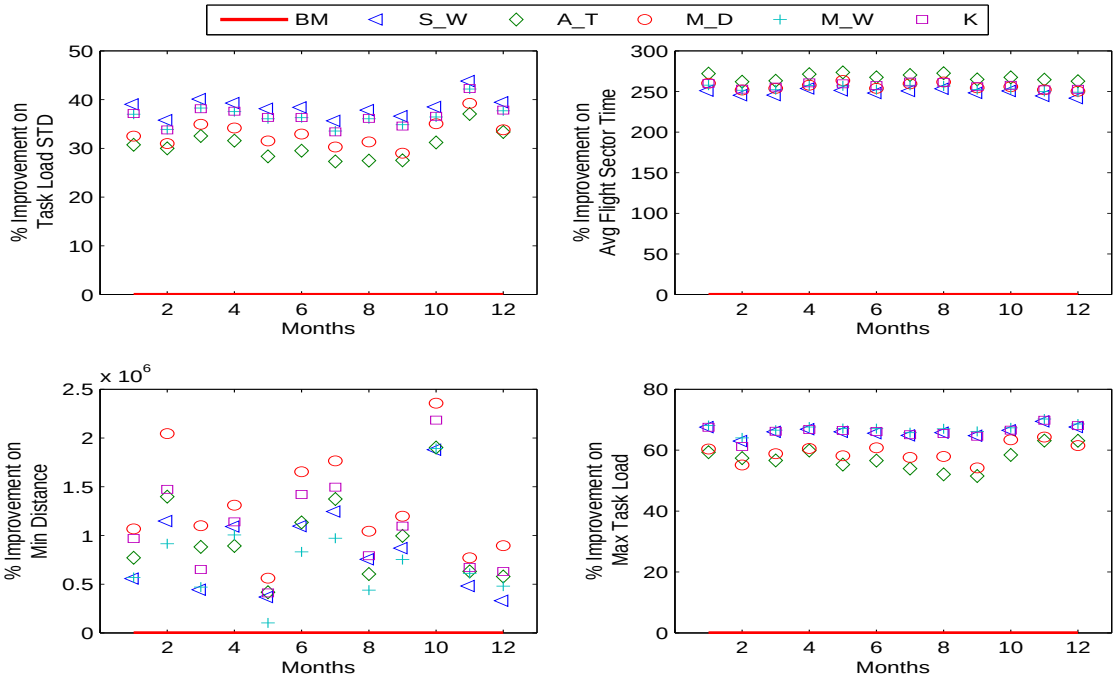


Figure 7.39: Comparison on the Monthly Averaged Objectives between the Current Australian Sector Configurations and Airspace Sectorization by CVDM in Scenario 2

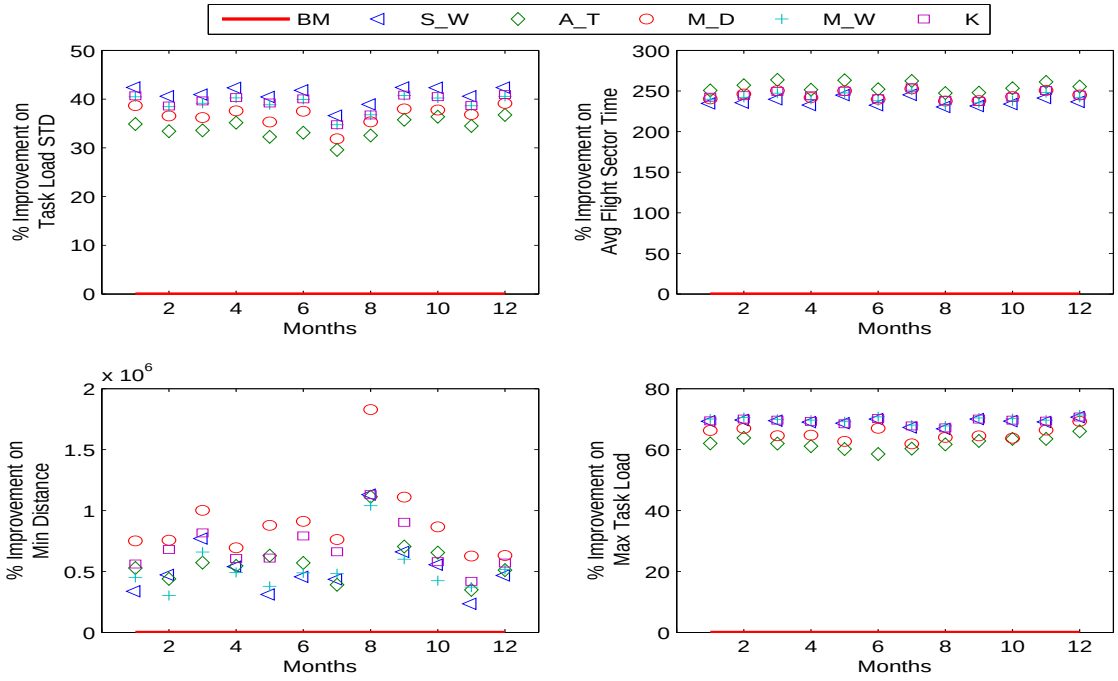


Figure 7.40: Comparison on the Monthly Averaged Objectives between the Current Australian Sector Configurations and Airspace Sectorization by CVDM in Scenario 3

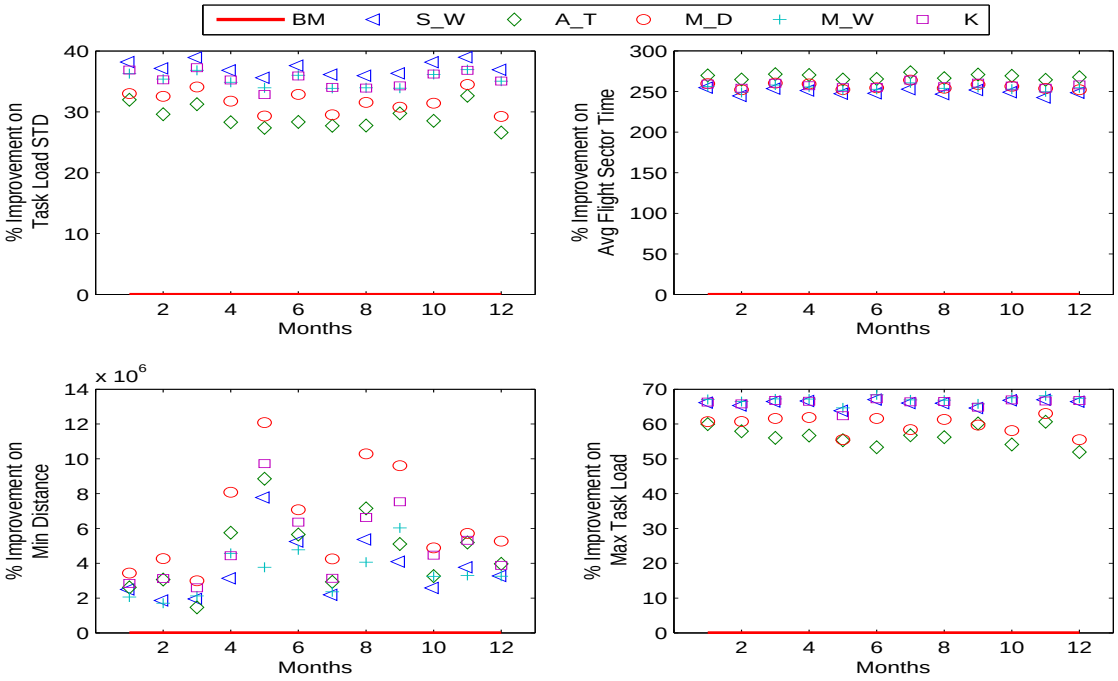


Figure 7.41: Comparison on the Monthly Averaged Objectives between the Current Australian Sector Configurations and Airspace Sectorization by CVDM in Scenario 4

Both models have big advantages on both average flight sector time and minimum distance between sector boundaries and traffic flow crossing points over the current airspace sectorization which are shown in both yearly and monthly comparisons. This is also consistent with the daily comparisons as shown in the following tables (from Table 7.10, to Table 7.17). These tables show the average improvement on each objective of each solution type from both models, and the best and worst improvements on each objective of them. For example, the solution (S_W) of SPBM in Scenario 1 has average improvements of 12.23% on task load standard deviation, 412.38% on average flight sector time, 86000% on minimum distance between sector boundaries and crossing points, and 49.11% on maximum task load as shown in Table 7.10. It is also showing that the best improvement of the solutions (S_W) on task load standard deviation is 37.34% and the worst of the solutions (S_W) is -32.37%.

Objectives	Solutions	Average Improvement(%)	Best Improvement on the Benchmark(%)				Worst Improvement on the Benchmark(%)			
			Task load STD	Avg Flight Sector Time	Min Distance	Max Task load	Task load STD	Avg Flight Sector Time	Min Distance	Max Task load
Task Load STD	S_W	12.23	**37.34**				**-32.37**			
	A_T	2.87	**30.58**				**-58.38**			
	M_D	3.27	**33.92**				**-103.26**			
	M_W	9.34	**35.72**				**-38.67**			
	K	9.99	**36.26**				**-34.18**			
Avg Flight Sector Time	S_W	412.38		**568.21**				**293.65**		
	A_T	449.64		**623.29**				**329.7**		
	M_D	416.91		**610.48**				**278.8**		
	M_W	417.19		**576.73**				**276.9**		
	K	429.22		**584.65**				**313.76**		
Min Distance	S_W	8.6e+04			**2.74e+07**				**-8.52e+01**	
	A_T	9.02e+04			**2.83e+07**				**3.19e+02**	
	M_D	1.38e+05			**3.49e+07**				**4.96e+03**	
	M_W	8.12e+04			**3.31e+07**				**-8.82e+01**	
	K	1.09e+05			**3.44e+07**				**1.38e+03**	
Max Task Load	S_W	49.11				**70.3**				**14.78**
	A_T	37.88				**67.76**				**-82.99**
	M_D	38.34				**71.06**				**-170.73**
	M_W	50.56				**71.35**				**14.9**
	K	49.29				**70.96**				**14.9**

Table 7.10: Average, Best, and Worst Improvements on each Objective by the Solutions (S_W, A_T, M_D, M_W, K) from SPBM in Scenario 1

Objectives	Solutions	Average Improvement(%)	Best Improvement on the Benchmark(%)				Worst Improvement on the Benchmark(%)			
			Task load STD	Avg Flight Sector Time	Min Distance	Max Task load	Task load STD	Avg Flight Sector Time	Min Distance	Max Task load
Task Load STD	S_W	10.46	**32.89**				**-26.52**			
	A_T	0.14	**31.08**				**-99.31**			
	M_D	1.49	**27.44**				**-125.28**			
	M_W	7.26	**30.21**				**-27.39**			
	K	8.06	**32.12**				**-34.9**			
Avg Flight Sector Time	S_W	421.63		**605.83**				**302.15**		
	A_T	456.53		**659.86**				**330.53**		
	M_D	425.3		**607.47**				**228.19**		
	M_W	423.61		**613.5**				**312.94**		
	K	437.87		**628.24**				**319.65**		
Min Distance	S_W	5.11e+05			**3.12e+08**				**1.83e+02**	
	A_T	4.97e+05			**1.78e+08**				**-1e+02**	
	M_D	8.76e+05			**3.8e+08**				**7.9e+04**	
	M_W	4.7e+05			**2.33e+08**				**5.07e+02**	
	K	6.93e+05			**3.18e+08**				**5.21e+03**	
Max Task Load	S_W	46.34				**68.03**				**12.72**
	A_T	34.35				**64.85**				**-169.37**
	M_D	35.44				**64.96**				**-181.7**
	M_W	48.08				**68.23**				**15.09**
	K	46.57				**68.03**				**5.78**

Table 7.11: Average, Best, and Worst Improvements on each Objective by the Solutions (S_W, A_T, M_D, M_W, K) from SPBM in Scenario 2

Objectives	Solutions	Average Improvement(%)	Best Improvement on the Benchmark(%)				Worst Improvement on the Benchmark(%)			
			Task load STD	Avg Flight Sector Time	Min Distance	Max Task load	Task load STD	Avg Flight Sector Time	Min Distance	Max Task load
Task Load STD	S_W	13.7	**34.27**				**-24.88**			
	A_T	4.84	**30.98**				**-43.15**			
	M_D	5.98	**33.48**				**-42.47**			
	M_W	10.61	**32.97**				**-26.42**			
	K	11.33	**32.31**				**-25.66**			
Avg Flight Sector Time	S_W	400.39		**569.35**				**244.44**		
	A_T	437.65		**591.47**				**299.05**		
	M_D	406.52		**568.85**				**272.44**		
	M_W	404.94		**572.41**				**244.44**		
	K	416.41		**564.85**				**295.08**		
Min Distance	S_W	4.98e+05			**3.61e+08**				**-8.17e+01**	
	A_T	1.09e+06			**1.07e+09**				**-8.35e+01**	
	M_D	1.21e+06			**1.12e+09**				**1.87e+04**	
	M_W	3.56e+05			**1.48e+08**				**-8.17e+01**	
	K	1.02e+06			**9.63e+08**				**-4.42e+01**	
Max Task Load	S_W	50.7				**69.66**				**8.58**
	A_T	40.91				**66.6**				**-98.1**
	M_D	41.99				**68.51**				**-79.52**
	M_W	52.27				**69.66**				**9.56**
	K	50.86				**68.82**				**-4.31**

Table 7.12: Average, Best, and Worst Improvements on each Objective by the Solutions (S_W, A_T, M_D, M_W, K) from SPBM in Scenario 3

