Strategic planning of aircraft trajectories

Supatcha Chaimatanan

THÈSE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par :
Université Toulouse lll Paul Sabatier (UT3 Paul Sabatier)

Discipline ou spécialité :
Mathématiques appliquées et Informatique

Présentée et soutenue par
Supatcha CHAIMATANAN

Soutenue le :
21 juillet 2014

Titre :
Planification stratégique de trajectoires d'avions

Strategic planning of aircraft trajectories

Ecole doctorale :
École Doctorale Aéronautique Astronautique (EDAA)

Unité de recherche :
Laboratoire de Mathématiques Appliquées Informatique et Automatique pour l'Aérien (MAIAA), École Nationale de l'Aviation Civile (ENAC)

Directeurs de Thèse :
Professeur Marcel MONGEAU, ENAC
Professeur Daniel DELAHAYE, ENAC

Rapporteurs :
Professeur Patrick SIARRY, Université Paris-Est Créteil Val-de-Marne
Professeure Hamsa BALAKRISHNAN, Massachusetts Institute of Technology

Autres membres du jury :
Professeur Éric FERON, Georgia Institute of Technology (Examinateur)
Professeure Marie-Pierre GLEIZES, Université Toulouse lll Paul Sabatier (Examinatrice)
Professeur Claus GWIGGNER, Universität Hamburg (Examinateur)

Résumé

Afin de pouvoir satisfaire la demande sans cesse croissante du trafic aérien, le futur système de gestion du trafic aérien utilisera le concept d'opérations basées sur les trajectoires (*Trajectory Based Operations*), qui augmentera la capacité du trafic aérien, en réduisant la charge de travail du contrôleur. Pour ce faire, les tâches de détection et de résolution de conflits seront transférées depuis la phase tactique vers la phase stratégique de la planification.

Dans le cadre de ce nouveau paradigme pour le système de gestion du trafic aérien, nous introduisons dans cette thèse une méthodologie qui permet d'aborder ce problème de planification stratégique de trajectoires d'avion à l'échelle d'un pays ou d'un continent. Le but de la méthodologie proposée est de minimiser l'interaction globale entre les trajectoires d'avion, en affectant de nouveaux créneaux de décollage, de nouvelles routes et de nouveaux niveaux de vols aux trajectoires impliquées dans l'interaction. De plus, afin d'améliorer la robustesse du plan stratégique de vols obtenu, nous prenons en compte l'incertitude de la position de l'avion et de son heure d'arrivée à un point donné de la trajectoire de l'avion.

Nous proposons une formulation mathématique de ce problème de planification stratégique conduisant à un problème d'optimisation discrète et un problème d'optimisation en variables mixtes, dont la fonction objectif est basée sur le nouveau concept d'interaction. Un algorithme efficace en termes de temps de calcul pour évaluer l'interaction entre des trajectoires d'avion pour des applications de grande taille est introduit et mis en oeuvre.

Des méthodes de résolution basées sur des algorithmes de type métaheuristique et métaheuristique hybride ont été développées pour résoudre ces problèmes d'optimisation de grande taille. Enfin, la méthodologie globale de planification stratégique de trajectoires d'avion est mise en oeuvre et testée sur des données de trafic, prenant en compte des incertitudes, pour l'espace aérien français et l'espace aérien européen, impliquant plus de 30 000 vols. Des plans de vols 4D sans conflits et robustes ont pu être produits avec des temps de calcul acceptables dans un contexte opérationnel, ce qui démontre la viabilité de l'approche proposée.

Mots-clés: Planification stratégique, trajectoire d'avion 4D, gestion du trafic aérien, optimisation discrète, optimisation en variables mixtes, métaheuristique, métaheuristique hybride.

Strategic planning of aircraft trajectories

Abstract

To sustain the continuously increasing air traffic demand, the future air traffic management system will rely on a so-called Trajectory Based Operations (TBO) concept that will increase air traffic capacity by reducing the controllers workload. This will be achieved by transferring tactical conflict detection and resolution tasks to the strategic planning phase.

In this future air traffic management paradigm context, this thesis presents a methodology to address such strategic trajectory planning at nation-wide and continent scale. The proposed methodology aims at minimizing the global *interaction between aircraft trajectories* by allocating alternative departure times, alternative horizontal flight paths, and alternative flight levels to the trajectories involved in the interaction. To improve robustness of the strategic trajectory planning, uncertainty of aircraft position and aircraft arrival time to any given position on the trajectory are considered.

