Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

A Novel Approach for Conflict Detection and Resolution for Trajectory-Based Operations in 4D-Navigation using NoSQL Databases and Local Search Algorithms

Vitor Filincowsky Ribeiro

Documento apresentado como requisito parcial
para conclusão do Doutorado em Informática

Orientador
Prof. Dr. Li Weigang

Brasília
2019

Universidade de Brasília — UnB
Instituto de Ciências Exatas
Departamento de Ciência da Computação
Programa de Pós-Graduação em Informática

Coordenador: Prof. Dr. Bruno Macchiavello

Banca examinadora composta por:

Prof. Dr. Li Weigang (Orientador) — CIC/UnB

Prof.ª Dr.ª Maria Emilia M. T. Walter — CIC/UnB

Prof. Dr. Anderson Ribeiro Correia — Membro Externo, IEI/ITA

Prof.ª Dr.ª Yaeko Yamashita — Membro Externo, ENC/UnB

Prof.ª Dr.ª Alba Cristina M. A. de Melo — CIC/UnB

CIP — Catalogação Internacional na Publicação

Ribeiro, Vitor Filincowsky.

A Novel Approach for Conflict Detection and Resolution for Trajectory-Based Operations in 4D-Navigation using NoSQL Databases and Local Search Algorithms / Vitor Filincowsky Ribeiro. Brasília : UnB, 2019. 140 p. : il. ; 29,5 cm.

Tese (Doutorado) — Universidade de Brasília, Brasília, 2019.

1. *navegação 4D, NoSQL, detecção de conflitos, busca local*

CDU 004

Endereço: Universidade de Brasília

Campus Universitário Darcy Ribeiro — Asa Norte

CEP 70910-900

Brasília–DF — Brasil

Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

A Novel Approach for Conflict Detection and Resolution for Trajectory-Based Operations in 4D-Navigation using NoSQL Databases and Local Search Algorithms

Vitor Filincowsky Ribeiro

Documento apresentado como requisito parcial
para conclusão do Doutorado em Informática

Prof. Dr. Li Weigang (Orientador)
CIC/UnB

Prof.ª Dr.ª Maria Emilia M. T. Walter
CIC/UnB

Prof. Dr. Anderson Ribeiro Correia
Membro Externo, IEI/ITA

Prof.ª Dr.ª Yaeko Yamashita
Membro Externo, ENC/UnB

Prof.ª Dr.ª Alba Cristina M. A. de Melo
CIC/UnB

Prof. Dr. Bruno Macchiavello
Coordenador do Programa de Pós-Graduação em Informática

Brasília, 25 de junho de 2019

Dedicatória

Dedico este trabalho à minha preciosa família, que sempre me apoiou e me ajudou a superar cada obstáculo em minha vida.

Agradecimentos

Primeiramente, agradeço a Deus por seu amor e graça infinitos, e por ser meu sempre presente socorro.

À minha amada esposa Sabrina e meu filho Alonso, por seu amor e compreensão em minhas muitas horas de ausência durante esta jornada, e por serem luz em minha vida.

Aos meus pais Josué e Eliane, por sempre me fornecerem o suporte necessário, e pelos preciosos ensinamentos de vida

Ao professor Dr. Li Weigang, por nunca deixar de acreditar em meu potencial.

À equipe Boeing Research and Technology - Brazil, pelo fundamental apoio à esta pesquisa.

Aos colegas do TransLab, por seu companheirismo e pelas experiências compartilhadas.

Ao Ten. Cristiano Garcia e ao professor Dr. Geraldo Filho, pelo trabalho de revisão da presente tese.

Aos meus chefes Adauto Freitas e Newton Machado, pelo fundamental apoio às minhas atividades acadêmicas.

Onde estavas tu, quando eu fundava a terra? Faze-mo saber, se tens inteligência.
Quem lhe pôs as medidas, se é que o sabes?
Ou quem estendeu sobre ela o cordel?

Jó 38:4,5

Resumo

O principal objetivo das Operações baseadas em desempenho é o gerenciamento de trajetórias de voo com base na capacidade operacional das aeronaves com a finalidade de diminuir os custos inerentes ao voo. Concomitantemente, a maximização da eficiência e capacidade do espaço aéreo devem ser observadas sem prejuízo das restrições operacionais de segurança do espaço aéreo. Novas tecnologias para previsão de trajetórias e detecção e resolução de conflitos (CD&R) em voo são essenciais para estabelecer um novo paradigma na aviação, de instruções de controle de tráfego aéreo (ATC) a operações baseadas em trajetórias (TBO). A implementação das TBO vem modernizando a estrutura do Gerenciamento de Tráfego Aéreo (ATM) avançado. Diversos algoritmos e metodologias para CD&R vêm sendo desenvolvidos para a comunidade da aviação, abrangendo desde busca extensiva a Programação Inteira e algoritmos evolucionários. O problema é então descobrir um esquema eficiente para armazenar e gerenciar as trajetórias na complexa rede de tráfego, com massiva quantidade de dados, para então ser possível detectar e resolver os conflitos. Nesta tese, um *framework* para CD&R é desenvolvido para o gerenciamento de trajetórias quadridimensionais (4DT) previstas. Com a utilização de esquemas especiais em bases de dados NoSQL, as trajetórias 4D de voos comerciais são apresentadas e os eventuais conflitos são detectados em complexidade computacional na ordem de $O(nlog_2(n))$ no caso médio, o que comprovadamente supera técnicas atualmente praticadas. Uma aplicação cliente de um Previsor de Trajetórias é apresentada. Esta modelagem permite a classificação automática de cada ponto 4D amostrado nos voos, onde conflitos entre as trajetórias são detectados via consultas aos bancos de dados. A inovação da abordagem reside no fato de que a computação é majoritariamente reservada ao TP, sem necessidade de conhecimento detalhado sobre as trajetórias executadas. Por fim, algoritmos de busca local para resolução dos conflitos são desenvolvidos para ajustar os atributos do plano de voo das aeronaves a fim de resolver os conflitos eventualmente encontrados no planejamento estratégico dos voos.

Palavras-chave: *navegação 4D, NoSQL, detecção de conflitos, busca local*

Abstract

The main goal in Performance Based Operations is the management of flight trajectories in order to optimize the operating capability of the aircraft and lower the overall cost of the flight. At the same time the maximization of airspace efficiency/capacity needs to be addressed considering local airspace requirements and constraints. New technologies for Trajectory Prediction and conflict detection and resolution (CD&R) are paramount to establish a new paradigm in aviation, from ATC instructions to Trajectory-Based Operations (TBO). The implementation of TBO has been updating the structure of the advanced Air Traffic Management (ATM). Several methodologies and algorithms for CD&R have been developed to the aviation community, ranging from extensive search to Integer Programming and Evolutionary algorithms. The legacy problem is to find an efficient scheme to store and manage the trajectories in the complex network with massive data and further to detect and resolve the conflicts. In this research a CD&R framework is developed for the management of predicted 4-Dimensional Trajectories (4DT). Using special schemes of Not Only SQL (NoSQL) databases, the 4D trajectories of commercial flights are presented and the eventual conflicts are detected with the computation complexity in the order of $O(nlog_2(n))$ in the average case, which is proven to outperform state-of-art techniques. A client application for a Trajectory Predictor is hereby presented. This modeling allows the automatic classification of each 4D point sampled during the flight, in which conflicts are detected via queries to the database. The innovation in this approach lies on the fact that the computation is mainly reserved to TP, without the need of detailed knowledge about the performed trajectories. Finally, local search algorithms for conflict resolution are developed to adjust the parameters in the aircraft's flight plan in order to resolve the conflicts eventually found during the strategic planning phase.

Keywords: *4D navigation, NoSQL, conflict detection, local search*

Acronyms

4D - Four-dimensional Trajectory

A-CDM - Airport Collaborative Decision Making

ACC - Area Control Center

ADS-B - Automatic Dependent Surveillance-Broadcast

AHP - Airborne Holding Problem

AIDL - Aircraft Intent Description Language

AIS - Aeronautical Information Service

AMAN - Arrival Management

APP - Approximation Control

ATC - Air Traffic Control

ATCO - Air Traffic Control Officer

ATFM - Air Traffic Flow Management

ATM - Air Traffic Management

ATS - Air Traffic Service

CDA - Continuous Descent Arrival

CDO - Continuous Descent Operations

CDM - Collaborative Decision Making

CGNA - Air Navigation Management Center (*Centro de Gerenciamento da Navegação Aérea*)

CPVR - Repeated Flight Plans Center (*Central de Planos de Voo Repetitivos*)

CTA - Control Transfer Area

CTOT - Calculated Takeoff Time

DECEA - Airspace Control Department (*Departamento de Controle do Espaço Aéreo*)

DMAN - Departure Management

EOBT - Estimated Off-Block Time

ETA - Estimated Time of Arrival

ETOT - Estimated Takeoff Time

FAA - Federal Aviation Administration

FAB - Brazilian Air Force (*Força Aérea Brasileira*)

FCFS - First come, first served

FIFO - First in, first out

FMS - Flight Management System

FPL - Flight Plan

GHP - Ground Holding Problem

IAF - Initial Approach Fix

IATA - International Air Transport Association

ICAO - International Civil Aviation Organization

IFR - Instrumented Flight Rules

NoSQL - Not only SQL

PBN - Performance Based Navigation

PM - Point Merge

RDBMS - Relational Database Management System

RBS - Ration by Schedule

RNAV - Area Navigation

RPL - Repeated Flight Plan

SA - Simulated Annealing

SBT - Shared Business Trajectories

SESAR - Single European Sky ATM Research

TBO - Trajectory-Based Operations

TMA - Terminal Control Area

TOBT -Target Off-Block Time

TOD - Top of Descent

TP - Trajectory Predictor

TransLab - Laboratory for Computation Modeling for Air Transport (*Laboratório de Modelo Computacional para Transporte Aéreo*)

TTOT - Target Takeoff Time

TW - Time Window

TWR - Control Tower

VFR - Visual Flight Rules

VNAV - Vertical Navigation

List of Figures

List of Tables

List of Algorithms

Contents

Chapter 1

Introduction

The latest assessments of International Air Transport Association (IATA) identified that the demand for worldwide air transportation is expected to double up to 2037 [1]. Recent efforts from NextGen [2] and SESAR [3] programs are already addressing this scenario by identifying enhancement opportunities for safer and more efficient air traffic management and air space control procedures [4].

Such programs have as premise the fact that not only the technological evolution is needed, but the proceedings and algorithms used in the supporting systems must evolve as well. Furthermore, the stakeholders on ground or airborne should have common awareness about the aircraft trajectories by means of information sharing and management. System Wide Information Management (SWIM) infrastructure is part of an implementation effort in order to address this requirement [5]. The SWIM concept comprises standards, infrastructure, and governance to enable an architecture of interoperable services for qualified parties. This enables the management of air traffic related information and its exchange between the stakeholders [6] and strongly contributes for a collaborative and fair environment in which aircraft trajectories are directly managed [7].

An essential requirement for efficient Air Traffic Management (ATM) is the quality of available information, which must be up-to-date, highly accurate, and reliable. This allows the user adequate decision making at the right time [7], such as scenario forecast, resource allocation and support for trajectory management operations.

Trajectory-Based Operations (TBO) emerges as a novel technology that defines strategic trajectories for long-term conflict resolution combining safety and efficiency [8]. TBO systems are developed in order to support the air traffic controllers in the management of the air traffic flow during the execution of every flight. The implementation of supporting technologies for trajectory prediction is a high-priority task in this new paradigm since the control is based on aircraft trajectories instead of ATC clearance. Therefore, the aircraft's trajectory must be completely modeled and interpreted by the decision support

tools available to the controllers [9].

Different ground and airborne stakeholders must share a common view of the aircraft trajectories [10]. This task is made possible by TBO, which requires the computation of four-dimensional trajectories to implement the 4D navigation, which is a method that adds to the spatial constraints the time factor, thus favoring the accurate prediction of the aircraft's position at a given time [9]. This new paradigm requires the development of novel and efficient computational methods to calculate, store, present and process trajectories to enable the planning and execution of all flight operations supported by the Air Traffic Service (ATS) providers.

Aircraft Intent Description Language (AIDL) is a language designed for this purpose, where aircraft trajectories are mathematically modeled. The language alphabet in AIDL is built according to mathematical rules that define the possible combinations among flight instructions grouped according to their effects on the behavioral profile of aircraft. The final product of trajectory computation from a predictor using AIDL is a unique, reproducible trajectory that can be interpreted and shared by any agent [10].

The main purpose of this research is to deliver a framework for conflict detection and resolution of 4D trajectories. The framework is designed to be compliant with the SWIM paradigm and takes advantage of an AIDL trajectory prediction service. Conflict detection is performed by evaluation of sampled AIDL points conveniently stored in NoSQL databases and a local search algorithm is modeled for conflict resolution. In this modeling, each aircraft is regarded as agents who must have their departure times evaluated so that the departure queue is managed by updates in the flight plans such that the airspace resources and the operational costs are reallocated to accomodate a complete set of conflict-free 4D trajectories.

1.1 Motivation

NextGen and SESAR programs are looking for a shift in the Air Traffic Control/Management method moving for Trajectory-Based Operations (TBO) [2] [3]. The introduction of Performance Based Navigation procedures (PBN) is an attempt to optimize and enhance the usage of air space resources. Studies published by Boeing [11] predict an increase of 146% of the commercial fleet only in Latin America; 84% of them are single-aisle[1], most commonly used in domestic flights.

Considering the airspace usage forecast for the next years, the Brazilian Air Force already started an effort to implement the PBN in order to meet the safety and performance

[1] A single-aisle or narrow-body aircraft is arranged along two columns of seats separated by a single aisle.

needs of the national ATM program [12]. Following this demand, the proposed system adds collaborative functionality currently not featured by the systems used by Brazilian ATM organizations, such as X-4000 and SAGITARIO, which only integrate and present consolidated data to human controllers.

The ATM system is continuously evolving, with the expectation that global air traffic will increase three fold by 2025 [13]. Current systems and procedures need to evolve altogether with the air space demands, decreasing the air traffic controller workload whilst increasing air traffic capacity. Current methods for constrain aircraft are limited to clearances issued by the air traffic controller to the airborne crew, which must strictly obey the guidance instructions, regarding the aircraft performance limitations [14]. These clearances may have the format of required time of arrival (RTA) in order to reduce the along-track flight uncertainty. Although separation is ensured by the air traffic control (ATC) clearances, current research is being developed to optimize the results in the trajectory guidance procedures.

A 4D (four-dimensional) trajectory is the precise description of an aircraft's path in space and time. Waypoints are used to represent specific steps along the path, and are earth-referenced with a proper latitude and longitude. In 4D trajectory, the path contains altitude description for each waypoint and indications about the time at which the trajectory will be executed. Some waypoints in the 4D trajectory path may be associated with Controlled Time of Arrivals (CTA) or Required Time of Arrival (RTA).

Air traffic controllers (ATCOs) can count on systems that support the flow management in operations. Nevertheless, such operations in Brazil are performed without any computational support for decision making for treating conflicts and scheduling flights. This task is fully performed by the controller, who must rely on his expertise [15].

1.1.1 Problem identification and hypothesis definition

The decision making process performed by air traffic controllers is error-prone and there is not a mechanism that should assess the quality of the conflict detection and resolution concerning the performed trajectories. Furthermore, an underlying problem is the detection and resolution of conflicts among flights performing 4D trajectories, and how new methodologies can contribute to the increase of performance in the air traffic management.

The problem that is expected to be solved by this work is specifically the 4D trajectory prediction used to detect potential conflicts to be solved.

The hypothesis hereby presented is that big data management architectures, specifically using NoSQL databases, can be modeled to efficiently store trajectory data that can be input for conflict detection, and local search algorithms are suitable technologies for conflict resolution, which is a well-known high complexity problem.

1.1.2 Current efforts

The Brazilian Airspace Control Department (DECEA) is currently working on the implementation plan for the national ATM [16]. This implementation plan aims at establishing the strategic evolution of the performance-based National ATM System in order to comply with the national needs and ensure an harmonic integration to the International Civil Aviation Organization (ICAO) plannings and requirements.

The Brazilian Air Force recognize the national limitations concerning the national air traffic system [17] and already started an effort to implement improvements. PBN is expected to be accomplished by SIRIUS Program [12], but 4D Navigation concepts were not specified so far.

Translab, the Laboratory for Computation Modeling for Air Transport at the University of Brasília, has successfully started several works addressing air traffic management issues. More recently, Four-dimensional Trajectories and PBN have become promising subjects for research since DECEA started the national ATM plan. The scope of this work is specifically the development of an integrated framework that should support the controllers' operations for maintaining a safe, efficient air traffic transport system in Brazil and abroad. The current research can be presented as a major contribution to the National ATM Plan, and therefore is intended to have significant social impact.

1.2 Objectives

The main objective in this research is to develop a new computational solution for conflict detection and resolution for 4D trajectories, which should result in a more efficient management of trajectories for commercial aircraft under 4D navigation paradigm. The expected product of this proposal is a modeling that will be able to allocate 4D trajectories and promote the coordination between the air traffic controllers and the flight management systems on-board the aircraft.

Secondary objectives are enumerated below:

1. Develop algorithms for air traffic resource allocation compliant to the fairness concepts. More importantly, the algorithms need to effectively address the 4D navigation paradigm;
2. Implement a simulation environment in which aircraft should develop efficient flight profiles;
3. Evaluate the implemented algorithms in terms of effective allocation of air traffic resources;

4. Provide a SWIM-compliant architecture to manage the necessary data to be produced and used by the framework;
5. Elaborate algorithms using NoSQL databases to find trajectory conflicts and local search for exploring solution spaces;
6. Evaluate the developed technologies in order to identify advantages, limitations and performance.

1.3 Research methodology

The methodological concepts in which this work is developed consist in optimization algorithms. A brief description of the main aspects involved in the expected achievements is given below.

1.3.1 Research description

This work is developed according to the following research procedures:

- *Technical revision*: The technical revision comprises the survey of the main concepts inherent to the air traffic operations, airline business and ATC/ATM requirements. Study of manuals from aircraft manufacturers and reports from air traffic and airport authorities are also paramount to provide a mapping of the current technologies, resources and procedures deployed in the field.
- *State-of-the-art research*: This step comprises the study of methods for 4D trajectory prediction and conflict resolution. Some previous related works are studied in order to bring an overview concerning the already developed solutions for the proposed scenario and the techniques currently being used. This research step is important in the sense that some problems may not be fully addressed by previous approaches, and improvements may be eventually identified.
- *Definition of methodological approach*: The survey performed in the previous steps allowed the identification of important requirements, such as big data storage and processing, collaborative decision making and resource allocation. The definition of the proposed methodologies addressed these needs regarding the provision of up-to-date data and optimization of NP-Hard problems.
- *Requirements gathering*: in this stage the requirements are collected altogether with the users and stakeholders, that is, Brazilian Air Force representatives, aircraft

manufacturers and air traffic coordinators. The new resources needed in order to develop the solution are also enumerated.

- *Information gathering*: several meetings with the stakeholders took place. Information about the procedures, expectations, limitations and technical issues were collected in order to produce data and knowledge for modeling the proposed environment.
- *Architecture definition*: The prototype architecture is established once the research scope is well-defined. The implementation language is selected, the databases, classes and data structures are modeled and the datalink modules that will simulate the communication between the entities is described.
- *Implementation*: the prototype implementation should comply with the previous definitions and requirements. The studied methodologies will be applied in the databases and algorithms that will be evaluated in the treatment of the posed problems.
- *Simulation*: The scenarios selected for simulations comprise a comprehensive set of commercial flights and a set containing the routes connecting the busiest airports in the country. The test cases for the simulation are defined and the respective data sets to be used are compiled and analyzed. The efficiency of the proposed model is evaluated. Finally, the success and drawbacks discovered are exposed.

1.3.2 Decision making processes

In order to have an efficient decision making process, the agents involved in the modeling need to have access to complete and up-to-date information, that should be available always when needed. This is the core principle of Collaborative Decision Making (CDM), which has become a common goal for practically every air traffic provider in the world. Such decision making paradigm allows the information sharing among the entities and the optimization in the resource allocation. The primary consequence is the fair cost distribution among all the interested parties.

1.3.3 Big data management

NoSQL (Not Only SQL) databases constitute an approach developed to management of big data to be distributed among several nodes in a network [18]. Some characteristics of NoSQL databases are project simplicity, horizontal scalability and high performance and availability. However, NoSQL does not conform with ACID (Atomicity, Consistency, Isolation, Durability) properties.

There are some distinct storage structures for NoSQL databases, viz. Key-value, columns, graphs and documents. In the currently researched approach, a column-oriented database and a document-oriented database are developed, namely Cassandra (column-based) and MongoDB (document-based).

1.3.4 Local search algorithms

Some specific problems demand modeling environments that should not be discrete, nor finite. In such situations, where extensive search becomes cumbersome due to high algorithmic complexity or prohibitively large search spaces, local search algorithms emerge as an adequate approach for finding good solutions within acceptable time. For this reason, a local search algorithm is used in order to have accomplished the proposed goals of this work.

Simulated Annealing (SA) is an implementation that emulates the heating of a substance until it thaws, then this substance is gradually cooled down until its freezing point. Low energy states achieved in this cooling process can be considered as solutions, where random annealing is performed in order to disturb states that eventually get stuck in local optima. A low energy state is the goal to be achieved, which represents a low cost solution.

1.4 Brief review of previous works

Several works were already developed concerning 4D Navigation. Not only trajectory optimization is important, but also computational techniques that decrease algorithmic complexity, data storage or data traffic.

The Aircraft Intent Description Language (AIDL) was firstly proposed as a standardized formal method to express aircraft intent information allowing interoperability among heterogeneous decision support tools concerning trajectory prediction [9]. This is a direct benefit of AIDL, which allows supporting systems leading to flight profiles that are more efficient increasing the ATM capacity.

Trajectory-Based Operations (TBO) involve different stakeholders in the air traffic management (ATM) that wish to increase the predictability of the aircraft behavior by coordinating a decision-making process that takes the individual preferences into account. For this reason, continuous communication of the aircraft's intent is paramount for trajectory prediction [13]. This results in conflict-free trajectories. In order to have operational compatibility between ATC and aircraft, the same performance models should be used by these endpoints. A proper description of 4D trajectories is by the AIDL langague [19].

Accurate trajectory prediction is paramount for decision support tools [20]. Historical data and stochastic models can be used to effectively predict trajectories and deliver consistent representations of future scenarios. This task has substantial impact on ATM and airspace flow management, allowing air traffic authorities to efficiently plan and allocate air traffic resources.

Flight management systems on board should include new capabilities for TBO that support 4D trajectory execution and navigation profiles for required or controlled time of arrival (RTA/CTA) at one or more points defined in the air space, as well as situation awareness. The later capability clearly depends on an efficient data-link communication [21] [22].

One of the main ideas toward the future of 4D Navigation is the Point Merge system concept. It is a novel air space design in which extensive use of lateral guidance is enabled by the on-board flight management systems [23]. This approach designates virtual merge points near Terminal Manoeuvring Areas (TMAs) around airports where arriving routes converge. Traffic coordination in such points is essential for successful 4D operations.

The Top of Descent (TOD) point is the designation of the location with the highest altitude of an aircraft's trajectory prior the beginning of descent trajectory. In other words, the aircraft starts its landing path in the TOD point. An appropriate TOD calculation is very important to the efficient flight path, due to the fact that optimum descent profiles count on idle thrust in the descent phase and therefore the descent rate must take TOD into account [24].

Exploratory search in complex deconfliction is still a matter of concern [25]. Conflict detection for predicted trajectories can be performed by spatial data structures with the purpose to generate the state-space representation of the network and to perform conflict detection. Optimality in air traffic deconfliction is a highly combinatorial problem for which constraints and data pruning must be applied to narrow down the search space.

1.5 Document organization

Chapter 2 describes the main procedures and services concerning the air traffic management and the air traffic control, and presents a sensitive description of the 4D navigation concept.

Chapter 3 presents some state-of-art solutions already developed and tested concerning 4D operations, conflict detection and resolution, and efficient arrival coordination.

Chapter 4 is presented in order to show how the 4D Navigation concept is used in a trajectory prediction tool.

Chapter 5 introduces the theoretical aspects of the methodology for implementing the solution proposed. Local search algorithms for combinatorial search spaces are described with the techniques of Simulated Annealing.

Chapter 6 describes the modeling aspects of the prototype. The proposed methodologies are used as main techniques under which the algorithms are conceived. The decision making process in the proposed prototype is also detailed in this chapter.

Chapter 7 describes a test scenario for the simulation procedures. The implemented system is tested for conflict detection and the yielded results are shown. The main conclusions about its effectiveness are evaluated in this chapter.

Chapter 8 presents a secondary case study in this work, a technology developed to aid the CD&R implementation for arrivals. Findings show that the majority of conflicts happen during arrival phase, and therefore an Arrival Management (AMAN) system is implemented as a potential add-on to the main framework.

At last, Chapter 9 discusses the findings and brings some overview about the current status of this research. Further steps and future work are identified.

Chapter 2

Concepts and methods in ATFM and 4D Navigation

This chapter is intended to present the reader some major concepts about air traffic flow management and air traffic control prior further reading of this work. The main Air Traffic Services (ATS) provided are presented in Section 2.1, while ATM (Air Traffic Management) and ATFM (Air Traffic Flow Management) are described in Section 2.2.

Section 2.3 depicts all the steps involved throughout the aircraft movements, from takeoff to landing. Special attention is given to the landing phase, since this is the main operation covered by this work.

A Flight Plan (FPL) is the document that thoroughly describes the flight parameters, for instance the source and destination airports, the route to be followed etc. Section 2.4 is devoted to this document.

Section 2.5 completes the chapter with an explanation about what is the 4D Navigation concept and with a description of some historical background and technical aspects inherent to this relatively new *modus operandi* of air traffic transportation provision.

2.1 Air Traffic Services

Airspace congestion has become a problem to be addressed for every operator in the world, provided that airspace usage has experienced a massive growth since the last decades [26].

Because of this increase in the complexity of the worldwide airspace network, new technologies and services were developed so the controllers could diminish the number of incidents and mitigate error-prone, human-based procedures, which can bring disastrous consequences if failures occur. Several rules for flight execution emerged, as new instruments arise as additional aid to the operators on control centers, which act on the majority

of the flights performed nowadays. Flights operating under navigation aid instruments guidance are called *IFR flights*, where IFR stands for *Instrumented Flight Rules* [27].

Air Traffic Services (ATS) are intended to enable the integration between the air traffic controller and the aircraft [28]. ATS must provide technology and communication resources that allow the accomplishment of the objectives of the air transportation entities, like grant the safety separation among the aircraft, provide full information needed by the pilots and control centers and so on [29]. Brazil is a country in which air traffic services cover the full extension of its territory, including the oceanic area under Brazilian administration.

ATS include a vast range of services that should enable efficient operations in several areas, for commercial, military and cargo flights. Such services are divided in four categories [27]:

1. Flight information services;
2. Alert services;
3. Instrumented navigation aid;
4. Air Traffic Control services.

Air Traffic Control services are divided in Area Control, Approach Control and Aerodrome Control. ATS providers have the responsibility of taking management actions for grant the quality and safety of the air transportation environments. Such actions are intended to apply procedures that should prevent and/or correct failure states [28].

2.2 Air Traffic Management and Flow Management

Air Traffic Management (ATM) allows the aircraft to accomplish its predicted departure and arrival schedules, according to well-defined and published flight plan. Nevertheless, operational safety is a premise that should be observed in the first place. This task is performed through an adequate management of the air traffic flow subject to the Air Traffic Control (ATC) capabilities and to the limitations imposed by the ATS providers [27].

Air traffic control can be either ground control or airspace control. ATC providers must avoid as long as possible reliance on empirical information, since such providers have the responsibility of prevent air traffic incidents [28]. This premise motivates several research works throughout the world, in which enhancement opportunities are prospected.

The main task of ATM is then to establish structures, procedures and protocols for usage of airspace, based on the current and future air traffic demand. ATM must also identify the operational and technical needs, and define the structures for an efficient

airspace usage. This involves the correct definition of airways, climb, cruise and descent procedures and the design of the controlled areas, as well as the adequate actions and protocols for each of the flight phases.

Air Traffic Flow Management (ATFM) is one of the air traffic services, intended to control the flow of the airspace network. Its main goal is to deliver a flight according to the expected parameters defined in its flight plan, provided the departure and arrival schedule are accomplished. The critical phase of the flight occurs when it reaches a terminal area [26]. The enforcement of safety procedures during this phase often introduce unexpected delays and eventually expensive rerouting maneuvers.

As expected, this situation is a source of complications in the air transportation framework, provided that ATFM must allow the flights to safely perform their intended operations with the smaller delay as possible. Restrictive, constraining measures eventually might be applied to some flights so these safety constraints can hold, for example ground holding, speed control, vectoring or airborne holding. Such measures often cause undesired delays in order to reduce airspace congestion and grant safety, but the more expensive situation is the Airborne Holding Problem (AHP), which occurs mainly when the aircraft is prevented from landing upon arrival at an aerodrome until it receives further instruction from the control tower.

Delays, undesired changes of speed profiles and unpredicted trajectories are sources of cost increase for any flight, since such actions represent high financial impact concerning fuel consumption or time-related cost [30]. One of the difficulties in the application of constraining measures resides in determining the fair distribution of the associated costs, based on specific criteria to define the priority of the aircraft.

At this point, computational aid needs to be developed as additional support for ATS providers. Such technological resources are intended to help controllers in the task of safely guide the aircraft throughout their routes without inserting any unnecessary sources of cost increase. For evaluation purposes, any impact caused by changes in the original parameters of the flight may be a source of increased operational cost.

The main goal of ATFM service is the pursuit of an optimum air traffic flow, where the traffic demand overwhelms the available structure destined to the air traffic flow. The attempt of correctly apply control measures is an inherent role of the Air Traffic Flow Management service [31].

2.3 Phases of the Flight

A flight path is mainly structured in airways, which are aerial navigation routes beginning near the source aerodrome and ending at the destination aerodrome. Aerodromes are

specific areas in which aircraft moves are restricted to departure and landings.

Several operation phases compose the whole flight of an aircraft [32]. These phases begin at the flight planning and endure until the flight is completed, that is, until the landing of the aircraft and disembarkation of all passengers.

The reader must be acquainted to the main procedures involved in each flight phase in order to understand the difficulties and goals of this work, thus a brief and summarized description of the main flight phases and the acting operators is presented in this section.

- Pre-flight: involves the pre-tactical and tactical planning phases.
- Taxi out: the aircraft is authorized to *push-back*, that is, is allowed to get off the parking slot and take position in a holding point of the departure runway. In this phase the engines are started and the aircraft is guided by Tower controllers until its final position for taking off, where it needs to wait for departure clearance.
- Take off: Control Tower clears the aircraft for departure. From the moment when the aircraft leaves the ground, the Control Tower looses jurisdiction under the aircraft and the controlling task is transferred to the Approach Control (APP), which is responsible to guide the aircraft upwards into the terminal area (TMA).
- Climbing: the aircraft continues its climbing trajectory towards cruise altitude, according to its flight plan. The aircraft leaves the TMA and enters in the airway when the specified altitude is reached.
- Cruise (en route): the aircraft adjusts its speed and altitude, and it is controlled by the Area Control Center (ACC) from this moment on.
- Descent: aircraft starts to reduce its speed until it reaches the destination TMA. Should the TMA be congested, restraining measures shall be applied to the aircraft in order to delay its entrance in the TMA. Aircraf under this condition need to be queued up in a logical sequence, and in more undesirable (but not uncommon) cases, they need to fly in circles. This is specifically the Airborne Holding Problem (AHP), a very expensive situation that increases the overall cost for the affected airlines.
- Approach and landing: the aircraft starts to be once more controlled by the APP when it reaches the destination TMA. From this moment on, the aircraft adjusts its descent rate in order to reach the landing runway with appropriate speed for touch down. Continuous descent operations allow the aircraft to perform efficient descent profiles toward the runway.