Objectives	Solutions	Average Improvement(%)	Best Improvement on the Benchmark(%)				Worst Improvement on the Benchmark(%)			
			Task load STD	Avg Flight Sector Time	Min Distance	Max Task load	Task load STD	Avg Flight Sector Time	Min Distance	Max Task load
Task Load STD	S_W	11.85	**38.43**				**-30.57**			
	A_T	2.61	**35.89**				**-112.95**			
	M_D	3.19	**35.39**				**-80.03**			
	M_W	8.46	**35.95**				**-35.62**			
	K	9.41	**36.85**				**-32.88**			
Avg Flight Sector Time	S_W	408.52		**610.05**				**267.66**		
	A_T	444.3		**648.31**				**328.92**		
	M_D	412.01		**612.47**				**281.99**		
	M_W	411.17		**600.14**				**272.64**		
	K	426.33		**638.83**				**283.87**		
Min Distance	S_W	1.74e+06			**8.82e+08**				**-2.03e+00**	
	A_T	2.04e+06			**7.16e+08**				**6.98e+02**	
	M_D	3.11e+06			**8.87e+08**				**4.03e+04**	
	M_W	1.76e+06			**6.38e+08**				**5.45e+02**	
	K	2.17e+06			**7.16e+08**				**2.29e+02**	
Max Task Load	S_W	47.36				**69.15**				**-15.74**
	A_T	35.61				**69.43**				**-145.79**
	M_D	36.59				**69.28**				**-113.14**
	M_W	48.93				**70.24**				**-14.08**
	K	47.76				**70.08**				**-14.25**

Table 7.13: Average, Best, and Worst Improvements on each Objective by the Solutions (S_W, A_T, M_D, M_W, K) from SPBM in Scenario 4

Objectives	Solutions	Average Improvement(%)	Best Improvement on the Benchmark(%)				Worst Improvement on the Benchmark(%)			
			Task load STD	Avg Flight Sector Time	Min Distance	Max Task load	Task load STD	Avg Flight Sector Time	Min Distance	Max Task load
Task Load STD	S_W	41.49	**59.53**				**9.1**			
	A_T	34.27	**54.87**				**-30.72**			
	M_D	37.19	**55.72**				**-19.1**			
	M_W	39.47	**58.67**				**6.31**			
	K	39.66	**57.61**				**6.31**			
Avg Flight Sector Time	S_W	235.09		**336.28**				**169.63**		
	A_T	254.26		**368.84**				**179.83**		
	M_D	242.95		**356.91**				**179.83**		
	M_W	240.71		**330.25**				**170.54**		
	K	243.55		**336.22**				**163.82**		
Min Distance	S_W	7.1e+04			**8.12e+07**				**-9.22e+01**	
	A_T	7.87e+04			**7.67e+07**				**-7.79e+01**	
	M_D	1.61e+05			**1.4e+08**				**2.46e+03**	
	M_W	7.69e+04			**8.16e+07**				**-7.41e+01**	
	K	9.43e+04			**8e+07**				**-6.35e+01**	
Max Task Load	S_W	68.14				**79.69**				**48.51**
	A_T	61.02				**78.76**				**-31.53**
	M_D	63.95				**78.01**				**-48.13**
	M_W	68.88				**79.73**				**49.66**
	K	68.35				**79.72**				**44.82**

Table 7.14: Average, Best, and Worst Improvements on each Objective by the Solutions (S_W, A_T, M_D, M_W, K) from CVDM in Scenario 1

Objectives	Solutions	Average Improvement(%)	Best Improvement on the Benchmark(%)				Worst Improvement on the Benchmark(%)			
			Task load STD	Avg Flight Sector Time	Min Distance	Max Task load	Task load STD	Avg Flight Sector Time	Min Distance	Max Task load
Task Load STD	S_W	38.58	**61.06**				**9.6**			
	A_T	30.58	**56.34**				**-47.8**			
	M_D	32.98	**57.91**				**-35.38**			
	M_W	36.67	**56.46**				**6.42**			
	K	36.71	**58.31**				**5.42**			
Avg Flight Sector Time	S_W	248.75		**346.66**				**178.07**		
	A_T	267.75		**365.1**				**195.65**		
	M_D	256.32		**354.21**				**187.31**		
	M_W	253.92		**350.46**				**186.36**		
	K	257.28		**352.17**				**189.41**		
Min Distance	S_W	3.12e+05			**2.14e+08**				**1.31e+01**	
	A_T	4.67e+05			**3.88e+08**				**-7.08e+01**	
	M_D	6.14e+05			**4.48e+08**				**6.5e+03**	
	M_W	4.25e+05			**4.04e+08**				**-7.69e+01**	
	K	5.51e+05			**4.47e+08**				**1.6e+02**	
Max Task Load	S_W	65.86				**79.96**				**43.17**
	A_T	56.96				**79.61**				**-26.67**
	M_D	59.02				**80.63**				**-19.85**
	M_W	66.82				**80.71**				**43.31**
	K	65.87				**80.63**				**19.3**

Table 7.15: Average, Best, and Worst Improvements on each Objective by the Solutions (S_W, A_T, M_D, M_W, K) from CVDM in Scenario 2

Objectives	Solutions	Average Improvement(%)	Best Improvement on the Benchmark(%)				Worst Improvement on the Benchmark(%)			
			Task load STD	Avg Flight Sector Time	Min Distance	Max Task load	Task load STD	Avg Flight Sector Time	Min Distance	Max Task load
Task Load STD	S_W	41.01	**60.89**				**14.17**			
	A_T	34.03	**57.42**				**-27.78**			
	M_D	36.75	**57.93**				**-5.64**			
	M_W	39.11	**60.22**				**13.36**			
	K	39.27	**58.61**				**12.67**			
Avg Flight Sector Time	S_W	236.74		**321.33**				**170.79**		
	A_T	255.7		**342.25**				**193.56**		
	M_D	244.87		**326.59**				**185.15**		
	M_W	241.99		**321.71**				**172.81**		
	K	244.96		**322.06**				**187.17**		
Min Distance	S_W	2.29e+05			**2.36e+08**				**-9.2e+01**	
	A_T	3e+05			**3.37e+08**				**-9.72e+01**	
	M_D	4.26e+05			**3.37e+08**				**2.02e+03**	
	M_W	2.4e+05			**2.25e+08**				**-6.71e+01**	
	K	3.47e+05			**3.37e+08**				**-3.86e+01**	
Max Task Load	S_W	68.98				**78.72**				**30**
	A_T	61.95				**78.35**				**-64.89**
	M_D	64.97				**78.65**				**-6.47**
	M_W	69.65				**79.38**				**30**
	K	69.17				**79.33**				**27.94**

Table 7.16: Average, Best, and Worst Improvements on each Objective by the Solutions (S_W, A_T, M_D, M_W, K) from CVDM in Scenario 3

Objectives	Solutions	Average Improvement(%)	Best Improvement on the Benchmark(%)				Worst Improvement on the Benchmark(%)			
			Task load STD	Avg Flight Sector Time	Min Distance	Max Task load	Task load STD	Avg Flight Sector Time	Min Distance	Max Task load
Task Load STD	S_W	37.33	**58.12**				**0.76**			
	A_T	29.25	**54.53**				**-29.29**			
	M_D	31.81	**56.08**				**-28.39**			
	M_W	35.36	**56.28**				**-1.61**			
	K	35.41	**57.37**				**-11.31**			
Avg Flight Sector Time	S_W	249.36		**370.27**				**183.09**		
	A_T	268.37		**419.49**				**209.17**		
	M_D	256.57		**390.6**				**179.49**		
	M_W	254.27		**388.88**				**190.31**		
	K	257.62		**344.35**				**200.97**		
Min Distance	S_W	1.26e+06			**1.22e+09**				**-8.5e+01**	
	A_T	1.64e+06			**2.12e+09**				**-8.07e+01**	
	M_D	2.43e+06			**2.18e+09**				**2.34e+03**	
	M_W	1.28e+06			**8.19e+08**				**-3.96e+01**	
	K	1.78e+06			**1.39e+09**				**7.67e+01**	
Max Task Load	S_W	65.85				**77.17**				**38.34**
	A_T	56.25				**77.1**				**-55.27**
	M_D	59.58				**77.03**				**-55.77**
	M_W	66.8				**77.73**				**39.94**
	K	65.91				**77.63**				**10.73**

Table 7.17: Average, Best, and Worst Improvements on each Objective by the Solutions (S_W, A_T, M_D, M_W, K) from CVDM in Scenario 4

In summary, my proposed models can achieve better overall performance than the current airspace configuration as illustrated above. Both models have much better results of the average flight sector time. However, other objectives have some bad cases that have been identified. An analysis on these cases is addressed in Section 7.3. These comparisons also imply that CVDM has better performance on the task load balancing than SPBM but with lower average flight sector time, which are investigated in the next section.

7.2.3 Comparison between SPBM and CVDM

I compare these two models on two aspects: performance and efficiency. At first, the number of days with the best performance when comparing two models are listed in the Table 7.18, 7.19, 7.20, and 7.21. As shown in the table, CVDM has better performance on task load standard deviation and maximum task load than SPBM on the most days in all four scenarios. However, SPBM has better

performance on the average flight sector time almost all days. Moreover, it achieves larger minimum distance than CVDM in around two-thirds year in all scenarios.

Solution Types	Task load Standard Deviation		Avg. Flight Sector Time		Min Distance		Max Task load	
	SPBM	CVDM	SPBM	CVDM	SPBM	CVDM	SPBM	CVDM
S_W	2.19%	97.81%	100.00%	0.00%	67.40%	32.60%	3.56%	96.44%
A_T	4.93%	95.07%	100.00%	0.00%	66.30%	33.70%	13.97%	86.03%
M_D	3.56%	96.44%	100.00%	0.00%	64.38%	35.62%	10.14%	89.86%
M_W	2.19%	97.81%	100.00%	0.00%	63.29%	36.71%	3.56%	96.44%
K	2.19%	97.81%	100.00%	0.00%	63.01%	36.99%	3.01%	96.99%

Table 7.18: Percentage of Better Performance Days of Comparison between SPBM and CVDM in Scenario 1

Solution Types	Task load Standard Deviation		Avg. Flight Sector Time		Min Distance		Max Task load	
	SPBM	CVDM	SPBM	CVDM	SPBM	CVDM	SPBM	CVDM
S_W	1.37%	98.63%	100.00%	0.00%	67.40%	32.60%	5.48%	94.52%
A_T	4.38%	95.62%	100.00%	0.00%	68.49%	31.51%	14.79%	85.21%
M_D	3.56%	96.44%	99.73%	0.27%	69.32%	30.68%	14.79%	85.21%
M_W	1.37%	98.63%	100.00%	0.00%	67.12%	32.88%	4.38%	95.62%
K	1.37%	98.63%	100.00%	0.00%	70.14%	29.86%	4.93%	95.07%

Table 7.19: Percentage of Better Performance Days of Comparison between SPBM and CVDM in Scenario 2

Solution Types	Task load Standard Deviation		Avg. Flight Sector Time		Min Distance		Max Task load	
	SPBM	CVDM	SPBM	CVDM	SPBM	CVDM	SPBM	CVDM
S_W	1.37%	98.63%	100.00%	0.00%	62.47%	37.53%	2.19%	97.81%
A_T	3.84%	96.16%	100.00%	0.00%	67.12%	32.88%	11.51%	88.49%
M_D	3.29%	96.71%	100.00%	0.00%	66.03%	33.97%	6.85%	93.15%
M_W	1.10%	98.90%	99.73%	0.27%	61.64%	38.36%	2.19%	97.81%
K	1.64%	98.36%	100.00%	0.00%	63.29%	36.71%	2.19%	97.81%

Table 7.20: Percentage of Better Performance Days of Comparison between SPBM and CVDM in Scenario 3

Solution Types	Task load Standard Deviation		Avg. Flight Sector Time		Min Distance		Max Task load	
	SPBM	CVDM	SPBM	CVDM	SPBM	CVDM	SPBM	CVDM
S_W	2.47%	97.53%	99.73%	0.27%	59.73%	40.27%	6.58%	93.42%
A_T	7.40%	92.60%	100.00%	0.00%	62.19%	37.81%	15.62%	84.38%
M_D	4.38%	95.62%	100.00%	0.00%	59.45%	40.55%	11.23%	88.77%
M_W	1.64%	98.36%	99.73%	0.27%	60.00%	40.00%	4.93%	95.07%
K	3.84%	96.16%	100.00%	0.00%	57.26%	42.74%	5.75%	94.25%

Table 7.21: Percentage of Better Performance Days of Comparison between SPBM and CVDM in Scenario 4

The quality of the objective achievements of both models are compared as shown in Figure 7.42, 7.43, 7.44, and 7.45. The blue boxes represent SPBM and the green ones are the objective values generated from CVDM.