This thesis proposes a mathematical formulation of this strategic trajectory planning problem leading to a discrete-optimization and a mixed-integer optimization problem whose objective function relies on the new concept of interaction between trajectories. A computationally efficient algorithm to compute interaction between trajectories for large-scale applications is introduced and implemented. Resolution methods based on metaheuristic and hybrid-metaheuristic algorithms have been developed to solve the above large-scale optimization problems. Finally, the overall methodology is implemented and tested with air traffic data taking into account uncertainty over the French and the European airspaces, involving more than 30,000 trajectories. Conflict-free and robust 4D trajectory planning are produced within computational time acceptable for the operation context, which shows the viability of the approach.

Keywords: Strategic planning, 4D aircraft trajectory, air traffic management, discrete optimization, mixed-integer optimization, metaheuristic, hybrid metaheuristic.

Contents

Introduction

We first present here an overview of the current air traffic management system and its limitations. After that, future trends of the air traffic management concept are discussed. Then, objectives, scope, and the contributions of this study are presented. Finally, the structure of this thesis is given.

Air traffic management system

Air traffic management (ATM) is a system that assists and guides aircraft from a departure airport to a destination airport in order to ensure its safety, while minimizing delays and airspace congestion. It manages the air traffic through the management of the three following complementary systems: airspace management (ASM), air traffic flow management (ATFM), and air traffic control (ATC).

The ASM manages the usage of airspace. Its primary objective is to maximize the utilization of available airspace by segregating the airspace among various airspace user's needs in order to prevent interference from all users and to facilitate the flow of air traffic.

The ATFM manages the air traffic flow in order to minimize delays and to prevent congestion. In Europe, this system is managed by the Central Flow Management Unit (CFMU) of Eurocontrol. Every (non-military operation) flight performing under instrument flight rules[1] in Europe must submit a *flight plan* to the CFMU. The CFMU then analyzes the compatibility of the request. If the request is not compatible with the airspace structure or the capacity limit, the CFMU will suggest alternative routes, then distributes the accepted flight plan to all local air traffic control centers overflown by that particular flight in Europe.

This flight plan includes the following information:

- aircraft identification number, aircraft type, and navigation equipment installed on board;
- departure airport;
- proposed time of departure;
- requested cruising altitude (*flight level*[2]);
- requested route of flight;

[1]Instrument Flight Rules (IFR) is a set of regulations concerning aircraft operated when the pilot is unable to use visual reference navigation.

[2]Flight level (FL) is a pressure altitude, expressed in hundreds of feet, e.g. and altitude of 32,000 feet is referred to as FL 320.

- cruising airspeed, climb and descent profiles, and speed schedules;
- destination airport.

The ATC then controls the air traffic in real time. It uses the flight plan information to predict the traffic situation, then issues necessary changes to the flight plan in order to ensure aircraft separation, and to maintain the order of air traffic flow, while satisfying as much as possible the pilot's request. For this purpose, the airspace is partitioned into different *sectors*, each of which is assigned to a group of controllers monitoring the air traffic. In order to prevent the controller from being overloaded, the number of aircraft allowed to enter a given sector at any given time is limited. When the number of aircraft reaches this limit, the corresponding sector is said to be *congested*.

Generally, congestion in air transportation can be categorized into two groups according to the part of airspace it involves. *Terminal congestion* is the congestion that occurs around the terminal control area[3] (TCA, or TMA outside the U.S. and Canada). *En-route congestion* is the congestion involved in the en-route section of the flight between TMAs. In the U.S. the congestion occurs more often in the terminal areas, whereas the en-route congestion is more critical in Europe due to the fragmented nature of its airspace where there are extra difficulties for coordinating the air traffic across the boundaries.

Air traffic regulations impose that aircraft must always be separated by some prescribed distance, noted N_v for the vertical separation and N_h for the horizontal separation. Current ATC regulations require aircraft operating in the terminal maneuvering area (TMA) to be vertically separated by at least N_v = 1,000 feet and horizontally separated by a minimum of N_h = 3 nautical miles. In the en-route environment, for aircraft operating up to (and including) FL 410, the horizontal minimum separation is increased to 5 nautical miles; for aircraft operating above FL 410, the vertical separation is increased to 2,000 feet [71]. Aircraft are considered to be in *conflict* when these *minimum separation* requirements are violated. Such conflict situations would not necessary lead to a collision; however, it is a situation that controllers must avoid. One can consider that at any given time, each aircraft has a bounded and closed reserved block of airspace defined by a three-dimensional cylinder, as shown in Figure 1, in which other aircraft are not allowed to enter.