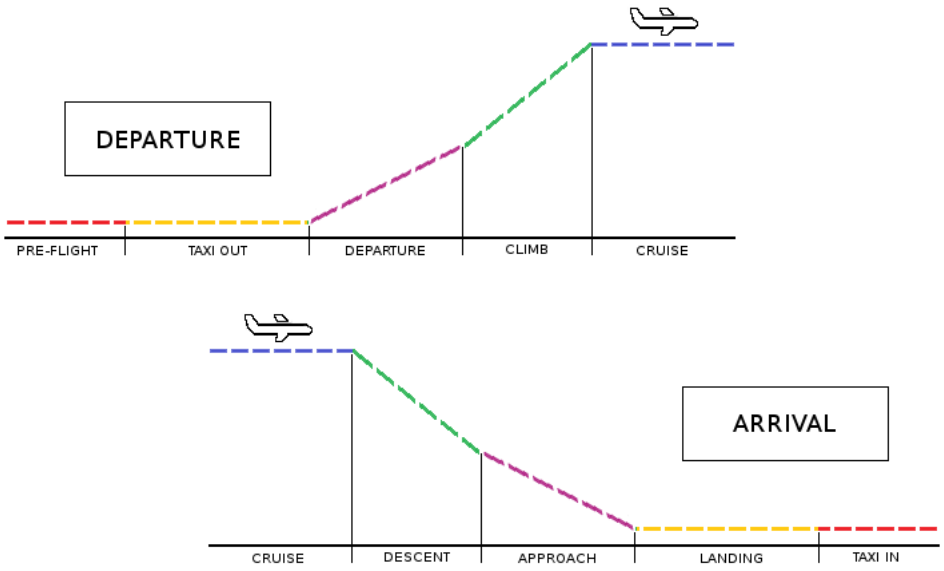

Figure 2.1: Operational phases involved in a flight.

- Taxi in: after successful touch down, the aircraft slows down and is maneuvered toward the arrival gate. The ground control is responsible to safely lead the aircraft all the way through the airport yard until it parks and the engines are turned off.

Figure 2.1 summarizes the described flight phases.

2.4 Flight Plan

A flight plan is a document conceived to indicate all the information inherent to the intended movement of the aircraft. This document enables the aircraft to be assigned an ATC slot, that is, this is the way in which air traffic resources are planned and allocated to that aircraft [33].

In most cases, the flight plan is mandatory [34]. The following situations require previous publishing of flight plans:

- Aircraft that should execute a flight under Instrumented Flight Rules (IFR);
- Aircraft that depart from aerodromes for which air traffic services are provided;
- Aircraft that is capable of communicating with ATS providers, after departing from sites for which air traffic services are not provided;
- Aircraft that should cross international borders.

A Repeated Flight Plan (RPL) is a category of flight plans that describes regular, commercial flights. Such flights usually operate very often and with identical basic features, repeatedly using the same ATS resources at the same time. RPL are used for flights

```
LOC: SBBR
--------------------------------------------------------------------------------
   VALIDO VALIDO DIAS OP  IDENT  TIPO   ADEP     VEL   FL  ROTA
   DESDE   ATE   STQQSSD   ANV     TURB     EOBT
--------------------------------------------------------------------------------

   210716 UFN    0234560 AZU4421 E190/M SBBR0000 N0444 350 DCT MUGIS UZ35
                                                          MEBLU DCT

-----------------------------------------------------
           DEST         OBSERVACOES
               EET
-----------------------------------------------------

           SBCF0055 EQPT/SDFGHIRWY PBN/B1D1O1S2
```

Figure 2.2: An example of a Repeated Flight Plan.

operating at least once a week, with a minimum of 10 flights, when there is intention for at least two months validity. RPL documents are very reliable in the sense that the provided data has a great degree of stability, so eventual changes that should be demanded can be easily applied. Naturally, RPL are typical for commercial aircraft that frequently execute the same route in strict schedules.

Flight plan proposals should be presented to the ATS providers with a minimum antecedence of 10 days from the intended beginning of the validity period [33]. In Brazil, this provider is the Repeated Flight Plan Center (CPVR) in the Air Navigation Management Center (CGNA) [34]. After processing, the ATS provider publishes this information to the Area Control Centers (ACC) involved in the intended movement.

The Brazilian Air Navigation Management Center grants free access to the RPL database in its webpage [1] for all flights crossing the Brazilian skies. An example of RPL can be seen in Figure 2.2.

This RPL is published for flight 4421 of Azul Linhas Aéreas. The string 0234560 means that this flight is executed from Tuesday to Saturday, with undefined validity. The intended route is also described (each token in *ROTA* parameter is a waypoint along the route). Further parameters complete the flight description:

- Origin aerodrome: SBBR (Brasília - DF);
- Destination aerodrome: SBCF (Belo Horizonte - MG);
- Departure time: 00h00m (midnight);
- Estimated flight duration (en route time): 0h55m;

[1] URL: `http://www.cgna.gov.br/?page_id=148`

- Aircraft model: Embraer 190;
- Average cruise speed: 444 knots, or 444 nautical miles per hour;
- Flight level (cruise altitude): FL 350 (35000 feet).

2.5 4D Navigation concept

In recent years, the NextGen [2] and SESAR [3] programs are looking for a shift in the Air Traffic Control/Management method moving for Trajectory-Based Operations (TBO). According to Nolan [35], a trajectory can be defined as the four-dimensional (4D) flight path of an aircraft through space (three-dimensions) and time (one-dimension).

2.5.1 What is 4D Navigation

Four-dimensional (4D) trajectory can be understood as a precise description of an aircraft's path in space and time, including position uncertainty in all four dimensions [2].

Trajectory Based Operations (TBO) aims to integrate an aircraft's navigation capability in space and time to improve efficiency and predictability in the Air Traffic Management (ATM) system [36]. This objective requires mainly four key components in order to hold:

- stored reference trajectories
- a continuously recomputed capture trajectory
- electronic situation displays
- a control system to follow the overall trajectory in space and time.

Aircraft must meet specified timing constraints at designated waypoints along their route. Thus the trajectories eventually need to be recomputed to selected waypoints on the reference trajectory so as to achieve the desired time of arrival [37].

This paradigm is subject to some research efforts worldwide. 4D flight management systems (4D FMS) are designed to guide aircraft along a pre-specified flight path such that the aircraft would arrive at the approach gate at a time specified by the ATC controller [37]. Such systems implement flight planning capabilities that include horizontal and vertical path definition for cost efficient operation [38].

2.5.2 Historical background

One of the first efforts of United States Government to implement 4D Navigation was performed by NASA in 1975 [37]. The research team implemented a digital integrated avionics system named STOLAND, which was installed on a CV-340 airplane. Although the 4D system was designed primarily for automatic operation, it was flight tested in a flight director mode, in which the pilot follows the flight director commands. The flight test showed that the pilot achieved the objectives of path tracking and time of arrival control with only moderate workload, even in the flight director mode. The system also permitted controlled delay of the time of arrival by path stretching, taking advantage of the continuously changing capture trajectory to predict the time of arrival, which was controlled to within less than 5 seconds.

While trajectory prediction systems were being implemented, NASA concluded that differences between the ATC-generated profiles and those generated by the airborne 4D FMS may introduce system problems, which represent a significant design challenge to ensure compatibility between future airborne and ground-based systems. For this reason, in 1991, Williams and Green [38] conducted an experiment to explore integration of a 4D-equipped aircraft into a 4D ATC system. In further experiments, the research team linked the NASA Langley Transport Systems Research Vehicle (TSRV) cockpit simulator in real time to the NASA Ames Descent Advisor (DA) ATC simulation. Three active airline pilots were set as test subjects at Langley and six active air traffic controllers and one research controller as test subjects at Ames participated for approximately 30 hours of simulation.

Candidate procedures for handling 4D-equipped aircraft were devised and traffic scenarios established which required time delays absorbed through speed control alone or in combination with path stretching. Dissimilarities in 4D speed strategies between airborne and ATC-generated trajectories were tested in these scenarios, based on the Denver Air Route Traffic Control Center (ARTCC) northeast arrival sectors. After the simulations, the arrival accuracy at the metering fix was indicated by a time error with standard deviation of 2.9 seconds and negligible mean. Further clarity in the time clearance procedures and integration of the time guidance into the airline cockpit were cited as future needs. In addition, the amount and nature of the information transmitted during a time clearance were found to be a drawback when using the voice radio communication channel, indicating that a digital data link for transmission of time clearances and restrictions are required for successful and efficient system operation.

In 2003, the US Government signed a law establishing the Joint Planning and Development Office (JPDO) to carry out the mission to design and deploy an air transportation system to meet the nation's needs in 2025 [2]. The concerns of this endeavour address air

space capacity, safety and efficiency, among other national issues.

In 2005, the JPDO developed a high-level vision to communicate the key operating principles and characteristics of the Next Generation Air Transportation System (NextGen). The goal of NextGen is to significantly increase the safety, security, capacity, efficiency, and environmental compatibility of air transportation operations, improving the overall economic well-being of United States [2].

In the European front of research, the Single European Sky ATM Research (SESAR) by EUROCONTROL defined performance objectives in order to respond to the predicted increase of the European airspace usage in the next years [3]. It is expected an extension of the arrival management horizon and the introduction of free routing. This will bring more flexibility to airspace operations in an increasingly complex environment.

The Mission Trajectory concept has been created in 2003 by European Air Traffic Management (EATM) to respond these demands and to line up the next generation ATM objectives with some military needs not covered by the Business Trajectory. Information sharing about trajectories is a core concept in this new paradigm in every flight phase, from planning to execution. This will allow situation awareness (SA) between civil and military actors, which will be beneficial in terms of flexibility, predictability and safety for all airspace users.

In 2008, Konyak *et al.* [13] applied the Aircraft Intent Description Language (AIDL) in the experiment performed by Williams and Green [38] as a demonstration of the communication of aircraft intent and its effect in ensuring conflict-free trajectories. The AIDL is a formal language developed by Boeing Research & Technology that describes and exchange information related to aircraft trajectories in an unequivocal way. The language is characterized by an alphabet and a grammar formed by a set of instructions used to model the basic commands, guidance modes or control strategies at the FMS to guide the operation of the aircraft. The whole aircraft motion profile is accurately described by AIDL and real-time simulators or trajectory generators are able to define and compute the aircraft flight unambiguously.

The premiere flight operated under Initial-4D was performed by an Airbus-320 departing from Tolouse to Copenhagen in February 2012 [2]. The purpose of this test flight was to evaluate synchronization between airborne and ground systems under the 4D paradigm. Aircraft and Air Traffic Control (ATC) agree far in advance a target time to the aircraft to reach a merging point, provided that the aircraft is allowed to fly its optimum descent profile up to that point.

[2]EUROCONTROL (2015). Retrieved from `http://publish.eurocontrol.int/articles/towards-i4d-initial-four-dimensional-operations`, accessed in 07/sep/2015

A premise is that the communication between airborne and ground segments is not performed by voice, but by data link message exchanges [21]. While in cruise phase the ground control sends a proposed two-dimentional trajectory up to the runway, including a merging point located at the destination terminal area for which a time constraint will be defined.

This route is loaded into the FMS, which receives wind and temperature updates and computes target times for waypoints along the proposed trajectory. Particularly, the FMS is supposed to calculate a time window in which the aircraft will reach the merging point whilst maintaining the optimum descent profile. This time window is sent to ground control by data link. ATC is then able to select a target time within the time window that is preferable concerning traffic flow optimization, and this time is sent back to the aircraft as a target that the aircraft has to comply with.

The same procedure was performed again in a sequential flight from Copenhagen to Stockholm, then from Stockholm back to Copenhagen and Toulouse. At the ending of the experiment, findings show that fuel consumption was decreased and delays were absorbed in enroute airspace before entering in terminal areas. In addition, data link communication was fairly efficient and comfortable to the flight crew and ground control, since trajectory optimization was responsibility of the FMS. This strategy was also beneficial due to the fact that there was no need to ATC to send any vectoring instructions to the aircraft.

All key concept elements were tested on each flight leg. Avionics interoperability was shown through the use of two independently developed FMS prototypes. Time window and airborne trajectory calculation by the supporting application to the downlink of FMS were reliable and the integration of the airborne trajectory in the ATM automation systems improved the trajectory prediction and provided achievable Controlled Times of Arrival (CTAs).

2.5.3 Continuous Descent Approach (CDA)

The Continuous Descent Approach (CDA) is a procedure established to allow aircraft to obtain an optimum descent profile while arriving at an airport. It is an aircraft operating technique aided by appropriate airspace and procedure design, and appropriate ATC clearances [39]. It is a fuel-efficient descent operation which starts at cruise level exactly at the point where usage of idle or near idle thrust is allowed during the entire descent operation [40].

Descent operations are held from the Top of Descent (TOD) to touch down on the airport yard. The optimum vertical profile should work in a continuously descending path, with a minimum of level flight segments through elimination of level-offs [41]. Arriving aircraft should descend from an optimal position with minimum thrust, avoiding level

flight to the extent permitted by the safe operation of the aircraft and yet in compliance with published procedures and ATC instructions.

Main goals in CDA operation are the execution of a flight profile optimized to the operating capability of the aircraft, reduction of fuel burn and emissions during descent and maximization of operational efficiency while still addressing local airspace requirements and constraints [39]. In other words, this means that aircraft should operate under low engine thrust settings and low drag configuration in order to save fuel and reduce the operational costs inherent to the flight.

The type of aircraft, their actual weight and wind conditions are essential parameters in the optimum vertical path angle calculation [24]. Dynamic parameters, such aircraft speed and altitude also account to determine the TOD point, which should be at a given distance from the airport, usually 100 to 150 nautical miles from the destination airport [42].

Airspace design should consider CDA route profiles for every aircraft type and flight conditions, as well as CDA altitude constraints compatible with speed constraints, and altitude constraints defined and expressed under intervals instead of fixed prescribed altitude. The provision of timely accurate distance-to-touchdown information and updated distance-to-go (DTG) at regular time intervals as appropriate allow the operation to be 4D-compliant during the descent phase.

2.5.4 Air Traffic Coordination in 4D

Aircraft must meet specified timing constraints at designated waypoints along their route. Thus their trajectory eventually needs to be recomputed to selected waypoints on the reference trajectory so as to achieve the desired time of arrival [37].

Conventional ATC fine tunes the landing sequence and integrate traffic flows from different fixed points in order to avoid unnecessary gaps at the runway threshold and to improve flexibility. The drawback is the rise of high traffic load conditions, high workload both for flight crews and controllers, difficulty to optimize vertical profiles and difficulty to contain dispersion of trajectories. ATC constraints and *ad hoc* vectoring limit the ability of effective use of on board avionics due to lack of adequate traffic planning [43].

Airspace congestion upon TMA becomes a problem when several aircraft are scheduled to arrive at close timing. Hence ATC should take actions in order to detect and solve conflicts that arise when two or more aircraft paths intersect during their flights. Speed and altitude constraints are then applied to each flight involved in the conflict, thus vertical and horizontal separation are granted.

This configuration does prevent safety incidents, but the straightforward side effect is the increase of cost operation for some aircraft, since they are prevented from performing

their optimum descent profile [44]. The solution for this situation is expected to be found in a way in which total cost is minimized, although the airspace remains safe and efficiently coordinated.

One can understand the arrival coordination task as a scenario in which aircraft want to arrive at a time within an optimum time window, provided that safe operations are ensured by ATC. It means that ATC must address and eliminate every en-route conflict among aircraft through correctly sequencing the aircraft that will reach a specific merging point, i.e., a confluence point which collects incoming aircraft from several airways.

Airlines want to enhance efficiency in their operations by diminishing delays and enforcing efficient flight profiles for their aircraft. Obviously, sometimes they are prevented from accomplishing this objective, because ATC needs to issue measures for the sake of safety, like inserting separation times, re-routing or airborne holding.

Therefore, arrival coordination becomes even more complicated when individual costs are also a matter of concern. Efficient operations then should mean that arrival safety will be effectively ensured, although with a minimum, fair cost propagation among the aircraft involved.

Chapter 3

Previous work on 4D Navigation procedures

Several works on Air Traffic Flow Management (ATFM) were already developed, ranging from architecture definitions to trajectory management. This chapter presents some of the most important works in the field.

Section 3.1 presents some advanced techniques for data management in 4D algorithms. Approaches that diminish data transfer volume or that uses efficient storage methods are presented. Also, a data mining technique is described in order to collect statistics about previous flights for future predictions.

Section 3.2 introduces the very important concept of Point Merge in trajectories of arriving aircraft and further techniques for time-based navigation for discrete and continuous 4D guidance.

Section 3.3 shows an approach for calculating efficient 4D departures and arrivals, and technologies to optimize vertical trajectories are also described.

Section 3.4 introduces some techniques applied for optimization of 4D trajectories. Finally, Section 3.5 presents some advancements to be delivered by the current research.

3.1 Data management in 4D

Despite improved hardware and software, computational requirements for data storage and analysis are steeply increasing.

Wandelt and Sun [45] realized that data compression is a key technology to address this situation. Existing compression techniques could be simply extended from 3D to 4D storage, but the precision of decompressed trajectories still depends on user-defined error bounds. It means that one could adjust the algorithms to a higher compression ratio, but the decompressed information would be less precise. On the other hand, loss-

less compression in such manner is not adequate for the purpose of data compression effectiveness.

If $p = (x, y, z, t)$ is a 4D-trajectory point (x, y and z are spacial dimensions and t is time), $tr = [p_1, ..., p_m]$ is the complete 4D-trajectory, which is a sequence of four-dimensional points. The first compression technique analyses a set of samples and computes a prediction for the most likely successor coordinate. By treating trajectories as strings, it is possible to verify edit distances in a pair (tr_1, tr_2) by replacing, removing or adding points in t_1 so it turns equal to t_2. A 3D-trajectory projection tr^{3D} of a 4D-trajectory may be understood as a concatenation of (x, y, z) and tr_2^{3D} if tr results from concatenation of (x, y, z, t) and tr_2 ($(x, y, z, t) \circ tr_2$). The redundancy existent in a set of trajectories is used to decrease the storage cost.

The first compression technique is based on statistical compression. The hypothesis is that several aircraft with the same origin and destination share very similar routes, differing mainly in the *time* dimension, where a reference trajectory database is transformed into a labeled graph which provides statistics about route segments frequency and the average time spent on these segments. The referential compression technique compresses a 4D-trajectory as a collection of sub-trajectory vectors into a reference trajectory. The compression can be done by encoding the trajectory as a concatenation of sub-trajectories from a given reference trajectory. Each match is encoded as a quintuple named Referencial Match Entry (RME), composed of the start position of a match, the length of a match, the base time, a list of temporal differences, and the first point following the match.

A database is formed by the combination of the previous techniques, provided that the first insertion is always an uncompressed trajectory. Further trajectories are evaluated before insertion and the least space-consuming compression strategy is used, which requires adequate order of traversal. This compression technique was named 4DTC (4D Trajectory Compression). The input for simulations were files containing almost 9.5 million flights. The default compression method is 7zip, which presented stationary compression ration of 8.3:1, while 4DTC compression ratio increases with the number of flights, ranging from 20:1 up to 77.3:1. Uncompressed storage of 60.3 GB could be compressed to 780 MB.

In order to extract the most reliable information record of the flight course to analyze the pattern of the trajectories and to further improve the trajectory prediction accuracy, Song *et al.* [46] propose data mining algorithms that can be used to process radar data to abstract a typical trajectory library. Typical trajectories are used as the intent information to update the flight mode transition probability matrix and to propagate the nominal trajectory instead of the flight plan paths.

Clustering algorithms can be used to group trajectories with the same space-time characteristics. A typical trajectory can then be defined as the kernel of one group of clustered

trajectories. It represents the common pattern of the cluster. Among several other information, hidden information within clustered trajectories can be used to derive inferences about the future flight intent. The process for constructing the typical trajectory library is implemented in three steps: abstract trajectory data from radar; data preprocessing; and cluster analysis and typical trajectory abstraction. Detailed algorithms for these steps are given in [46].

Flight plan waypoints are replaced with typical trajectory points, which are used to calculate the flight intent. The flight mode transition probability is updated with the typical trajectory, then the probability density function is calculated to each flight mode and flight state at time $t + 1$. This process is repeated to propagate the state probability density function into the future.

By analyzing historical radar data from a database totalizing $4,2 \times 10^6$ point records, 119 clusters were obtained and the corresponding 119 typical trajectories were stored as the typical trajectory library. Predictions of 10 minutes are calculated at the initiation time with radar sampling each 15 seconds. The prediction performance of every single trajectory and the average prediction performance are compared to the actual flight performance. Results show that average reduction in prediction errors were reduced $7,7\%$. The best case reduced prediction errors up to $10,2\%$.

3.2 Point Merge

Airborne Spacing - Flight deck Interval Management (ASPA-FIM) is a concept being developed by the Federal Aviation Administration (FAA). It is a procedure for merging and sequencing aircraft by constructing a queue where spacing is enforced by means of ADS-B position reports from the leading aircraft. Such strategy is a miles-in-trail implementation, and is characterized by the alignment of all flights into a single linear flow toward an airport [47].

Similarly, EUROCONTROL developed a technique to address airspace congestion called Point Merge (PM). It aims at optimizing the use of available airspace in terms of capacity, environmental aspects, and track distance flown. The main objectives are efficient landing sequencing, increase of landing throughput, minimization of ground and airborne delays and traffic management with varying capability with minimum air/ground communication.

Zúñiga *et al.*, [43] and Ruiz *et al.* [48] presented a discrete event model for Conflict Detection and Conflict Resolution algorithm which mainly comprises the arrival phase. The main motivation is the overflow in the airspace near large airports. Despite many earlier successful attempts to alleviate airspace congestion, non-efficient procedures such

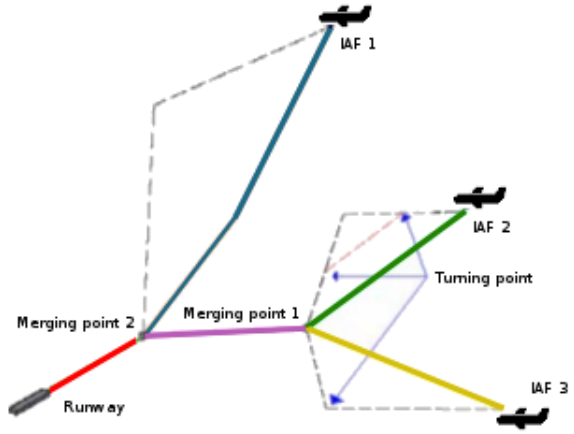

Figure 3.1: Point merge implementation (Zúñiga *et al., 2010*).

as the use of holding trajectories are still used to avoid conflicts. The proposed model considers different alternative predefined turning points for each flight evaluating path shortening/path stretching of all trajectories upwards the merging point in a TMA.

They propose the usage of Colored Petri Nets (CPN) as to establish a model for path stretching/shortening technique in order to compute the exact turning points according to the type of aircraft. The predictive model computes and detects Medium Term Conflicts each time there is a new TMA arrival at any entry point by evaluating the passing time at the merging points and checking if the time between two consecutive aircraft is smaller than the minimum separation standard (MSS) previously parameterized according to the leading and following aircraft weights. Control actions for conflict resolution are divided in evaluation of an alternative trajectory (change of vector), aircraft speed up and aircraft slow down.

The objective of the change of vector is to stretch the distance to be flown from the TMA to the merging point so an aircraft can arrive at the expected time maintaining its speed profile. Therefore, the delay required is absorbed by the alternative trajectory proposed. For simulation purposes, the direction change is standardized in 45 degrees. Further actions do not modify the trajectory to be flown but the speed of aircraft solely. By doing so, the time to be flown within the TMA is reduced or increased in order to adjust the MSS between the aircraft in case of early conflict detection. This configuration can be viewed in Figure 3.1.

The CPN is modeled according to the following elements:

- Places: queues and logical conditions. Graphically represented by circles;
- Transitions: events of the system. Graphically represented by rectangles;
- Input Arc Expressions and Guards: type of tokens that fire transitions;

- Output Arc Expressions: indicate the system state change;
- Color Sets: types, operations and functions that can be used;
- State Vector: the smallest information needed to predict the events that can appear. The state vector represents the number of tokens in each place, and the colors of each token.

The CPN is designed in a manner such that the color sets allow the specification of entity attributes. The output arc expressions allow specification of which actions should be coded in the event routines associated with each event (transition). The input arc expressions allow specifying the event preconditions. The state vector will allow understanding why an event can appear and consequently to introduce or remove preconditions in the model, or change some variable or attribute values in the event routines to disable active events.

The discrete event approach has been specified using seven colors, five places and nine transitions. Three different control actions attached to each merging point could be fired. These events are represented by different transitions in the CPN model. Event *Change trajectory* is fired when a conflict between two aircraft is detected in a merging point. Event *Change trajectory + decrease speed* is fired when a conflict is detected but it cannot be solved only by a changing vector procedure. And event *Speed up* is fired when the separation between the aircraft is greater than the MSS.

The test scenario was selected from Gran Canaria Airport, where there are three different approaching routes from Europe and two different merging points. A synthetic traffic workload of 35 arrival aircraft has been designed. The arrival traffic sequence is assumed to have been determined upstream in the extended TMA by an arrival manager (AMAN). Sequencing is implicitly defined in the traffic preparation input file.

Each entry point has associated a fixed re-route computed by a turn of 45 degrees from the arrival route at the Initial Approach Fix (IAF). The re-routing methodologies differ in terms of the use of path shortening or path stretching technique.

The solution is obtained using the reachability tree in CPN. The proposed model has been tested using three different IAFs and two merging points, providing good results for a traffic peak. The causal CPN model could be extended to a variable number of IAF and merging point configurations, but scalability problems appear when inserting multiple point merge systems.

3.2.1 Time-based CDO and Mixed 4D

De Prins *et al.* [44] conducted a research on time-based navigation defined in the form of discrete and continuous 4D guidance. Discrete 4D guidance leads the aircraft to a target

time at a single waypoint with a desired target accuracy, whilst Continuous 4D guidance confines the aircraft within a desired temporal accuracy along the contracted trajectory, not only at the target waypoint.

Researchers implemented a guidance reference calculation in the FMS (which is henceforth called RFMS) in order to allow the prediction of a descent trajectory using the provided atmospheric conditions. Aircraft performance and desired cruise and descent speed profiles are used in this prediction, which should comply with regulatory altitude and speed constraints published for the waypoints along the planned route. This makes possible the determination of a Top of Descent (TOD) location.

A guidance around the reference is also implemented in the RFMS, in which the required time of arrival (RTA) algorithm iterates the cost index that yields a trajectory that complies with the target time at an indicated waypoint. This cost index is updated when the aircraft approaches the target waypoint, considering the time error exceeds a dynamic threshold, and a new vertical descent profile is then applied.

The first ground automation model is a TMA-based Scheduler for Mixed Avionics in charge of constructing a conflict free arrival sequence at the runway and at predefined metering fixes. The traffic volume is balanced over the available runways and target arrival times at the meter fix and at a specified runway are scheduled to each landing aircraft. This schedule routine is limited by the fact that the algorithm is only able to delay flights.

The single-stage arrival management process optimizes the traffic flow to the TRACON metering fix according to Figure 3.2. Additionally, fairness [49] is ensured by the principle that speed reduction and path stretching are constrained to a specific maximum delay for each flight. Periodic scheduling rounds are performed in order to build the arrival schedule, provided that each round updates the estimated time of arrival (ETA) of the flights located within the prediction horizon at all candidate runways and the metering fixes. Flights positioned between the influence and freeze horizon are taken into account to create a proposed arrival sequence at runway and metering fix, and flights that crossed the freeze horizon remain unchanged in the sequence.

Continuous Descent Operation (CDO) from top-of-descent is said to be successful if ATC does not break the Optimal Profile Descent (OPD) with interventions that change the descent profile. AMEBA Trajectory Computation Infrastructure is a three degree-of-freedom point-mass kinetic trajectory computation toolbox developed by Boeing Research & Technology Europe which is capable of executing trajectory calculations based on the provided sequence of aircraft intent operations [44]. The calculated aircraft states include all aerodynamic forces, thrust, fuel consumption and aircraft configuration positions. This

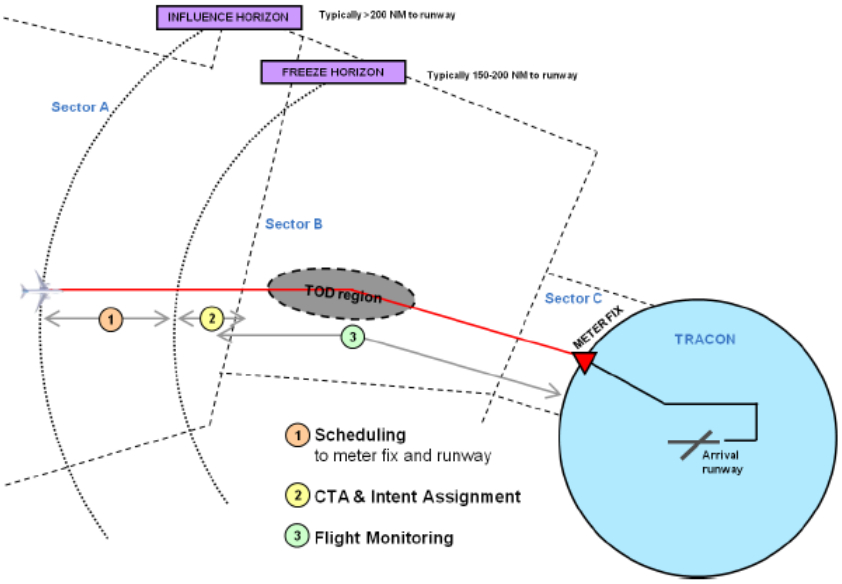

Figure 3.2: Single-stage strategic arrival management (De Prins *et al.*, 2010).

toolbox was used along with BADA, from where aircraft performance models were collected for simulation with Boeing aircraft solely.

Discrete and continuous 4D FMS guidance methods are combined with single-stage arrival management in the experiments to quantify the operational performance of the proposed model. The experiment aims to evaluate the relationship between maximizing ODP success and airport arrival throughput, as well as the effect on fuel consumption. The experiments reproduced the traffic demand faced into Atlanta International Airport (KATL) for February 13th 2009, in which the traffic scheme managed 300 flights.

Runway spacing buffers and atmospheric conditions are independent variables and the performance of the 4D FMS guidance methods is evaluated in terms of ODP success rate, arrival throughput, fuel consumption, and temporal and vertical trajectory confinement, defined as the time deviation at multiple points along the flown trajectory with respect to a reference 4D trajectory and a reference vertical path, respectively.

Results indicate that the proposed single-stage arrival management concept is able to perform OPD flights from TOD to runway with a success rate greater than 75%, i.e., without ATC interventions in vectoring or controlled speed changes. Onboard 4D guidance enhances the success rate up to 8% to the runway and 3% to the meter fix for the evaluated scenario.

Simulations were also forced to increase the rate of intact ODPs up to 90%, but fuel consumption increased up to 8% for the 4D guidance methods and even 20% for the open-loop VNAV PATH guidance, with the additional side effect of arrival throughput reduction up to 8% for the 4D guidance methods and 15% for the open-loop VNAV PATH

guidance.

3.3 Departure and Arrival procedures

Arrival procedures take place when the aircraft approach to their destination aerodrome. This stage comprises the final part of the flight trajectory and is critical for a successful and safe flight.

A successful arrival depends on efficient procedures and predictability accuracy of trajectories and time-based operations. Some previous work have been developed in this field in order to enhance safety and operational efficiency. This section summarizes the state of art about the study of arrival management.