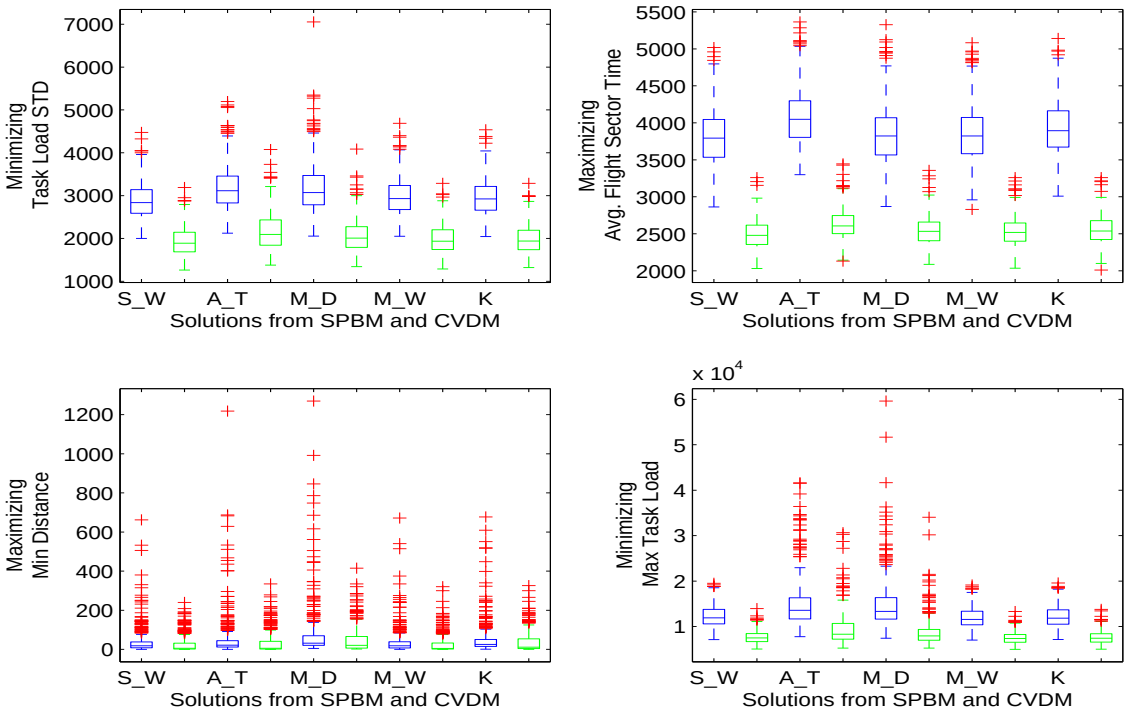


Figure 7.42: Comparisons on Objective Achievements between the SPBM and CVDM in Scenario 1

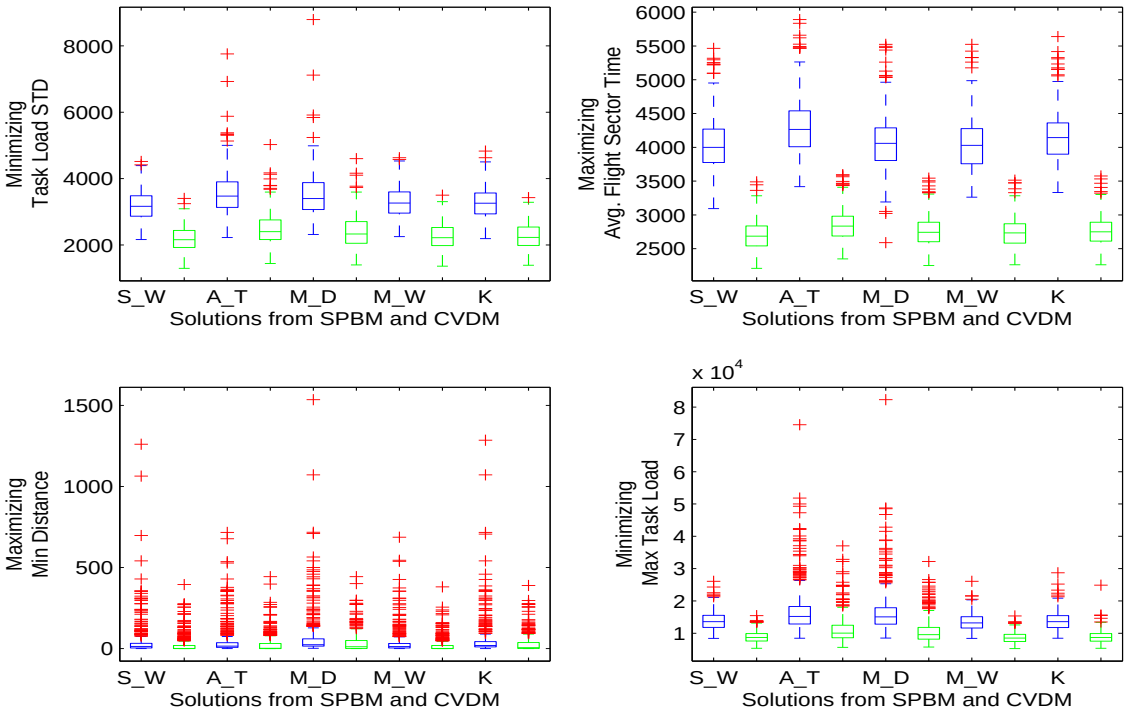


Figure 7.43: Comparisons on Objective Achievements between the SPBM and CVDM in Scenario 2

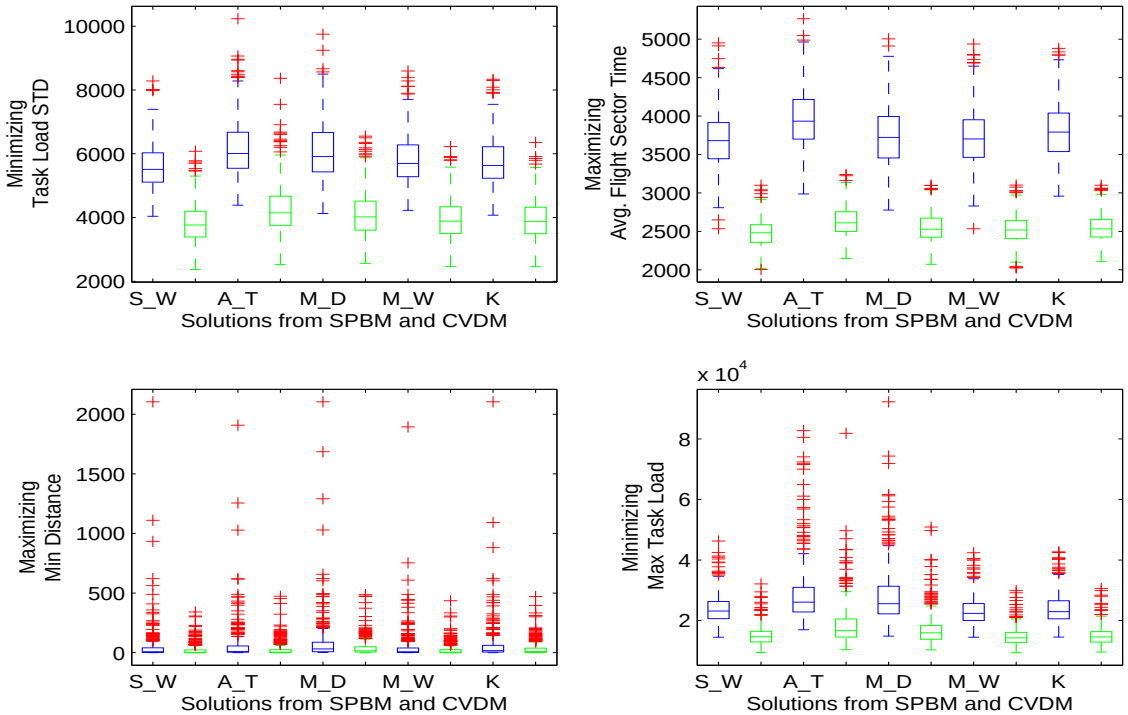


Figure 7.44: Comparisons on Objective Achievements between the SPBM and CVDM in Scenario 3

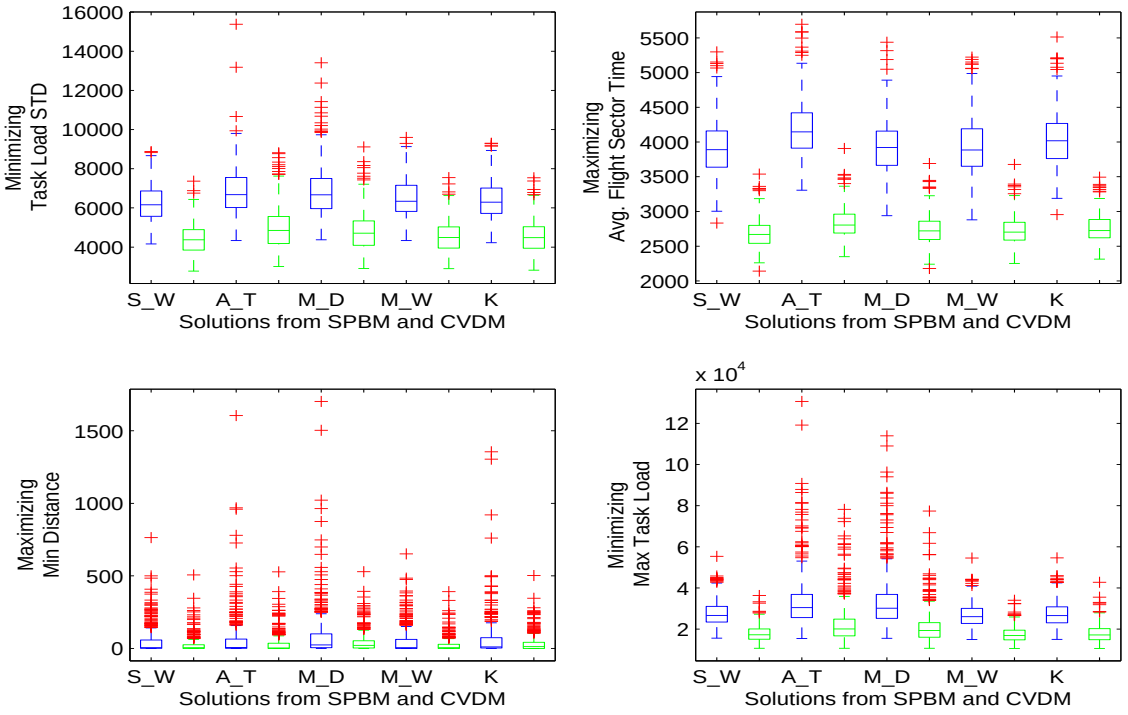
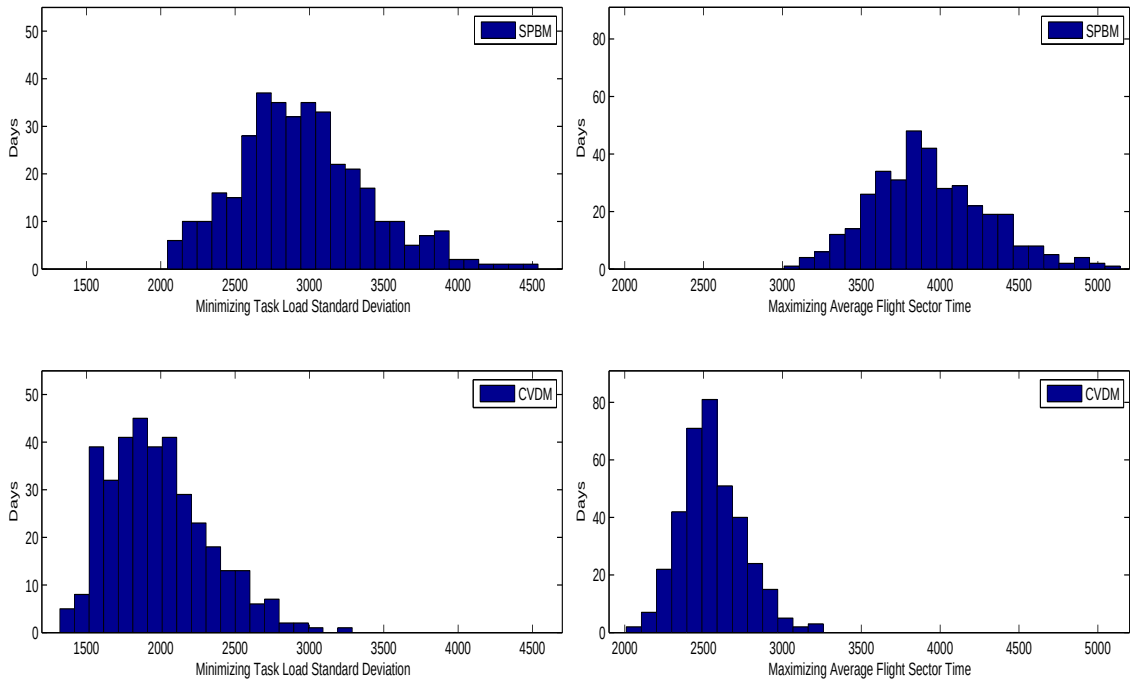


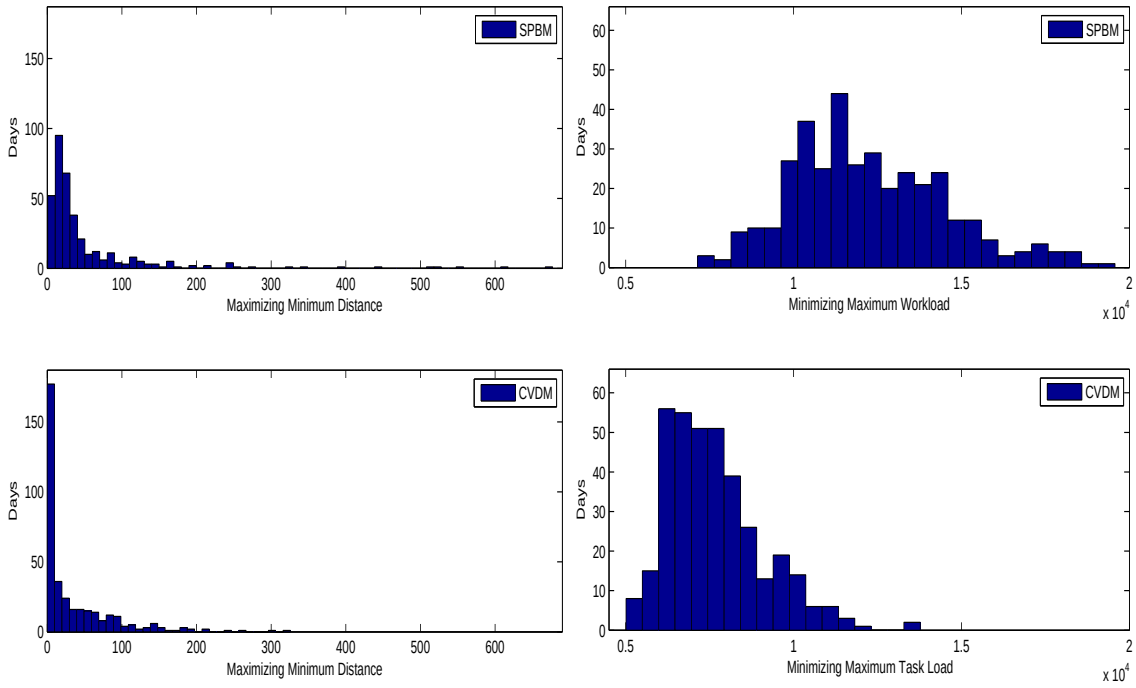
Figure 7.45: Comparisons on Objective Achievements between the SPBM and CVDM in Scenario 4

The difference on the minimum distance between sector boundaries and traffic flow crossing points are small in all scenarios between the two models as shown in the figures although SPBM is better than CVDM. However, the differences of the other three objectives between the two models are large. SPBM has longer average flight sector time which is two times of CVDM in all scenarios. On the contrary, CVDM achieve the half task load standard deviation and maximum task load as the results by SPBM. This situation maybe caused by the different sector shapes generated by the two models. CVDM can generate more varied sector shapes than SPBM which helps distribute task load. Whereas, it also potentially leads to more traffic flow cuts and restricts flights to stay within a sector longer. This will be investigated in the next section.