Because airspace, aircraft, ground systems, and human operators are limited resources which are very costly to extend, the usage of these resources has to be optimized through an effective planning. A good planning allows the ATM process to conform with the airspace user's requirement, and to be robust against unexpected events. The ATM process is performed through the following three planning phases:

- **Strategic planning.** This phase is performed around one year down to one week before real-time operations of the flight. This process aims to predict the air traffic load, and design the air route structure in order to balance capacity and demand.
- **Pre-tactical planning.** This phase takes place six days down to one day before real-time

[3] A terminal control area (also known as a terminal maneuvering area) is a controlled airspace surrounding major airports, generally designed as a cylindrical or up-side-down wedding-cake shape airspace of 30 to 50 mile radius and high of 10,000 feet.

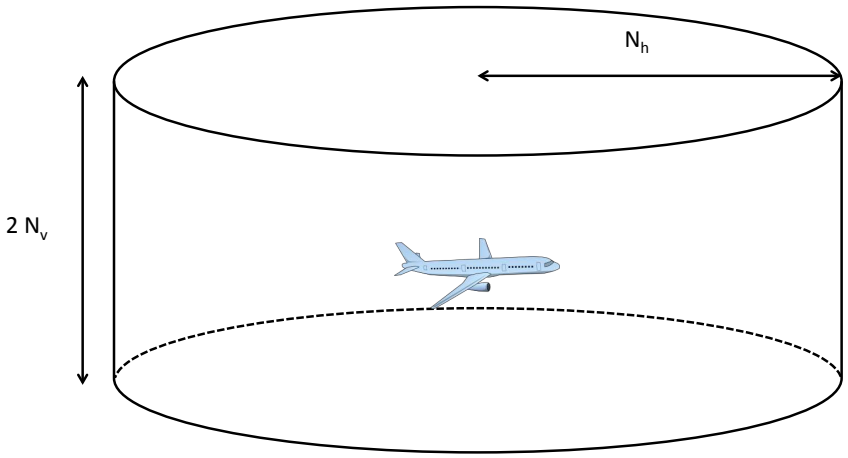

Figure 1: The cylindrical *protection volume*

operations. Its primary aims are to optimize the overall ATM network performance, minimizing delays and costs by fine-tuning the strategic plan using more-update information of expected traffic condition, traffic demand, available capacity and weather forecast.

- **Tactical planning.** This phase is carried out on the day of operations. In this phase, individual aircraft departure slots are to be provided, re-routings and alternative flight profiles can also be issued in order to avoid bottlenecks (congested sector) and to maximize airspace capacity according to real-time traffic demand.

As the air-traffic demand keeps on increasing, the airspace becomes more and more congested. Over past decades, several methods have been proposed to address the air traffic management problem aiming at balancing air traffic demand and airspace capacity, and to prevent airspace congestion. There are two frequently-used air traffic *decongestion* strategies.

The first one adapts the airspace capacity to the increased demand. This includes reducing of separation norms (e.g. the reduced vertical separation minima - RVSM - approach), constructing new airports, increasing the number of runways, etc. These methods, however, involve high building costs, and the space for airport construction is limited. Another method to increase the capacity is to reduce the size of airspace sectors, in order to decrease the controller's workload. This method is considered, for instance, in [34, 81, 70]. Unfortunately, this re-sectorization strategy has its structural limits since the controller needs sufficient space and time in order to manage the traffic which imposes a constraint on how small the size of the sector can be. Besides, the controller handles only the aircraft of his sector; therefore, he must collaborate with the pilot to ensure flight safety as the aircraft passes from one sector to another. Reducing the size of sectors, in fact, increases the number of sectors and therefore increases the traffic transfer workload.

The second air traffic decongestion strategy is to regulate the air traffic demand to the current capacity. This strategy focuses on decongesting the ATM system through several approaches, such as: allocating delays to each aircraft in order to reduce congestion in sectors or at destination airports, re-routing flights, or regulating aircraft speed in order to avoid congestion mainly in TMA sector, etc. These approaches are considered, for instances, in [72, 51, 84, 24, 22].