Idle-thrust descent is assumed by on-board FMS so the location of the top of descent can be calculated. However, air traffic controllers still need to level flight segments in order to estimate the relative speeds of two incoming aircraft. Controllers can have situation awareness and minimize conflicts if ground automation arrival management is able to accurately predict three-dimensional descent trajectories.

Arrival time at the meter fix and vertical profile predictions are paramount to ensure vertical separation from aircraft at different altitudes. In this sense, ATC may be prevented from interrupting FMS-controlled efficient descents.

3.3.1 Top of descent

Stell [24] investigated factors that affect Top-of-Descent (TOD) location in idle-thrust descents and randomness in these TOD locations. In his research, the procedures analyzed the horizontal flight path, cruise and meter fix altitudes, while extracted the TOD location from recorded radar data. Additional data concerning the flight, e.g. aircraft weight and wind information were manually provided by the pilots. Multiple regression analysis of the pilot-provided information was performed for Airbus A319/320 and Boeing B757 aircraft.

Top of descent location was modeled as a function of aircraft type, cruise altitude, cruise Mach number descent calibrated air speed (CAS) meter fix altitude and speed constraints, aircraft weight and wind. One of the objectives in this study was to determine the shape of this function, from where the relevance of these predictive factors can be evaluated, henceforth the noise can be correctly estimated in the TOD prediction.

From Equation 3.1, the variable S_{TOD} represents the path distance of TOD location along the estimated horizontal trajectory relative to the meter fix. This modeling makes the dependent variable and the independent variables scalar, whilst the independent variables (exceptions to aircraft type and wind) are also scalar. S_{TOD} was desired to be a continuous scalar as a function of scalar variables based on empirical data, so it was

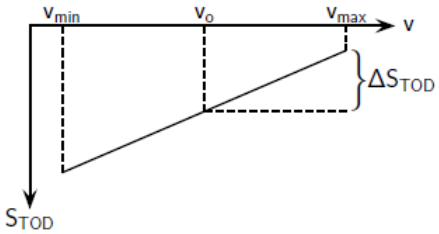

Figure 3.3: Error in calculation of path distance of TOD (Stell, 2010).

needed to transform wind to a scalar variable and analyze aircraft types separately. A Taylor polynomial approximation around a point in the range of the observed values of the predictive factors yielded the following equation to the model, where $\beta_0, ..., \beta_n$ are coefficients of regression model and the $x_i \in X$ were chosen from the predictive factors:

$$S_{TOD} \approx \beta_0 + \sum_i \beta_i x_i \tag{3.1}$$

When described in matrix form, multiple regression uses the least square solution and the model Z for the observed values of S_{TOD} is:

$$Z = X\beta + \epsilon \rightarrow Z = Xb + r \tag{3.2}$$

The goal is now to determine a linear approximation of S_{TOD} as a function of x. In order to evaluate the relative importance of CAS and cruise altitude coefficients in predicting S_{TOD}, Stell considered the error in S_{TOD} using a nominal value for descent speed from which a curve can be derived, as shown in Figure 3.3. The slope of this curve is the regression coefficient b_v of the descent speed.

The same method can be applied in a straightforward manner to all other predictive factors and the final error in S_{TOD} calculation is then

$$(\Delta S_{TOD})_i = \frac{1}{2}|b_i|(x_{i,max} - x_{i,min}) \tag{3.3}$$

The nominal values for the factors can be used to find new lower bounds to TOD location prediction error. Main conclusions are that a predictor needs to use accurate values for the predictive factors and by this method the excessive noise can be neglected, since the regression residuals have absolute value less than 5 nm.

3.3.2 Vertical Trajectory Optimization

In earlier research, CDA has been shown to reduce operating costs. Continuous descent from the TOD with engines at idle power and elimination of flight levels at low altitudes

are proven to reduce fuel burn. Additionally, flight time is also reduced due to the higher airspeeds closer to the runway.

One limitation to the effective implementation of CDA is the fact that air traffic controllers still have inaccurate predictions of the aircraft trajectories. FMS is able to calculate a vertical flight profile that satisfies preprogrammed constraints, which allow the aircraft to fly along the RNAV path. This enables the ATC to predict the future position of the aircraft, thus improving the air traffic flow management capability.

Simplified energy state equations can be used to describe optimal trajectories in terms of direct operating cost. In this sense, the energy parameter replaces the time as the independent variable in aircraft dynamics equations.

Park and Clarke [50] conducted a research in which several optimal trajectories for a CDA are presented that maximize the benefits in terms of operating cost such as fuel consumption or flight time. Optimal trajectories are obtained by formulation of multiple phase optimal control problems with a fixed range. These problems included consideration of both operating conditions and speed constraints. The optimal trajectory is divided in cruise and descent, and TOD and the optimal descent path are obtained simultaneously. Their model also allows suboptimal trajectories for an RNAV CDA based on the optimal trajectory results.

A point mass model is used and flight path angle dynamics are ignored as to balance the trade-off between model accuracy and computational burden. Two right-hand coordinate frames are used to derive the aircraft's equations of motion. The origin of the North-East-Down (NED) is the runway threshold with the North-East plane tangent to the Earth's surface, provided that the X axis points to the North, Y axis points to the East and Z axis points downward. The second frame is the aircraft center of gravity, where X_w axis points in the true airspeed direction. This is the Relative wind frame.

The first trajectory segment comprises the space between the fixed range and the TOD point. Hence this segment represents the cruise phase, in which the aircraft flies at cruise altitude and constant speed. The second segment starts at the TOD point to a final point or termination of the CDA procedure. This means that the aircraft performs a continuous descent with idle power, with mild trajectory variations due to deceleration for flap extension.

Two performance indexes are defined as to represent the concerns with fuel burn and flight time. They are:

- Flight time performance index;
- Fuel consumption performance index.

Both of them are divided in two parts in order to address the performance in both trajectory segments. The maximum range, the fuel flow rate at a given cruise speed during the cruise phase and idle power fuel flow rate at altitude during CDA compose the equations' indexes. Operating conditions such as flap speed schedule, landing gear extension, regulated speed restrictions, and bounded control input were also considered in the constraints formulation.

In the simulations, Park & Clarke used two aircraft types (B737 and B767) with predetermined flap speed schedules. BADA is used in order to provide aircraft performance data to the analysis. The aircraft mass chosen was 52000 kg for the B737 and 158800 kg for the B767. Cruise altitude and speed were respectively FL360 and 260 kt, with maximum speed allowed of 350 kt.

The aircraft flies the Instrumental Landing System (the ILS approach) after reaching the final approach fix. In this phase aircraft maintain the 3 degree flight path angle. Therefore, the final point of the CDA is the final approach fix instead of the runway threshold. The initial along track distances are set as 120 nm for the B737 and 130 nm for the B767. Initial and final point conditions are summarized in Table 3.1.

Table 3.1: Initial and Final point conditions (Park and Clarke, 2012).

Type	**Initial condition**			**Final condition**		
	H (ft)	CAS (kt)	Dist. (nm)	H (ft)	IAS (kt)	Dist. (nm)
B737-500	36000	260	-120	3000	180	-8
B767-400	36000	260	-130	3000	180	-8

A reference VNAV CDA trajectory generated by the FMS descent algorithms was used for comparison purposes. This vertical trajectory was used for a flight test conducted at Louisville International Airport in September 2004. The findings show that in the minimum time profile, the aircraft flies with maximum performance to reduce flight time. In the minimum fuel trajectory, the aircraft flies with idle power as long as possible, which implies that the aircraft TOD is earlier than the first case.

Comparing the graphics with and without wind conditions, the TOD curve variation with wind for the minimum fuel trajectory is steeper than that for the minimum time trajectory, which implies that the minimum fuel optimal profile is more sensitive to wind effects than the minimum time optimal profile. Summarized results for B737 and B767 are shown in Table 3.2.

Table 3.2: Numeric comparison for B737 and B767 trajectories (Park and Clarke, 2012).

Type	**B737-500**			**B767-400**		
	H (ft)	CAS (kt)	Dist. (nm)	H (ft)	IAS (kt)	Dist. (nm)
Performance index	TOD (nm)	Fuel burn (kg)	Flight time (s)	TOD (nm)	Fuel burn (kg)	Flight time (s)
Minimum time	-91.93	290.9	1038.7	-106.84	503.9	1134.6
RNAV (min. time)	-91.93	290.9	1038.7	-106.84	503.9	1134.6
Minimum fuel	-109.65	235.03	1219.8	-121.60	401.9	1299.00
RNAV (min. fuel)	-109.02	236.59	1197.5	-121.89	402.6	1315.4

3.4 Multiobjective 4D optimization

Some work on strategic de-confliction has been developed. Ruiz [25] introduces a new approach based on causal modeling in order to evaluate the Shared Business Trajectories against Reference Business Trajectories of aircraft. The term Strategic De-confliction is often used to define actions taken when the SBT takeoff time is known with sufficient accuracy, i.e., tactical decisions that demand immediate responses are not part of the scope.

It is known that the problem of Conflict resolution with a global optimization scope is not tractable in polynomial time, often demanding combinatorial search through the search space.

The evolution of the system through different possible states can be graphically represented by the reachability tree, where a new branch is opened for each of the different possible state evolution starting from a previously known state. The final states are represented by the leaves of the reachability tree found at the end of each branch; a subset of the final states could be feasible solutions belonging to the solution space of the system and others are non-feasible final states.

The computation of online metrics (of efficiency or any other criteria) during the causal exploration and their comparison among the branches that belong to the same level of the reachability tree allows a driven search via a hill-climbing minimal-gradient algorithm,

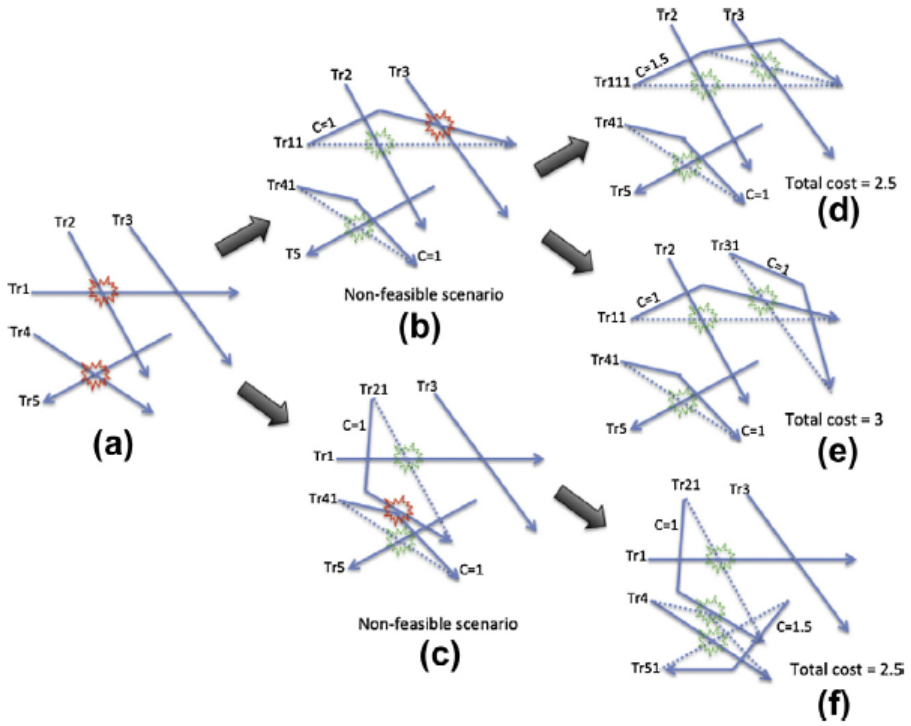

Figure 3.4: Causal exploration on a reachability tree (Ruiz *et al.*, 2014).

which outputs the feasible final states with better metrics first, i.e., more efficient scenarios are found first. Figure 3.4 illustrates how a causal exploration is performed with this method.

Over than 23000 flights impact the European airspace every day. Currently, the airspace is partitioned into sectors in order to organize the air traffic flow and to balance demand and capacity. The development of a sectorless air traffic management (ATM) with the concept of trajectory-based individual flight control is a current research topic, in which Zillies *et al.* implement a model that is capable of determining optimal trajectories for scheduled flights [51]. This task should be performed under the separation restrictions, such that aircraft maintain adequate separation distances from other aircraft and from regions presenting hazardous weather conditions.

The Single Flight Routing Problem (SFRP) is treated as a dynamic single commodity network flow problem, i.e., each commodity is a single aircraft. The modeling of this problem is based on the theory of network flows and linear programming. The airspace is discretized with a four-dimensional network, where each node represents a possible 4D waypoint an aircraft could comply with during its flight. This is accomplished by extending the two-dimensional airspace grid to a 3D airspace by adding heights to the waypoints and then adding a time dimension to each one.

Take-off and landing are not subject to this study, but the cruise time at the default

flight altitude only. Then, the origin and destination points in the trajectory are respectively the first and last points of the route at the desired flight level. Input data and parameters for each aircraft comprise position values, speeds, distances between points (nodes), arcs, time discretization and fuel consumption characteristics.

The cost function takes into account the different objectives and performance parameters of an aircraft. The objectives included into the cost function are minimal fuel consumption, minimal distance and minimal time. Each objective inside the cost function is weighted with a parameter contained in the vector $\mu = (\mu_d, \mu_{ff}, \mu_t)$, which represent respectively distance, fuel consumption and time weighting parameters and can be arbitrarily set. If an aircraft is technically unable to take an arc, this arc is infeasible and its cost is set to infinite.

The SFRP is formulated as a dynamic minimum cost single commodity flow problem and it becomes a shortest path problem which can be solved by artificial intelligence methods such as A* algorithm. In this approach, the node to be selected is the one with the minimum cost value $c(i) = d(i) + \pi(i)$, where $d(i)$ is the distance label and $\pi(i)$ is an heuristic value named as potential. It is calculated as follows, where ff_{min} is the minimum cost for fuel flow, v_{min} is the minimum speed and *dest* is the destination point:

$$\pi_d(i) = directdist(i, dest); \tag{3.4}$$

$$\pi_{ff}(i) = ff_{min}\pi_d(i); \tag{3.5}$$

$$\pi_t(i) = \frac{\pi_d(i)}{v_{min}}; \tag{3.6}$$

$$\pi(i) = \mu_d\pi_d(i) + \mu_{ff}\pi_{ff}(i) + \mu_t\pi_t(i); \tag{3.7}$$

Due to prohibitive space complexity [52], unnecessary nodes are removed from processing before the algorithm is started, for better runtime performance. The resulting subgraph represents a corridor from the origin to the destination point, but this corridor needs to be extended to a wider range of grid points if the algorithm does not yield a solution after a certain number of iterations. The main steps in the algorithm are: Potential calculation, Node selection, Node generation (expansion) and distance updates of connected nodes.

The Multiple Flight Routing Problem (MFRP) addresses the problem in a slightly different way, since more aircraft are added to the scenario and capacity of the airspace is now a constraint. In this multicommodity algorithm, the capacity of each segment is

restricted to one aircraft, i.e. only one aircraft can affect a segment within a particular time interval defaulted to five minutes. Conflicts occur if a segment is affected by at least two aircraft or if two aircraft violate separation requirements.

Lagrangian relaxation uses the idea of relaxing the explicit linear constraints by bringing them into the objective function with associated Lagrange multipliers. The resulting objective function has a cost equal to the cost of the path plus the summation of all individual Lagrange multipliers related to each cube-shaped segment around a specific node at time t, which represent penalties to affected segments. It means that a segment can be affected at most by one flight path. Accordingly, the penalty related to a specific arc can be determined as the sum of all Lagrangian multipliers associated to the segments affected by this arc.

In each iteration of MFRP the algorithm sets the Lagrangian multipliers to fixed values for all nodes in the search space. The Problem decomposes into k separate shortest path problems for any fixed values of the Lagrangian multipliers, where k is the number of demanded flights (commodities). The cost function is then calculated for each commodity k, considering that this function evaluates the cost for sending one commodity along the arc connecting the current node to each neighbor. Thus, SFRP solution can be used to evaluate the path cost for each commodity k in particular.

The solution of the Lagrangian subproblem is used to obtain valid lower bounds on the original problem. The algorithm keeps the value of the sharpest lower bound found so far and updates the value of the lower bound to the maximum value obtained from the Lagrangian relaxation in each iteration. The solution provides a collection of paths with respective associated costs, which are subject to feasibility evaluation, i.e. conflict detection. The algorithm subsequently improves the values of the Lagrangian multipliers and the number of conflicts caused by the solutions are expected to decrease. If the current solution for a commodity k of the Lagrangian relaxation does not cause any constraint violation, the corresponding path is accepted and the cost of each arc lying on that path is updated for all accepted paths. Then a solution which causes the smallest number of collisions with other paths is selected and the corresponding commodity is properly rerouted through the modified network. The algorithm terminates if the best feasible solution achieved so far is within a required accuracy.

The developed model was tested in German Aerospace Center (DLR) simulation environment. Each scenario is composed of: a file containing all scheduled departures, flight plans of each aircraft and weather data. The developed tool was designed to provide optimal routes for each scheduled flight, which are sets of new constraint points used to change the original flight plans. MFRA was tested with a traffic scenario containing 200 flights and the scheduled departure times cover a 15 hours time frame.

Upon startup, Lagrangian multipliers in the Lagrangian subproblem are set to zero. The initial solution to the Lagrangian subproblem was found after 5 seconds with a resulting objective function value of 374101 (km). The solution produced 85 conflicts involving 70 different flights. The waypoints obtained within successive iterations are stored and the search space to certain points around these waypoints is reduced. A solution was found after 66 seconds. The algorithm sequentially rerouted 33 flights and generated a feasible solution with a total cost of 374166 (km), which is within 0.0175% of optimality. At the end, the affected flights were rerouted at a minimum cost.

Ayhan *et al.*[20] introduce a new approach for modeling the air space. The aircraft trajectories are modeled as sequences of cubes centered at the aircraft sampled waypoints. In this approach, the cubes form safety spaces that surround the aircraft in each waypoint along the route. Therefore, a conflict is found whenever two cubes overlap in a given time.

The cube-shaped 4D trajectory presentation is formed in the trajectory prediction phase. The system learns from descriptive patterns of historical trajectories and pertinent weather observations. A Hidden Markov Model (HMM) is developed to predict the flight path for every aircraft.

The conflict detection is performed by a pairwise comparison among the cubes that form the trajectory. The aircraft trajectories are iteratively added to the empty scenario. For every new trajectory to be added, the conflict detection procedure is applied among the trajectories that are already present in the scenario, and a new conflict-free trajectory is calculated to the current aircraft before it is added to the set.

The drawback in this implementation is that there is no guarantee that the prescribed trajectories form the most efficient configuration, although the resolution is very fast. The quality of the resolution procedure is dependent on the order in which the aircraft are evaluated.

Legrand [53] presents a new technique to measure the potential benefit produced by sharing wind/temperature data between aircraft and between aircraft and ground stations. To reach this objective, a new Wind/Temperature Networking concept (WTN) was proposed. The aircraft needs to measure the local atmospheric data, calculate the local wind vector $W(x, y, z, t)$ and the local air density $\rho(x, y, z, t)$, and broadcast the assessments to the further aircraft and ground stations. Having such distributed weather information, each aircraft is able to compute an enhanced local wind/temperature map as a function of location (3D) and time. These updated wind/temperature fields can be shared with other aircraft and ground systems. Using this enhanced weather information, each aircraft is able to improve drastically its own trajectory prediction.

In order to achieve the best flight performance in terms of flight time and fuel consumption, airlines may adjust the flight trajectories based on en-route wind profiles. Legrand

addresses the robust trajectory planning in presence of wind with some uncertainties, as weather forecast usually propose several possible situations by producing Ensemble Prediction. This is achieved by the computation of wind optimal trajectories with a focus on building the network with a clustering algorithm. The simulation results obtained by applying this new concept to weather data with different dispersion over the Atlantic Ocean.

The main relevant contribution is the Wind/Temperature Networking concept based on modern aircraft capacity to measure atmospheric data through their Air Data Computers. Two algorithms were developed and implemented to simulate the WTN concepts, in which the methodology for Wind/Temperature measures interpolation was also developed. The algorithm has been tested on a realistic French airspace with 8000 flights, including short, medium and long-haul ones. The improvement on both Wind/Temperature estimate and trajectory prediction has been demonstrated with very promising results.

3.5 Advancements of current research

Methodologies for strategic CD&R were already presented to the aviation community, ranging from extensive search to Integer Programming and Evolutionary algorithms [54]. The size of the search space may be a matter of concern in such approaches, hence solution attempts are limited to parameterized look-ahead periods in time. Furthermore, pairwise comparisons between trajectories tend to be computationally expensive, which demands the appropriate definition of time windows to constrain the processing scope.

Other techniques for collaborative conflict resolution may regard user-preferred trajectories. Narrowing down the search space is paramount due to computational resources limitations, therefore storing reference trajectories and defining sampling neighborhood for solution sets are also a matter of concern for conflict resolution [25], [55]. The hereby proposed methodology does not compare reference trajectories, and narrows even more the search space by checking time-sequenced waypoints without the need to pre-shape the airspace.

The proposed approach is designed to gracefully handle an undetermined amount of data about predicted trajectories and further to evaluate the available data in order to determine potential conflicts. While some of the presented techniques attempt to store and process trajectory data by pre-analyzing the trajectory characteristics, the proposed framework stores the trajectories as-is, as a sequence of waypoints that should be reached by the airborne aircraft at a specific time in the future.

Convenient data models enable the retrieval of the waypoints in a fashion that they can be easily evaluated for conflict detection. Chapter 6 demonstrates the efficiency of

Table 3.3: Comparison among the current research and state-of-art techniques.

Institution	**Maryland U., USA**	**ENAC, France**	**UnB, Brazil**
Detection	Pairwise ($O(n^2)$)	Pairwise ($O(n^2)$)	$O(nlog(n))$
Resolution	Pairwise ($O(n^2)$)	-	Exploratory
DB associated	Not clear	Not clear	NoSQL
Data type	Historical	Online weather	FPL
Methods	HMM	OR, IA	SA
Trajectory model	AIDL	Not clear	AIDL
Innovation	Cube-shaped 4D trajectory	Wind/Temperature Networking (WTN)	DB Associated 4D trajectory presentation

the proposed approach, which steps out of the problem of extensive trajectory search for conflict detection by efficiently traversing the waypoints queue. Table 3.3 compares the current proposal against the most prominent solutions found for trajectory management.

Chapter 4

Trajectory Prediction in 4D Navigation

Flight data that is viewed by several stakeholders are represented in a shareable system instances known as Flight Objects (FO) [4]. Specific implementations might have internal, private data not shareable with other systems, such as internal flight plan data, internal events, data from internal sub-systems among other ones. This is the reason for the existence of Flight Objects, whose shared information includes (among other data) aircraft identity, flight performance parameters and the flight plan. Once the flight is being executed, the flight plan in the FO includes also the cleared flight profile, current aircraft position and near-term intent information. Availability of the FO to the stakeholders will be ensured by the SWIM network.

Section 4.1 describes the role of the flight plans in the modeling of an aircraft intent. Section 4.2 presents a fundamental technology for trajectory prediction based on the generated aircraft intents.

Section 4.3 describes the Aircraft Intent Description Language (AIDL) as a standardized method to uniquely express the intent information of an aircraft. Section 4.4 goes deeper into the intent description architecture, comparing the different hierarchies of trajectory description languages.

4.1 Flight Intent and Aircraft Intent

The main document used to predict the flight intent is the Flight Plan. This document indicates all the information inherent to the intended movement of the aircraft. It enables then the aircraft to be assigned an ATC slot being the way in which the air traffic resources are planned and allocated to that aircraft [33].

Intent information can be defined as the path and the constraints dictating the aircraft motion in the future, including sequences of control actions, flight plans, ATC constraints and airline preferences [14]. The constraints are a very efficient way to describe the trajectory requirements reducing the solution space. They may be related to limitations in the aircraft performance, airspace or aerodrome capacity, weather conditions or time restrictions. Although constraints are used to model the aircraft behavior limiting the state variables, the usual mechanism to constrain the aircraft behavior is applying aircraft performance models.

The atomic unity of a predicted trajectory is a 2D segment composed of initial and final 2D waypoints. A set $W = (w_1, w_2, \ldots, w_n)$ composed by several 2D waypoints describes the full trajectory of an aircraft provided that:

1. w_1 is the initial waypoint and w_n is the final waypoint;
2. w_i is directly connected to w_j if, and only if, $i = j - 1$;
3. there is always a connected path between w_i and $w_j, i \neq j$.

Hence, W expresses a set of connected segments in a linear trajectory on a 2D grid. This trajectory representation is not appropriately accurate, since it is still possible to enhance this model to accommodate additional state or control variables. Attributes such as flight path and course, aircraft mass, aircraft velocity, or a generic combination of state variables affect the overall flight profile and they should be regarded as variables to any trajectory predictor [14].

The flight intent represents the constraints and preferences applicable to the specific flight described by a Flight Object. This data describes the objectives to be achieved in the flight execution, as the objectives of the aircraft operator and the compliance with constraints inherent to the aircraft configuration, the airport and airspace resources and safety requirements. Several trajectories might be possible to a given flight intent, provided that they meet the specified mission objectives. The flight intent is then an ambiguous definition of an aircraft's trajectory to be refined.

The aircraft intent is the complete description of a constrained aircraft trajectory. In this context, completeness means that the aircraft trajectory is modeled through an unambiguous mathematical description [13].

A ground-based trajectory predictor (TP) must use aircraft intent information into the TP process to compute a trajectory. Provided that aircraft and weather models are the same, any TP application must be able to work with synchronized intent information at the input level. Thus, the aircraft intent ensures interoperability among such automation systems, through an input format named as Flight Script [4].

The Flight Script provides the data required by a TP client to perform a trajectory prediction. The predicted flight trajectories are not supposed to be necessarily identical, but must be consistent. Therefore, this approach could support a trajectory manager (TM) that intends to adjust some aircraft's trajectory allowing the TM to request changes to the Flight Script, although the changes are not proposed directly to the intended trajectory. Typically, these changes are the addition of constraints to the Flight Script, and then the aircraft intent is properly updated.

In a nutshell, the aircraft intent is the operation plan inherent to an aircraft defining precisely how the aircraft intends to comply with the constraints and parameters defined in the Flight Intent. This precise, unambiguous description of the intended flight path allows the interoperability among the stakeholders like trajectory predictors and CD&R (conflict detection and resolution) tools to evaluate scenarios in a shared scenario awareness.

Effective and continuous communication of aircraft intent is required by the decision support tools, like CD&R and TP clients. Communication between airborne aircraft and ground-based service providers can be achieved by the implementation of Automatic Dependent Surveillance - Broadcast (ADS-B) technology [56] [57]. Ground-based operations performed by ATS providers must be able to calculate conflict-free trajectories and communicate them back to the airborne aircraft. This is why interoperability is paramount in the TBO concept, ensuring the common view of weather models, aircraft performance data and a language unambiguously describing the aircraft intent.

4.2 Trajectory computation

Usually, the output of the trajectory generation process is a list of time-referenced samples of the kinetic state and control variables of the aircraft. This process is established as a set of methods to discover suitable trajectories compliant to the flight specification. Obviously, several different trajectories might fulfill the flight script requirements, and some optimization procedure might take place in order to select the most satisfying trajectory [14].

An objective function is used to minimize or maximize optimization criteria. The criteria aim to optimize the flight time range, fuel consumption, airline direct operating cost, amount of performed maneuvers, track deviations or the magnitude of maneuvering accelerations, due to the direct impact in the aircraft passengers comfort. The Cost Index (CI) can be used for this purpose, since it is an attribute showing the trade-off between the costs associated with the flight time and the fuel consumption determining the flight profile to be performed by the aircraft according to the flight variable costs [58].

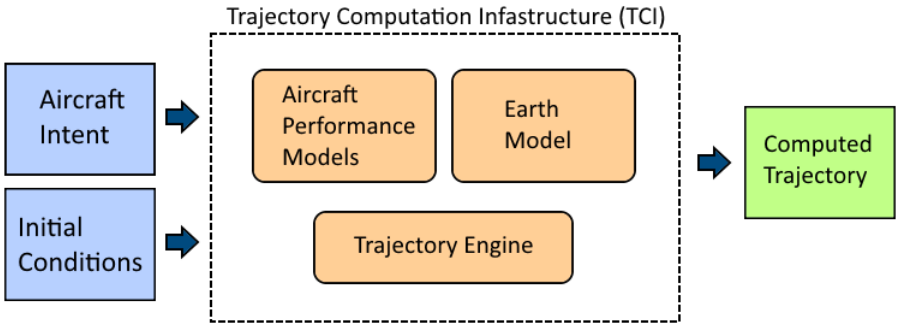

Figure 4.1: Scheme of a trajectory computation framework.

The core of trajectory computation process is a Trajectory Engine integrating the equations of motion into a trajectory based on an initial state and a primary aircraft intent description, obtained from a flight plan. Moreover, to properly calculate the equations of motion, the trajectory engine uses aircraft-specific configuration data provided by an aircraft performance model, such as drag, lift, aircraft weight and so on [19]. The aircraft performance model provides specific characteristics of aircraft performance, and usually is aggregated to the airline procedure model (ARPM) to generate the full Aircraft Model. ARPM contains airline-specific operation parameters, thus most operational systems do not implement such functionalities, and the aircraft model can be simply restricted to the aircraft performance model (APM) [59].

The data set regarding the wind, temperature and air pressure are also provided by an Earth model delivering proper and up-to-date weather data to the trajectory engine. Figure 4.1 shows a representation of how these functionalities are related to the task of aircraft trajectory computation.

The aircraft trajectory may be computed using different sets of equations of motion varying in complexity and in the effects of the degrees of freedom. The formulation of the aircraft intent depends on the actual form of the equations of motion, because the same variables in the equations are the ones in the description of the aircraft intent [19]. AIDL regards the aircraft as a point-mass object to be placed in a 3D position in space. The aircraft coordinates are defined by three parameters:

- latitude (ϕ);
- longitude (λ);
- altitude (h).

The aircraft's attitude can also be defined by three parameters:

- aerodynamic pitch angle (γ_{TAS});

- aerodynamic yaw angle (χ_{TAS});
- aerodynamic bank angle (μ_{TAS}).

A set of equations is then obtained applying the given parameters (Equations 4.1 to 4.3) reproducing the aircraft's evolution along its trajectory, concerning all aspects of the state of the aircraft. They are referred to as the Aircraft Motion Model (AMM) and are employed at the core of a trajectory engine.

Equations 4.1 to 4.3 model the dynamics of the aircraft, where WA_1, WA_2 and WA_3 are the wind gradients, T is the total thrust, D is the total drag, L is the total lift, W is the aircraft weight and m is the aircraft mass:

$$\frac{dv_{TAS}}{dt} = \frac{T - D - Wsin\gamma_{TAS}}{m} + WA_1 \tag{4.1}$$

$$\frac{d\gamma_{TAS}}{dt} = \frac{1}{v_{TAS}}\left[\frac{Lcos\mu_{TAS} - Wcos\gamma_{TAS}}{m} + WA_2\right] \tag{4.2}$$

$$\frac{d\chi_{TAS}}{dt} = \frac{1}{v_{TAS}cos\gamma_{TAS}}\left[\frac{Lsin\mu_{TAS}}{m} + WA_3\right] \tag{4.3}$$

Equation 4.4 models the mass variation, where F is the instantaneous fuel consumption and m is the aircraft mass:

$$\frac{dm}{dt} + F = 0 \tag{4.4}$$

Equations 4.5 to 4.7 model the navigation evolution, where M and N are the meridian and prime vertical radius of curvature, respectively, and w1 and w2 are wind components:

$$\frac{d\gamma}{dt} = \frac{1}{(N+h)cos\phi}(v_{TAS}cos\gamma_{TAS}sin\chi_{TAS} + w_2) \tag{4.5}$$

$$\frac{d\phi}{dt} = \frac{1}{M+h}(v_{TAS}cos\gamma_{TAS}cos\chi_{TAS} + w_1) \tag{4.6}$$

$$\frac{dh}{dt} = v_{TAS}sin\gamma_{TAS} \tag{4.7}$$

This AMM is a system composed of seven non-linear ordinary differential equations and ten dependent variables. This system has then three degrees of freedom. An unambiguous trajectory only can be defined if these three degrees of freedom are given so they must be defined externally. Furthermore, adding the three degrees of freedom from the aircraft configuration, namely the landing gear (δ_{LG}), the high-lift devices (δ_{HL}) and the speed brakes (δ_{SB}), the system has its six degrees of freedom.