I take the solutions with the balanced objectives (K) from both models to compare them further, which is illustrated in histograms based on the distributions of the number of days and objective values in the Figure 7.46, 7.47, 7.48, and 7.49.

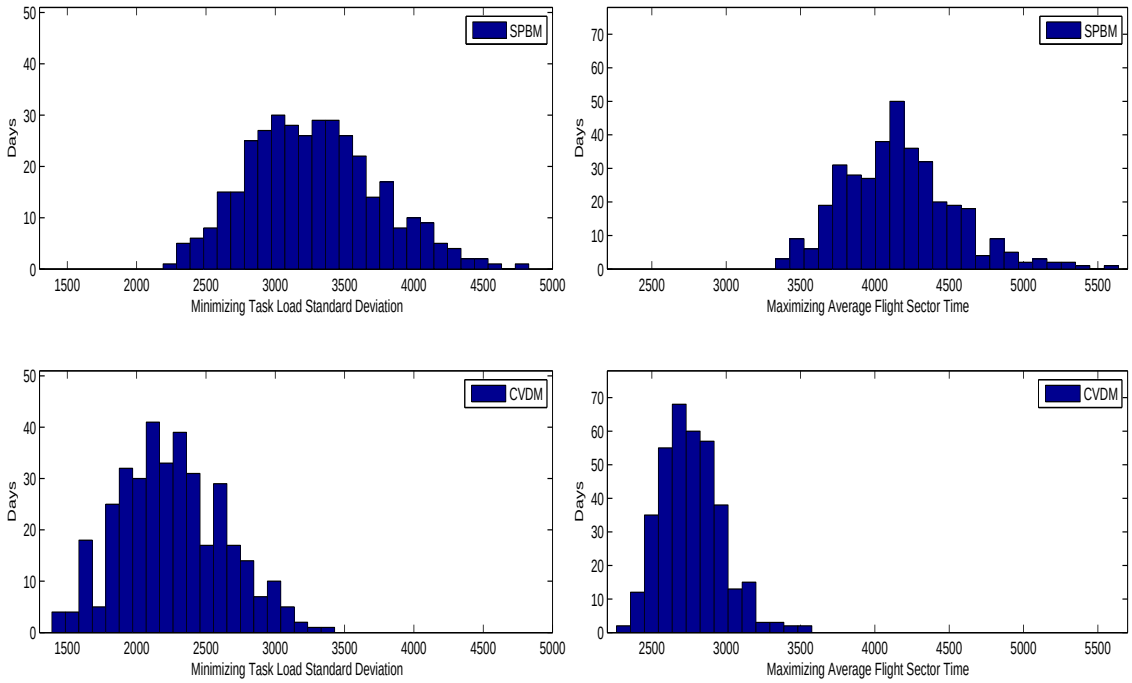


(a) Distribution of Minimizing Task load STD (b) Distribution of Maximizing Avg Flight Sector Time

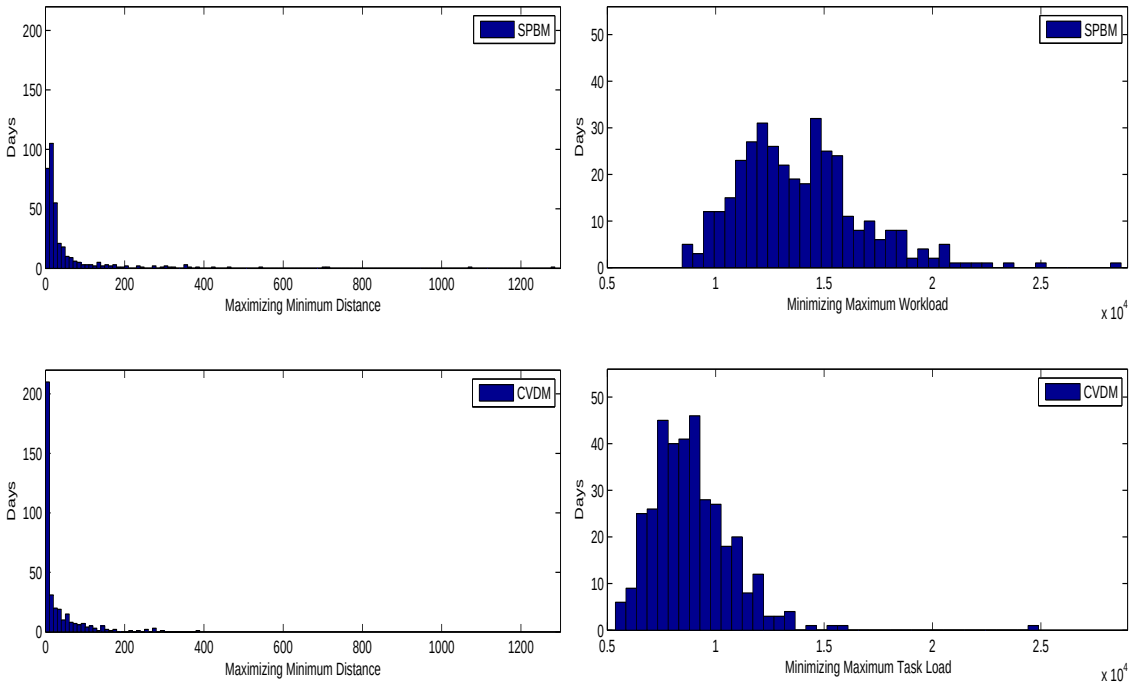


(c) Distribution of Maximizing Min Distance (d) Distribution of Minimizing Max Task load

Figure 7.46: Distributions of Solutions with the Balanced Objectives by SPBM and CVDM in Scenario 1

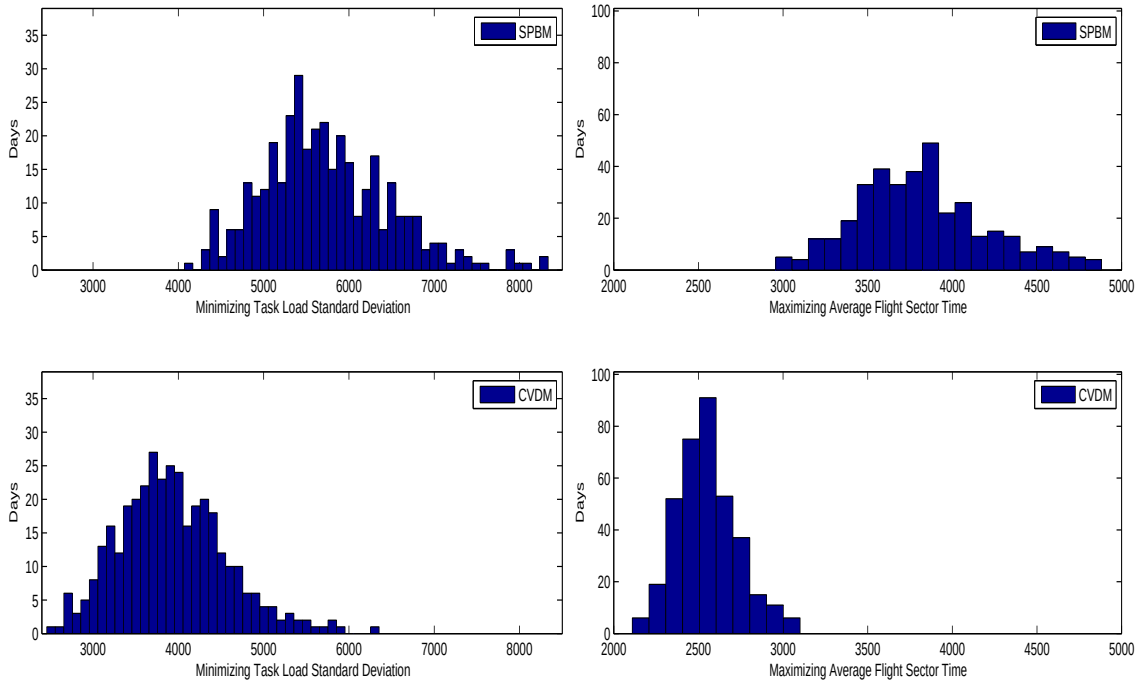


(a) Distribution of Minimizing Task load STD (b) Distribution of Maximizing Avg Flight Sector Time

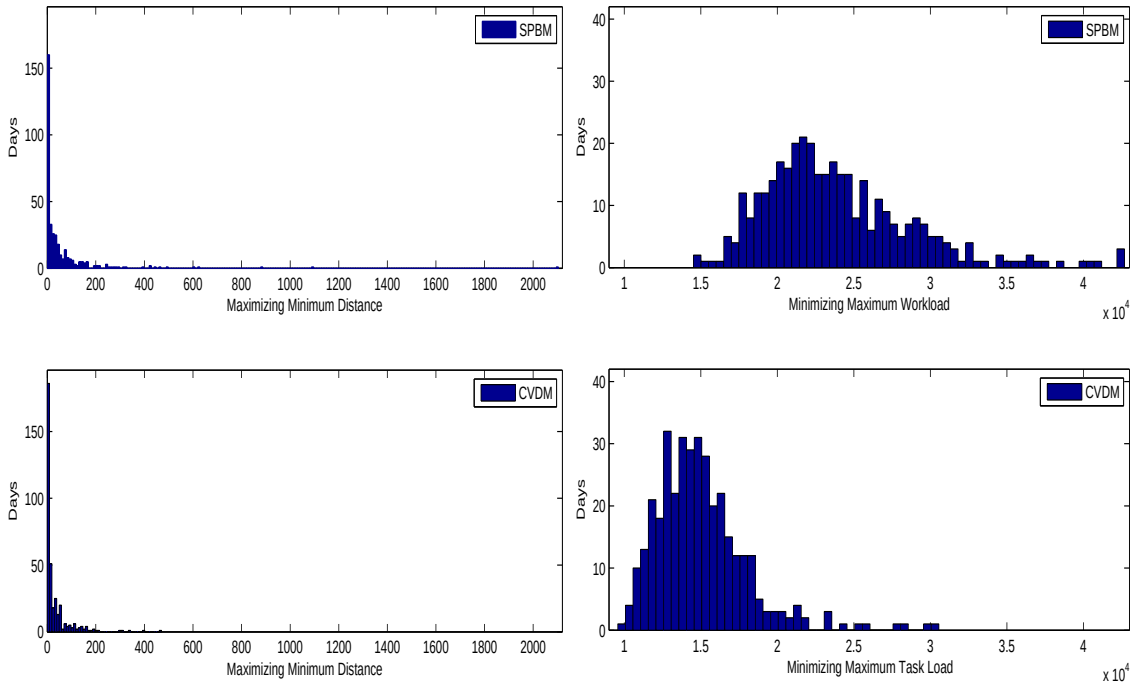


(c) Distribution of Maximizing Min Distance (d) Distribution of Minimizing Max Task load

Figure 7.47: Distributions of Solutions with the Balanced Objectives SPBM and CVDM in Scenario 2

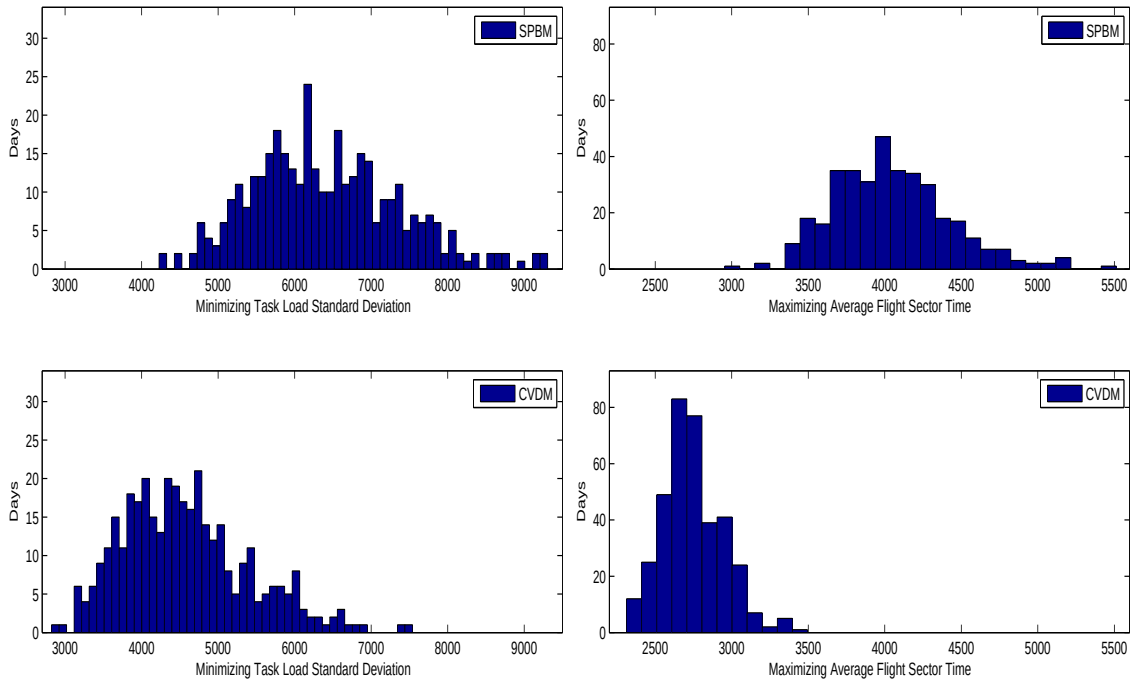


(a) Distribution of Minimizing Task load STD (b) Distribution of Maximizing Avg Flight Sector Time