Despite the use of the above-mentioned methods, the capacity of today's air traffic management system is still constrained by procedures, and technologies that require air traffic controllers to operate within inefficient guidelines. The usage of the airspace is limited by the inflexible route structure based on the location of *beacons*.[4] In order to accommodate the increasing air-traffic demand in the near future, in an already saturated airspace, ATM requires a major improvement that allows more automations and more efficient usage of the airspace.

Future trends in air traffic management system

Currently, the world's major ATM systems (i.e. European and U.S. ATM systems) are being modernized. The Next Generation air transportation system (NextGen) is a project aiming at transforming the national airspace system (NAS) in the United States towards a satellite-based air traffic management and control system. The Single European Sky ATM Research (SESAR) system is a major collaborative project aiming at modernizing the European air traffic management system.

With the soon-coming technologies that will enable more powerful communication systems, more precise surveillance, and more reliance automated support tools, the new ATM system will improve safety, reduce delay and aviation emission, as well as maximize airspace capacity.

The new ATM system will rely on a concept of Trajectory Based Operations (TBOs) which will focus more on adapting the airspace user's demand to the current airspace capacity. In this new ATM paradigm, air traffic will no longer be constrained by artificial boundaries such as airspace sectors, national borders, locations of beacons, etc. Instead, ATM will focus on trajectory management together with an adapted airspace design. An aircraft flying through the airspace will be required to follow a negotiated *conflict-free* trajectory, accurately defined in 4 dimensions (3 spatial dimensions and time).

This new route structure will not be limited by the location of beacons as in the current ground-based ATM system. Therefore, aircraft will be able to fly more efficient and more direct routes. Besides, this will increase flexibility in aircraft trajectory design.

In addition, improved surveillance equipment will provide the controller precise positions and trajectory data of every aircraft, while advanced avionics technology will allow each aircraft to follow its assigned 4D trajectory with very high accuracy. This will significantly improve the predictability of the air traffic.

Moreover, this improvement of air traffic predictability implies a possibility to automate the conflict detection and resolution tasks. Furthermore, because the uncertainties of the air traffic is decreased, this will also allow us to shift the tactical conflict management tasks to the strategic and pre-tactical planning levels. The concept of 4D strategic deconfliction that aims to generate conflict-free trajectories for aircraft from origin to destination airports is introduced in the Innovative Future Air Transport System (IFATS) project [86] and the 4 Dimension Contract-Guidance and Control (4D CO-GC) project [55]. This 4D trajectory concept will significantly reduce the need of controller's intervention during the tactical phase. The tactical workload will

[4]Beacon is a device that transmits radio signal that could be received by overflown aircraft. It allows aircraft to determine their bearing and pinpoint their exact locations.

involve more monitoring than conflict prediction and resolution. Therefore, a controller will be able to accommodate more flights in a given airspace at a given time.

Objectives, scope, and contributions of this study

In the perspective of the future ATM system, such as the one proposed by the 4D CO-GC project [55], the key factor to improve the ATM capacity is an efficient strategic 4D trajectory planning methodology to compute a conflict-free 4D trajectory for each aircraft. In this work, we propose a methodology to address such a strategic planning of trajectories at national and continent scale. The goal of the proposed strategic planning method is to separate the given set of aircraft trajectories in both the three dimensional space and in the time domain by allocating an alternative flight plan (route, departure time, FL) to each flight.

Instead of trying to satisfy the capacity constraint, we focus on minimizing the global *interaction* between trajectories. An interaction between trajectories occurs when two or more trajectories have an effect on each other; for instance, when trajectories occupy the same space at the same period of time. Therefore, contrary to the concept of conflict, the measurement of interaction does not only refer to the violation of minimum separation requirements. It also allows us to take into account other separation criteria such as minimum separation time between aircraft crossing at the same point, minimal distance between trajectories, topology of trajectory intersection, etc.

In real-life situations, aircraft may not be able to follow precisely the assigned 4D trajectory due to external events, such as passenger delays, wind conditions, etc. Besides, aircraft may not be able to fly at their optimal speed profile in order to satisfy the hard constraints imposed on the 4D trajectory. To improve robustness of the deconflicted trajectories and to relax the 4D trajectory constraints, uncertainties of aircraft position and arrival time will also be taken into account in the strategic trajectory planning process introduced in this thesis.