The variables in the resulting six-degree of freedom system of equations can be organized as follows:

- state variables (v_{TAS}, γ_{TAS}, χ_{TAS}, ϕ, λ, h, m);
- control variables (μ_{TAS} L, δ_T);
- configuration variables (δ_{LG}, δ_{HL}, δ_{SB}).

The model shown above considers also the time evolution to yield a solution returning a unique trajectory to the aircraft. This is achieved by rewriting the motion equations in the state-space form, as follows:

$$\dot{v}_{TAS} = f(v_{TAS}, \gamma_{TAS}, \chi_{TAS}, \delta_T, L, \delta_{LG}, \delta_{HL}, \delta_{SB}, \delta, \theta, g, w, t)$$

$$\dot{\gamma}_{TAS} = f(v_{TAS}, \gamma_{TAS}, \chi_{TAS}, \mu_{TAS}, L, \delta_{LG}, \delta_{HL}, \delta_{SB}, \delta, \theta, g, w, t)$$

$$\dot{\chi}_{TAS} = f(v_{TAS}, \gamma_{TAS}, \chi_{TAS}, \mu_{TAS}, L, \delta_{LG}, \delta_{HL}, \delta_{SB}, \delta, \theta, w, t)$$

$$\dot{m} = f(v_{TAS}, \delta_T, \delta, \theta, t)$$

$$\dot{\lambda} = f(v_{TAS}, \gamma_{TAS}, \chi_{TAS}, \lambda, \phi, h, w, t)$$

$$\dot{\phi} = f(v_{TAS}, \gamma_{TAS}, \chi_{TAS}, \lambda, \phi, h, w, t)$$

$$\dot{h} = f(v_{TAS}, \gamma_{TAS}, \lambda, \phi, h, w, t)$$

Here, variables δ, θ, g and w form the Earth Model $E = [\delta\theta gw]$ and represent, respectively, the pressure, temperature, gravity force and wind velocity. Components of E depend on position and time (Equation 4.8), where X is the state vector given by $X = [v_{TAS}\gamma_{TAS}\chi_{TAS}\lambda\phi hm]$.

$$E = E(X, t) = f(\lambda, \phi, h, t) \tag{4.8}$$

Table 4.1: Aircraft motion constraints.

Constraint Label	**Parameter**	**Affected aspect**
C1	δ_{HL}	High lift devices
C2	δ_{SB}	Speed brakes
C3	δ_{LG}	Landing gear
M1	δ_T	Throttle
M2	μ_{TAS}	Bank angle
M3	v_{TAS}	Speed
M4	γ_{TAS}	Path angle
M5	χ_{TAS}	Yaw angle
M6	h	Altitude
M7	$v_{TAS} \cos \gamma_{TAS}$	Horizontal speed
M8	$v_{TAS} \sin \gamma_{TAS}$	Vertical speed
M9	$g(\lambda, \phi)$	Horizontal track
M10	$g(\lambda, \phi, h)$	Vertical track

Moreover, a control vector $u = [\mu_{TAS}\ L\ \delta_T]$ and a configuration vector $\Delta = [\delta_{HL}\ \delta_{LG}\ \delta_{SB}]$ are used altogether to form the equations of motion in a compact state-space form as seen in Equation 4.9:

$$\dot{X} = f\Big(X, u, E(x,t), \Delta, t\Big) = f(X, u, \Delta, t) \tag{4.9}$$

One should note that u and Δ contain the six inputs needed to close the six degrees of freedom showing a unique aircraft trajectory. It is also possible to close the system formed in Equation 4.9 through constraints, represented by Equation 4.10, being a mathematical abstraction of any possible aircraft guidance configuration:

$$g = (X, u, \Delta, t) = 0 \tag{4.10}$$

Finally, a set of constraints is obtained for aircraft control and guidance to integrate the AIDL framework, as shown in Table 4.1.

4.3 Aircraft Intent Description Language (AIDL)

The Aircraft Intent Description Language (AIDL) was firstly proposed as a standardized formal method to express aircraft intent information allowing interoperability among decision support tools concerning trajectory prediction [9]. Thus, an intent described in AIDL is input for a consistent trajectory computation.

A direct benefit of AIDL is the interoperability among heterogeneous decision support tools leading to flight profiles that are more efficient increasing the ATM capacity. The ground-based tools for an aircraft under its operational coverage must then be consistent with the trajectory predicted by the FMS onboard [19].

In this sense, two or more different trajectory predictors can yield results that can be interpreted as the same trajectory constrained to a given time interval into the future. Thus, given an initial state, the trajectory predictor must compute a unique trajectory describing the intended operation of the aircraft.

The aircraft intent must describe how an aircraft must be controlled. Trajectory computation requires a set of equations of motion, models of the aircraft performance and the environmental conditions, such as wind speed, temperature and pressure.

Differential Algebraic Equations are the basis theory behind AIDL. Mathematical rules are modeled to define the alphabet and the grammar rules of the language. AIDL grammar rules define the possible combinations of the alphabet instructions, in a unique, unambiguous manner [60]. Then, AIDL allows the description of a trajectory before it is actually computed, so the trajectory predictors have the possible motion profile of the aircraft as input for computation.

4.3.1 AIDL Instruction Set

AIDL instructions are the lexemes of AIDL, i.e., they form the lexicon of this language. Furthermore, they define the constraints used to close the degrees of freedom (DOF) of the aircraft motion model given by the motion equations [19].

AIDL instructions can be divided into five groups according to their effect in the motion profile of the aircraft, that is, the mode in which they influence the aircraft behavior [13] [10]:

- Set instructions model the change of a given parameter of aircraft motion or configuration from its initial value towards a target value. This transition is governed by the specific aircraft performance model and capture the evolution of configuration states.

AIDL Alphabet - Σ_{AIDL}					
#	Keyword	Instruction	#	Keyword	Instruction
1	SL	Speed Law	19	OLT	Open Loop Throttle
2	HS	Hold Speed	20	SBA	Set Bank Angle
3	HSL	Horizontal Speed Law	21	BAL	Bank Angle Law
4	HHS	Hold Horizontal Speed	22	HBA	Hold Bank Angle
5	EL	Energy Law	23	OLBA	Open Loop Bank Angle
6	HE	Hold Energy	24	CL	Course Law
7	VSL	Vertical Speed Law	25	HC	Hold Course
8	HVS	Hold Vertical Speed	26	TLP	Track Lateral Path
9	SPA	Set Path Angle	27	SHL	Set High Lift devices
10	PAL	Path Angle Law	28	HLL	High Lift devices Law
11	HPA	Hold Path Angle	29	HHL	Hold High Lift devices
12	OLPA	Open Loop Path Angle	30	SSB	Set Speed Brakes
13	AL	Altitude Law	31	SBL	Speed Brakes Law
14	HA	Hold Altitude	32	HSB	Hold Speed Brakes
15	TVP	Track Vertical Path	33	OLSB	Open Loop Speed Brakes
16	ST	Set Throttle	34	SLG	Set Landing Gear
17	TL	Throttle Law	35	HLG	Hold Landing Gear
18	HT	Hold Throttle			

Figure 4.2: Full set of AIDL instructions (Konyak *et al.*, 2008).

- Law instructions model flight commands that control any given parameter as a function of dynamic state, i.e., the aircraft motion, such as horizontal/vertical speed, energy, path angle, altitude and others. Such parameters can be measured by the aircraft systems and sensors with specific functions programmed in the FMS.
- Hold instructions maintain a specific variable constant by the coordinated action of the aircraft control and configuration settings, independently of its value. They are a simplified version of Law instructions.
- Open Loop instructions model the direct actions and commands issued by the pilot or the FMS over the available controls. These instructions do not depend on the state variables, just the time.
- Track instructions model the guidance along a geometric path. These instructions contain geometric aspects of the trajectory solely and specify the geometry of the aircraft trajectory to be followed.

Some AIDL instructions might also include specifiers indicating the type of the parameter to be controlled by the instruction, e.g., which speed (CAS, TAS, Mach) is the target parameter to be governed [61].

The full instruction set of AIDL can be viewed in Figure 4.2.

Speed guidance

Speed guidance instructions are executed in a Longitudinal thread. These instructions are Speed Law (SL) and Hold Speed (HS). The target variation is v_{TAS} and they are constrained by constraint M3.

The specifiers of speed guidance instructions are:

- M (Mach number);
- TAS (true air speed);
- IAS (indicated air speed);
- CAS (calibrated air speed);
- EAS (equivalent air speed).

Horizontal speed guidance

Horizontal speed guidance instructions are executed in a Longitudinal thread. These instructions are Horizontal Speed Law (HSL) and Hold Horizontal Speed (HHS). The target variation is $v_{TAS} \cos \gamma_{TAS}$ and they are constrained by constraint M7.

The unique specifier of horizontal speed guidance instructions is HS (horizontal speed).

Vertical speed guidance

Vertical speed guidance instructions are executed in a Longitudinal thread. These instructions are Vertical Speed Law (VSL) and Hold Vertical Speed (HVS). The target variation is $v_{TAS} \sin \gamma_{TAS}$ being given by the constraint M8. The specifiers of vertical speed guidance instructions are:

- VS (vertical speed);
- ROC (rate of climb).

Path angle control

Path angle control instructions are executed in a Longitudinal thread. These instructions are Set Path Angle (SPA), Path Angle Law (PAL), Hold Path Angle (HPA) and Open Loop Path Angle (OLPA). The target variation is γ_{TAS} being given by the constraint M4. The specifiers of path angle control instructions are:

- GEO (geometric path angle);
- AER (aerodynamic path angle).

Altitude guidance

Altitude guidance instructions are executed in a Longitudinal thread. These instructions are Altitude Law (AL) and Hold Altitude (HA). The target variation is h being given by the constraint M6. The specifiers of altitude guidance instructions are:

- GEO (geometric path angle);
- PRE (pressure altitude).

Vertical position guidance

Vertical position guidance instructions are executed in a Longitudinal thread. The unique instruction is Track Vertical Path (TVP). The target variation is ϕ, λ, h being given by the constraint M10. Vertical position guidance instructions do not have specifiers.

Energy guidance

Energy guidance instructions are executed in a Longitudinal thread. These instructions are Energy Law (EL) and Hold Energy (HE). The target variation is $\frac{dv_{TAS}}{dh}$. The specifiers of energy guidance instructions are:

- ESF (energy share factor);
- ASF (acceleration factor).

Throttle control

Throttle control instructions are executed in a Longitudinal thread. These instructions are Set Throttle (ST), Throttle Law (TL), Hold Throttle (HT) and Open Loop Throttle (OLT). The target variation is δT being given by the constraint M1. The specifiers of throttle control instructions are:

- MTKF (maximum take-off);
- GA (go around);
- MCNT (maximum continuous);
- MCBM (maximum climb);
- MCRZ (maximum cruise);
- HIDL (high idle);
- LIDL (low idle);

- THRO (throttle control parameter)
- CT (thrust coefficient).

Lateral directional control

Lateral directional control instructions are executed in the Lateral thread. These instructions are Set Bank Angle (SBA), Bank Angle Law (BAL), Hold Bank Angle (HBA) and Open Loop Bank Angle (OLBA). The target variation is μ_{TAS} being given by the constraint M2. The unique specifier of lateral directional control instructions is AER (aerodynamic bank angle).

Lateral directional guidance

Lateral directional guidance instructions are executed in the Lateral thread. These instructions are Course Law (CL) and Hold Course (HC). The target variation is χ_{TAS} being given by the constraint M5. The specifiers of lateral directional guidance instructions are:

- GEOMAG (magnetic bearing);
- GEOTRUE (true bearing);
- AERMAG (magnetic heading);
- AERTRUE (true heading).

Lateral position guidance

Lateral position guidance instructions are executed in the Lateral thread. The unique instruction is Track Lateral Path (TLP). The target variation is ϕ, λ. Lateral position guidance instructions do not have specifiers.

High lift configuration

High lift configuration instructions are executed in the High Lift Devices thread. These instructions are Set High Lift (SHL), High Lift Law (HLL) and Hold High Lift (HHL). The target variation is δ_{HL} being given by the constraint C1. The high lift configuration instructions do not have specifiers.

Speed brake configuration

Speed brake configuration instructions are executed in the Speed Brake thread. These instructions are Set Speed Brake (SSB), Speed Brake Law (SBL), Hold Speed Brake

(HSB) and Open Loop Speed Brake (OLSB). The target variation is δ_{SB} being given by the constraint C2. Speed Brake configuration instructions do not have specifiers.

Landing gear configuration

Landing gear configuration instructions are executed in the Landing Gear thread. These instructions are Set Landing Gear (SLG) and Hold Landing Gear (HLG). The target variation is δ_{LG} being given by the constraint C3. Landing Gear configuration instructions do not have specifiers.

4.4 Intent Description Architecture

Aircraft intent is expected to be synchronized across different decision support tools, and it must then be expressed by a commonly known format prior sharing through appropriate communication infrastructure [60]. Obviously, each trajectory predictor to be implemented must accept aircraft intent information as input.

Formal languages that describe aircraft intent are different in hierarchy and completeness. These languages depend on the clear division between aircraft intent and flight intent. The aircraft intent determines how an aircraft is intended to operate within a given time interval, while flight intent describes the objectives and restrictions expected to be met in the future time of the flight operation. The aircraft intent captures basic commands, guidance modes and control strategies enabling the aircraft to fulfill the requirements of the flight intent [14].

As a consequence, the flight intent is not sufficiently strong to yield a unique, unambiguous trajectory to the aircraft. Therefore, the flight intent is based in a higher level in the intent hierarchy, guiding the process to a lower level intent generation. This process calculates more accurate trajectory descriptions.

The approach described above has several advantages. Different hierarchies in intent description allow the representation of information in different levels of detail, specification of different flight portions and specification of restrictions and optimization criteria to be fulfilled by the desired trajectory [10].

The formal intent description languages can be divided in three main groups. AIDL is the lower level language, capable of describing unambiguous trajectories, modeling each detail of how an aircraft should behave. Above AIDL, the Intent Composite Description Language (ICDL) facilitates description of complex behavior, and at the higher level in the hierarchy, the Flight Intent Description Language (FIDL) expresses trajectory requirements in the form of constraints and objectives. Information of FIDL includes preferred routes, preferred climb rate and preferred cruise altitude and Mach number.

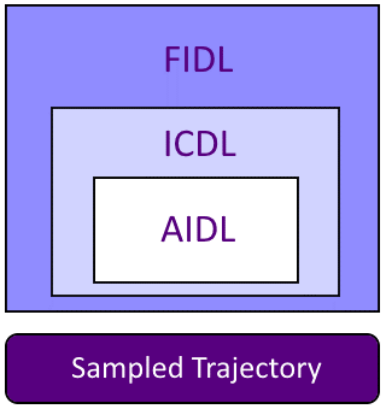

Figure 4.3: Language hierarchy.

Figure 4.3 illustrates how the three languages are related. ICDL can be viewed as an extension of AIDL, due to the fact that anything described in AIDL can also be described in ICDL, and FIDL is a language closer to the flight plan than to the trajectory specification.

AIDL is the formal language directly related to the physics of aircraft motion and to the degrees of freedom available to the pilot and the air traffic controller. The alphabet of AIDL consists on atomic flight operations and the proper rules to combining them in a way that all degrees of freedom are properly closed at any instant.

AIDL instructions may require parameters to define the effect of the instruction. These parameters may contain magnitude, a mathematical function or a specifier. The execution interval of AIDL instructions is defined by triggers, which are the accomplishment of the end condition for instructions. AIDL has five types of triggers [10]:

- *Fixed trigger*: a time or distance limit, independent of the aircraft motion state.
- *Floating trigger*: mathematical condition involving motion states or configuration variables.
- *Linked trigger*: represents synchronous ending with other parallel AIDL instructions.
- *Default trigger*: indicates the time at which the demanded change in motion or configuration is completed. Default triggers are always associated to Set instructions.
- *Auto trigger*: used to satisfy future conditions over the aircraft state.

Each degree of freedom in the aircraft motion must be closed by every AIDL instruction. Therefore, an AIDL sentence is composed by six parallel lists of AIDL instructions, known as AIDL threads. The six AIDL threads are divided as:

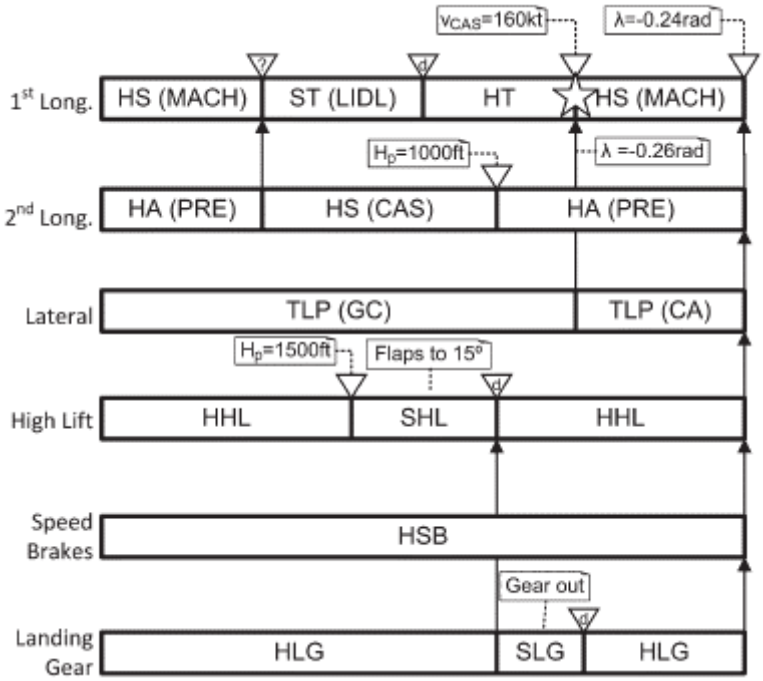

Figure 4.4: Example of AIDL sequence, with 6 threads and some triggers (Besada *et al.*, 2013).

- *two Longitudinal threads*, for instructions describing the aspects of aircraft speed, altitude, engine setting, or energy;
- *one Lateral thread*, for instructions defining the lateral aspects of the aircraft motion, such as geometric path, heading, bearing, or bank angle;
- *three configuration threads*, dedicated to high-lift devices, speed brakes and landing gear.

All execution threads must end at the same time, and for this reason, all final instructions of each thread (but one) must finish in linked triggers. They must point to the remaining thread defined as the Master trigger. An example of an AIDL instance is shown in Figure 4.4.

ICDL uses composition operations over the basic AIDL instructions, defining both AIDL effects and triggers in a parametric fashion [10]. ICDL is in a higher level of intent definition, and it is not required to define all the six threads of AIDL. Thus, ICDL is more flexible to support trajectory refinement, but is not able to univocally describe a trajectory.

The key elements of ICDL are:

- *Parameter intervals*: range of values being used for a specific parameter;

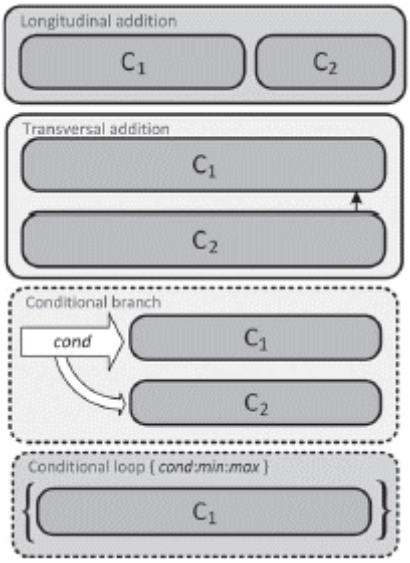

Figure 4.5: Graphical representation of composition operations (Frontera *et al.*, 2014).

- *Elementary composites*: ordered, aggregated AIDL instructions;
- *Composition operations*: constructive elements of the language, defining parallel (+) or serial (*) composition of successive composites;
- *Aggregated composites*: result as consecutive operations over the elementary composites.

ICDL composites can be combined in transversal or longitudinal composition. The transversal composition is both associative and commutative, allowing the aggregated composites to be linked between themselves, with all AIDL instructions ending synchronously. In longitudinal composition, the commutative property does not hold. The end trigger of the resulting aggregated composite coincide with the end trigger of the last composite in the right-hand side (time increases from left to right).

Composites may also be combined in conditional branches or conditional loops using parameterized execution intervals. ICDL can then be abstracted to an algorithmic understanding instead of the imperative, strict intent representation of AIDL.

Figure 4.5 shows an example of possible composite aggregations in ICDL, and Figure 4.6 shows an example of ICDL instance.

The highest level in the intent language hierarchy, the FIDL, is a generalization of the flight plan information, allowing the usage of ICDL. FIDL also connects the detailed specification level and the more abstract intent specification level. The FIDL lexicon works with flight segments (FS), conditions, constraints and objectives. Graphically, FIDL and ICDL can be expressed in analogous manner, and any valid ICDL instance with appropriate triggers represents also a valid flight segment in FIDL.

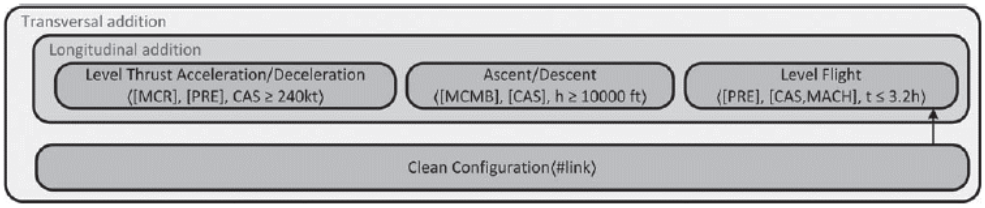

Figure 4.6: Graphical representation of an ICDL instance (Frontera *et al.*, 2014).

The flight segment is the intent of changing the aircraft dynamics toward a final state condition corresponding to the end trigger. The flight segment represents the aircraft behavior ranging from a fully unknown maneuver to a complete description of the aircraft's operation.

Chapter 5

Optimization through local search algorithms

Every problem to be solved consists of an environment where the search for a solution occurs. The state space composing the environment needs to be evaluated, so every position in this scenario can be considered as a probable solution.

When a valid solution is found, optimization can be performed in order to find even better solutions. This chapter is dedicated to present basic contextualization about optimization problems.

Section 5.1 explains the concepts of local search problems and the nature of incomplete search.

Section 5.2 introduces the Simulated Annealing as an implementation for reaching low energy states in the process of cooling substances. Finally, Section 5.3 introduces brief concepts in Game Theory and explains the equilibrium points as solutions to the game.

5.1 Local Search Algorithms

Local search algorithms are a class of problem-solving techniques that do not depend on the history of the solutions previously found through exploration of the search space. Therefore a single current state is evaluated so the system can reach an immediate neighboring state [52]. Furthermore, several problems in the real world are modeled for continuous environments, which means that they have infinite branching factor considering the reachability tree.

Usually, local search methods start in a state, which simply is an assignment of a value to each variable, and then it iteratively tries to find improvements to such assignment by taking random steps toward the neighborhood of the current state [62]. In such approach there is no guarantee that the search space will be fully explored, so the successive assign-

ments proceed until some halting criterion is achieved, which can be a satisfying solution found or a maximum of iterations (or processing time) reached.

Key advantages in such approach are the low memory usage and the confidence that satisfying solutions are guaranteed to be found even in very large search spaces, for which systematic and complete algorithms are impractical [52]. Generally, local search problems are modeled as optimization problems where an objective function is used in order to evaluate the quality of every reachable state.

A standard high-level algorithm for local search is described in Algorithm 1.

Algorithm 1: A local search algorithm

```
   Data: variables, domain, constraints
   Result: assignment
1  while true do
2  |  foreach var ∈ variables do
3  |  |  assignment[var] ← random(domain(var));
4  |  end
5  |  while criterion = false and isGoodEnough(assignment) = false do
6  |  |  varX ← any variable in variables;
7  |  |  value ← random(domain(varX));
8  |  |  assignment[varX] ← value;
9  |  end
10 |  if isGoodEnough(assignment) then
11 |  |  break;
12 |  end
13 end
14 return assignment
```

Random assignment is performed in lines 2 to 4 and random walk is performed in lines 5 to 9. One can notice that such algorithm is not guaranteed to halt if the yielded solutions are not evaluated as good enough, which forces a resetting of the initial assignment and another search try. Obviously, specific halting criteria can be established for each problem, although this algorithm is incomplete.

This is the reason why local search algorithms can also be understood as *evolutionary algorithms* [63]. Incremental mutations in the current state produces new solutions, that are kept if they are satisfying. This means that each iteration replaces the current solution by a better one, or at least a close one. Naturally, the neighborhood needs to be properly modeled and is highly dependent on the environment's structure [64].

In such paradigm, it is logical to think the solution space as a population composed of individuals that gradually get fit for the environment constraints. Each assignment is a single individual that should be evaluated under a fitness function that will give a hint if

this individual is good enough to be passed to the next generation. In short words, fitter individuals are more likely to be selected as solutions.

Although local search may lead to valid solutions, sometimes (if not always) the goal is to find even better solutions. This process is called *optimization* and consists of yielding solutions that maximize the payoff in the scope where the optimization function is valid. Eventually, the solution will reach an equilibrium state, meaning that no further movements should yield better solutions [65].

In order to get a model for an optimization problem, one must firstly define a set of variables (and their respective domains), an objective function that evaluates assignments for real numbers and an optimality criterion, which consists in finding solutions that increase the payoff [63]. If the problem is cost optimization, the quality of solution depends on finding assignments that minimizes the objective function. On the other hand, if the problem is related to profit increase, then the objective function must be maximized.

5.2 Simulated annealing

Some classes of problems are too difficult to be solved by efficient algorithms. This is a common situation in optimization algorithms, for which there are no guarantee that solutions will be found in polynomial time. For this reason, optimization problems often can be cast to simpler decision problems by bounding the value to be optimized [66].

This means that the problem of picking the best value in the full extent of possibilities might by intractable. An effective way to bound optimization problems is limiting the set of alternatives in the search space [64]. Obviously, this new finite set does not provide any confidence that the best solution should be within it, but indeed there is a best solution confined to this search space. This is a local optimum.

In local search algorithms, randomization is often used in order to select a local move when several are available [63]. This strategy prevents the solution to get stuck in sub-optimal states in the search space and avoids premature or biased convergence. Furthermore, randomization can be used to getting a starting point in the search procedure, which should be interpreted as an initial solution for the problem.

Since the local optima are not known in advance, and they can be many, random moves provide a way to establish a probability for finding the best solution. May p be the probability of finding a satisficing local optimum. Then a good solution is expected to be found within $O(1/p)$ time. If this does not happen, the local search can be restarted from a different initial solution, which is given by reseeding the randomization algorithm [63]. It is now clear that the quality of the final solution depends on the starting point.

Kirkpatrick *et al.* [67] firstly introduced the concept of Simulated Annealing as an analogy with thermal equilibrium in condensed matter physics. This analogy is based on the principle that when solid materials are heated past the melting point they can be cooled again, so the structural properties of the material are changed according to the cooling rate, and the residual energy of the resulting configuration can be evaluated [64]. The fact is: the lower the energy state, the higher the quality of the material.

The rate of cooling is strongly important to the residual energy of the material. A low energy state results from a slow cooling process and high energy states are obtained from fast cooling. In the later case, the thermal shock may result in an imperfect material. At a temperature T, for a substance in equilibrium, the probability of an increase in energy of magnitude ΔE is given by Equation 5.1, where k is the Boltzmann's constant.

$$P(\Delta E) = e^{-\Delta E/kT} \tag{5.1}$$

The movements in the search space S are modeled as a Markov chain, where $x(t) = i$ is the current state and j is one random element in the neighborhood $S(i)$ of i [68]. The acceptance of j as the next move is conditioned to $\Delta E = C(j) - C(i)$, where C is a cost function, according to the following conditions:

$$\begin{cases} x(t+1) = j & \text{if } C(j) \leq C(i) \\ x(t+1) = j \text{ with probability } P(\Delta E) & \text{if } C(j) > C(i) \end{cases} \tag{5.2}$$

Each moment in the cooling process may be considered as an iterative step in which an atom of the substance is displaced to a new position, which makes the energy of the solid system vary in ΔE. If this variation is negative, the result is accepted as a better configuration than the previous one. If it is positive, the new configuration is accepted with a given probability.

In fact, Simulated Annealing algorithm is conceived as a hill-climbing algorithm where random walk sometimes takes place. This is done because a pure hill-climbing approach is guaranteed to be incomplete (because local optima may trap the search) [52]. The minimization characteristic of the algorithm allows the mirroring of hill-climbing to a gradient descent search, where some uphill moves sometimes are performed. The probability of uphill moves decreases as the temperature gets lower, which means that bad moves are more likely to be accepted in the beginning of the cooling process.

In the algorithm, the material is the problem to be solved. The energetic state of the material is considered as a cost function for the instantaneous configuration, that is, each configuration corresponds to a solution whose cost is measurable. Leading the material to the lowest energy state is equivalent to minimizing the cost of the solution.

The system should be concisely described prior modeling a problem as a Simulated Annealing optimization task. Each state should be evaluated with a cost function C, which is problem-specific, and the annealing schedule must be defined [67]. As any local search algorithm, a neighborhood must also be delimited for each state i and the movement function must be implemented. This function is mainly a random selection of a neighboring element. Other important decisions should be taken [64]: the starting temperature, the rate of cooling and the stopping condition. These decisions directly influence the speed and the quality of solutions.

The Markov property asserts that previous states are irrelevant to the next moves, such that the current state has the only information available for reasoning. Thus, reaching s' from s, where $s, s' \in S$ are states in the set S, do not depend on the history of earlier states [52]. A Markov Decision Process (MDP) is a decision making process in which the Markov property holds. Equation 5.3 describes this situation, summarizing that, given a MDP X_n, the MDP $X_r, r > n$ is not influenced by $X_m, m < n$.

$$P_r\{X_{n+1} = j \mid X_0 = s_0, X_1 = s_1, ..., X_{n-1} = s_{n-1}, X_n = s_n\} = P_r\{X_{n+1} = j \mid X_n = s_n\} \\ \forall n \in N, \forall s_0, ..., s_{n-1}, s_n, j \in S \tag{5.3}$$

In order to properly design X_n, the following project decisions must be defined [69]:

- a set of allowed states;
- a set of allowed actions;
- a reward function for every state;
- a transition model $P(s'|s, a)$, where each element in $s' \in S$ is reached from $s \in S$ with a given probability.

Hajek's theorem assesses the convergence of the algorithm by analysing the system's Markov chain [68], considering that state i communicates with the set of global minima S^* at height h if there exists a path in S (with each element in the path being a neighbor of the preceding element) that starts at i and ends at some element in S^*, provided that the highest value of C along the path is $C(i) + h$. Also, let b be the smallest number such

that every $i \in S$ communicates with S^* at height b. Then, the algorithm converges if and only if:

$$\begin{cases} \lim_{t\to\infty} T(t) = 0 \\ \sum_{t=0}^{\infty} e^{\frac{-b}{T(t)}} = \infty \end{cases} \tag{5.4}$$

The efficiency of the Simulated Annealing algorithm is highly dependent on the length of the Markov chains, which reflect the duration of the cooling schedule [70]. As said before, the cooling should be slow enough, so erratic behavior is avoided. The termination criterion usually is when the temperature reaches the freezing point, but there is no freezing in computational algorithms. Thus, *freezing* can hereby be understood as the lowest energy state, or the less costly solution. This happens when the obtained cost is considered good enough, or when no more solutions are accepted during a certain number of consecutive cooling steps.