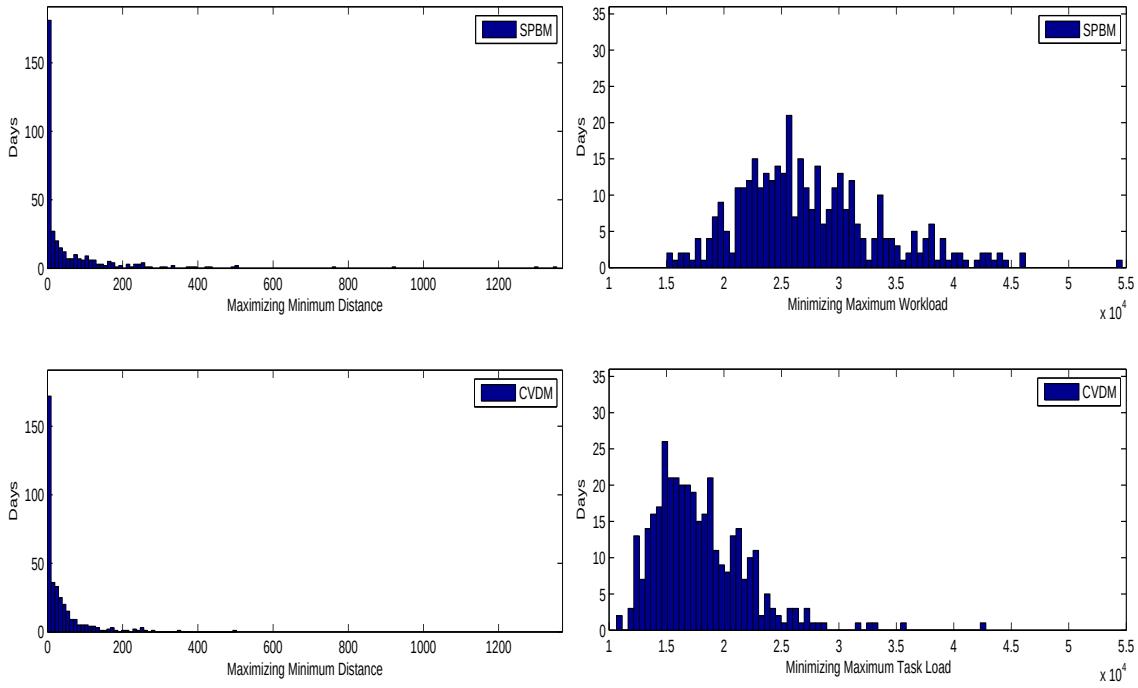


(c) Distribution of Maximizing Min Distance (d) Distribution of Minimizing Max Task load

Figure 7.48: Distributions of Solutions with the Balanced Objectives by SPBM and CVDM in Scenario 3



(a) Distribution of Minimizing Task load STD (b) Distribution of Maximizing Avg Flight Sector Time



(c) Distribution of Maximizing Min Distance (d) Distribution of Minimizing Max Task load

Figure 7.49: Distributions of Solutions with the Balanced Objectives by SPBM and CVDM in Scenario 4

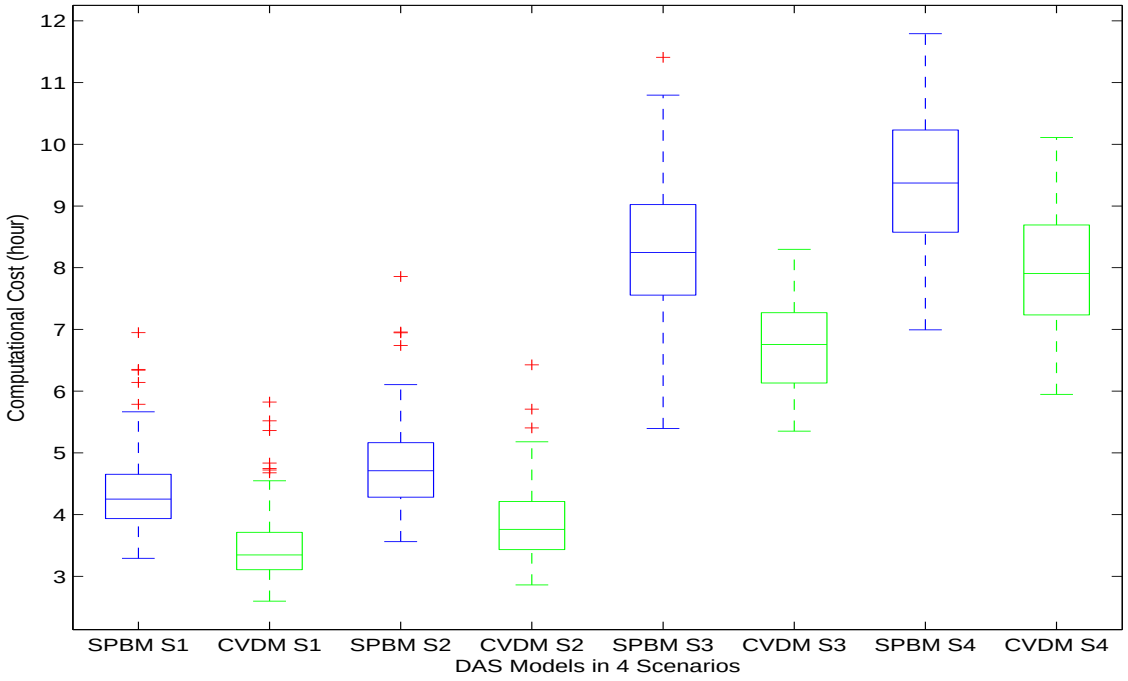


Figure 7.50: Computational Costs By SPBM and CVDM in 4 Scenarios

Figure 7.50 shows the time spent by both models in four scenarios. CVDM has less computational cost than SPBM in all scenarios which is consistent with the experiments conducted in the last chapter. The costs for Scenario 1 and 2 are around half of the costs for Scenario 3 and 4 because the traffic demands are doubled in Scenario 3 and 4 but the numbers of sectors remained the same. It conforms to the computational costs ($O(S \times N_F)$) for grouping flight track hits into sectors estimated in Chapter 4 for both models. The costs for the scenarios (Scenario 2 and 4) with flexible flight tracks are a little higher than the cost for the scenarios (Scenario 1 and 3) only allowing the classic flight routs respectively as shown in the figure. It is caused by the flexible tracks which make flight more easily jump from one sector to another with different parent nodes and it leads to more searching time to group flight track hits into sectors. The relationship between the computational cost of my models and the air traffic volume is almost linear.

In summary, the differences on the objective achievement exist among the two models although they both have better performances than the current airspace sectorization. SPBM is good at increasing the flight sector time, but CVDM is better

in balancing the task load. Moreover, CVDM has less computational costs than SPBM too.

7.3 Case Study

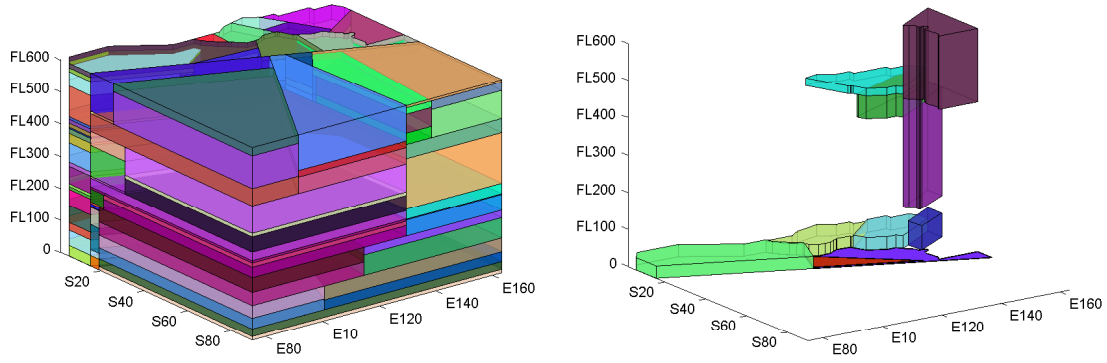
In the last section, the cases with a poor performance when compared to the current airspace sectorization are found in both models. Theses cases have only worse performance minimizing task load standard deviation and minimizing maximum task load of activated sectors, therefore, only the worst of them from both models are selected and analysed in this section. Secondly, the different behaviours between SPBM and CVDM are also investigated by a case study here.

7.3.1 Worst Case Analysis

As shown in the comparison, CVDM has a better task load balancing than the current sectorization when the solutions are aiming to minimize the task load standard deviation and to minimize the maximum task load. The model only performs poorly when it needs longer flight sector time because of the trade-off between the task load balancing and flight sector time as explained in the last chapter. Some worse cases happen when the algorithm aims at increasing minimum distance between sector boundaries and traffic flow crossing points. To achieve this objective, there are more restrictions on the sectorization. However, SPBM has some days when it can't achieve the better results even though the solutions are biased towards the task load balancing. Therefore, the worst case study for the poor task load balancing focuses on SPBM only.

Two solutions on two days in two different scenarios are selected from the SPBM results, which have a worse standard deviation than the current sectorization. Figure 7.51 is showing the first airspace sectorization on Day 315 in Scenario 1. It has the task load standard deviation value at 4476.83 which is 32.37% higher than the value (3381.94) from the current sectorization but the maximum task load (18053.69)

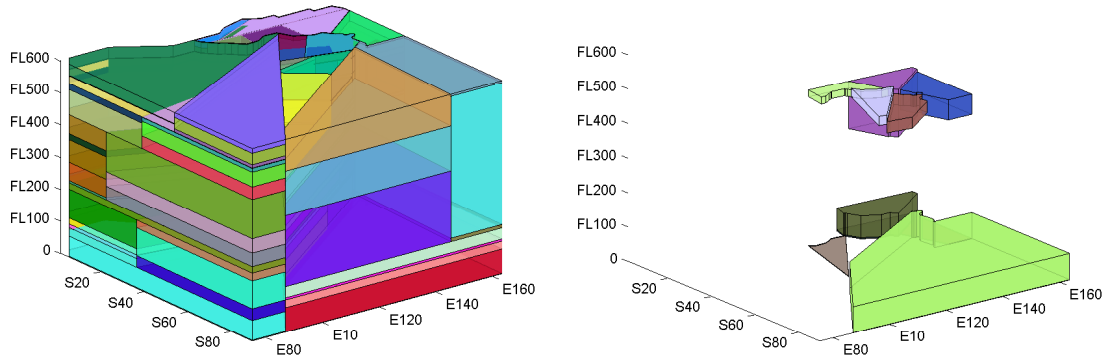
is better than the current sectorization (28245.07). The airspace sectorization is visualized in Figure 7.51a. Another example from SPBM happening on Day 19 in Scenario 4 is shown in Figure 7.52a, which has the standard deviation of 8491.22 and the maximum task load of 55376.73. Both them are 30.57% and 15.74% worse respectively than the current sectorization which are 6503.40 and 47844.41.



(a) Sectorization on Day 315 in Scenario 1 (b) Top 10 Sectors with Highest Task load

Figure 7.51: Airspace Sectorization by SPBM on Day 315 in Scenario 1

As shown in Figure 7.51b, the 10 sectors with the highest task load among sectors are neighbouring with each other vertically or laterally and are clustered in two parts of the airspace. This restricts SPBM to adjust the boundaries for task load balancing. This is also found in another example as shown in Figure 7.52b.



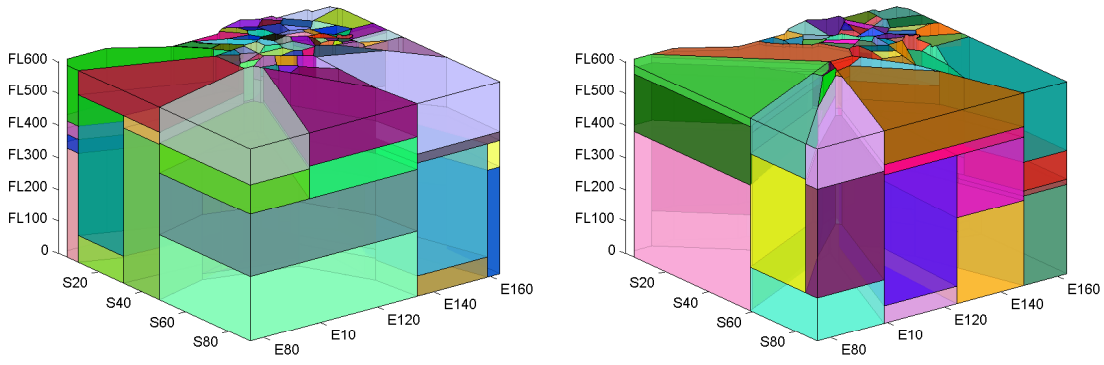
(a) Sectorization on Day 19 in Scenario 4 (b) Top 10 Sectors with Highest Task load

Figure 7.52: Airspace Sectorization by SPBM on Day 19 in Scenario 4

The sectors with highest task load congested at one place restricts the model to adjust the sector boundaries for task load balancing or reduction. On the other hand, SPBM used by the model has less flexibility to configure the sector lateral shapes. Instead of lateral sectorization, SPBM shows the preference to use vertical divisions for task load balancing. As shown in the figures, many thin vertical layers exist in both examples. These make task load balancing a hard job for SPBM. Meanwhile, these sectors are all along the boundaries of the two control centres as shown in the figures, which also restricts the flexibility of the sector shape tunings to minimizing task loads for these sectors. However, the vertical layers in the en-route airspace where flights fly at certain altitudes help it to achieve the higher flight sector time. In both examples, the average flight sector time is 4044.39sec for the first and 3847.75sec for the second which are 5 times of the current airspace sectorization.