More precisely, the strategic trajectory planning problem under consideration can be presented as follows:

- We are given a set of flight plans for a given day associated with a nation-wide scale or continent-scale air traffic.
- For each flight, i, we suppose that the following elements are known:
  - a set of possible routes;
  - a set of possible flight levels;
  - a set of possible departure times;
  - the features of the uncertainties of aircraft position and arrival time.

The proposed strategic planning methodology consists of four main modules: a 4D-trajectory generator, a conflict-detection module, an interaction evaluation module, and an optimization module (Figure 2). The 4D-trajectory generator is used to compute a 4D trajectory given an alternative route, an alternative flight level, and an alternative departure time. The conflict detection module computes the conflicts encountered by a given set of 4D trajectories. Then, the

interaction-evaluation module computes the level of interaction between trajectories at a nation-wide or continent scale. The objective of the optimization algorithm is to provide alternative routes, flight levels, and departure times that attempt at minimizing the interaction between trajectories. The optimization algorithm therefore manages the search of an optimal set of alternative routes, alternative flight levels, and alternative departure times.

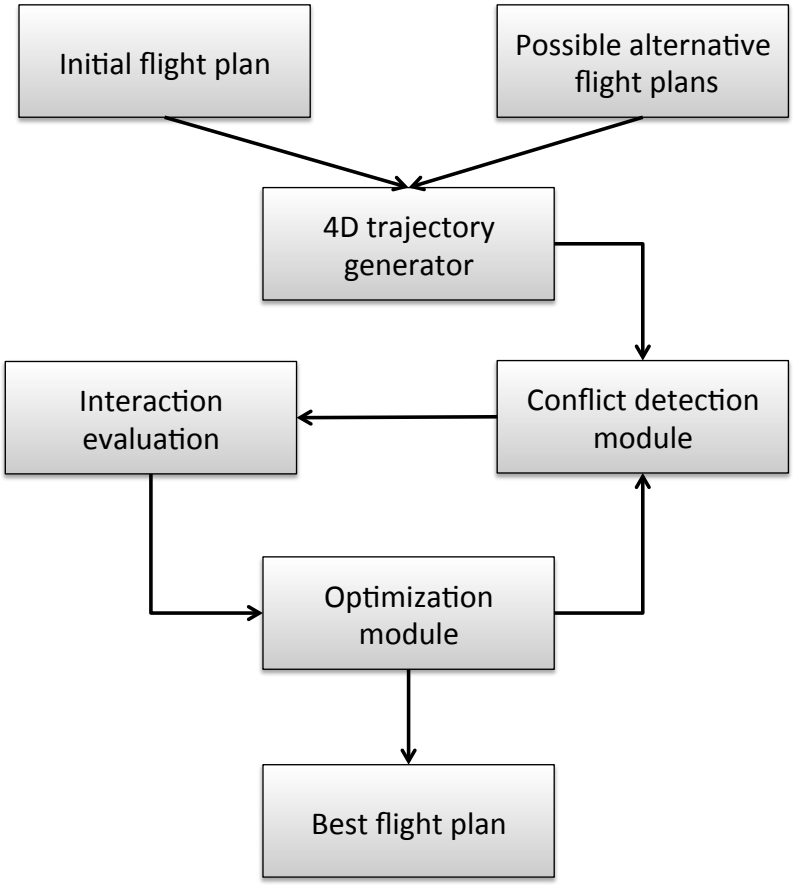

Figure 2: Proposed strategic trajectory planning procedure.

In this thesis, we contribute to the area of air traffic management in the framework of future ATM paradigm. More precisely, we introduce a global strategy to solve conflicts and reduce interaction between 4D aircraft trajectories at a large scale. We also introduce mathematical formulations of the strategic trajectory planning problem under the form of a discrete optimization problem and a mixed-integer optimization problem.

Moreover, we develop a computationally efficient method to detect conflicts and to compute interaction between aircraft trajectories. We propose a metaheuristic optimization algorithm and a hybrid metaheuristic optimization algorithm to solve this large scale and complex problem. Finally, we introduce an approach to consider uncertainties of aircraft trajectory in the spatial and temporal domains during the optimization process.