5.3 Game Theory

Agents in any environment may acquire, transfer, distribute or loose resources. Any action taken by agents or groups of agents impacts the environment in any way, and may even affect the other players. Hence, the future states of interactive environments result from decisions (movements) performed by agents or from probabilistic events.

Game Theory is the formal study of decision making processes in which players make decisions in order to maximize the expected reward, transiting from an initial state toward potentially better states [71]. The existence of several agents may motivate competitive behavior, which occurs when there is a conflict for desired resources conflict at a given time and place in the environment, or in the game.

Strategic behavior is the rule of thumb in interactive scenarios. A strategy may be understood as any possible action for a player. A complete action plan of an agent constitutes the strategic profile, which is the policy that specify what the agent should do in any state [52]. Pure strategies are a simple definition of this behavior, and mixed strategies are given by probabilistic distributions over the pure strategies [65].

The best strategy is the one that should yield the best reward to the player. This reward is defined by a function (called the *payoff*), which is a real number that expresses the player's preference about a given situation [72]. Let P_i be this player. Then, its pure strategies $\pi_{i\alpha}$ are defined by Equation 5.5, where $p_{i\alpha} \geq 0$ and $\sum_\alpha p_{i\alpha} = 1$. In other words,

$\pi_{i\alpha}$ is the α-th strategy of player P_i, which will be used with probability p_i.

$$s_i = \sum_{\alpha} p_{i\alpha}\pi_{i\alpha} \tag{5.5}$$

Equation 5.5 makes clear that mixed strategies are non-deterministic. Then, the n-tuple of mixed strategies $\sigma = (s_1, s_2, ..., s_n)$ and a function $u_i(\sigma) = u_i(s_1, s_2, ..., s_n)$ can be properly defined, provided that $u_i(\sigma)$ is the payoff function associated to the mixed strategies of player P_i.

Some environments allow the agents to simply make use of their reward system to take discrete decisions, like a matrix with preset payoffs. More complex scenarios may present dynamic or stochastic events, and they demand enhancements on the reward system. Additionally, most cases in the real world cannot be efficiently modeled as discrete scenarios, so the agents must rely on dynamic reward systems in order to take decisions. Hence, the expected payoff is modeled as an utility function, which is well-defined, bounded to a domain and differentiable in the given state space [73].

5.3.1 Equilibrium points

A rational player should then act in a way that its payoff is always maximized [71]. This means that the player uses its behavior policy in order to traverse the state space in a way that makes the overall process yield the best reward, or namely, an equilibrium point.

Another characteristic of rational players is that they reason about the rationality of other players as well. Thus, each player must take decisions that increase the payoff, but also respond to the decisions performed by the other players. The selected strategic profile is such that, if all players draw their decisions simultaneously, none of them would have a greater reward if its strategy was changed [74]. This is the Nash Equilibrium, defined by the state in which the players cannot unilaterally decide for presumably better strategies [72].

The fact that mixed strategies for player P_i are build from a distribution of probabilities over $\pi_{i\alpha}$ allows the substitution of any mixed strategy for another one by adjusting the probability mapping. Then, a valid substitution of a given mixed strategy r_i in σ is:

$$(\sigma; r_i) \rightarrow (s_1, s_2, ..., s_{i-1}, r_i, s_{i+1}, ..., s_n) \tag{5.6}$$

Finally, the strategic profile σ is an equilibrium point if, and only if, Equation 5.7 is valid [65].

$$u_i(\sigma) = max_{\forall r_i}[u_i(\sigma; r_i)] \tag{5.7}$$

Chapter 6

Modeling of Conflict detection and resolution algorithms for 4D navigation

Usually, a conflict in the airspace can be defined as a situation in which two or more airborne aircraft violate the minimum separation from another aircraft. The minimum separation must be established both in the vertical and longitudinal axis.

This chapter is dedicated to define a conflict and propose a method for detecting potential separation violations. Section 6.1 defines mathematically how a conflict occurs.

Section 6.2 briefly describes the quality attributes expected to be held by the proposed implementation.

Section 6.3 exposes the main technologies used in the proposed method for conflict detection. Section 6.4 describes the implementation of such techniques.

Finally, Section 6.5 describes the implementation of a conflict resolution algorithm.

6.1 Definition of conflict

The minimum vertical distance between two aircraft must be equal or greater than 1000 feet (approximately 305 meters, for flight level below 29000 feet high) or 2000 feet (approximately 610 meters, for flight level above 29000 feet high), and the minimum longitudinal distance must be equal or greater than 5 nautical miles (9,26km). This longitudinal distance corresponds to approximately 15 minutes of flight between two aircraft in the same route.

In a 4D scenario, one should regard also the temporal restriction, so the conflict can be redefined as the violation of minimum separation between aircraft in a given fragment of airspace, within a given time window [75]. Figure 6.1 illustrates the safety space that

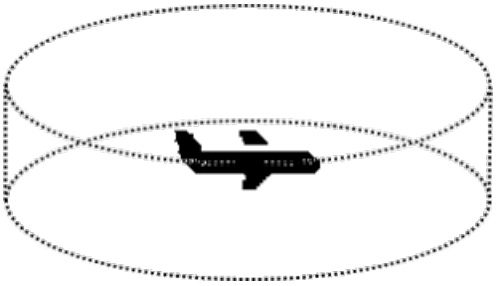

Figure 6.1: Separation area surrounding an airborne aircraft.

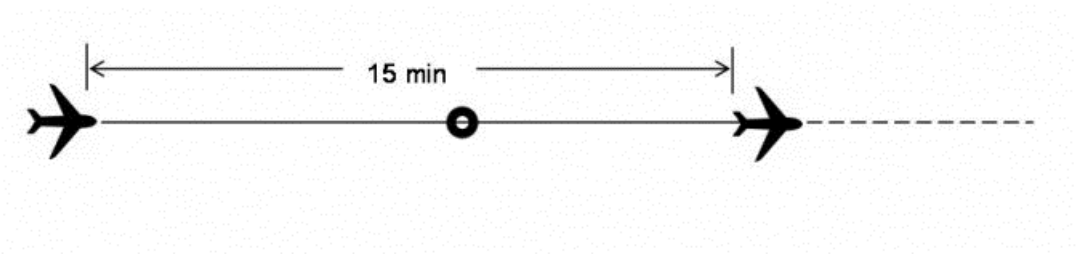

Figure 6.2: Longitudinal separation between trailing aircraft.

must be granted to each en-route aircraft and Figure 6.2 displays the minimum distance between aircraft in the same route.

Then, it is possible to identify the logical relation that defines the existence of a conflict c between aircraft A_i and A_j at an instant t, according to Equation 6.1:

$$c^{A_i,A_j}(t) \iff \Big(d_h^{A_i,A_j}(t) < S_h\Big), \Big(d_v^{A_i,A_j}(t) < S_v\Big) \tag{6.1}$$

Here, $d_h^{A_i,A_j}(t)$ is the longitudinal distance between A_i and A_j at instant t, $d_v^{A_i,A_j}(t)$ is the vertical distance between A_i and A_j at instant t, and S_h and S_v are respectively the minimum longitudinal and vertical required separations.

The longitudinal distance between points w_i and w_j is given by the Haversine formula, given by Equation 6.2. The equation computes the geodesic distance between the projections of 2D points to the Earth's sphere surface, where latitude and longitude are expressed in radians and R is the Earth's radius.

$$\begin{aligned}
&\Delta\phi = |lat_i - lat_j| \\
&\Delta\lambda = |long_i - long_j| \\
&a = sin^2\Big(\frac{\Delta\phi}{2}\Big) + cos(lat_i) \cdot cos(lat_j) \cdot sin^2\Big(\frac{\Delta\lambda}{2}\Big) \\
&c = 2 \cdot arctan2\Big(\sqrt{a}, \sqrt{1-a}\Big) \\
&d_h^{w_i,w_j} = R \cdot c.
\end{aligned} \tag{6.2}$$

6.2 System overview

Some quality standards must hold as to allow the evaluation of the proposed implementation after the simulations. These standards are enumerated as:

1. Usability: the system must be user-friendly, permitting a smooth interaction with the final user and providing adequate and intuitive mechanisms for handling input and output;
2. Safety: software engineering disciplines the aspects of integrity, availability and dependability as major concerns during the development phase of any computational system;
3. Performance: performance assessment evaluates if the answer time is adequate and proper for real-time processing. Tasks should be executed within deterministic time and the algorithmic complexity of the implemented features must be measured;
4. Scalability: this system is supposed to aggregate further functionality as the research proceeds. Also, availability in case of expansion and easiness of maintenance in case of requirement changes must be granted.

6.3 Key technologies

6.3.1 Trajectory predictor (TP)

Trajectory prediction is the process to determine the future trajectories of the aircraft through computation [76]. The TBO concept envisioned by NextGen and SESAR demand that the several decision support tools operating in the stakeholders' environments have similar performance requirements. Variation in accuracy, uncertainty, response time and input data requirements have to be common to achieve interoperability among such systems making them compliant to the TBO concept requirements.

A generic structure of a trajectory predictor must interface with the flight object and decide how to apply the given information. Four main processes are implemented in this generic structure, namely, preparation, computation, update and export [76].

The preparation process builds a behavior model from initial conditions, input state and intent information. The behavior model is the description of all execution steps that the aircraft intends to perform in order to meet the specified mission constraints. This model is described by a set of maneuvers in an ordered list of execution. The computation process uses the behavior model to generate a predicted trajectory.

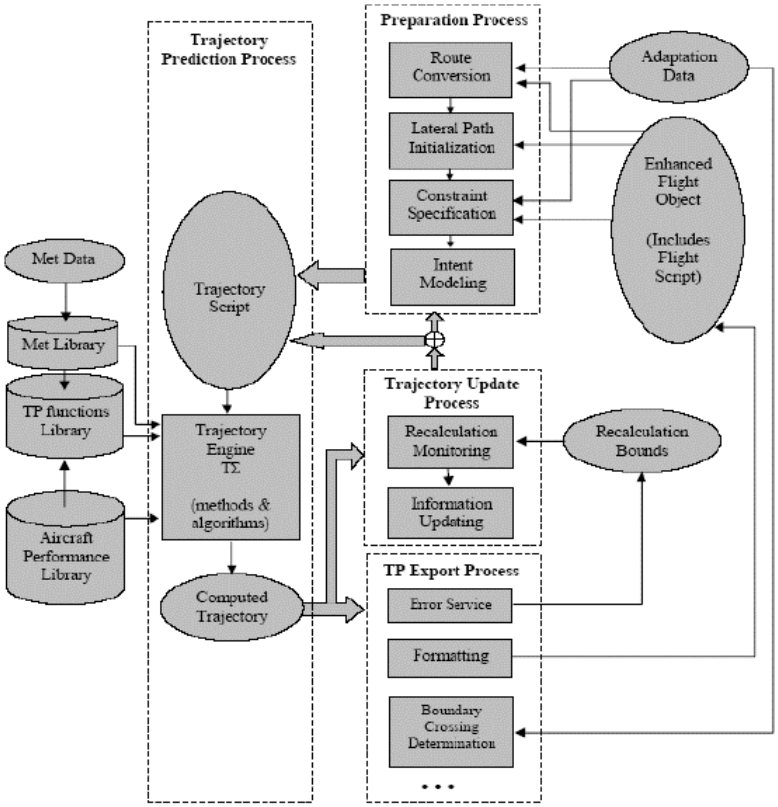

Figure 6.3: TP structure (FAA and EUROCONTROL, 2006).

The update process assesses the computed trajectory to evaluate if it is compliant with the constraints of the input flight intent. The process changes the behavior model and the flight intent if needed, so a more adequate trajectory is achieved in the re-computation process.

Finally, the export process is responsible to make the results available to the TP client (which is any system requiring support of a trajectory predictor). The output includes the predicted trajectory, warnings about availability and quality of output data, the updated behavior model (which must also be updated in the flight script) and any information about relaxed preference and constraints.

The overall trajectory predictor structure can be seen in Figure 6.3.

6.3.2 SWIM

The System Wide Information Management (SWIM) network includes standards, infrastructure, and governance enabling the management of ATM information and exchange between qualified parties via interoperable services [77]. SWIM helps to address the challenges to implement a reliable, integrated, and interoperable air-ground communication

network system regarding three aspects: air navigation service providers, governance, and technology.

The principles of SWIM include accessibility, equity, scalability, openness, standardization, governance, Service-Oriented Architecture (SOA), global applicability, and security and integrity. The accessibility allows ATM stakeholders to offer and to consume ATM information based on common service interfaces and on network connectivity, while equity means that nobody can dominate or constrain what might be offered or consumed by other stakeholders. Scalability means that new services can be added or the existing ones can asynchronously change, as well as the system can dynamically adapt to the demand workload providing adequate response time for the users. In this case, SWIM solution may depend on cloud computing, which enables resources to be dynamically provisioned or released on-demand and in any quantity.

ATM stakeholders are responsible for guaranteeing the quality, the integrity, the security, and the availability of the services based on shared interfaces and technology infrastructures. Hence, SWIM comprises openness systems supporting operational, technical, and institutional evolutions. Furthermore, it aims to minimize the usage of individually specialized interfaces and connectivity. Therefore, SWIM is a complex system supported by a dynamic evolution to reduce one-to-one connectivity.

SWIM then employs a Service-Oriented Architecture (SOA) based on standard interfaces for information, content, processes, and provision of the services. A successful SWIM implementation enables interconnectivity between the SWIM network of various air navigation service providers making use of one or more message services. SOA defines a strategy to integrate applications running across heterogeneous and distributed platforms based on industry-wide acceptable standards [7].

6.3.3 NoSQL Databases

NoSQL (Not Only SQL) databases constitute an approach developed to management of big data to be distributed among several nodes in a network [18]. Some characteristics of NoSQL databases are project simplicity, horizontal scalability and high performance and availability. However, in order to comply with these essential requirements for big data processing, NoSQL databases compromise should be made concerning some ACID properties (Atomicity, Consistency, Isolation, Durability) inherent to relational database management systems (RDBMS) [78].

The CAP Theorem demonstrates the trade-offs when selecting the desired properties of a NoSQL database [79]. It imposes a *pick-two choice* among Consistency, Availability and Partition tolerance. Figure 6.4 demonstrates the effects of feature compromise.

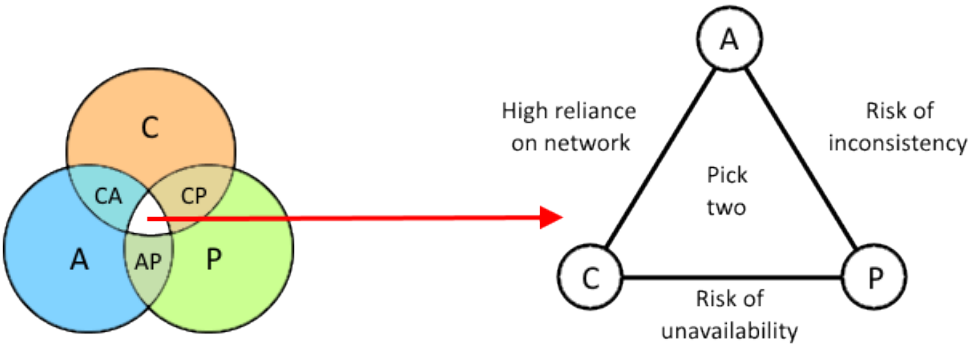

Figure 6.4: CAP Theorem in NoSQL databases.

There are some distinct storage structures for NoSQL databases, viz. Key-value, columns, graphs and documents. In the currently researched approach, Cassandra and MongoDB databases are presented. While Cassandra is a column-oriented database, MongoDB is a document-based database. The current research aims at taking advantage of the main features of such databases in order to store and retrieve all the necessary data for the intended purposes of the proposed framework.

Cassandra

A Cassandra database is designed to be a peer-to-peer management system through communication among several distributed nodes in a network [80]. Each node has the same hierarchy and workload, and each one is able to write and read data. Application data is partitioned through all nodes in the cluster and data replication is fairly accepted in this distributed data management system.

Unlike relational models, a data model in Cassandra does not define the concept of Entity-Relationship (ER). Instead, data is structured in columns allowing the classification of every detail of an item in a single row, indexed by its key. This implementation favors the performance of read and write operations when the application is intended to big data.

In a Cassandra database, data is partitioned in column families, in a similar fashion regarding ER databases. A column family is simply a collection of rows labeled by a name. Column families are logically grouped in a structure called Keyspace. Keyspaces work like an isolated scope for the names of column families. Figure 6.5 exemplifies the structure of a keyspace.

Apache Cassandra (or simply Cassandra) is a NoSQL database allowing high scalability, specifically designed for big data management. Data distribution among the nodes

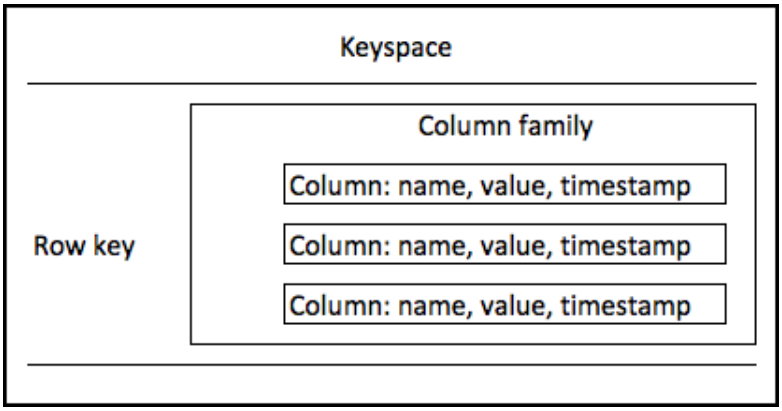

Figure 6.5: Keyspace representation in Cassandra.

of network is transparent to the administrator, provided that no programming effort is demanded in this task [80].

The nodes in a Cassandra database are distributed in a ring topology. The architecture does not present a single point of failure and data availability is granted even if a node becomes out of service. Furthermore, the processing capacity grows linearly in relation to the number of nodes, that can be added or removed online. Data replication is configured according to the application needs.

As any NoSQL database, Cassandra does not support ACID transactions. The standard behavior does not grant data consistency in all replicas, i.e., if at least one node owns a given data, the transaction is regarded as completed [18]. This happens to diminish read/write time, as these are extremely efficient in this architecture.

Common commands in relational data models like joins and subqueries are not allowed in Cassandra. For this reason, SQL language is not used in Cassandra queries. Instead, CQL (Cassandra Query Language) is defined as the query language. Furthermore, the framework limits the reference to *where* clauses to the columns part of the primary key of the corresponding column family [80].

MongoDB

MongoDB is a schema-free document store database where documents are grouped into collections. The documents are stored in BSON (Binary JSON) format and in order to increase performance, data files are mapped in memory. By default, data is sent to disk every 60 seconds and when new files are created, everything is flushed to the disk, releasing memory. MongoDB uses indexes on collections in a similar fashion as relational databases and supports map-reduce for complex aggregations across documents to increase performance [81]. Each document is identified by a *_id* field and an unique index is created over this field.

Apart from automatic index creation on *_id* field, additional indexes can be created by the database administrator. For example, an index can be defined over a field within a specific collection to work with a specific key. This feature of MongoDB is called *compound index*. However, all indexes use the same B-tree structure, and each query uses only one index chosen by the query optimizer mechanism, giving preference to the most efficient index [82].

The choice for MongoDB was based on its performance when compared to other relational and NoSQL databases. Many studies show that MongoDB performs better when compared to SQL databases and other types of NoSQL databases [81]. Depending on the size of the collections and the modeling adopted, MongoDB can achieve the best performance in read operations when compared to other databases optimized for big data [83].

MongoDB uses a scale-out scheme, which is flexible against hardware expansion, and supports auto-sharding, which is the MongoDB configuration of horizontal data partitioning and scalability. Thus, the automatic distribution of data over a number of servers can be conveniently carried out. Replica sets ensure high availability, safety, and data consistency. They also enable distributed expansion for data processing involving large amounts of data [78].

The main functionalities of the MongoDB components are as follows:

- *mongod*: handles data requests, manages data access and performs background management operations;
- *mongos*: processes queries from the application layer and determines the location of these data in the shared cluster;
- *shard*: stores data. For easy availability and data consistency, each shard is a replica set;
- *config server*: stores the cluster's metadata. These data contain a mapping of the cluster's dataset to the shards;
- *replica sets*: a group of mongod processes that maintain the same dataset. If the primary mongod is unavailable, the replica set will elect a new primary mongod.

Even distribution of data among shards is controlled by a shard key. A shard key is either a single indexed field or an indexed compound field that exists in every document. Data in MongoDB is split into chunks, which are logical units of stored data, according to the shard key by using either range-based partitioning or hash-based partitioning. Hence, appropriate shard key selection is a very important decision factor in the MongoDB design [78].

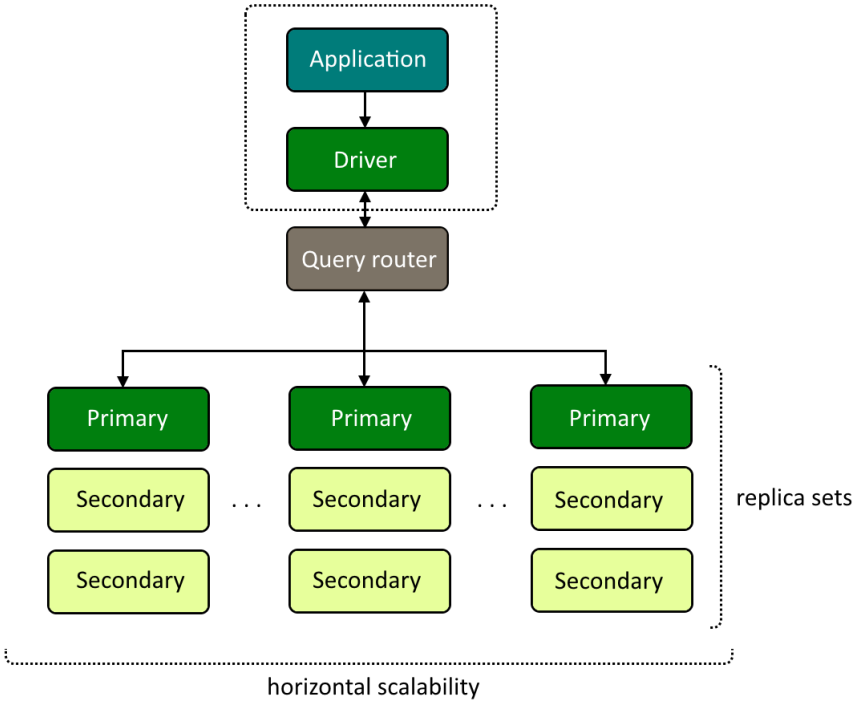

Figure 6.6: Interaction between MongoDB components.

6.4 Proposed methodology for Conflict Detection

The framework developed for conflict detection and resolution consists of a client for a trajectory predictor, NoSQL databases for trajectory storage and a local search module for conflict resolution.

This section describes how these entities are implemented and how they are connected.

6.4.1 TP Client

The TP Client function is the interface among the TP and the higher level services in the ATM system. The interface between TP Client and TP very often presents implementation issues, e.g. the TP may or may not be embedded within the TP Client function. Hence the interface definition can be specific to the system developer. The definition of unambiguous TP performance requirements is key to ensure that the TP client will be capable of meeting its own higher level performance requirements [84].

The Trajectory Predictor developed by Portas and Frontera [85] will be used in this research[1]. This tool, named simply as TP, receives as input initial conditions specifying the state of the aircraft at the start of the predicted trajectory. The initial conditions are provided to the TP as a XML file containing a series of magnitudes. For each mag-

[1]This tool was kindly provided by Boeing Research and Technology - Madrid.

nitude, the user must provide its name, a scalar value, and the unit used to quantify the magnitude.

Aircraft intent and flight intent are provided to TP using AIDL and FIDL respectively, modeled in XML input files. Weather model contains wind, temperature and pressure information are also provided by a XML file. The initial conditions of TP have a reference time matching the time provided in the weather model.

The aircraft performance model uses the BADA files to provide sufficient performance information to the trajectory prediction. BADA stands for Base of Aircraft Data providing an aircraft performance model (APM) based on a kinetic approach to aircraft performance modeling [2].

The TP uses a settings file with a set of parameters that model the behavior of TP. These parameters only affect the TP if a trajectory is predicted from the flight intent. The optimization process is also parameterized in this file.

As output, the TP yields the predicted trajectory of the aircraft. This trajectory is delivered to the user in the format of a sequence of points in the aircraft path. A KML file can be exported if the user desires to view the predicted trajectory in Google Earth interface. These samples indicate the position of aircraft in respect to its latitude, longitude and altitude, plus the time relative to the time provided in the initial conditions. The resulting samples can include attitude, velocity, accelerations, mass, local winds, and other magnitudes.

When provided a flight intent, the TP computes also the aircraft intent corresponding to the predicted trajectory. This flight intent is specified using AIDL. The Trajectory Predictor is able to calculate trajectories described in this language and as result, TP yields the predicted 4D trajectory for each aircraft.

This trajectory is presented as a sequence of waypoints that form the flight path for the aircraft, with respect to the latitude, longitude, altitude and estimated time. Furthermore, every waypoint has inherent attributes viz. speed, acceleration and current mass, among others. The output files are the aircraft intent and the sampled trajectory, given in a detailed XML file, and simplified in a KML file, in which the waypoint coordinates are featured in order to be displayed in some appropriate tool like Google Maps. Figure 6.7 is an example of a predicted trajectory for a flight between São Paulo and Rio de Janeiro, Brazil.

Thus, the sample point attributes (x, y, h, t) of a predicted trajectory can be used as input to any conflict detection tool. The searched implementation demands the aircraft

[2]EUROCONTROL. *Base of Aircraft Data (BADA).* [Online]. Available: https://www.eurocontrol.int/services/bada. [retrieved: Apr., 2017]

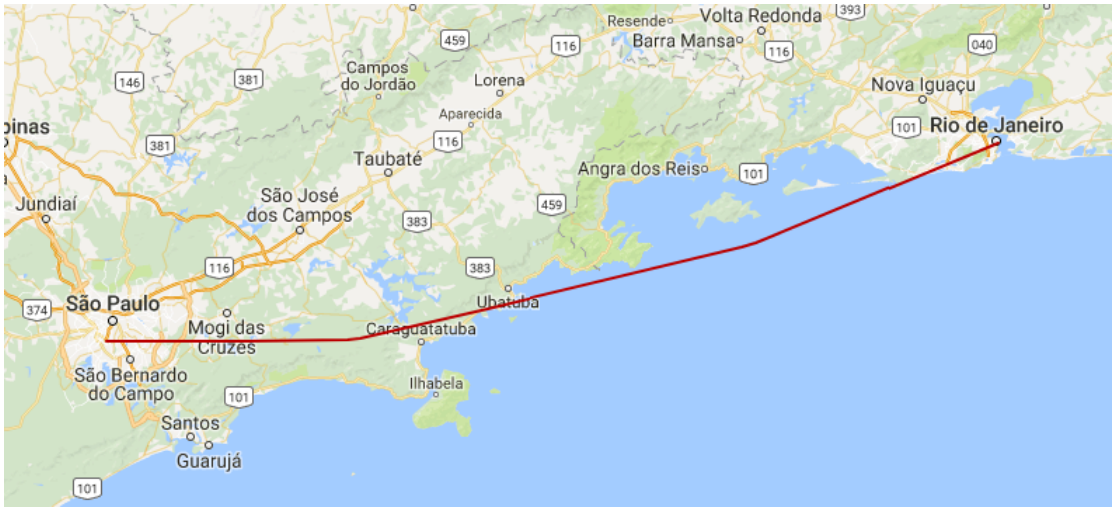

Figure 6.7: Flight path of an aircraft departing from Santos Dumont to Congonhas airport.

trajectories to be stored in specific column families or BSON documents being indexed by a time constraint (a time interval describing instantaneous airspace occupation).

6.4.2 Cassandra Database

A very important conceptual difference between Cassandra and traditional RDBMS models is the way in which data is described. In a relational database, schemas rely on tables representing particular entities. The entities relate to each other through foreign key sharing. In Cassandra, this level of relationship does not exist. The model relies on column families, whose primary keys are divided in two. The first part of the key is called the partition key, and the second part is called cluster key [86].

The partition key, intuitively, defines the partition where the data will be based. The chosen node in the network is done by a hash function (Consistent Hashing). The cluster key is conceptually closer to the common definition of a primary key in a relational database, provided that it univocally identifies a row within a partition. Therefore, the choice of good primary keys is paramount for an adequate modeling.

The row structure classifies every detail of a specific item within a unique row. Particularly, Cassandra allows up to 2 billion registries (columns) per line. Furthermore, a row in a column family can be properly indexed by its key.

In a Cassandra database, the quality of the modeling process resides in the fact that not only the stored information must be modeled, but also how this information can be retrieved. A proper modeling in Cassandra must consider the most probable queries for the persisted data.

The flexible design structure of a Cassandra database makes it fit to properly model a functionality for conflict detection in 4D trajectories. Data can have different structuring

levels and both primary and secondary indexation is supported, meaning that the column tags can be also modeled to store data. Columns contain a name, a value and a timestamp, and the collection of columns can be tagged by their names.

A Cassandra database is compliant to the SWIM paradigm, since it can be used to massive trajectory storage, weather models and aircraft data in any detail level. In order to implement the functionality of conflict detection in 4D trajectories using a Cassandra database, one should define the data type to be retrieved. Relationships among the entities are not possible through foreign keys, thus every distinct relationship depends on the implementation of a distinct column family. This might result in data replication, but in Cassandra, it is actually regarded as a feature instead of a drawback.

Probable queries desired in this implementation should be:

- Flight plans of aircraft;
- Trajectory of aircraft;
- Points in a trajectory;
- Points occupied at time t;
- Trajectory conflicts;

6.4.3 MongoDB Database

In order to obtain a better performance to read data, since the conflict detection algorithm needs to manipulate a large amount of data, each information domain was modeled as a distinct collection of documents, and each document in the collection represents the information, corresponding to a record in an SQL table. For the complete CDR process the following collections were modeled:

- *waypoints*: contains the fixed points of the routes;
- *airports*: domain information about airports;
- *runways*: stores information about the runways of each airport;
- *sid*: collection that gathers the documents describing the take-off procedures;
- *flightPlan*: contains documents that represent the intended flight of the aircraft;
- *acTrajectory*: stores the points of the trajectories computed by the trajectory predictor, ordered by the timestamp;

- *pointsByTime*: similar to the *acTrajectory* collection, this is a set of collections that organizes the documents according to the altitude of each point, a fundamental feature to decrease the search space of the conflict detection algorithm;
- *conflicts*: stores the detected conflicts.

The shard key selection is similar to the partition keys selected to the Cassandra database.

6.4.4 4D Trajectory storage proposal

Trajectory Based Operations (TBO) aims to integrate an aircraft's navigation capability in space and time to improve efficiency and predictability in the Air Traffic Management (ATM) system [36]. This objective requires mainly four key components in order to hold:

- stored reference trajectories;
- a continuously recomputed capture trajectory;
- electronic situation displays;
- a control system to follow the overall trajectory in space and time.

Characteristics of the historical flight data and abstracted patterns of trajectories can be analyzed through usage of data mining algorithms [46]. Therefore, some database structure must be conceived in order to handle aircraft trajectories and perform any eventual operation based on the historical trajectory data. NoSQL databases Cassandra and MongoDB are conceived for this task.

A NoSQL Cassandra keyspace was created to accomodate data about the information above. For prototyping purposes, the whole database is kept in a single node, but it should easily be adapted to a traditional cluster topology. This means that the proposed system abides to Consistency and Availability properties in the CAP triangle.