7.3.2 Differences between DAS Models

When I compare SPBM and CVDM, I found that each of them specialise on some objectives. Two examples are shown in Figure 7.53 which are generated by the same day air traffic used to investigate SPBM in the previous section.



(a) Sectorization on Day 315 in Scenario 1 (b) Sectorization on Day 19 in Scenario 4

Figure 7.53: Airspace Sectorizations from CVDM on Day 315 in Scenario 1 and on Day 19 in Scenario 4

CVDM achieves a task load standard deviation of 2008.38 and the maximum

task load of 9735.81 on Day 315 in Scenario 1 and 4038.84 and 15594.96 for both respectively on Day 19 in Scenario 4, which are all much better than both SPBM and the current airspace sectorization. On the other hand, with the results on the average flight time (2101.30sec) is also better than the current sectorization, they are not as good as the results from SPBM.

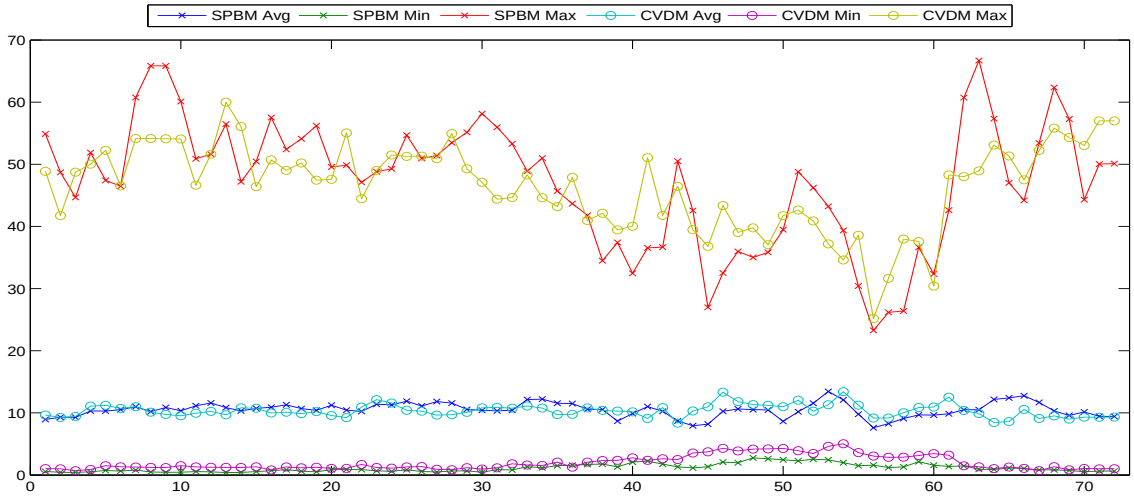
It is obvious that both models have different behaviors to partition the airspace when comparing Figure 7.53 with Figure 7.51 and Figure 7.52. CVDM partitions the airspace more in lateral than SPBM, but SPBM is keen to layer the airspace vertically. Both models partition the airspace alternately in lateral and vertical dimensions as described in Chapter 4. SPBM only produces two lateral subspaces at each turn of lateral partitioning but CVDM can generate more than two lateral subspaces. Both models use only one point to partition the airspace vertically. Theoretically, SPBM has $\frac{2}{3}$ chances to generate lateral partitions because 2 points are used for a lateral partition and 1 point is used for a vertical partition. However, CVDM has $\frac{s}{s+1}$ probability to partition the airspace laterally, where s is the number of sites and is greater than 2. Therefore, SPBM has less chances to partition an airspace in the lateral dimension than CVDM.

Different behaviours of the two models results in their different performances on objectives. The bigger lateral blocks can keep flights staying within the blocks longer, the smaller lateral blocks can balance the task load easily but has to cut the traffic flows. That is why SPBM is better on flight sector time while CVDM has advantages to balance the task load and to push down the maximum task load. Meanwhile, the big lateral sectorization helps SPBM to get larger minimum distance between sector boundaries and traffic flow crossing points.

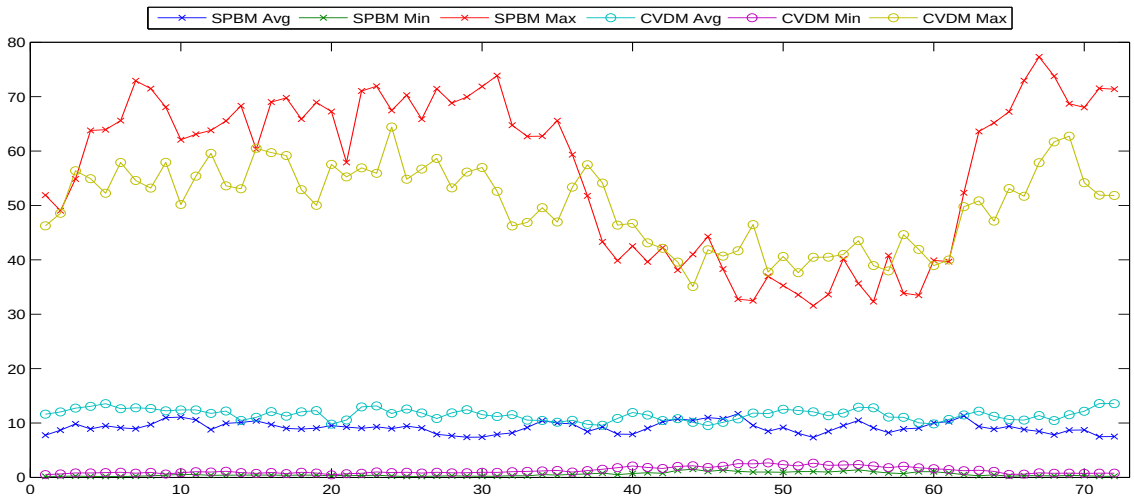
7.3.3 Dynamic Density Metrics

The air traffic of Day 315 in Scenario 1 and Day 19 in Scenario 19 and the solutions (K) with the balanced performance from both models are further investigated in TOP-LAT by the Complexity Components. The DD Metrics excluding

the duplicated measurements, which are used in this investigation, are summarized in Table C.1 in the Appendix. An equally weighted sum of these metrics is used as the final result. Figure 7.54 is illustrating the average, minimum, and maximum of DD metrics calculated by the airspace sectorization from both SPBM and CVDM in 20 minutes interval.



(a) DD Metrics on Day 315 in Scenario 1



(b) DD Metrics on Day 19 in Scenario 4

Figure 7.54: Dynamic Density Metrics for SPBM and CVDM: the average, maximum, minimum of DD metrics in every 20 minutes interval.

As shown in the figure, CVDM has better results than SPBM although the difference between them is small. The variances between minimum and maximum values are large for both models, but the average DD values remain stable all time.

The results show that the task loads measured by traffic hits are different from DD metrics, with the latter providing a better picture of workload.

7.4 Chapter Summary

In this chapter, a number of experiments are undertaken to investigate my proposed DAS models. A comparison against the benchmark derived from today's sectorization is conducted. As the results show, both models have advantages in terms of the specified objectives, although some worse cases happened in task load balancing. An investigation on these cases shows that SPBM is lacking the flexibility to configure the lateral boundaries of sectors. CVDM is better in balancing each objective achievements as described in this chapter but it is not as good as SPBM if the target is to achieve a longer flight sector time. In addition, the computational costs of both models grow linearly with the increase in air traffic volume, which means both models have good scalability to handle large airspace and traffic. Meanwhile, it also shows that the DAS models can provide airspace sectorization suggestion to users through TOP-LAT.

Chapter 8

Conclusion

8.1 Summary

In this thesis, I presented and investigated four models based on agent or geometric methods for 3D Dynamic Airspace Sectorization. I further investigated two of them, the Support Plane Bisection Model (SPBM) and Constrained Voronoi Diagram Model (CVDM), in an Air Traffic monitoring and advisory system with different air traffic of a year within the Australian national airspace. From the investigation, the proposed models have achieved the intended results.

The development of the four 3D DAS models is formulated in three stages. In the first stage, the state-of-art DAS model (ABM) is experimentally investigated and critical gaps are identified. These gaps include sector design violation and high computational cost, which limit the feasibility and scalability of this model. According to the findings, I have developed an improved agent based model (iABM) which overcomes all limitations of ABM, such as right prism violation, embedded sectors, and reduces the computational cost. However, the grid based iABM has shortcomings when it comes to the variations in the convexity of sector shape. It isn't worth it to introduce a new agent rule for convexity while increasing the computational cost because the grid based approach can only generate cuboid sectors when convexity is required. Therefore, in the second stage, three new geometric based models are

proposed. The KD-tree based Model is investigated first. Although it can satisfy sector design constraints including convexity, right prism, and others, the sector shapes are limited to boxes. Hence, the Support Plane Bisection Model (SPBM) and Constrained Voronoi Diagram Model (CVDM) are developed to introduce variety of sector geometric shapes. These three models are much more efficient than the agent based model and meet all requirements of sector design but these models don't work on the raw air traffic hits directly. For this reason, an algorithm to group traffic hits into sectors according to spatial locations of the hits is implemented. The complexity of it increases linearly with the increase in the number of active flights and the number of sectors. The objectives of DAS, such as task load balancing and traffic flow alignment, can be calculated by the output of the algorithm. In the final stage, I integrate NSGA-II into all four DAS models in order to optimize for airspace sectorizations for three objectives including minimizing standard deviation of task load across sectors, maximizing average sector flight time, and maximizing distance between sector boundaries and traffic flow crossing points. Evaluations of the four models by sampling air traffic data within an airspace section are conducted and the results demonstrate that the objectives are optimized by all proposed models although each model has its own advantages over the others. Two of them, SPBM and CVDM, have better overall performances in all objectives as well as good efficiency. It was necessary to investigate both models further on a large scale national airspace level with different traffic situations. A real time air traffic monitoring and advisory system, called TOP-LAT, is designed and implemented. Artificial air traffic scenarios reflecting different air traffic demands and prototyping advanced ATM concept are designed for the evaluation. The results show that SPBM and CVDM have significant improvements on the proposed objectives. Different behaviours of the models result in different outputs.

The findings and summary of the work carried out in this thesis are as follows:

1. By experimental evaluation of ABM, some limitations of it on the sector design are found, such as right prism violation and embedded sectors. In addition, the computational cost of ABM is $O(N^2)$, which rises with the square of the

number of cells (N) in a given airspace. This shows that ABM is not an efficient model for airspace sectorization, especially when the size of the airspace is increasing. All the identified gaps of ABM restrict its use in practice.

2. An improved agent model (iABM) is developed according to the findings from the evaluation of ABM. iABM overcomes the limitations of ABM. It guarantees sectorization to satisfy principles of sector design, such as the right prism constraint and avoidance of embedded sectors. It also reduces the computational cost to $O(N)$, which increases linearly with the number of cells (N) in a given airspace although it is increased dynamically by the Gap Filling Rule on occasions. iABM has advantages over ABM in both sector design and efficiency.

3. To address the convexity of sector shape while accommodating for other constraints, three geometric based models, KD-tree, Support Plane Bisection Model (SPBM), and Constrained Voronoi Diagram Model (CVDM), are developed and investigated. KD-tree is the most efficient model whose computational cost is $O(N \log N)$ where N is the number of the given points to partition the airspace. However, it produces only cuboid sector shapes. Therefore, the SPBM and CVDM are developed to introduce variety of sector shapes. The computational cost of SPBM is $O(S \times N)$ while $O(S \times N \log N)$ is the cost of CVDM. N is the number of given points and S is the expected sectors number. All three models have significant improvements on the computational cost compared with iABM. An algorithm to assign air traffic hits into sectors according to their locations is necessary. Based on the data structure (tree) of these models and flight movements, an efficient search algorithm based on BFS is developed. The computational cost of it is $O(S \times N_F)$ where S is the number of sectors and N_F is the number of the flights flying in the airspace.

4. The four proposed models with NSGA-II are used to find optimal airspace sectorization. In the experiments for an airspace section, all four models achieved the expect results for three objectives: minimizing standard deviation of task

load across sectors, maximizing average sector flight time, and maximizing distance between sector boundaries and traffic flow crossing points. The results show that the trade-offs between objectives, specially the task load standard deviation and average flight sector time. Moreover, their performance varies and the comparisons of the models show that both SPBM and CVDM have better overall performances.

5. An evaluation environment (TOP-LAT) is developed in order to evaluate SPBM and CVDM further for the Australian airspace with different air traffic demands and other advanced ATM concepts (UPT). Comparing with the current airspace sectorization, both models have achieved better overall results for all objectives. Different behaviours are found in the two models, which results in different sectorization and performance on each objective. The SPBM prefers to slide the airspace into many layers vertically while the CVDM generates more lateral partitions than vertical divisions. This evaluation also shows the capability of TOP-LAT to provide advice for ATM to decision makers.

8.2 Future Work

Various avenues of further research stem from the work carried out in this thesis. Some open research questions have already been highlighted in the respective chapters where they directly follow on from the work completed in the experiments. Here we outline more diverse future research directions.