Thesis structure

This thesis addresses two problems. First, we formalize mathematical models of the strategic trajectory planning problem assuming that the aircraft is able to fly the resulting 4D trajectory with high precision. This strategic trajectory planning problem, without uncertainty, will be referred to, in the sequel of this thesis, as the *strategic trajectory planning* problem. A methodology to detect conflict and to compute the interaction between trajectories is introduced. Optimization algorithms are developed in order to solve this strategic trajectory planning problem.

We shall then focus on improving robustness of the resulting 4D trajectory plan, by taking into account uncertainties of aircraft position and arrival time. This variation of the strategic trajectory planning problem, taking into account uncertainty, will be referred to, in the sequel of this thesis, as the *robust strategic trajectory planning* problem.

The remaining parts of this thesis are organized as follows. Chapter 1 discusses previous works related to air traffic decongestion and trajectory deconfliction problem. Chapter 2 describes the nominal strategic trajectory planning problem in a mathematical framework. Trajectory separation maneuvers to separate the trajectories are introduced. The concept of measuring interaction between trajectories is explained. Mathematical formulations of the nominal strategic trajectory planning problem are presented. Issues related to the complexity and the size of the formulations are discussed. Then, we present an efficient algorithm to detect conflicts and to compute interaction between trajectories in a large-scale application context. In Chapter 3, resolution algorithms to solve the strategic trajectory planning problem are presented. Numerical examples are given, and the performances of the proposed algorithms are discussed. In Chapter 4, we consider the extension of the proposed methodology in order to improve the robustness of the resulting 4D trajectory plan. The concept of robust optimization is briefly described. The uncertainties of the aircraft position and the arrival time are modeled. Mathematical formulations of the robust strategic trajectory planning problem are proposed. The conflict-detection and the interaction-computation algorithms are adapted in order to take into account uncertainties. The optimization algorithm adapted to solve the robust strategic trajectory planning problem is presented, and numerical examples are given. Finally, we present conclusion and perspectives.

Chapter 1

Literature review: Air traffic decongestion and trajectory deconfliction problems

In this chapter, we present existing methods in the literature considering air traffic decongestion and trajectory deconfliction problems. First, the main approaches to alleviate air traffic congestion are presented. Then, we present strategies to detect and solve conflict between aircraft. Finally, we discuss the main methods to address trajectory deconfliction problem that can be applied on large-scale air traffic. We refer the reader interested by a survey on modeling and optimization in air traffic to the recent book [37]. Another survey on mathematical optimization models for air traffic management problems based on different air traffic management strategies is provided in [2].

1.1 Air traffic decongestion methodologies

In this section, we present existing approaches in the literature to address air traffic decongestion problem. Congestion is a situation where the number of aircraft in a given airport or in a given airspace exceeds the maximum number of aircraft that are allowed to enter in the airport or in the airspace respectively. Several researches aiming at minimizing congestion have been conducted in the recent decades. The main goal is to regulate the air traffic demand to fit the current capacity. The air traffic decongestion approaches can be roughly categorized into two categories: ground holding approaches and air traffic flow management approaches.

1.1.1 Ground holding

One of the simplest approaches to regulate the air traffic demand is to attribute delays to the initially-planned aircraft departure times. This is commonly referred to as *ground delay* or *ground holding*. The main idea of the ground holding approach is to limit the number of airborne aircraft at any given time in order to reduce the controller workload, and/or to respect the capacity constraints of the arrival airport. Ground holding strategies transfer airborne delay to the ground at the departure airport, because it is safer and less expensive. Indeed, aircraft

can thereby avoid flying extra distance to avoid congested areas or flying in a holding pattern around congested airport, both of which induce extra fuel consumption.

In general, the ground holding problem assumes that capacity of the arrival airport is known in advance, and that the capacity of both the departure airport and the airspace are unlimited. Alternative route options are not considered. The given data of the problem include: a set of flights, initial preferred departure times, scheduled arrival times to destination airports, a set of possible delay slots, associated delay costs, capacity of the arrival airports for the given scenario, etc. The set of decision variables includes: the arrival times to the destination airport, the ground delay attributed to each flight, etc. The objective function typically involves minimizing the total congestion and delay costs (relative to the total ground holding and to the airborne delays for the flights), while satisfying the capacity constraints of the arrival airports.