MongoDB is schema-free, therefore there is no need to a pre-conception of database schemas [78]. The BSON documents are created on demand and the objects do not need to have the same structure or fields. Therefore, the common fields do not need to have the same type, thus allowing a flexible schema storage [83].

The first entity is created in order to hold the data about the flight intent of the aircraft. The flight plans are parsed and inserted into the table *flightplan*, whose creation script is shown Figure 6.8. For Cassandra, the partition key for this table is the departure airport and the clustering key is the flight number. An index was created on the flight number and in the strategy attributes, which allows the fetch requests to be performed

```
CREATE TABLE IF NOT EXISTS flightplan (
    adep text,
    flightnum text,
    strategy int,
    actype text,
    dest text,
    opdays text,
    eobt time,
    eet time,
    velocity int,
    flightlevel int,
    route text,
    obs text,
    updated boolean,
    cost double,
    PRIMARY KEY (adep, flightnum, strategy));
CREATE INDEX ON flightplan( flightnum );
CREATE INDEX ON flightplan( strategy );
```

(a) Cassandra


```
{
   "flightNum":"AZU2518",
   "adep":"SBSV",
   "strategy":"0",
   "acType":"E190/M",
   "cost":8221.882141,
   "dest":"SBKP",
   "eet":"02:14:00.000000000",
   "eobt":"20:05:00.000000000",
   "flightLevel":320,
   "obs":"EQPT/SDFGHIRWY PBN/A1B1D1O1S2 EET/SBBS0107",
   "opDays":"7",
   "route":"[SBSV-DCT-BLOCK-UZ10-PUBAV-UZ16-POSMU-DCT-CNF-DCT-XOMOD-UZ30-ENTIT-DCT-SBKP]",
   "updated":false,
   "velocity":443
}
```


(b) MongoDB

Figure 6.8: Creation of entity *flightplan*.

only by adding the column *flightnum* or *strategy* as parameters to the WHERE clause. Otherwise, the departure airport should also be specified, due to the fact that one cannot query for cluster keys without constraining the query by the partition key.

The Trajectory Predictor application is able to calculate trajectories described in AIDL and generate a KML file as output. This file uses XML syntax to represent each sample point in a 4D trajectory. Thus, the sample point attributes (x, y, h, t) of a predicted trajectory can be used as input to any conflict detection tool. It is important to create a table from which the aircraft trajectory can be fetched. This entity is called *actrajectory* and its conception is shown in Figure 6.9. The partition key is formed by the flight number and the trajectory strategy. A clustering order on the column time is applied so that the

```
CREATE TABLE IF NOT EXISTS actrajectory (
    flightnum text,
    strategy int,
    time time,
    altitude double,
    latitude double,
    longitude double,
    PRIMARY KEY ((flightnum, strategy), time))
    WITH CLUSTERING ORDER BY (time ASC);
```

(a) Cassandra


```
{
   "flightNum":"ONE6060",
   "strategy":"4",
   "time":"12:00:41.000000000",
   "altitude":30.50076,
   "localization":{
      "coordinates":[
         -47.91962,
         -15.88796
      ],
      "type":"Point"
   }
}
```


(b) MongoDB

Figure 6.9: Creation of entity *actrajectory*.

trajectory waypoints are ordered by their time in ascending order.

The entities depicted in Figure 6.9 can be queried to bring the whole set of (ordered) waypoints that form some aircraft's trajectory, but it is ineffective if one desires to fetch the instant airspace occupation at a given time. This is why the entity *pointsbytime* is created. In Cassandra, this table has as partition key the composed key (hour, minute), which represents a timestamp with precision in minutes, and the clustering keys are the the altitude and the time with precision in seconds. Thus, the table stores every sampled point of each aircraft trajectory, partitioned by their timestamp. This column family is shown in Figure 6.10(a) and aggregates the airspace occupation during a given period, e.g. a whole operation day.

For MongoDB, a different collection is dynamically created for every minute of simulation. The collection name is set to *h[HH]m[MM]*, where *HH* gives the current hour and *MM* gives the current minute. Figure 6.10(b) is an example of an entry retrieved from collection *h21m00*.

This entity is paramount for the proposed conflict detection algorithm because the model design grants that every waypoint is ordered by the given set of keys: in Cassandra, by a specific partition key, and in MongoDB, by a dynamic collection conveniently labeled. Therefore, the evaluation for conflict detection between two waypoints can be done by comparing constrained sets of neighboring waypoint entries in the fetch result (see Section 6.4.5 for more detailed description). Figure 6.10 shows how this entity is created.

Some further auxiliary entities are also straightforwardly modeled, as follows:

- *waypoints* just holds the fix waypoints in the predefined routes. Only the partition key is defined, so that a fix point can be fetched by its name;

```
CREATE TABLE IF NOT EXISTS pointsbytime (
    hour int,
    minute int,
    time time,
    altitude float,
    latitude float,
    longitude float,
    flightnum text,
    strategy int,
    PRIMARY KEY ((hour, minute), altitude, time));
CREATE INDEX PBT_IDX ON pointsbytime( flightnum );
```

(a) Cassandra


```
{
  "seconds":24,
  "hour":21,
  "minute":0,
  "altitude":30.48472,
  "time":"21:00:24.000000000",
  "flightNum":"TAM3727",
  "localization":{
    "coordinates":[
      -47.91965,
      -15.88824
    ],
    "type":"Point"
  },
  "strategy":"0"
}
```


(b) MongoDB

Figure 6.10: Creation of entity *pointsbytime*.

- *airports*, which holds important information concerning the airport location. The primary key is the IATA identifier of each airport;
- *runways* store the properties of the runways present in the simulated airports;
- *sids* manages the Standard Instrumented Departure (SID) routes for every airport. This entity stores a collection of points that constitute the departure routes. The partition key is the origin airport name and the cluster key is a sequential number that indexes the position of every waypoint in the departure route. For simulation purposes, each airport has only one SID route;
- *conflicts* stores conflicting waypoints, as well as the number of the subject flight number and the conflicting aircraft number. More than the 4D attributes that define the conflict, further information is added to the conflict parameters, which will be used as variables to define the heuristics for the conflict resolution procedure in a future step of this research.

The last entities are created in order to store information about the pairs of summarized information about the detected conflicts. The partition key is the tuple *(flightnum, interceptorid)* and the cluster keys are the strategies executed by the first and the second flight, respectively. Further information are the costs of the performed trajectories. The aircraft IDs are ordered within every entry in the table. This ordering has no direct influence in the efficiency of data insertion, but is a convenience for further steps in the resolution algorithm. The creation script for this table is shown in Figure 6.11.

```
CREATE TABLE IF NOT EXISTS conflictpairs (
    flightnum text,
    interceptorid text,
    strategy1st int,
    strategy2nd int,
    cost1st double,
    cost2nd double,
    PRIMARY KEY ((flightnum, interceptorid),
        strategy1st, strategy2nd
    ));
CREATE INDEX ON conflictpairs( interceptorid );
```

(a) Cassandra


```
{
  "flightNum":"GLO1461",
  "interceptorId":"TAM4537",
  "strategy1st":"1",
  "strategy2nd":"1",
  "cost1st":4100.97481,
  "cost2nd":4414.68067
}
```


(b) MongoDB

Figure 6.11: Creation of entity *conflictpairs*.

6.4.5 Conflict detection algorithm

After trajectory prediction from TP, the client application collects the trajectory output and inserts the corresponding data into the databases. The overall flight costs are also updated for each calculated trajectory.

The most important structures to be filled in the databases are the ones that handle waypoints information. For conflict detection, a system parameter indicates if the detection should be performed to a constrained time window of one hour, or to the entire day of operation. At first, the total set of waypoints within the given time window is recovered from the databases.

Now, the database models are paramount to guarantee the efficiency of the detection algorithm. A premise is that the full set of waypoints is conveniently ordered within the database, therefore they can be queued according to the following parameters, in this specific order:

1. timestamp;
2. performed altitude.

Within the recovered waypoint queue, every waypoint w_i is checked against its next neighbors in order to evaluate the conflict constraints (time, vertical and longitudinal separations), until a waypoint w_j is found to grant vertical separation. Two advantages come from evaluating the vertical separation in the first step: first, horizontal separation is a complex calculation that involves geodesic distances in two dimensions, and this calculation can be ignored if the vertical distance is already ensured. Second, if the property of vertical separation holds for w_j, there is no need to evaluate the further waypoints $w_k, k > j$, because all of them hold the vertical separation with w_i as well.

In order to make the queue traversal even more efficient, one should realize that it is possible to appear a sequence of waypoints $w_i, ..., w_j \in W$ related to the same flight. Such

points cannot be compared against themselves, so a *jump* index is attributed to w_j every time a sequence is found. Then, in the subsequent iterations, waypoints $w_k, i < k \leq j$ are never compared to each other.

Algorithm 2 summarizes this innovative conflict detection technique that explores the NoSQL database features to conveniently store and retrieve trajectory information.

6.4.6 Complexity analysis

To demonstrate that the proposed conflict detection algorithm is a candidate to outperform most of state-of-art algorithms, a complexity analysis must be presented concerning the execution time.

Worst case

The worst-case running time of the algorithm is the upper bound on the running time without prior concern about the input characteristics [66]. The worst case in Algorithm 2 would happen when every waypoint should be compared to each other in a pairwise fashion. Notwithstanding, the usage of the proposed databases delivers a set of waypoints that follows a specific ordering, firstly by timestamp, then by altitude. Hence, the traditional pairwise evaluation is now reduced to a queued evaluation in which the waypoints are only checked against the ones ahead of it in the queue.

An example of worst case scenario is the one in which all waypoints are sampled at the same moment in time, provided that waypoints $w_1, .., w_n \in W$ do not hold vertical separation. In this hypothetical situation (practically impossible), the number of evaluations in the waypoint data set is given by Equation 6.3.

$$
\begin{aligned}
&= n - 1 + n - 2 + ... + n - (n - 1) + 0 \\
&= n - 1 + n - 2 + ... + 1 \\
&= \sum_{k=1}^{n-1} k \\
&= n\Big(\frac{n-1}{2}\Big) \\
&< \frac{n^2}{2} \\
&\in O(n^2).
\end{aligned}
\tag{6.3}
$$

Figure 6.12 illustrates how the iteration of waypoint comparisons are executed in a full queue traversal.

Algorithm 2: Conflict detection

Data: a given time window
Result: set of conflicts

```
1  tw ← timeWindow;
2  sh ← horizontalDistance;
3  wpList ← selectPointsByTime(timeWindow);
4  jump ← 1;
5  canJump ← true;
6  for i from 0 to (wpList.size() − 1) do
7      wp1 ← wpList.get(i);
8      if i ≥ jump then
9          canJump ← true;
10         jump ← jump + 1;
11     end
12     for j from jump to wpList.size() do
13         wp2 ← wpList.get(j);
14         flight1 ← wp1.flight;
15         flight2 ← wp2.flight;
16         if flight1 = flight2 then
17             if canJump = true then
18                 jump ← jump + 1;
19             end
20             continue;
21         end
22         else
23             canJump ← false;
24         end
25         if verticalDistance(wp1, wp2) ≥ minVerticalDistance or
            timeSeparation(wp1, wp2) ≥ minTimeSeparation then
26             break;
27         end
28         if horizontalDistance(wp1, wp2) < minHorizontalDistance then
29             database.insertConflict(flight1, flight2);
30         end
31     end
32 end
```

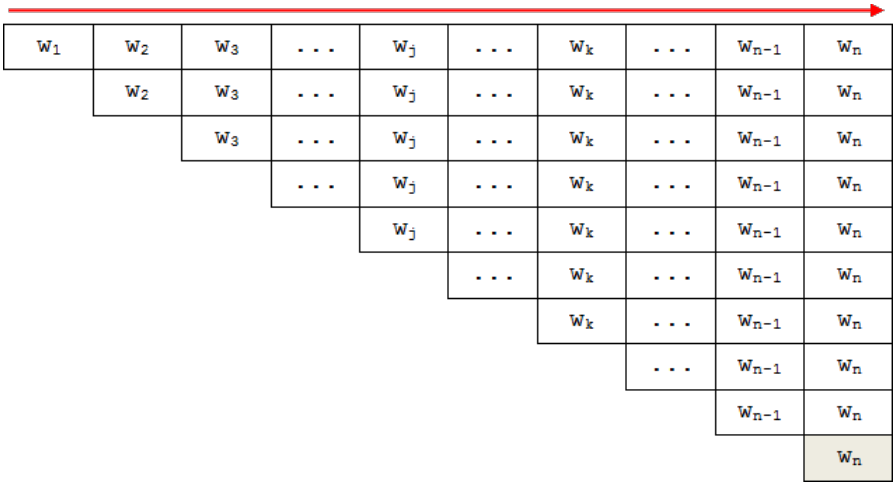

Figure 6.12: Waypoint queue traversal scheme.

Average case

The definition of what could be the average case for the detection algorithm can be somewhat cumbersome. Firstly, one should identify *what* should be the average input, i.e., what would be expected to happen in a typical case for the problem.

A probabilistic analysis would be suitable for this case, therefore we must have previous knowledge about the distribution of the inputs. A known fact is that the first check-up to determine the existence of a conflict is the vertical separation, and the queue of waypoints is ordered by altitude. Then, we must analyse the maximum amount of comparisons that will be executed for every waypoint.

Indicator random variables can be used to indicate a vertical violation, as given by Equation 6.4. Equation 6.5 analyses the set of indicator random variables.

$$\begin{aligned} X_i &= I\{\text{vertical violation between } w_i \text{ and } w_j\} \\ &= \begin{cases} 1, \text{if } w_i \text{ and } w_j \text{ violate vertical constraint,} \\ 0, \text{otherwise} \end{cases} \end{aligned} \tag{6.4}$$

$$X = X_1 + X_2 + ... + X_n = \sum_{i=1}^{n} X_i \tag{6.5}$$

The expected value of an indicator random variable associated with an event A is equal to the probability that A occurs [66]. Equation 6.6 represents the expected value of a vertical violation to occur between two different waypoints, where $x \in 0, 1 = X$, 0 meaning *separation ok* and 1 meaning *separation violated*. Let us assume that the

probability of a waypoint w_i violates the vertical separation with waypoint w_j is 50%:

$$E[X] = \sum_x xPr\{X = x\} = (0).\frac{1}{2} + (1).\frac{1}{2} = \frac{1}{2} \tag{6.6}$$

Then, for every $w_i \in W$, vertical separation is expected to be violated for half of the waypoints $w_j \in W, i \neq j$. Therefore, the queue would be evaluated for every waypoint up to $w_k, i < k \leq \frac{n}{2} + i$. In such case, the total number of comparisons is given by Equation 6.7, which holds for every $n > 2$:

$$= k\Big(\frac{k+1}{2}\Big) = \frac{n}{2}\Big(\frac{\frac{n}{2}+1}{2}\Big) = \frac{n}{2}\Big(\frac{n+2}{2}\Big) = \frac{n^2+2n}{4} \in \Theta(n^2). \tag{6.7}$$

Although the proposed assumptions yield a growth order bound to n^2, one should have in mind that the range of possible altitudes for a flight is far wide (for instance, the set of flight plans selected for the simulations ranges up to FL420, or 42000ft). This means that assuming a probability of 50% for vertical separations is extremely conservative and demonstrates the difficulty in finding a typical scenario, provided that the typical violation of vertical separation is defined for a distance less than 1000ft (see Section 6.1.)

Let us then rephrase the problem by equally dividing the ranges of altitudes among the n waypoints. Suppose that the waypoints are equally distributed by $log_2(n)$ altitude ranges. Note that this estimation is very realistic for a typical database of 10^6 waypoints, describing about 20 altitude ranges of about 2100ft interval. Furthermore, the initial probability of 50% for vertical separations holds for each range.

By this realistic assumption, the queue would be evaluated for every waypoint up to $w_k, i < k \leq log_2(n) + i$. Hence, a maximum of $log_2(n)$ comparisons will take place for every $i = 1, ..., n$. Therefore, the time complexity for the average case of Algorithm 2 is given by Equation 6.8.

$$g(n) = \sum_{i=1}^{n-1} log_2(n) = (n-1)log_2(n) \in \Theta(nlog_2(n)). \tag{6.8}$$

Best case

The best case scenario happens when all waypoints hold pairwise separation concerning time or altitude. In this situation, horizontal distance would never be checked, because the separation constraints that ensure logical ordering were already successfully evaluated. Then, every waypoint would be compared only once with its immediate neighbor (in the

queue), in a maximum of $n-1 < n$ comparisons. The best case complexity is given by Equation 6.9.

$$g(n) = \sum_{i=1}^{n-1} n = n - 1 \in \Omega(n)). \tag{6.9}$$

6.5 Conflict resolution

If conflicts are detected, the conflict resolution algorithm is activated. The ATC must search a solution in which the conflicts are completely solved, but at the smallest cost as possible. Obviously, changes in the intended flight paths or insertion of delays will invariably increase the cost for some aircraft, therefore increasing the total cost to the aircraft scenario. However, the cost distribution must be fair among the affected aircraft.

6.5.1 Simulated annealing

The resolution algorithm consists of an adaptation of the Simulated Annealing (SA) algorithm. When the SA optimizer is started, the conflicts are coded into simulation cells, which hold information about the flight parameters. The total energy is conceived as the total cost for the conflict set. The algorithm iteratively adjusts the flight parameters of conflicting flights in order to reduce the system energy to the lowest, stable state, which means to resolve the scheduling scenario to the less expensive configuration in terms of aggregated cost. The basic algorithm can be viewed in Algorithm 3.

The neighboring solution is selected by randomly picking one of the conflicting aircraft and adjusting the flight parameters so that the given trajectory is changed. If the selected solution has a lower energy state than the instantaneous best solution, then the current solution is updated. Else, the current solution is updated with probability given in Equation 6.10, where k is called the Boltzmann constant and T is the current temperature. This is done to avoid the system to be stuck in sub-optimal results. Eventually, this probability is low enough to guide the solution to an immutable state, given the transition constraints. Once the annealing result is ready, the yielded solution is applied to the aircraft set.

$$P(\delta E) = exp\Big(\frac{-\delta E}{kT}\Big) \tag{6.10}$$

6.5.2 Client application

A client application was implemented in Java. This application is responsible to parse the flight plans, the known points along the route, the airport data and consolidate these

Algorithm 3: Simulated Annealing algorithm for aircraft conflict resolution

```
Data: coalition
Result: solution
currentSolution ← coalition.allocation();
currentEnergy ← calculateEnergy(currentSolution);
minT ← 1;
while true do
    for T = initTemp() → minT do
        solution ← selectNeighbor(currentSolution);
        energy ← calculateEnergy(solution);
        ΔE ← energy − currentEnergy;
        if ΔE < 0 then
            currentSolution ← solution;
            currentEnergy ← energy;
        end
        else
            rand ← random(0, 1);
            if rand ≥ probability(ΔE) then
                currentSolution ← solution;
                currentEnergy ← energy;
            end
        end
        updateTemperature();
        if conflicts = 0 then
            break;
        end
    end
    if conflicts = 0 then
        break;
    end
    reset();
end
return currentSolution;
```

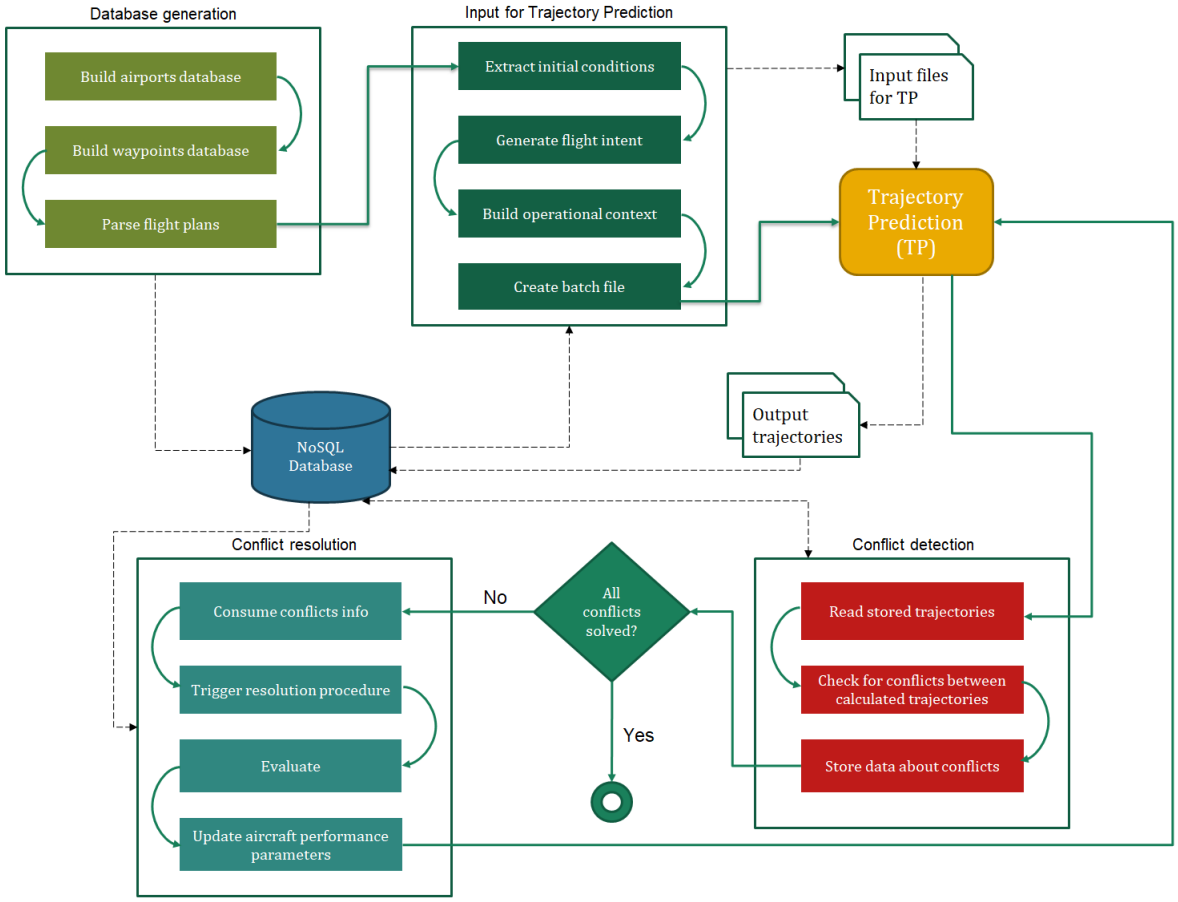

Figure 6.13: Sequence diagram for conflict detection and resolution.

data in a NoSQL database. The complete workflow for the proposed conflict detection and resolution is shown in Figure 6.13.

From the database, flight information is extracted in order to generate the input files for TP, and a shell command file is generated with the appropriate commands to be executed by each aircraft subject to analysis.

The application is divided into the following distinct packages:

- *db_cassandra*: classes that manage the connection to the Cassandra database and the interactions with it;
- *db_mongo*: classes that manage the connection to the MongoDB database and the interactions with it;
- *data*: classes developed for data treatment, such as parsing input files and other operations of reading and writing to disk;
- *entity*: classes that represent the objects to be managed by the database;

- *settings*: classes that manage the configuration of the execution behavior and user preferences for the application;
- *tp*: classes that generate the input files for TP and read access the output trajectories;
- *resolution*: classes that manage the resolution procedures;
- *util*: global helpers.

Chapter 7

Simulations on strategic conflict detection

This chapter presents the scenario selection to simulate the trajectories to be developed and the implementation of the conflict detection and resolution procedures.

Section 7.1 describes the scenario used in the case study and section 7.2 presents the results obtained in after the simulations.

7.1 Simulation scenario

The simulation counts with nine selected airports in Brazil, namely:

- SBBR: Brasília-DF;
- SBGR: Guarulhos-SP;
- SBSP: Congonhas (São Paulo-SP);
- SBKP: Campinas-SP;
- SBGL: Galeão-RJ;
- SBRJ: Santos Dumont-RJ;
- SBCF: Confins-MG;
- SBSV: Salvador-BA;
- SBPA: Porto Alegre-RS.

The choice for these airports was motivated by the fact that they insert the highest workload in the national Air Traffic Flow Management. In fact, in 2016 these nine airports

Figure 7.1: Selected airports for simulation.

responded for 56% of the national air traffic [87]. The distribution of these airports through the country is show in Figure 7.1.

The repeated flight database was collected from the freely accessible site of the Brazilian Air Navigation Management Center (CGNA) and comprehends the valid flights to June 2017. Among them, 1221 subject flights were filtered for simulation, provided that the flights have as origin and destination airports among the set of selected airports.

After running the trajectory prediction module, it was possible to have a comprehensive overview of the scenario occupation. Figure 7.2 depicts the trajectory network found. The selected airports can also be identified.

Figure 7.3 depicts the overall airspace occupation by time, i.e., how many waypoints are being occupied at each moment. The presented graphic groups the discovered waypoints per each hour of the day.

The computing resource used for the experiments is a HP Z220CMT BR Workstation equipped with a Intel® Xeon® CPU E3-1270 V2 3.50GHz, 32GB DDR3 SDRAM 800MHz and Microsoft Windows 7 Professional 64-bit as operating system.

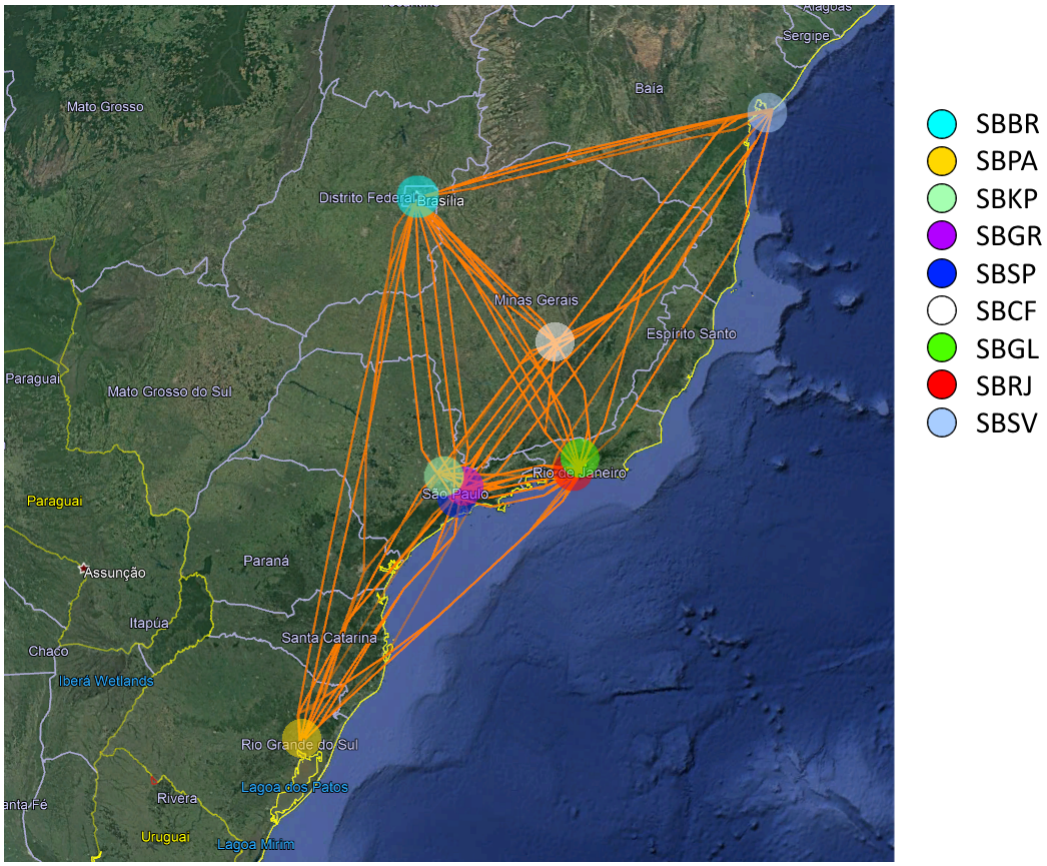

Figure 7.2: Network of developed trajectories.

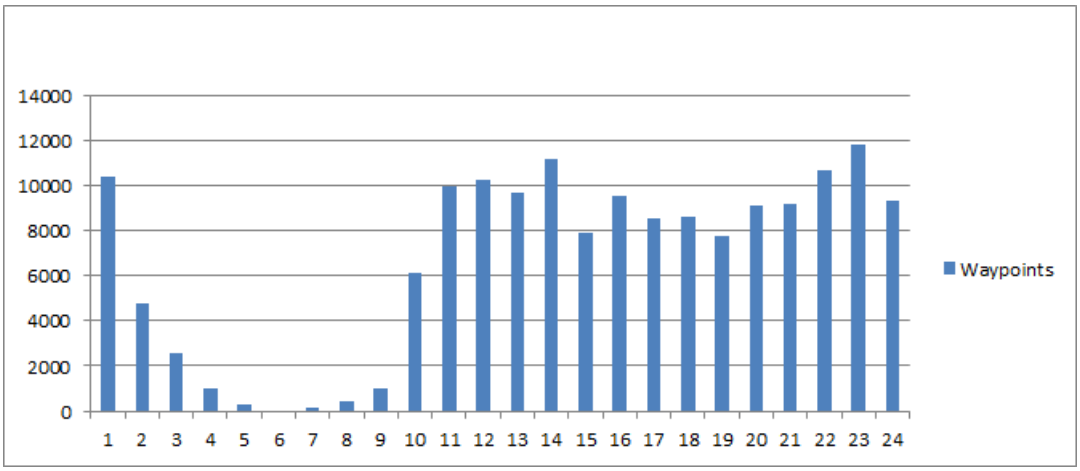

Figure 7.3: Airspace occupation by time.

7.2 Conflict detection

Once the database is fully populated, all flight plans (RPL) are fetched and submitted to the engine that generates the input files for TP. Then, the batch script executes the trajectory prediction for each flight. The client application accesses the output trajectories and inserts every sampled waypoint into the database. Finally, the conflict detection procedure is triggered and the conflicts found are inserted into the database as well.

The waypoint database is then queried by time window and the points occupying the airspace in the given moment of time are returned. For simulation purposes, three conflict detection procedures were performed:

- *Default conflicts*: the conflicts naturally found if the flight plans are performed as originally defined by the RPL;
- *Strategic conflicts*: the conflicts found among all original trajectories plus the alternate trajectories;
- *Time-window conflicts*: the conflicts are evaluated within a specific look-ahead time (one hour);

7.2.1 Performance evaluation

The conflict detection procedure was performed both in serial and parallel execution and the most efficient method was selected to provide input for the conflict resolution procedure.

Table 7.1 presents the performance of conflict detection using the default trajectories only. The execution was performed using up to 2^n threads, where $0 \leq n \leq 5$.

Similarly, the strategic approach was also evaluated. Table 7.2 show the performance of conflict detection using the combination of all possible trajectories to be performed by the aircraft, with different combinations of flight plan executions.

Figure 7.4 shows an example of a conflict between two flights. The trajectory represented by an orange path belongs to a flight departing from Galeão to Congonhas, whilst the trajectory represented by a teal line is performed by a flight departing from Santos Dumont to Congonhas. Although their distinct cruise levels (FL340 and FL300 respectively) grant the safe separation for most part of the flight, the separation loss is detected when the aircraft are about to begin the descent procedure.

It is important to clarify that the proposed conflict prediction procedure comprises a whole day of operations, thus every scheduled flight is evaluated. For each flight, four more strategies are provided, meaning that every aircraft has five different alternate trajectories to be performed in case of conflict. This justifies the large amount of conflicts

Table 7.1: Performance of conflict detection among default trajectories.