The proposed models focus on independent airspace sectorization according to the air traffic demands. Since ATC has to be trained to be familiar with a sector configuration, (boundaries, geographic characteristics, etc.), smaller changes on a sector configuration are easier for ATC to adapt to the new sector configuration. Therefore, Dynamic Airspace Re-sectorization, where DAS is frequently used in the operational environment, has to consider and measure the changes among the consecutive airspace sectorizations. The measurement on sector configuration changes

has to be investigated before it is introduced into Dynamic Airspace Re-sectorization approaches.

Although the traffic hits count is commonly used as workload measurement in DAS approaches, Dynamic Density (DD) may be a more accurate workload measurement. However, there are two problems that need to be solved before using DD for workload measurement in DAS. First, an accurate DD prediction method is necessary because DAS works on the predicted DD to produce airspace sectorization in advance of the actual occurrence of the air traffic. DD metrics are derived from flight trajectories which include calculations on flight headings, speed changes, climb or descent rates, potential conflicts and so on. All of these require an accurate flight trajectory prediction method. However, current trajectory prediction methods can produce accurate measurements only up to 20 minutes (Kopardekar and Magyarits, 2003). It is impossible to change airspace sectors in every 20 minutes. Therefore, an accurate trajectory and DD prediction method is required by DAS approaches if workload is measured by DD metrics. Second, DD requires large computational resources to conduct the calculation of complexity metrics. An efficient method for DD calculations is required by DAS.

Although the proposed advantages of DAS models are demonstrated in the computational evaluations, cognitive experiments by ATC on the proposed DAS models may be needed to investigate them further from a human factors perspective. The feedback from ATC provides perspectives in addition to the computer based measurements, that will help to improve the models.

Appendix A

The Australian Flight Information Region

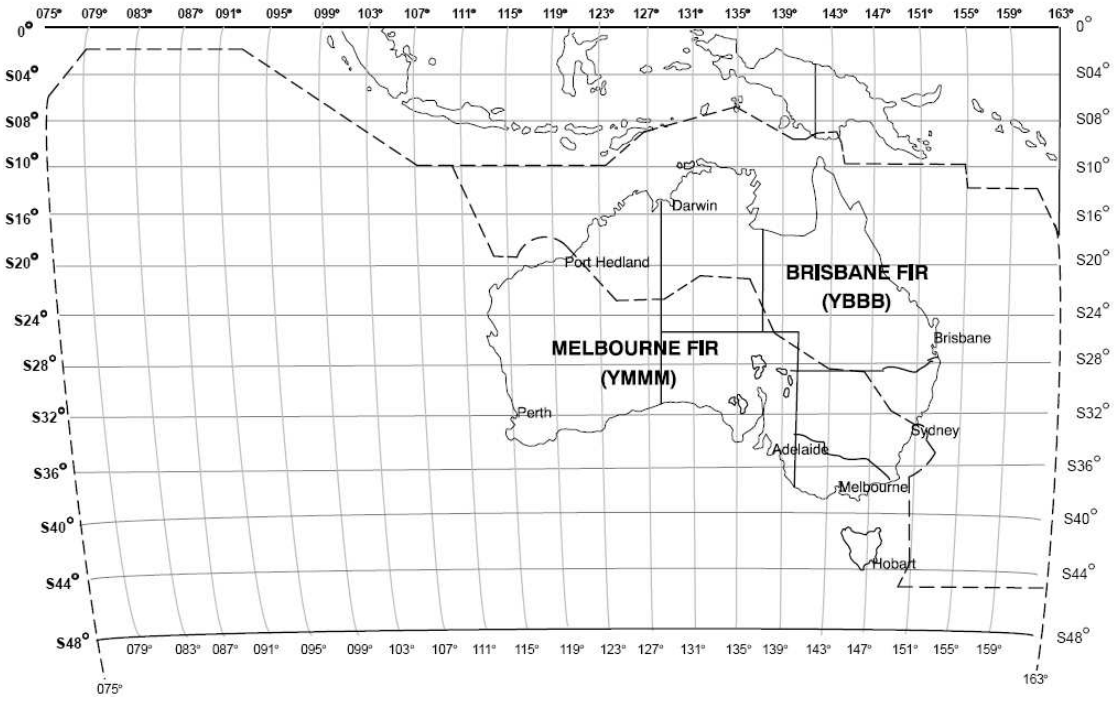


Figure A.1: The Australian Flight Information Region and the Two Control Centres

Appendix B

BADA parameters and their descriptions

Table B.1: BADA parameters and their descriptions.

Category	**Parameter and Description**
aircraft type	n_{eng} number of engines [-] engine type Jet/Turboprop/Piston wake category Heavy/Medium/Light
mass	m_{ref} reference mass [t] m_{min} minimum mass [t] m_{max} maximum mass [t] m_{pyld} maximum payload [t]
flight envelope	v_{MO} max. operating speed [kt] M_{MO} max. operating Mach number [-] h_{MO} max. operating altitude [ft] h_{max} max. altitude at MTOW and ISA [ft] G_W weight gradient on max. altitude [ft/kg] G_t temp. gradient on max. altitude [ft/C]
aerodynamics	S reference wing surface area [m] $C_{D0,CR}$ parasitic drag coefficient (cruise) [-] $C_{D2,CR}$ induced drag coefficient (cruise) [-] $C_{D0,AP}$ parasitic drag coefficient (approach) [-] $C_{D2,AP}$ induced drag coefficient (approach) [-] $C_{D0,LD}$ parasitic drag coefficient (landing) [-] $C_{D2,LD}$ induced drag coefficient (landing) [-] $C_{D0,\Delta LDG}$ parasitic drag coef. (landing gear) [-] C_{M16} Mach drag coefficient [-] $(V_{stall})i$ stall speeds for TO,IC,CR,AP,LD [kt] $C_{Lbo(M=0)}$ Buffet onset lift coef. [-] *jets only* K Buffeting gradient [1/M] *jets only*
engine thrust	$C_{Tc,1}$ 1st max. climb thrust coefficient [N] $C_{Tc,2}$ 2nd max. climb thrust coefficient [ft] $C_{Tc,3}$ 3rd max. climb thrust coefficient $[1/ft^2]$ $C_{Tc,4}$ 1st thrust temperature coefficient $[C]$ $C_{Tc,5}$ 2nd thrust temperature coefficient $[1/C]$ $C_{Tdes,low}$ low alt. descent thrust coefficient [-] $C_{Tdes,high}$ high altitude descent thrust coef. [-] h_{des} transition altitude [ft] $C_{Tdes,app}$ approach thrust coefficient [-] $C_{Tdes,ld}$ landing thrust coefficient [-] $V_{des,ref}$ reference descent speed [kt] $M_{des,ref}$ reference descent Mach number [-]
fuel flow	C_{f1} 1st TSFC coefficient [kg/min/kN] C_{f2} 2nd TSFC coefficient [kt] C_{f3} 1st descent fuel flow coefficient [kg/min] C_f4 2nd descent fuel flow coefficient [ft] C_{fer} cruise fuel flow correction coefficient [-]
ground operation	TOL take-off length [m] LDL landing length [m] span wingspan [m] length aircraft length [m]

Appendix C

Dynamic Density Metrics and their descriptions

Table C.1: Summary Dynamic Density Metrics and their descriptions.

Metric	**Description DD Metrics**	
AD2	Aircraft density 2 - number of aircraft/sector volume	WJTHC Titan System Metric
SV	Sector volume	WJTHC Titan System Metric
ACSQ	Square of aircraft count	WJTHC Titan System Metric
C1	Number of aircraft	NASA Metric 1
C2	Number of climbing aircraft	NASA Metric 1
C3	Number of cruising aircraft	NASA Metric 1
C4	Number of descending aircraft	NASA Metric 1
C14	Variance of speed	NASA Metric 1
C15	Ratio of standard deviation of speed to average speed	NASA Metric 1
NH	Number of aircraft with Heading Change greater than 15o	NASA Metric 2
NS	Number of aircraft with Speed Change greater than 10 knots or 0.02 Mach	NASA Metric 2
NA	Number of aircraft with Altitude Change greater than 750 feet	NASA Metric 2
S5	Number of aircraft with 3-D Euclidean distance between 0-5 nautical miles excluding violations	NASA Metric 2
S10	Number of aircraft with 3-D Euclidean distance between 5-10 nautical miles excluding violations	NASA Metric 2
S25	Number of aircraft with lateral distance between 0-25 nautical miles and vertical separation less than 2000/1000 feet above/below 29000 ft Number of aircraft with lateral distance between	NASA Metric 2
S40	Number of aircraft with lateral distance between 25-40 nautical miles and vertical separation less than 2000/1000 feet above/below 29000 ft Number of aircraft with lateral distance between	NASA Metric 2
S70	Number of aircraft with lateral distance between 40-70 nautical miles and vertical separation less than 2000/1000 feet above/below 29000 ft	NASA Metric 2
WBPROX	Count of number of aircraft within a threshold distance of a sector boundary (e.g., 10 miles).	Metron Aviation Metric

Bibliography

Ahlstrom, U. and Jaggard, E. (2010). Automatic identification of risky weather objects in line of flight (airwolf). *Transportation Research Part C: Emerging Technologies*, **18**(2), 187 – 192.

Alam, S., Abbass, H., and Barlow, M. (2008). ATOMS: Air traffic operations and management simulator. *IEEE Transactions on Intelligent Transportation Systems*, **9**(2), 209–225.

Ball, M., Hoffman, R., Chen, C., and Vossen, T. (2001). Collaborative decision making in air traffic management: Current and future research directions. In *New Concepts and Methods in Air Traffic Management*, pages 17–30. Springer.

Barricelli, N. (1962). Numerical testing of evolution theories. *Acta Biotheoretica*, **16**, 69–98. 10.1007/BF01556771.

Basu, A., Mitchell, J. S. B., and Sabhnani, G. K. (2009). Geometric algorithms for optimal airspace design and air traffic controller workload balancing. *J. Exp. Algorithmics*, **14**, 2.3–2.28.

Baughcum, S. L., Tritz, T. G., Henderson, S. C., and Pickett, D. C. (1996). Scheduled civil aircraft emission inventories for 1992: Database development and analysis, NASA contractor report 4700. Technical report, NASA.

Bentley, J. (1980). Multidimensional divide-and-conquer. *Communications of the ACM*, **23**(4), 214–229.

Birant, D. and Kut, A. (2007). ST-DBSCAN: An algorithm for clustering spatial-temporal data. *Data & Knowledge Engineering*, **60**(1), 208 – 221. Intelligent Data Mining.

Bloem, M. and Kopardekar, P. (2008). Combining airspace sectors for the efficient use of air traffic control resources. In *AIAA Guidance, Navigation, and Control Conference*, pages 1–15, Honolulu, HI. AIAA.

Brian, K., Gregg, F., Sathya, B., Andrew, M., Joosung, L., Ian, W., Kelly, K., Maryalice, L., Curtis, H., Angel, M., Edward, M., and Warren, G. (2005). Federal Aviation Administration: SAGE — system for assessing aviation's global emissions, version 1.5, global aviation emissions inventories for 2000 through 2004, FAA-EE-2005-02. Technical report, Federal Aviation Administration.

Brinton, C. and Pledgie, S. (2008). Airspace partitioning using flight clustering and computational geometry. In *Digital Avionics Systems Conference, 2008. DASC 2008. IEEE/AIAA 27th*, pages 3.B.3–1 –3.B.3–10.

Bugayevskiy, L. and Snyder, J. (1995). *Map Projections: A Reference manual*. Taylor & Francis, London.

Careño, V. and Muñoz, C. (2007). Safety and Performance Analysis of the Non-Radar Oceanic/Remote Airspace In-Trail Procedure. Technical Report NASA/TM-2007-214856, NASA.

CASA (2005a). *Required Navigation Performance 4 (RNP 4) Operational Authorisation*. Civil Aviation Safety Authority, Canberra, ACT, Australia. AC 91U-3(0).

CASA (2005b). *Required Navigation Performance 4 (RNP 4) Operational Authorisation*. Civil Aviation Safety Authority, Canberra, ACT, Australia. AC 91U-2(0).

CASA (2008). *Airspace Chage Manual*. Civil Aviation Safety Authority, Canberra, ACT, Australia, 1.0 edition. CASADOC 061.

Chatterji, G. and Sridhar, B. (2001). Measures for air traffic controller workload prediction. In *the First AIAA Aircraft Technology, Integration, and Operations Forum, Los Angeles, CA*.

Clarke, J., Ho, N., Ren, L., Brown, J., Elmer, K., Tong, K., and Wat, J. (2004). Continuous descent approach: Design and flight test for Louisville International Airport. *Journal of Aircraft*, **41**(5), 1054–1066.

Conker, R. S., Moch-Mooney, D. A., Niedringhaus, W. P., and Simmons, B. T. (2007). New Process for Clean Sheet Airspace Design and Evaluation. In *The 7th US/Europe ATM Seminar, Barcelona, Spain*, pages 1–10.

Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine Learning*, **20**, 273–297.

Coulouris, G., Dollimore, J., Kindberg, T., and Blair, G. (2011). *Distributed Systems: Concepts and Design (5th Edition)*. Addison Wesley, 5 edition.

Deb, K. and Goyal, M. (1996). A combined genetic adaptive search (geneas) for engineering design. *Computer Science and Informatics*, **26**, 30–45.

Deb, K., Pratap, A., Agarwal, S., and Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE transactions on evolutionary computation*, **6**(2), 182–197.

Deb, K., Sindhya, K., and Okabe, T. (2007). Self-adaptive simulated binary crossover for real-parameter optimization. In *GECCO '07: Proceedings of the 9th annual conference on Genetic and evolutionary computation*, pages 1187–1194, New York, NY, USA. ACM.

Delahaye, D., Alliot, J.-M., Schoenauer, M., and Farges, J.-L. (1995). Genetic algorithms for automatic regrouping of air traffic control sectors. In *Proceedings of the 4th Annual Conference on Evolutionary Programming*, pages 657–672.