			Cassandra	**MongoDB**
Threads	Waypoints	Conflicts	Time (s)	Time (s)
1	160022	330	0,936	0,518
2	160022	330	0,516	0,269
4	160022	330	0,331	0,2
8	160022	330	0,286	0,159
16	160022	330	0,298	0,19
32	160022	330	0,25	0,207

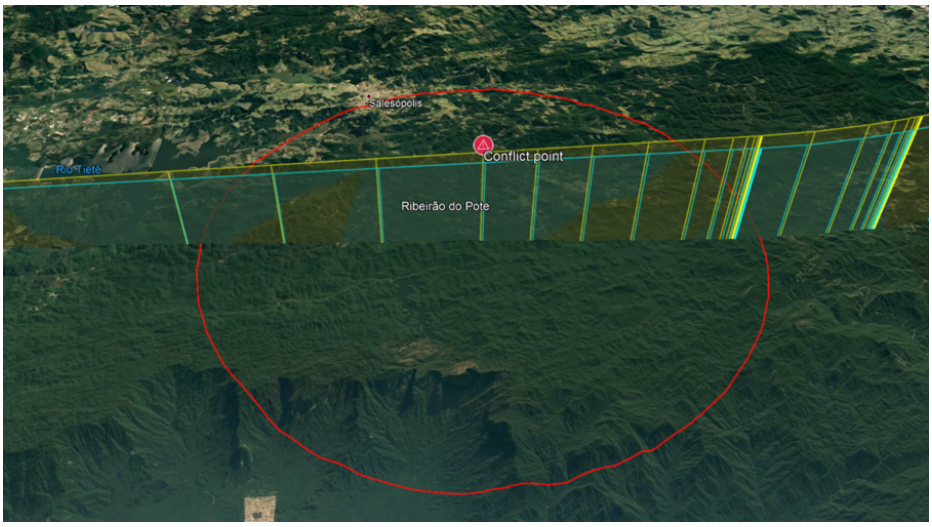

Figure 7.4: A conflict detected between two aircraft.

found: Firstly, the default flight plans insert 330 conflicts in the scenario throughout the day. The novel conflict detection methodology presented in this work also takes into consideration the probable conflicts that should appear in every possible combination of pre-selected alternate trajectories summing up to 6105 trajectories and 833072 waypoints.

Although this procedure completes the strategic conflict prediction in a whole day, further assessment was performed for sequential time windows comprising the evaluation of one hour from the current moment. This evaluation is important to demonstrate the efficiency of the algorithm in a dynamic scenario where conflicts should be detected on demand, typically in a short-term tactical operation, to be implemented as part of a future work. Table 7.4 shows the performance obtained for conflict detection for default trajectories only, and Table 7.5 shows the results including alternate trajectories. Both

Table 7.2: Performance of conflict detection among all strategic trajectories.

			Cassandra	**MongoDB**
Threads	Waypoints	Conflicts	Time	Time
1	833072	8466	1m12,025s	1m20,319s
2	833072	8466	0m38,126s	0m42,293s
4	833072	8466	0m23,075s	0m24,802s
8	833072	8466	0m15,798s	0m19,169s
16	833072	8466	0m15,498s	0m18,605s
32	833072	8466	0m16,216s	0m17,042s

Table 7.3: Performance comparison with threaded speed-up.

Threads	Speed-up in Cassandra	Speed-up in MongoDB
1	1	1
2	1,889130777	1,8991086
4	3,121343445	3,238408193
8	4,559121408	4,190046429
16	4,647373855	4,317065305
32	4,441600888	4,713003169

scenarios were performed in parallel execution with 16 threads.

Algorithm 2 for conflict detection was verified to be far more efficient than the algorithms found in the state-of-art survey concerning the execution time. In fact, the NoSQL database approach dismisses the need of data pre-processing by the application and therefore the provided data set is regarded as an already pruned search space.

Concerning the addition of threads to speed up the execution, consistent results were found. As expected, Figure 7.5 shows that the execution time dropped consistently up to 8 threads due to the CPU core virtualization feature. Table 7.3 confirms this information.

Figure 7.6 compares the execution performance of the proposed databases showing the time (x axis) needed to read the waypoints (y axis). By observing the execution time of the algorithms concerning both Cassandra and MongoDB databases, the first conclusion is that the performance is very similar. MongoDB presented better performance for small

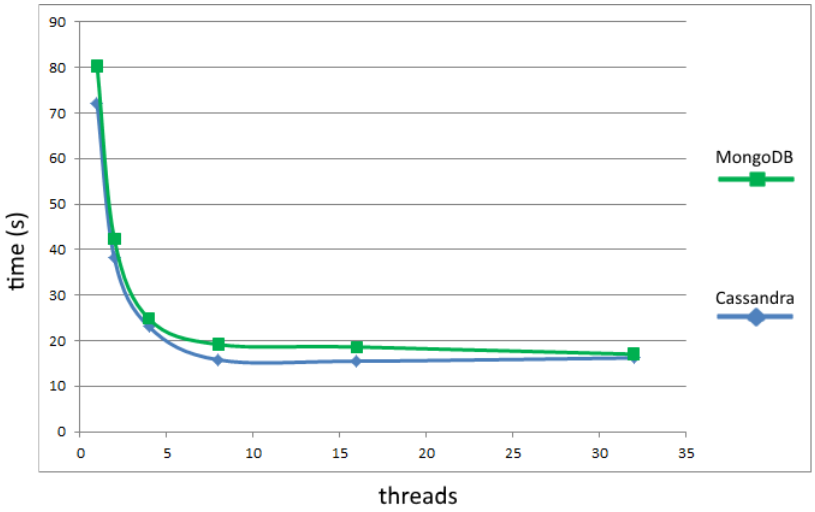

Figure 7.5: Performance comparison with threaded speed-up.

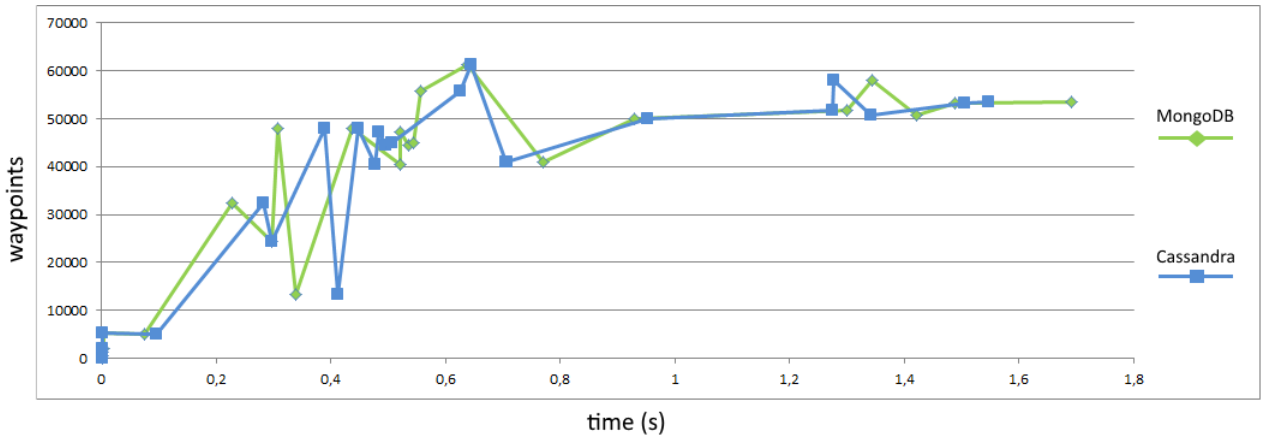

Figure 7.6: Performance comparison of Cassandra and MongoDB databases.

amounts of data, while Cassandra has a slight advantage when dealing with larger chunks of data. This can be explained by the indexation characteristic of Cassandra databases, provided that partition and cluster keys are properly defined.

Table 7.4: Time-window conflict detection for default trajectories.

			Cassandra	**MongoDB**
Hour	Waypoints	Conflicts	Time (s)	Time (s)
0:00	10357	23	0,016	0,003
1:00	4764	2	0,012	0,007
2:00	2600	4	0,026	0,002
3:00	1018	0	0,002	0,002
4:00	274	0	0,001	0,001
5:00	0	0	0,002	0,002
6:00	119	0	0,002	0,002
7:00	405	0	0,002	0,001
8:00	975	0	0,001	0,032
9:00	6086	8	0,015	0,016
10:00	9965	20	0,018	0,014
11:00	10220	22	0,013	0,012
12:00	9706	22	0,018	0,014
13:00	11135	30	0,021	0,008
14:00	7914	11	0,011	0,016
15:00	9530	24	0,012	0,01
16:00	8514	12	0,02	0,009
17:00	8613	14	0,017	0,024
18:00	7734	13	0,014	<0,001
19:00	9112	16	0,017	0,008
20:00	9143	21	0,018	0,012
21:00	10692	29	0,02	0,011
22:00	11826	37	0,061	0,026
23:00	9320	22	0,033	0,046
Total	160022	330	0,372	0,278

Table 7.5: Time-window conflict detection for all strategic trajectories.

			Cassandra	**MongoDB**
Hour	Waypoints	Conflicts	Time (s)	Time (s)
0:00	53416	611	1,548	1,691
1:00	24316	96	0,296	0,228
2:00	13286	87	0,411	0,338
3:00	5133	11	0,096	0,074
4:00	1358	0	0,001	0,001
5:00	0	0	0,002	0,001
6:00	606	0	0,002	0,002
7:00	2158	0	0,002	0,002
8:00	5204	0	0,002	0,003
9:00	32484	179	0,282	0,307
10:00	51690	593	1,276	1,344
11:00	53166	563	1,505	1,421
12:00	50721	520	1,343	1,489
13:00	58029	711	1,277	1,3
14:00	41044	268	0,705	0,771
15:00	50101	673	0,952	0,929
16:00	45054	327	0,507	0,639
17:00	44495	401	0,496	0,521
18:00	40409	278	0,477	0,556
19:00	47306	410	0,484	0,52
20:00	47955	482	0,448	0,437
21:00	55789	768	0,626	0,544
22:00	61287	938	0,644	0,536
23:00	48065	550	0,39	0,297
Total	833072	8466	13,772	13,951

Chapter 8

An AMAN system for CD&R in 4D arrivals

Trajectory-Based Operations (TBO) and Departure Management (DMAN) systems are developed in order to support the air traffic controllers in the management of the air traffic flow and the departures in the airports. Such systems must be integrated to AMAN (Arrival Management) systems and, therefore, there is a need for implementation of supporting technologies for landings and arrivals as well.

One can observe that the arrival phase is critical for conflict detection. This chapter is destined to evaluate an ancillary solution that specifically addresses this flight phase.

Section 8.1 explains how 4D compliant arrivals will be implemented in the proposed AMAN system. The way in which time windows and Top of Descent points are calculated, and the main parameters that will define the performance decisions of the aircraft are presented.

Section 8.2 presents the decision making process of the air traffic control. This reading shows how ATC should act and the main methodologies used to achieve its behavioral profiles are explained.

Section 8.3 describes the 4D-AMAN system implementation. The modeling and the resulting system are described.

Finally, Section 8.4 presents the AMAN Simulation scenario designed for this brief experiment.

8.1 4D-Compliant arrivals

The Continuous Descent Approach (CDA) is a procedure established to allow aircraft to perform an optimum descent profile while arriving at an airport. It is an aircraft operating

technique aided by appropriate airspace and procedure design, and appropriate air traffic control (ATC) clearances [39].

Descent operations are held from the Top of Descent (TOD) to touch down on the airport yard. The optimum vertical profile should work in a continuously descending path, with a minimum of level flight segments through elimination of level-offs [41]. Arriving aircraft should descend from an optimal position with minimum thrust, avoiding level flight to the extent permitted by the safe operation of the aircraft and yet in compliance with published procedures and ATC instructions.

Previous studies show that the type of aircraft, their actual weight and wind conditions are essential parameters in the optimum vertical path angle calculation [24]. Dynamic parameters, such aircraft speed and altitude also account to determine the TOD point, which should be at a given distance from the airport, usually 100 to 150 nautical miles from the destination airport [42].

In order to design 4D-compliant flights, one should remind that every point in an aircraft's trajectory is composed of a 4-tuple informing the three spatial dimensions, plus a timestamp. Thus, the flight management system (FMS) onboard the aircraft should calculate a time window for every waypoint along the route to be flown.

The arrival scenario can be figured as a Merge point implementation in which all arriving aircraft should cross a specific waypoint in order to perform the descent until the approach fix. Notice that many aircraft may cross the same merge points. Some research has been developed concerning the merge point concept, in which all arriving aircraft at the aerodrome should pass a specific waypoint in order to perform the descent until the approach fix. This last waypoint in effectively called the merging point, and such conception will be used here.

Figure 8.1 illustrates two flights, namely a and b, arriving at a given aerodrome. They are currently flying their assigned routes, but eventually they will reach the merging point, annotated with a black pin in the figure. Aircraft FMS need to calculate the Top of Descent (TOD) and issue the results to the ATC, which will evaluate all incoming aircraft and assign definitive instructions that need to be followed by the flight crew.

Data link communication between aircraft and ATC is paramount to effective descent operations as part of the arrival clearance for use by the FMS [39]. The controller must provide information about the distance-to-go, so the onboard FMS in the aircraft can calculate the top of descent (TOD) concerning specific performance parameters. As output, FMS yields a 4D slot, composed of waypoint specifications and a time window that constrain the instant in which the aircraft should cross the waypoint. This waypoint may be considered as a three-dimensional volume in airspace concerning the separation requirements (the waypoint is centered in the mass-center of such a volume).

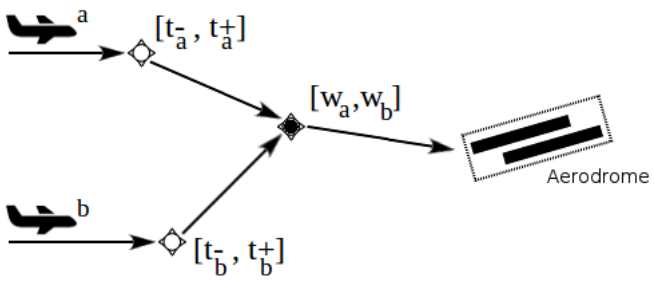

Figure 8.1: 4D flights arriving at a point merge.

Let us discretize the agents involved in the landing process. There are two entities, or two entity categories that take actions and decisions in this scenario: the aircraft and the ATC. One should realize that both entities assume a given behavior in this scenario, so we need to define also the protocols that shall rule these behavioral profiles.

8.1.1 Goals of 4D-compliant arrivals

The main goal in this implementation is the simulation of realistic scenarios to be found on terminal areas around aerodromes, with varying demand conditions and different sets of constraints. Furthermore, an adequate solution for arrival management comprises the development of algorithms for arrival scheduling compliant to the Ration by Schedule and Compression concepts. More importantly, the algorithms need to effectively address the 4D navigation paradigm.

The full decision making process should be modeled as a dynamic, multi-agent decision making process. All entities (agents) involved in this process have well-defined goals and their behavior must be guided by strategies that may make possible the achievement of the well-defined goals.

Aircraft major objective is the minimization of individual cost, expressed in fuel burn and flight time. In order to have this objective accomplished, two main tasks must be performed by the FMS:

- calculation of an optimal descent profile, in terms of TOD altitude and aircraft speed;
- calculation of a feasible time window in order to reach the point where CDA should take place.

It means that cost optimization should be performed through an efficient descent operation that depends on appropriate 4D slot calculation for each aircraft. This slot

is calculated by the FMS and represents the 4D value that minimizes the cost function specified by the aircraft.

ATC major objective is to keep the airspace safe [39]. Since this is a dynamic scenario where agents interact and may request the same resources at the same time, conflicts may arise and they need to be addressed by ATC. Thus ATC must resolve conflicts for aircraft that fly through common routes and have overlapping time windows for merging points.

ATC then should receive the FMS-calculated 4D slots and finally calculate new values for cruise speed and altitude for conflicting aircraft. New time windows should be assigned to each aircraft, provided that ATC-calculated time constraints comply with the desired individual time window preferences for each aircraft, when possible. Safety is now granted, but improvements can be made concerning the total cost to the scenario. This means that individual achievements of the aircraft may not be perfect, but they contribute to diminish the overall scenario cost.

8.1.2 Constraints

All calculations in this framework need to be constrained to performance specification, ATC regulations and capacity limitations. The following constraints apply:

- minimum and maximum speed per flight segment, regulated by ATC authority
- minimum and maximum speed per altitude, regulated by ATC authority;
- minimum separation between two aircraft, regulated by ATC authority;
- maximum descent rate;
- 4D time constraint should be within the calculated time windows always when possible
- aircraft flight path should always be continuous and progressive.

For model validation, re-routing is not performed in this stage of research.

8.1.3 Input

The main pillar of Collaborative Decision Making (CDM) is the sharing of complete and up-to-date information among all impacted entities. It means that all the information to the decision making process needs to be available always when needed [88].

The sources of information are specific databases, system parameters and radar input, plus the information exchanged among the entities. System Wide Information Management (SWIM) concepts may apply for data provisioning and information sharing [7], but this system is not subject of this particular study.

This information serves as input to calculations and are listed below:

- aircraft type;
- aircraft speed;
- aircraft position (x, y, z);
- distance-to-go;
- flight plans;
- runway thresholds;
- aircraft performance profiles.

8.1.4 Output

Onboard FMS need the distance-to-go, current aircraft speed and specific performance data in order to yield the optimum descent profile parameters, which involves the calculation of top-of-descent (TOD), optimum time window and descent angle. At the first moment, the time window is passed to the ATC, which adjusts it in case of potential conflicts and sends back updated values for altitude, speed and time window, which the aircraft must comply with.

After calculations, the expected output is as follows:

- optimum time windows (FMS);
- updated cruise speeds (ATC);
- updated cruise altitudes (ATC);
- descent angle (FMS);
- optimum TOD point (FMS);
- evaluation between TOD and PM;
- optimum arriving sequence to initial approach fixes (ATC).

8.1.5 Aircraft TOD and time window calculation

As said before, aircraft must behave in order to lower their operation costs. Individual costs for aircraft in practice are related to fuel burn and flight time. Thus we need to have accurate measures that evaluate these resources.

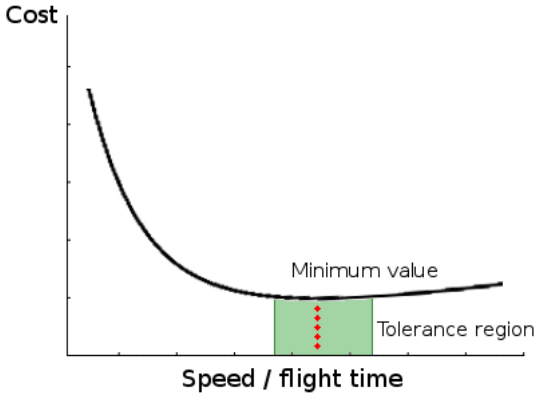

Figure 8.2: Tolerance region around an optimum value.

The Cost Index (CI) is an input parameter to the FMS which determines the flight profile that should be performed by the aircraft. FMS may choose to fly faster to recover en-route delays, or fly slower to minimize fuel burn [89].

This parameter is the ratio of the time-related operation costs and the fuel cost. The value of the CI reflects the relative effects of fuel cost on overall trip cost as compared to time-related direct operating costs [30]. The calculation of CI is according to the following equation:

$$CI = \frac{C_{time}}{C_{fuel}} \tag{8.1}$$

The FMS uses this number and other performance parameters to calculate economy climb, cruise and descent speeds. Low CI values are used to high fuel cost and large CI values are used when the delay cost is more impacting. Hence CI expresses the trade-off between the cost of delay and the cost of fuel consumption [89].

Although the state-of-art, commercial flight management systems are able to efficiently calculate a descent profile, a continuous descent operation under the 4D paradigm demands time constraint calculations that are not yet implemented in such onboard systems. This research proposal requires that the FMS uses the cost index value and further appropriate performance parameters to build a cost curve that expresses the direct operating cost (DOC) for the flight. This curve should show the overall cost as a function of the aircraft speed.

The minimum value in this curve is the optimum speed profile to be performed by the aircraft. This value might not be possible due to ATC restrictions, so a tolerance around this point should be calculated. Figure 8.2 shows an example that illustrates this tolerance region.

In order to make an arriving aircraft 4D-compliant, one needs to build a time window

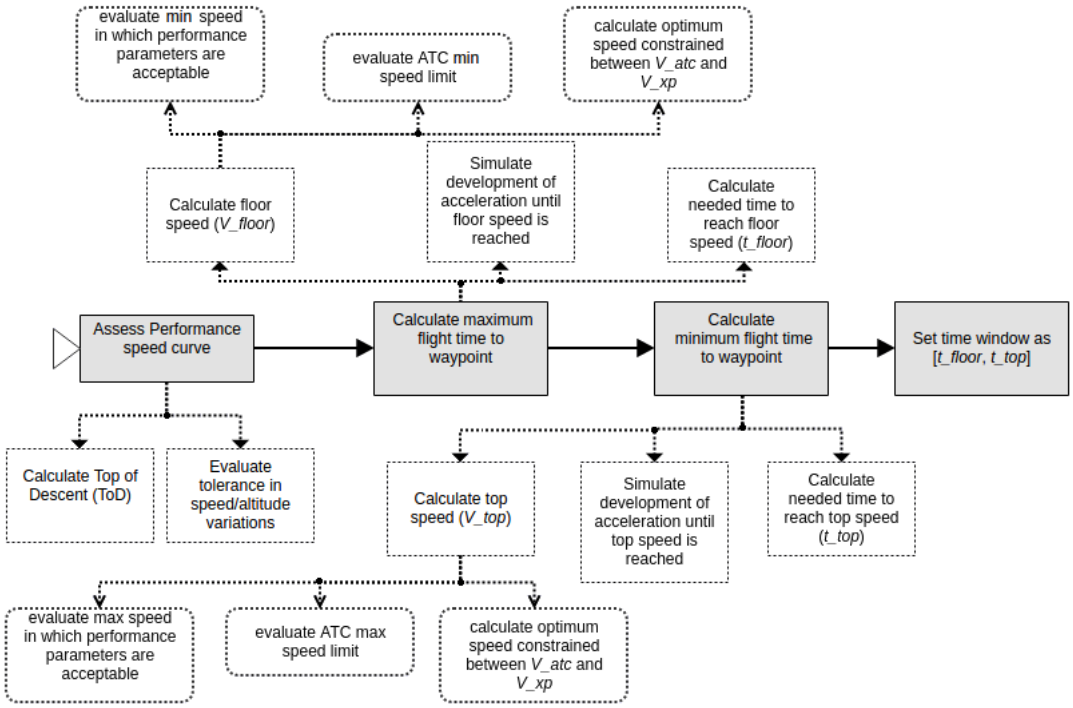

Figure 8.3: Time window calculation algorithm.

to be accomplished by the aircraft when crossing a given metering fix, i.e., a point in which the controlled aircraft will be evaluated. Flight time can easily be yielded once the speed value and flight path distance are known. Given the minimum and maximum values calculated for aircraft speed, FMS should extract the needed time to accommodate this operation profile. FMS then yields the time window from these values and this time window is submitted to ATC along with the top of descent point.

The algorithm developed for time window calculation is shown in the diagram of Figure 8.3.

8.2 ATC decision making process

ATC focuses on the safety of the airspace. It is responsible for monitor all the aerial movement and takes actions in order to eliminate risks that arise during the flight movements. Conflicts need to be detected with antecedence and separation rules must be applied to prevent aircraft from physically affect each other [90].

During the arrival phase, ATC must manage the aircraft that arrive to the destination TMA. This terminal area may have several entry points and ATC is responsible to sequence incoming aircraft in a First Come, First Served (FCFS) basis at each entry point. These entry points are some of the Merge points previously discussed. It is important to

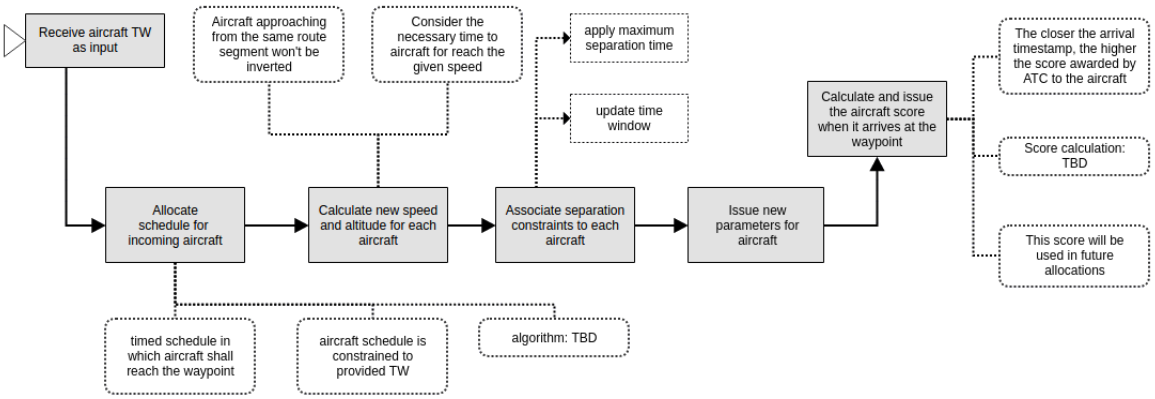

Figure 8.4: Sequencing and updated time window calculation algorithm.

state that TMA and aerodrome management is out of scope of this research, therefore we assume that aircraft that are already in CDO should not be interrupted and they are no more a matter for processing.

In the proposed 4D arrival procedure, ATC receives the time window and the top of descent point as input from aircraft. At first, they are enqueued by arrival time and the separation times are applied. Concerning aircraft that are in the same airway, i.e., when an aircraft is immediately followed by another one at the same airway, ATC must assume that these flights may not be inverted and their arrival order will be maintained. Therefore, ATC may change their speeds and change separation parameters, but the order in which the aircraft entered in the flight segment will be the same until the next airway intersection. Nevertheless, flights that converge from different flight segments may have their arrival order changed. This procedure is executed in order to grant the *ration by schedule* [29].

Once ATC receives the published time windows, conflicts are detected when overlapping time windows emerge at the 4D slots calculated by onboard FMS, which means that new time windows must be assigned for each conflicting aircraft. As long as it is possible, updated constraints should be compliant to the preferred slots.

In the pursuit of a safe and cost-efficient conflict resolution, ATC decision making process will be modeled as a Simulated Annealing combinatorial optimization problem.

Sequencing and updated time window calculation algorithm is shown in the diagram of Figure 8.4.

8.2.1 Simulated Annealing decision making

As another paradigm in which conflicts in the arrival schedule are treated, ATC implements an adaptation of the Simulated Annealing algorithm. The solutions in the search

space are represented by a vector $S = (x_1, ..., x_n)$, where n is the number of aircraft scheduled to cross the merge point. In this vector, x_i informs the Estimated Time of Arrival (ETA) of the aircraft. The neighborhood in each solution in the search space is also a vector $S_d = (x_1, ..., x_n) + (0, ..., y_d, ..., 0)$, where $x_d \neq y_d$. Hence, neighboring solutions displace the schedule of an aircraft in the arrival sequence.

The energetic state of the solution expresses the global cost for the scenario. ATC does not know specific performance parameters inherent to the aircraft, but the preferred time windows are known. Thus, the energy function is an ATC estimate about the solution quality concerning the aggregate individual costs, according to Equation 8.2.

$$E_j = \sum_{i=1}^{n} w_i^j \tag{8.2}$$

In this equation, value w expresses the estimated cost of the time window assigned to aircraft i by solution j. The magnitude of value w is directly proportional to the schedule displacement attributed to the aircraft according to the preferred time window. The minimum energy state is used in order to yield solution S, from where the updated time windows are extracted. Finally, after the optimization procedures, the aircraft are notified about the decisions taken by the ATC.

8.3 4D-AMAN system implementation

In order to evaluate the proposed technique as an adequate scheduler for 4D flights, a prototype system is expected to be implemented. Evolutionary algorithms were used in the implementation, and this chapter describes the yielded system.

The computational tool for scheduling of arrivals in 4D that was implemented can be defined as an AMAN (Arrival Management) system. Its main responsibility is to serve as an additional aid to the controller in his decision-making process.

This AMAN prototype system was implemented in Java language with Eclipse IDE Oxygen. The choice for this language is based on its popularity, easiness of maintenance and object-oriented programming capability.

8.3.1 Aircraft implementation

Each aircraft object is implemented in a way such that the same object represents an aircraft for the outer world, but internal methods perform the tasks inherent to the flight management system.

Aircraft in the prototype act as agents that have their own performance standards, which are used in the decision making process. After full parse of all flight plans, the

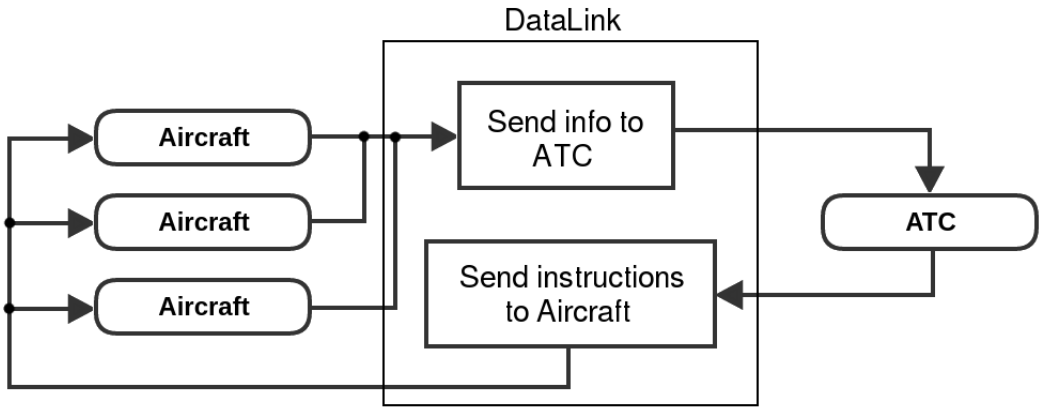

Figure 8.5: Datalink simulation among the entities of the prototype.

corresponding aircraft are instantiated. Thus, every flight plan that serves as input to the prototype is represented by an Aircraft object. Communication between aircraft and ATC is simulated by a class called *DataLink*, which invokes the appropriated methods in a point-to-point communication, according Figure 8.5.

Information present in the flight plan allow the aircraft's FMS to calculate the total flight distance, which will be later used to calculate the distance to the merge point. Also, flight plan allows the publicity of all information concerning the aircraft, like scheduled times for departure and arrival, the route to be followed [1], cruise speed and further on.

When the aircraft reaches a given distance to the merge point, it needs to calculate the time window to be sent to ATC. Merge point position is known by the aircraft and the distance to be flown is calculated according to a simple kinematic equation, since only the off-block time and the cruise speed are available to the aircraft.

Time window calculation is the main task of the onboard FMS. Minimum and maximum profile speeds are taken into account, and vary about 10% from the optimum speed in which the aircraft will start its descent procedures. It is assumed that an aircraft that crossed the merge point will perform the optimum descent profile according to the MP-crossing speed. See Figure 8.3 to see how this task is done.

8.3.2 ATC implementation

The Air Traffic Controller entity is implemented as a Singleton class that is responsible to establish the arrival schedules and send speeding instructions to the aircraft. Furthermore, ATC also checks the schedule conflicts by detecting time window overlaps.

At first, ATC receives all time windows through the implemented datalink mechanism and inserts the incoming aircraft in a queue. Each time window is in fact a time slot

[1] As future work, trajectory coordination shall also be a part of the 4D framework.

requested by the aircraft, thus the queue is transformed in a schedule ordered by arrival time. It is expected that slot requests conflict, so the selected optimization algorithm is invoked to update the schedule by eliminating the conflicts.

Simple separation could be applied to the schedule, but this is very ineffective in the sense that the later aircraft in the queue would have slower priority and therefore would suffer the greater impact. In fact, simulations showed that later aircraft in the queue could be assigned delays greater that two hours.