Delahaye, D., Schoenauer, M., and Alliot, J.-M. (1998). Airspace sectoring by evolutionary computation. In *IEEE World Congress on Computational Intelligence 1998.*, pages 218 –223.

Donovan, T. (2006). Concept for an integrated national surveillance and data communication infrastructure. In *Aerospace Conference, 2006 IEEE*, page 14.

Ehrmanntraut, R. and McMillan, S. (2007). Airspace design process for dynamic sectorisation. In *Digital Avionics Systems Conference, 2007. DASC '07. IEEE/AIAA 26th*, pages 3.D.2–1 –3.D.2–9.

Erzberger, H., Davis, T. J., and Green, S. (1993). Design of center-tracon automation system. In *AGARD Guidance and Control Symp. Machine Intelligence in Air Traffic Management*, pages 11.1–11.12, Berlin, Germany.

EUROCONTROL (2004a). Advanced airspace scheme concept document. Technical Report Edition 2.1, EUROCONTROL, Bretigny, France.

EUROCONTROL (2004b). User manual for base of aircrfat data (BADA). Technical Report Rev No:3.6, EUROCONTROL Experiment Center, Bretigny, France.

EUROCONTROL (2007). Control of User Preferred Trajectories in a constrained ATM environment. Technical Report Specification v1.0, EUROCONTROL, Bretigny, France.

EUROCONTROL (2008a). Challenges of growth 2008 summary report. Technical report, EUROCONTROL, Bretigny, France.

EUROCONTROL (2008b). EUROCONTROL long-term forecast: IFR flight movements 2008-2030. Technical Report Doc302 LTF08, EUROCONTROL, Bretigny, France.

EUROCONTROL (2009). EUROCONTROL safety regulatory requirement. Technical Report ESARR, EUROCONTROL, Bretigny, France.

FAA (2009). FAA Aerospace Forecast Fiscal Years 2009–2025. Technical report, Federal Aviation Administration.

FAA Advisory Circular 90-99 (2003). High altitude airspace design phase I. Technical Report FAA Advisory Circular 90-99, ATA-301, Federal Aviation Administration.

FAA Order 7210.3U (2006). Facility operations and administration. Technical Report FAA Order 7210.3U, Federal Aviation Administration.

FAA Order 7400.2F (2006). Procedure for handling airspace matters. Technical Report FAA Order 7400.2F, Federal Aviation Administration.

Federal Aviation Administration (2001). The measure of air traffic control sector complexity for the en route environment: Phase II experiment plan. FAA WJHTC internal document, Federal Aviation Administration.

Flener, P., Pearson, J., Ågren, M., Garcia-Avello, C., Celiktin, M., and Dissing, S. (2007). Air-traffic complexity resolution in multi-sector planning. *Journal of Air Transport Management*, **13**(6), 323 – 328.

Fortune, S. (1987). A sweepline algorithm for Voronoi diagrams. *Algorithmica*, **2**(1), 153–174.

Fraser, A. S. (1958). Monte Carlo Analyses of Genetic Models. *Nature*, **181**(Issue 4603), 208–209.

Garey, M. R. and Johnson, D. S. (1979). *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman and Company, New York, NY, USA.

Gianazza, D. and Alliot, J.-M. (2002). Optimization of air traffic control sector configurations using tree search methods and genetic algorithms. In *Digital Avionics Systems Conference, 2002. Proceedings. The 21st*, volume 1, pages 2A5–1 – 2A5–8 vol.1.

Goldberg, D. E. (1989). *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley Professional, Boston, MA, USA, 1 edition.

Hadley, J. A. and Sollenberger, R. L. (2001). Dynamic resectorization of airspace boundaries between adjacent air route traffic control centers. In *The 11th International Symposium on Aviation Psychology*. Columbus, OH: The Ohio State University.

Hoekstra, J., van Gent, R., and Ruigrok, R. (2002). Designing for safety: the Free Flight air traffic management concept. *Reliability Engineering and System Safety*, **75**(2), 215–232.

Holland, J. H. (1975). *Adaptation in natural and artificial systems*. MIT Press, Cambridge, MA, USA.

ICAO (1995). *ICAO Engine Exhaust Emissions Databank*. ICAO-9646-AN/943, International Civil Aviation Organization, Montreal, Quebec, Canada, 1st edition.

ICAO (1998). ICAO Annex 11: Air Traffic Services. *International Civil Aviation Organization*.

ICAO (1999a). ICAO Annex 2: Rules of the Air. *International Civil Aviation Organization*.

ICAO (1999b). *Manual on Required Navigation Performance(RNP)*. International Civil Aviation Organization, Doc 9613-AN/937, second edition.

ICAO (2004). *Manual of the ICAO Standard Atmosphere extended to 80 kilometers*. ICAO, third edition.

Joint Planning and Development Office (2007). Concept of Operations for the Next Generation Air Transportation System ver 2.0.

Khanna, S., Muthukrishnan, S., and Paterson, M. (1998). On approximating rectangle tiling and packing. In *Proc Symp. on Discrete Algorithms (SODA)*, pages 384–393.

Kicinger, R. and Yousefi, A. (2009). Heuristic method for 3D airspace partitioning: Genetic algorithm and agent-based approach. In *9th AIAA Aviation Technology, Integration, and Operations Conference (ATIO) and Aircraft Noise and Emissions Reduction Symposium (ANERS)*.

Klein, A. (2005). An efficient method for airspace analysis and partitioning based on equalized traffic mass. In *Proceedings of the 6th USA/Europe Air Traffic Management Research and Development Seminar*, pages 1–10.

Klein, A., Kopardekar, P., Rodgers, M., and Kaing, H. (2007). Airspace playbook: Dynamic airspace reallocation coordinated with the national severe weather playbook. In *Proceedings of the 7th AIAA Aviation Technology, Integration and Operations Conference*.

Kopardekar, P. (2000). Dynamic density: A review of proposed variables. Technical report, FAA WJHTC.

Kopardekar, P. and Magyarits, S. (2003). Measurement and prediction of dynamic density. In *Proceedings of the 5th USA/Europe Air Traffic Management Research and Development Seminar*.

Kopardekar, P., Bilimoria, K., and Sridhar, B. (2007). Initial concepts for dynamic airspace configuration. pages 18 – 20, Belfast, Northern Ireland. 7th AIAA Aviation Technology, Integration and Operations Conference (ATIO).

Laudeman, I., Shelden, S., Branstrom, R., and Brasil, C. (1999). Dynamic density: An air traffic management metric. Technical Report NASA/TMm1998-112226, NASA.

MacQueen, J. B. (1967). Some methods for classification and analysis of multivariate observations. In L. M. L. Cam and J. Neyman, editors, *Proc. of the fifth Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, pages 281–297. University of California Press.

Masalonis, A. J., Callaham, M. B., and Wanke, C. R. (2003). Dynamic density and complexity metrics for realtime traffic flow management. In *Proceedings of the 5th USA/Europe Air Traffic Management Research and Development Seminar Seminar*.

Mitchell, J. S. B., Sabhnani, G., Krozel, J. A., Hoffman, R. L., and Yousefi, A. (2008). Dynamic airspace configuration management based on computational geometry techniques. In *AIAA Guidance, Navigation and Control Conference and Exhibit*, number AIAA 2008-7225, Honolulu, HI.

Mitchell, M. (1996). *An Introduction to Genetic Algorithms*. MIT Press, Cambridge, MA.

MITRE (2011). User request evaluation tool. http://www.caasd.org/work/project_details.cfm?item_id=156. Accessed on 3/7/2011.

Mohleji, S. and Ostwald, P. (2003). Future vision of globally harmonized national airspace system with concepts of operations beyond year 2020. In *Digital Avionics Systems Conference, 2003. DASC '03. The 22nd*, volume 1, pages 4.A.4 – 41–14 vol.1.

Nanni, M. and Pedreschi, D. (2006). Time-focused clustering of trajectories of moving objects. *Journal of Intelligent Information Systems*, **27**, 267–289.

NCOIC (2008). Vision for a net centric aviation ecosystem. Technical Report 1.0, Network Centric Operations Industry Consortium.

Nuic, A. (2004). *User Manual for the Base of Aircraft Data (BADA) — Revision 3.6*. EUROCONTROL Experimental Centre, Bretigny, France, 3.6 edition.

Penner, J. E., Lister, D., Griggs, D. J., Dokken, D. J., and McFarland, M. (1999). *Aviation and the Global Atmosphere: A Special Report of the Intergovernmental Panel on Climate Change*. Cambridge University Press, 1 edition.

Prevot, T. (2002). Exploring the many perspectives of distributed air traffic management: The Multi Aircraft Control System MACS. In *Proceedings of the HCI-Aero*, pages 149–154.

Prevot, T. and Mercer, J. (2007). MACS: A Simulation Platform for Today's and Tomorrow's Air Traffic Operations. AIAA Modeling and Simulation Technologies (MST) Conference and Exhibit, Hilton Head, SC., AIAA-2007-6556.

P.U., L., Mercer, J., Gore, B., N., S., K., L., and R., H. (2008). Examining airspace structural components and configuration practices for dynamic airspace configuration. In *In proc. AIAA Guidance, Navigation, and Control Conference and Exhibit*, pages 18 – 21, Honolulu, HI.

Rechenberg, I. (1965). Cybernetic solution path of an experimental problem. In *Royal Aircraft Establishment Translation No. 1122, B. F. Toms, Trans.* Ministry of Aviation, Royal Aircraft Establishment, Farnborough Hants.

Rechenberg, I. (1973). *Evolutionsstrategie: optimierung technischer systeme nach prinzipien der biologischen evolution*. Frommann-Holzboog.

Sandrine, C. and Frank, J. (2006). GAES Advanced Emissions Model (AEM3) v1.5 — validation exercise 4. Technical Report EEC/SEE/2006/005, ENVISA.

Schmitt, A. and Brunner, B. (1997). Emissions from aviation and their development over time. in: Final report on the bmbf verbundprogramm, schadstoffe in der luftfahrt. Technical report, "DLR-Mitteilung 97-04, Deutsches Zentrum für Luft- und Raumfahrt, Oberpfaffenhofen and Cologne, Germany".

SESAR Consortium (2007). The ATM Target Concept: SESAR Definition Phase, Deliverable 3. Technical Report DLM-0612-001-02-00, EUROCONTROL, Brussels.

Shepherd, G., Byron, B., Mrdak, M., and Bills, K. (2007). Waypoint 2007 Airservices Australia air navigation service provision: Moving towards 2020. Technical report.

Sherali, H. D. and Hill, J. M. (2009). Reverse time-restricted shortest paths: Application to air traffic management. *Transportation Research Part C: Emerging Technologies*, **17**(6), 631 – 641.

Songchen, H. and Zhang, M. (2004). The optimization method of the sector partition based on metamorphic voronoi polygon. *Chinese Journal of Aeronautics*, **17**(1), 7–12.

Sridhar, B., Sheth, K., and Grabbe, S. (1998). Airspace complexity and its application in air traffic management. In *2nd USA/Europe Air Traffic Management R&D Seminar, Orlando, Florida.*

Sutkus, D. J., Baughcum, S. L., and DuBois, D. P. (2001). Scheduled civil aircraft emission inventories for 1999: Database development and analysis. Final Contractor Report NASA CR-2001-211216, National Aeronautics and Space Administration, Washington, DC 20546-0001.

Tien, S.-L. and Hoffman, R. (2009). Optimizing Airspace Sectors for Varying Demand Patterns using Multi-Controller Staffing. In *The Eighth USA/Europe Air Traffic Management Research and Development Seminar (ATM2009).*

Trandac, H. and Duong, V. (2002). A constraint-programming formulation for dynamic airspace sectorization. In *Digital Avionics Systems Conference, 2002. Proceedings. The 21st*, volume 1, pages 1C5–1 – 1C5–11 vol.1, Irvine, CA USA.

Trandac, H., Baptiste, P., and Duong, V. (2003). Airspace sectorization by constraint programming. In *IEEE International Conference on Research, Innovation & Vision for the Future (RIVF)*, pages 49–58.

Verlhac, C. and Manchon, S. (2001). Optimization of opening schemes. In *Proceedings of the fourth USA/Europe Air Traffic Management R&D Seminar*.

Weigang, L., Dib, M. V. P., Alves, D. P., and Crespo, A. M. F. (2010a). Intelligent computing methods in air traffic flow management. *Transportation Research Part*

C: Emerging Technologies, **18**(5), 781 – 793. Applications of Advanced Technologies in Transportation: Selected papers from the 10th AATT Conference.

Weigang, L., Dib, M. V. P., Alves, D. P., and Crespo, A. M. F. (2010b). Intelligent computing methods in air traffic flow management. *Transportation Research Part C: Emerging Technologies*, **18**(5), 781 – 793. Applications of Advanced Technologies in Transportation: Selected papers from the 10th AATT Conference.

Weiler, K. and Atherton, P. (1977). Hidden surface removal using polygon area sorting. *SIGGRAPH Comput. Graph.*, **11**, 214–222.

Wyndemere Inc. (1996). An evaluation of air traffic control complexity. Technical Report NAS2-14284, NASA.

Xue, M. (2008). Airspace Sector Redesign Based on Voronoi Diagrams. In *Proc. of AIAA Guidance, Navigation, and Control Conference and Exhibit*, pages 18–21.

Yousefi, A., Hoffman, R., Lowther, M., and Khorrami, B. (2009). *Trigger Metrics for Dynamic Airspace Configuration*. American Institute of Aeronautics and Astronautics, 1801 Alexander Bell Dr., Suite 500 Reston VA 20191-4344 USA,.

Zelinski, S. (2009). A comparison of algorithm generated sectorizations. In *In proc. Eighth USA/Europe Air Traffic Management Research and Development Seminar*, Napa, CA.

Zografos, K. G. and Madas, M. A. (2006). Development and demonstration of an integrated decision support system for airport performance analysis. *Transportation Research Part C: Emerging Technologies*, **14**(1), 1–17.