This situation can be addressed by ATC when it tries to estimate the implicit cost for the aircraft. Cost Index and specific performance parameters are unknown by the ATC, but the time parameters are well-known, since they are published in the flight plans. The cost is calculated as a measure that shows how displaced is the assigned schedule from the desired time window.

Each schedule slot is composed of a time cell and its respective flight number, so every time slot is univocally related to a flight. The Particle Swarm and the Simulated Annealing optimizers are invoked after the initial scheduling performed by the ATC. The selected optimizer is then initiated with the scheduled time cells as input.

8.3.3 Optimization through Simulated Annealing

When the Simulated Annealing optimizer is started, the total energy of the initial schedule is calculated. In the annealing algorithm, the total energy is conceived as the total cost for the scenario. This means that the goal of the optimization task of reducing the system energy to the lowest, stable state is equivalent to resolve the scheduling scenario to the less expensive configuration in terms of aggregated cost.

The algorithm that performs the calculation of the total energy is shown in Algorithm 4.

Setting the time cell probability in lines 8 and 11 in Algorithm 4 means that only conflicting flights have a chance to be chosen for update in the next iteration. This is equivalent to say that the solution's neighborhood is always another state where a conflicting aircraft is adjusted. This procedure bounds the probability for selection of bad moves. The consequence is that every solution attempt aims at yielding another state in which a current conflict is solved . The selection of a random neighbor is implemented according to Algorithm 5.

Once the initial solution is set, the system triggers the optimization process. At this moment, some important parameters must be defined:

- T: the initial temperature;
- k: the Boltzmann's constant;

Algorithm 4: Total energy calculation

```
   Data: time cells
 1 energy ← 0.0;
 2 conflicts ← 0;
 3 clearProbabilities();
 4 for each cell in timeCells do
 5 |   energy ← energy + calculateCost(cell.flight);
 6 |   if checkTimeConflict(cell.time) = true then
 7 |   |   conflicts ← conflicts + 1;
 8 |   |   probabilities[cell] ← 1;
 9 |   end
10 |   else
11 |   |   probabilities[cell] ← 0;
12 |   end
13 end
14 return energy;
```

Algorithm 5: Selection of a random neighboring substance

```
   Data: time cells
 1 rand ← random(1, conflicts);
 2 sum ← 0;
 3 index ← 1;
 4 while sum < rand do
 5 |   sum ← sum + probabilities[index];
 6 |   index ← index + 1;
 7 end
 8 aircraft = max(1, index);
 9 cell ← timeCells[aircraft];
10 time ← cell.time + randomInt(−1, 1);
11 neighbor ← timeCells;
12 neighbor[aircraft] ← time;
13 return neighbor;
```

- α: the temperature update function (cooling);
- ΔE: the selection probability.

The initial temperature T is set to 1000 and is updated by function `updateTemperature()` (see Algorithm 3), which returns the *alpha* value in the Equation 8.3. Small updates of 1% of the current temperature are expected to be slow enough to make the algorithm converge.

$$\alpha = 0.99T \tag{8.3}$$

The Boltzmann's constant is a physical constant, thus it can be essentially suppressed by making it equals to 1 [64]. Thus, the probability of an increase in energy of magnitude ΔE is given by Equation 8.4.

$$P(\Delta E) = e^{\frac{-\Delta E}{T}} \tag{8.4}$$

8.4 AMAN Simulation scenario

The proposed solution is modeled based on the traffic (arrival) scenario in the Presidente Juscelino Kubitschek International Airport in Brasília (SBBR). It is the third largest airport in Brazil, managing around 46 thousand passengers per day, distributed in an average of 267 landings and 266 departures. The airport has annual capacity for 11 million passengers [58].

As there is no actual implementation of the merging point concept in this airport, a virtual merge point is created by 150 nautical miles at landing runway extension for simulation purposes. In the developed algorithms, the minimum and maximum profile speeds are calculated according to the predefined parameters for optimum descent speed. The calculation of time window is straightforward, since the aircraft FMS knows the direct distance that should be traveled toward the merge point and the speed range in which this path should be performed.

All arriving flights are selected at SBBR from 11:00AM to 11:30AM during a regular workday. 26 flights are scheduled for this period according to their repeated flight plans (RPL) and their time windows are also calculated. The ATC is responsible to schedule all flights in a FIFO fashion according to the predicted time windows within one minute of separation. If a given pair of time windows overlaps, then the scenario is tagged as conflicting.

In fact, the ATC's perception regarding airlines operational costs is total time delay, despite airlines models and estimations. In this work, the ATC perspective is considered

and therefore time delay is used as proxy for operational costs estimation. Worth to mention that according to recent studies conducted by EUROCONTROL, each minute of delay inflight may represent an increase between 50 to 80 US$ in the direct operational costs for a flight, depending on equipment type [11].

8.4.1 Results with Simulated Annealing

In this model, a neighboring allocation is such that the time window of a conflicting aircraft is displaced. In order to make this possible, the energetic state evaluation triggers the attribution of uniform selection probability among all aircraft that fit in this constraint. In this way, the selected neighbor certainly will be a valid attempt for resolving a conflict shared between two aircraft.

Naturally, a new conflict might occur. In this case, the new solution should be accepted with a probability given by Equation 8.4, which triggers the update of the selection probability in the neighborhood.

In the simulation preformed, the algorithm converges to the lowest energy state after 578 iterations. At the end of processing, the solution values are extracted and the results show that the yielded schedule was able to comply with preferred time windows for 21 of 26 flights, that is, 80,77% of the evaluated flights were allowed to execute their desired time windows. Concerning further flights, only 4 needed to be delayed.

These results are shown in Table 8.1.

8.4.2 Remarks

An Arrival Management system (AMAN) counting on a solution for Continuous Descent Arrival (CDA) has been developed as a Proof of Concept. A Simulated Annealing algorithm was implemented to address the problem of arrival coordination considering air traffic safety and individual requirements in the extent of possibilities.

This case study revealed that the flights issued to later periods are more receptive to absorb delays, since their time windows are less strict. Therefore, the coordination with other air traffic services is needed for proper flow control in further phases of the flight.

Individual interests of aircraft were successfully combined with airspace control constraints. This case study has shown that Simulated Annealing is an effective methodology to model the conflict resolution in arrivals using 4D time windows. Previous experiments [64] demonstrated that modeling the cost function and the methodology for selecting valid neighbors are cumbersome tasks, but the proposed models were able to overcome this difficulty.

Table 8.1: Arrival schedule with time window yielded by SA algorithm.

Aircraft Id	**Preferred TW**	**Output**
AC_01	[0:12, 0:15]	00:12
AC_02	[0:14, 0:17]	00:14
AC_03	[0:18, 0:21]	00:16
AC_04	[0:18, 0:21]	00:18
AC_05	[0:18, 0:21]	00:22
AC_06	[0:19, 0:23]	00:24
AC_07	[0:21, 0:26]	00:26
AC_08	[0:26, 0:32]	00:28
AC_09	[0:26, 0:32]	00:31
AC_10	[0:28, 0:34]	00:33
AC_11	[0:29, 0:35]	00:35
AC_12	[0:30, 0:37]	00:38
AC_13	[0:33, 0:40]	00:41
AC_14	[0:36, 0:43]	00:43
AC_15	[0:46, 0:53]	00:48
AC_16	[0:46, 0:53]	00:50
AC_17	[0:46, 0:53]	00:52
AC_18	[0:56, 1:03]	00:57
AC_19	[0:56, 1:03]	00:59
AC_20	[1:02, 1:09]	01:04
AC_21	[1:03, 1:11]	01:06
AC_22	[1:04, 1:11]	01:08
AC_23	[1:06, 1:13]	01:10
AC_24	[1:08, 1:15]	01:13
AC_25	[1:09, 1:17]	01:15
AC_26	[1:18, 1:25]	01:21

Chapter 9

Discussion and Conclusion

The ATC conception about the operating costs for airlines is directly related to the estimated delay to the flights, for it is not possible to assess data from the airlines directly. This research is focused in the decision making process of ATC, which regards the estimated cost per delay minute while trying to eliminate all the conflicts by adjusting the departure time of aircraft.

In this proposed model, the neighborhood of solutions is designed to adjust some aircraft which is already involved in a conflict. Hence, the evaluation of the energetic state of solutions embeds the attribution of an uniform probability for the aircraft which fulfill this requirement. Therefore, every solution is an effective attempt to resolve a conflict between two aircraft. If a new conflict arises, the solution is accepted under the probability given in Equation 6.10.

The main contribution in this research is the development of a new computational solution for conflict detection and resolution in 4D trajectories that incorporates a trajectory predictor and databases specifically designed for big data. The developed application consumes information from several sources conveniently aggregated in NoSQL MongoDB and Cassandra databases, and invokes an external service for trajectory predictions, in a successful attempt to adequate the ATC operations to the SWIM architecture paradigm.

Finally, this application is not supposed to replace the ATCO whatsoever, but serves as a decision-making supporting tool for the human operator.

9.1 Research evaluation

Novel methods for 4D trajectory evaluation were presented. The proposed storage architecture is designed to comply with SWIM paradigm, as this architecture is able to manage and provide information to support the decision making process of air traffic stakeholders.

Some important algorithms were developed. Simulated Annealing was used to effectively deconflict aircraft during a critical phase in the flight, which is the arrival phase. The insertion of delays was performed by adjusting the arrival time windows in a fair manner concerning the distribution of scenario costs among the impacted aircraft.

The algorithms proposed for trajectory evaluation demonstrated to be very efficient. Although the simulation scenarios concern only the strategic planning, conflict detection procedure was found to be suitable even for tactical planning, provided that sampled trajectories are available in the database.

Some limitations were found during the experiments. The simulation scenarios concern only the strategic planning, therefore the algorithm must be extended for tactical planning. An important requirement for a more realistic scenario should also consider air traffic capacity constraints, which were not modeled in this stage of the research. Another caveat is related to the NoSQL compliance to the CAP theorem. Partitioning was compromised to make data available and consistent during the simulation, which could introduce problems related to resource scalability.

9.2 Future work

As a future work we intend to extend the implementation to update not only the departure time in the strategic deconfliction, but also observe the en-route trajectories and manage the aircraft motion profile by dynamically updating the flight parameters.

Sector management is a very important share of the ATCO workload, but we didn't evaluate the airspace occupation per sectors in this work. The notion of conflict herein presented does not include capacity constraints, therefore this is a pending matter or concern for our future implementations.

It is suggested to study en-route trajectory conflicts and capacity constraints, as well as the complete modeling of 4D trajectory management processes. Airport capacity was beyond the scope of this work, although this is a very important factor for efficiently address the requirements of constrained capacity. Also, *Compression* techniques were found to be important as an attempt to avoid the stabilization in sub-optimal local solutions and are expected to be implemented in further stages of this research.

Finally, modeling 4D procedures for every individual phase of the flight is a contribution to be achieved in a mid-term future.

9.3 Acknowledgements

This work has been partially supported by the Brazilian National Council for Scientific and Technological Development (CNPq) under the grant number 311441/2017-3 and also by Boeing Research & Technology/Brazil. By the initiative of Boeing, the project of 4D Modeling for Efficient Flight Path with Collaborative Trajectory Option Program in Brazilian Scenarios (4D EFP BR) is being developed in TransLab at the University of Brasília.

In the current stage of this research, the proposed methodologies were proven to be effective in the conflict resolution task for arrivals upon a terminal area. In order to have a functional 4D framework for the Brazilian scenario, some steps took place in the research workflow:

1. *4D Trajectories*: Aircraft Intent Description Language is a framework developed in order to uniquely describe an aircraft's trajectory intent in terms of space and time. Implementation of this solution aggregates to the Brazilian ATM system not only time window prediction, but also trajectory prediction and early conflict detection. This task was accomplished by this research.

2. *Traffic synchronization*: a complete 4D solution is desired as a final product of this research. All flight phases should be addressed and effectively integrated.

3. *Scenario coordination*: once the integrated 4D framework is fully implemented, several scenarios must be designed and the effectiveness of the framework must be evaluated concerning the totality of the Brazilian airports.

4. *Impact analysis*: experiments will be performed and the yielded results will be assessed. Detailed analysis will show the benefits of utilizing the implemented solution, that should be evaluated in terms of cost reduction, applicability, economic and social impact and feasibility.

This research looks toward into the future of Air Traffic Management (ATM) in Brazil. It addresses the 4D Navigation concept in a novel way, considering optimization algorithms for efficiently distribute air traffic resources and manage in-flight conflicts. The main social contribution is the greater reliability in the air traffic safety assurance, since the proposed 4D framework should emerge as an important decision making support tool, thus reducing the workload faced by air traffic controllers. The minimization of operational costs inherent to the air transportation network is another expected achievement, with the positive side effect of reduction of noise and toxic emissions due to idle thrust operations and fuel economy.

Very good results were obtained. For this reason, some conference and journal papers were submitted. Furthermore, this work yielded a patent application. The full set of offspring so far is listed below:

- *Modeling the Swarm Optimization to Build Effective Continuous Descent Arrival Sequences*, 19th International IEEE Conference on Intelligent Transportation Systems - ITSC 2016 (Qualis CC B1), Rio de Janeiro-RJ;
- *Gerenciamento de Aterrissagens em Navegação 4D Utilizando Arrefecimento Simulado para Resolução de Conflitos entre Aeronaves*, for the XXX Yearly Congress of Associação Nacional de Pesquisa e Ensino em Transportes - ANPET 2016, Rio de Janeiro-RJ.
- *Collaborative Decision Making in Departure Sequencing With an Adapted Rubinstein Protocol*, IEEE Transactions on Systems, Man, and Cybernetics: Systems (Volume: 46, Issue: 2, Feb. 2016) (Qualis CC B1);
- *Towards Intelligent System Wide Information Management for Air Traffic Management*, Lecture Notes in Computer Science, 1st edition. Springer International Publishing, 2017;
- *High Performance Deconfliction Algorithm on 4D Trajectories*: Commercial patent application altogether with Boeing Research & Technology Brazil (BR&TB) - ID 17-2677.
- *Conflict Detection and Resolution with Local Search Algorithms for 4D-Navigation in ATM*, 18th International Conference on Intelligent Systems Design and Applications - ISDA 2018, Vellore, India;
- *Air Traffic Conflict Resolution applying the Airplanes Total Operational Costs*, submitted to IEEE Intelligent Transportation Systems Transactions (ITS). Awaiting review process.

References

[1] IATA. Iata forecast predicts 8.2 billion air travelers in 2037, 2018. Press Release No.: 62. Date: October 24th, 2018. 1

[2] Joint Planning and Development Office. Operational concept for the next generation air transportation system (nextgen). Technical Report 3, JPDO, 2009. 1, 2, 16, 17, 18

[3] European Air Traffic Management (EATM). Mission trajectory detailed concept. Technical Report Document identifier DDS/CM/SPM/SESAR/12-042, SESAR, 2012. 1, 2, 16, 18

[4] EUROCONTROL and FAA. Sesar-nextgen aligned tp structure and terminology. Technical Report Version 1.0, Action Plan 16 White Paper, Jan 29th 2010. 1, 40, 41

[5] L. Weigang, A. Leite, V. Ribeiro, J. Fregnani, and I. de Oliveira. Towards intelligent system wide information management for air traffic management. In *Security, Privacy, and Anonymity in Computation, Communication, and Storage*, pages 584–593. Springer International Publishing, 2017. 1

[6] International Civil Aviation Organization. Manual on system wide information management (swim) concept. Technical Report Doc 10039, ICAO, 2015. 1

[7] J. E. Dieudonne, H. L. Crane, S. R. Jones, C. J. Smith, S. A. Remillard, and G. Snead. Neo (nextgen 4d tm) provided by swim's surveillance soa (sdn asp for rnp 4d ops). In *2007 Integrated Communications, Navigation and Surveillance Conference*, pages 1–12, April 2007. 1, 68, 101

[8] A. Rodríguez-Sanz, D. Álvarez, F. Comendador, R. Valdés, J. Pérez-Castán, and M. Godoy. Air traffic management based on 4d trajectories: A reliability analysis using multi-state systems theory. *Transportation Research Procedia*, 33:355 – 362, 2018. XIII Conference on Transport Engineering, CIT2018. 1

[9] M. Vilaplana, E. Gallo, and F. Navarro. Towards a formal language for the common description of aircraft intent. *24th Digital Avionics Systems Conference*, 1:3.C.5–3.1–9, 2005. 2, 7, 47

[10] G. Frontera, J. Besada, A. Bernardos, E. Casado, and J. López-Leonés. Formal intent-based trajectory description languages, Aug 2014. 2, 47, 52, 53, 54

[11] Boeing. Current market outlook 2015-2034. Technical Report Document 9931, AN/476, Boeing Commercial Airplanes, 2015. 2, 111

[12] DECEA. Implementação operacional do conceito de navegação baseada em performance (pbn) no espaço aéreo brasileiro. Technical Report Aeronautical Information Report 24/13, Brazilian Airspace Control Department, 2013. 3, 4

[13] M. A. Konyak, D. Warburton, J. López-Leonés, and P. C. Parks. A demonstration of an aircraft intent interchange specification for facilitating trajectory-based operations in the national airspace system. *AIAA Guidance, Navigation and Control Conference and Exhibit*, 2008. 3, 7, 18, 41, 47

[14] J. Besada, G. Frontera, J. Crespo, E. Casado, and J. López-Leonés. Automated aircraft trajectory prediction based on formal intent-related language processing. *IEEE Transactions on Intelligent Transportation Systems*, 14(3), Sep 2013. 3, 41, 42, 52

[15] A. M. F. Crespo. Módulo de avaliação e apoio à decisão: uma aplicação de aprendizagem por reforço no gerenciamento tático do fluxo de tráfego aéreo. Master's thesis, Universidade de Brasília - Departamento de Ciência da Computação, 2010. 3

[16] Brazilian Air Force (FAB). Plano de implementação atm nacional. Technical Report Portaria DECEA no. 37/DGCEA, Defense Ministry, 2012. 4

[17] Brazilian Air Force (FAB). Concepção operacional atm nacional. Technical Report Portaria no. 630/GC3, Defense Ministry, 2011. 4

[18] C. F. M. Teixeira Jr and G. R. Ferreira. Uma comparaçãoo entre sgbd's relacionais e nosql para dados de proveniência em workflows científicos, 2014. 6, 68, 70

[19] J. López-Leonés, M. A. Vilaplana, E. Gallo, F. A. Navarro, and C. Querejeta. The aircraft intent description language: A key enabler for air-ground synchronization in trajectory-based operations. In *2007 IEEE/AIAA 26th Digital Avionics Systems Conference*, pages 1.D.4–1–1.D.4–12, Oct 2007. 7, 43, 47

[20] Samet Ayhan and Hanan Samet. Aircraft trajectory prediction made easy with predictive analytics. In *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, pages 21–30. ACM, 2016. 8, 37

[21] L. H. Mutuel, P. Neri, and E. Paricaud. Initial 4d trajectory management concept evaluation. *Tenth USA/Europe Air Traffic Management Research and Development Seminar*, 2013. 8, 19

[22] L. Cruciol, J.-P. Clarke, and L. Weigang. Trajectory option set planning optimization under uncertainty in ctop. In *2015 IEEE 18th International Conference on Intelligent Transportation Systems*, pages 2084–2089, Sept 2015. 8

[23] B. Favennec, F. Vergne, and K. Zeghal. Point merge integration of arrival flows enabling extensive rnav application and continuous descent - operational services and environment definition. Technical Report 2, EUROCONTROL ATC Operations and Systems, 2010. 8

[24] L. Stell. Predictability of top of descent location for operational idle-thrust descents. *10th AIAA Aviation Technology, Integration, and Operations (ATIO) Conference*, 2010. 8, 20, 29, 99

[25] S. Ruiz, M. Piera, J. Nosedal, and A. Ranieri. Strategic de-confliction in the presence of a large number of 4d trajectories using a causal modeling approach. *Transportation Research Part C: Emerging Technologies*, 39:129–147, 2014. 8, 33, 38

[26] G. M. Camargo. Controle da pressão seletiva em algoritmo genético aplicado à otimização de demanda em infra-estrutura aeronáutica. Master's thesis, Escola Politécnica da Universidade de São Paulo - USP, 2006. 10, 12

[27] International Civil Aviation Organization. *International Standards and Recommended Practices - Air Traffic Services*, volume 11. International Civil Aviation Organization, 2001. 11

[28] Air Force Command. *Diretrizes para Implementação de Programas de Garantia de Qualidade nos Serviços de Tráfego Aéreo*, volume MCA 100-12. DECEA, 2004. 11

[29] M. O. Ball, R. L. Hoffman, D. Knorr, J. Wetherly, and M. Wambsganss. Assessing the benefits of collaborative decision making in air traffic management. *Progress in Astronautics and Aeronautics*, 193:239–252, 2001. 11, 105

[30] B. Roberson. *Fuel Conservation Strategies: Cost Index Explained*. Quarterly publication of Boeing Aeromagazine, 2007. 12, 103

[31] A. M. G. Silva. Sistema de simulação acelerado para análise de fluxo de tráfego aéreo. Master's thesis, Instituto Nacional de Pesquisas Espaciais - INPE, 2001. 12

[32] International Civil Aviation Organization. Phase of flight: Definitions and usage notes. Technical Report 1.3, ICAO Commercial Aviation Safety Team, April 2013. 13

[33] FAB. Plano de voo. Technical Report ICA 100-11, Brazilian Air Force - Electronics and Flight Protection Board, 2000. 14, 15, 40

[34] Brazilian Air Force Command. *Regras do Ar e Serviços de Tráfego Aéreo*, volume ICA 100-12. DECEA, 2006. 14, 15

[35] M. Nolan. *Fundamentals of air traffic control*. Cengage Learning, 2010. 16

[36] T.L. Teller. 4d fms tbo pilot-controller human-in-the-loop simulation. Technical report, Massachusetts Institute of Technology (MIT), 2001. 16, 76

[37] H.Q. Lee and G.H. Hardy. 4d area navigation system description and flight test results. Technical Report Technical Note no. 7874, NASA, 1975. 16, 17, 20

[38] D.H. Williams and S.M. Green. Airborne four-dimensional flight management in a time-based air traffic control environment. Technical Report Technical Memorandum no. 4249, NASA, 1991. 16, 17, 18

[39] International Civil Aviation Organization. Continuous descent operation (cdo) manual. Technical Report Document 9931, AN/476, ICAO, 2010. 19, 20, 99, 101

[40] C. Grabow. A method to design a tie-point-based optimized profile descent (opd) solution. In *2013 IEEE/AIAA 32nd Digital Avionics Systems Conference (DASC)*, pages 1D3–1–1D3–9, Oct 2013. 19

[41] L. Jin, Y. Cao, and D. Sun. Investigation of potential fuel savings due to continuous-descent approach. *Journal of Aircraft*, 50(3), 2013. 19, 99

[42] M. Wu and A. Sadovsky. Minimum-cost aircraft descent trajectories with a constrained altitude profile. Technical Report NASA Technical Memorandum 2015-218734, AMES Research Center, 2015. 20, 99

[43] C.A. Zúñiga, M.A. Piera, S. Ruiz, and I Del Pozo. A tma 4dt cd/cr causal model based in path shortening/path stretching techniques. *4th International Conference on Research in Air Transportation*, 2010. 20, 24

[44] J. De Prins, R.G. Ledesma, and M. Mulder. Towards time-based continuous descent operations with mixed 4d fms equipage. *11th AIAA Aviation Technology, Integration, and Operations (ATIO) Conference*, September 2011. 21, 26, 27

[45] S. Wandelt and X. Sun. Efficient compression of 4d-trajectory data in air traffic management. *IEEE Transactions On Intelligent Transportation Systems*, 16(2), 2015. 22

[46] Y. Song, P. Cheng, and C. Mu. An improved trajectory prediction algorithm based on trajectory data mining for air traffic management. In *Proceeding of the IEEE International Conference on Information and Automation*, pages 981–986, 2012. 23, 24, 76

[47] S. Torres. Swarm theory applied to air traffic flow management. *Procedia Computer Science*, 12:463–470, 2012. 24

[48] S. Ruiz, M. Piera, and C. Zúñiga. Relational time-space data structure to speed up conflict detection under heavy traffic conditions. *SESAR Innovation Days (SID)*, 2011. 24

[49] D. Bertsimas and S. Gupta. Fairness in air traffic flow management. *INFORMS Meeting, San Diego - CA, USA*, 2009. 27

[50] S.G. Park and J.-P. Clarke. Vertical trajectory optimization for continuous descent arrival procedure. *AIAA Guidance, Navigation, and Control Conference*, 2012. 31

[51] J.L. Zillies, A.R. Schmitt, and R. Vujasinovic. Multiobjective 4d optimization of a trajectory-based air traffic management. *Integrated Communications, Navigation and Surveillance Conference*, pages 1–11, 2013. 34

[52] S. Russel and P. Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall, Pearson Education, Inc, 3rd edition, 2010. 35, 57, 58, 60, 61, 62

[53] D. Delahaye K. Legrand, C. Rabut. *Correction and Optimization of 4D aircraft trajectories by sharing wind and temperature information*. PhD thesis, l'INSA de Toulouse conjointement avec l'École Nationale de l'Aviation Civile, 2019. 37

[54] Nicolas Durand, Cyril Allignol, and Nicolas Barnier. A ground holding model for aircraft deconfliction. *29th Digital Avionics Systems Conference*, 2010. 38

[55] Jan Berling, Alexander Lau, and Volker Gollnick. European air traffic flow management with strategic deconfliction. *Operations Research Proceedings*, pages 279–286, 2017. 38

[56] F. Kunzi and R. Hansman. Ads-b benefits to general aviation and barriers to implementation. Technical Report Report no. ICAT-2001-6, MIT International Center for Air Transportation (ICAT), 2011. 42

[57] M. Strohmeier, M. Schafer, V. Lenders, and I. Martinovic. Realities and challenges of nextgen air traffic management: the case of ads-b. *IEEE Communications Magazine*, 52(5):111–118, May 2014. 42

[58] V. F. Ribeiro, L. Weigang, V. Milea, Y. Yamashita, and L. Uden. Collaborative decision making in departure sequencing with an adapted rubinstein protocol. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 46(2):248–259, Feb 2016. 42, 110

[59] M. Paglione C.Garcia-Avello I. Bayraktutar J. Bronsvoort, G. McDonald and C.M. Young. Impact of missing longitudinal aircraft intent on descent trajectory prediction. *2011 IEEE/AIAA 30nd Digital Avionics Systems Conference (DASC)*, October 2011. 43

[60] M. Konyak, S. Doucett, R. Safa-Bakhsh, E. Gallo, and P. Parks. Improving ground-based trajectory prediction through communication of aircraft intent. *AIAA Guidance, Navigation, and Control Conference*, 2009. 47, 52

[61] J. Bronsvoort. *Contributions to Trajectory Prediction Theory and its Application to Arrival Management for Air Traffic Control*. PhD thesis, Escuela Técnica Superior de Ingenieros de Telecomunicación, Universidad Politécnica de Madrid, 2014. 48

[62] David Poole and Alan Mackworth. *Artificial Intelligence: Foundations of Computational Agents*. Cambridge University Press, 1st edition, 2010. 57

[63] S. Dasgupta, C. Papadimitriou, and U. Vazirani. *Artificial Intelligence: Foundations of Computational Agents*. McGraw-Hill Higher Education, 1st edition, 2006. 58, 59

[64] K. A. Dowsland and J. M. Thompson. *Simulated Annealing*, chapter 49, pages 1623–1655. Springer-Verlag Berlin Heidelberg, 2012. 58, 59, 60, 61, 110, 111

[65] J. F. Nash. *Non-cooperative games*. PhD thesis, Princeton University, 1950. 59, 62, 63

[66] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*, volume 2. The MIT Press, 2001. 59, 81, 83

[67] S. Kirkpatrick, C.D. Gelatt, and M.P. Vecchi. Optimization by simulated annealing. *SCIENCE, New Series*, 220(4598):671–680, 1983. 60, 61

[68] D. Bertsimas and J. Tsitsiklis. Simulated annealing. *Statistical Science*, 8(1):10–15, 1993. 60, 61

[69] V.F. Ribeiro. Decisão colaborativa com utilização de teoria dos jogos para o sequenciamento de partidas em aeroportos. Master's thesis, Universidade de Brasília - UnB, 2013. 61

[70] S Chaimatanan, D. Delahaye, and M Mongeau. A methodology for strategic planning of aircraft trajectories using simulated annealing. *1st International Conference on Interdisciplinary Science for Air Traffic Management (ISIATM)*, Jun 2012. 62

[71] Theodore L Turocy and Bernhard von Stengel. Game theory*: Draft prepared for the encyclopedia of information systems. Technical report, CDAM Research Report LSE-CDAM-2001-09, 2001. 62, 63

[72] J. F. Nash. Equilibrium points in n-person games. *Proceedings of the National Academy of Sciences of the United States*, 36:48 – 49, 1950. 62, 63

[73] S.W. Wilson. Classifier systems for continuous payoff environments. *Genetic and Evolutionary Computation Conference*, 6 2004. Seattle, USA. 63

[74] C. A. Holt and A. E. Roth. The nash equilibrium: A perspective. *Proceedings of the National Academy of Sciences of the United States*, 101(12):3999 – 4002, 2004. 63

[75] V. F. Ribeiro, D. A. Pamplona, J. A. T. G. Fregnani, I. R. de Oliveira, and Li Weigang. Modeling the swarm optimization to build effective continuous descent arrival sequences. In *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, pages 760–765, Nov 2016. 64

[76] EUROCONTROL and FAA. Human performance in air traffic management safety, Sep 2010. 66

[77] ICAO. 27th working group meeting. Technical Report ATMRPP-WG/27-WP/636, International Civil Aviation Organization, 2014. 67

[78] Y. Kang, I. Park, J. Rhee, and Y. Lee. Mongodb-based repository design for iot-generated rfid/sensor big data. *IEEE Sensors Journal*, 16(2):485–497, Jan 2016. 68, 71, 76

[79] . A. Brewer. Towards robust distributed systems (abstract). In *Proceedings of the Nineteenth Annual ACM Symposium on Principles of Distributed Computing*, PODC '00, pages 7–. ACM, 2000. 68

[80] Datastax. Apache cassandra 2.1 documentation. Technical report, Apache Cassandra, 2015. 69, 70

[81] A B M Moniruzzaman and S Hossain. NoSQL database: New era of databases for big data analytics - classification, characteristics and comparison. *Int J Database Theor Appl*, 6, 06 2013. 70, 71

[82] V. Abramova and J. Bernardino. Nosql databases: Mongodb vs cassandra. pages 14–22, 07 2013. 71

[83] Y. Li and S. Manoharan. A performance comparison of sql and nosql databases. In *2013 IEEE Pacific Rim Conference on Communications, Computers and Signal Processing (PACRIM)*, pages 15–19, Aug 2013. 71, 76

[84] FAA and EUROCONTROL. Flight object - a recommendation for flight script and trajectory description. Technical report, Action Plan 16: Common Trajectory Prediction Capabilities, Oct 30th 2006. 72

[85] J.A.B. Portas and G. Frontera. Trajectory prediction documentation. Technical report, TP Operation manual. Boeing Research and Technology - Spain, 2016. 72

[86] Datastax. Cassandra & datastax enterprise essentials documentation. Technical report, Apache Cassandra, 2015. 74

[87] DECEA. Anuário estatístico de tráfego aéreo. Technical report, Headquarters of Air Navigation Management (CGNA), 2017. 90

[88] M. O. Ball, R. Hoffman, and A. Mukherjee. Ground delay program planning under uncertainty based on the ration-by-distance principle. *Transportation Science*, 44(1):1–14, 2010. 101

[89] A. Cook, G. Tanner, V. Williams, and G. Meise. Dynamic cost indexing. *6th EUROCONTROL Innovative Research Workshops and Exhibition*, Dec 2007. 103

[90] Civil Aviation Authority of New Zealand. Wake turbulence. good aviation practice. Technical report, CAA, 2008. 104