The ground holding strategy was first studied in the U.S. in 1987 by A.R. Odoni [72]. This work considers flight planning in real time, aiming at minimizing congestion costs. In [11], the ground holding problem involving multiple aircraft that take off from different airports and that land at a common arrival airport at the same time was studied. A comparison of formulations for the single-airport ground holding problem with landing time-window constraints is discussed in [51]. More applications of the ground holding strategy can be found, for example, in [12, 72, 76, 79, 84, 88].

Despite the advantages of the ground holding approach in terms of reduced airport congestion, with increasing demand, significant delays have to be assigned to a large number of aircraft in order to meet all airport capacity constraints. Besides, the ground holding approach is more effective for situations where congestion is more likely to occur at the airports, which is not the case in Europe, where most congestion occurs in the airspace sectors.

1.1.2 Air traffic flow management

Another commonly-used approach to address the air traffic demand involves managing the flow of air traffic within the airspace. It boils down to study the so-called *Air Traffic Flow Management (ATFM)* problem. In addition to controlling the departure time of each flight, ATFM also seeks optimal arrival times to each airspace sector, taking into account airspace capacity constraints. In other words, ATFM controls not only the departure time, but also the flight itself, throughout its duration [24]. An overview of the air traffic flow management problem is provided in [73].

The ATFM problem assumes that the capacities of both the airports and the airspace are known. The flight precedence constraints and turnaround time (time during which aircraft must remain at the airport) requirements are usually considered. Change of flight speed profiles are however not taken into consideration. The problem inputs include, for instance, a set of initial flight plans (scheduled departure times, routes, flight levels, etc.), a set of possible alternative delay slots, a set of possible alternative routes, associated ground delay and airborne delay costs, etc.

The decision variables includes, for instance, the ground delay times, the air delay times, the time at which each flight arrives at a given airspace or at a given waypoint, the time at which each flight arrives at a given destination airport, alternative routes, and alternative flight levels

associated to each flight, etc. Typically, optimization criteria involve minimizing the total cost of deviation from the initial flight plans, while respecting the airport and the airspace capacity constraints.

In general, the ATFM problem represents the air traffic network by a directed graph, where the nodes correspond to a set of capacitated elements (e.g. airports and airspace sectors), and the arcs represent the sequence relations. For instance, an arc from node i to node j exists if i and j are adjacent sectors such that an aircraft can reach sector j immediately after flying through sector i. An example of a set of possible routes from one origin airport to one destination airport based on this directed graph model, considered in [23], is illustrated in Figure 1.1.

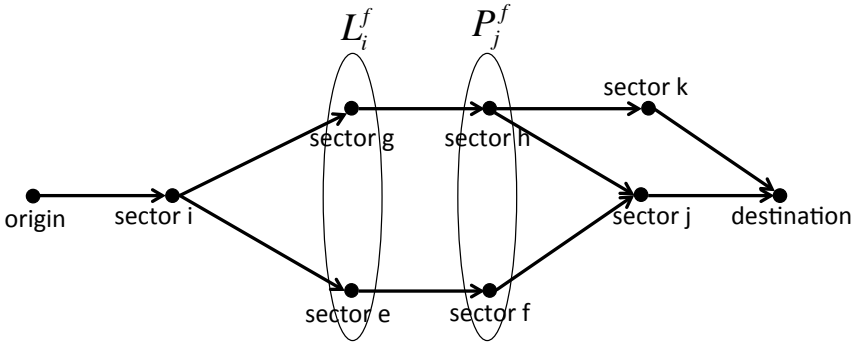

Figure 1.1: Example of possible routes from one origin airport to one destination airport modeled as a directed graph [23], given a flight f, the set L_i^f of sectors that follow sector i, and the set P_j^f of sectors that precede sector j.

In [24], a 0-1 integer optimization model for the ATFM problem considering re-routing options is presented. The objective of this model is to find ground holding and airborne delays that minimize the total delay cost. To take the re-routing option into account, the proposed formulation relies on two possible approaches. The first one is a path approach that decides, for each aircraft, which route to fly among a set of possible routes. The second one is a sector approach that determines, for each flight on its route, which sector to enter next. The problem is shown to be NP-hard[1]. Numerical results for problems involving approximately 1,000 flights are obtained.

In [22] an integer program for ATFM taking into account all phases of each flight (i.e. take-off, cruising, and landing) is presented. The origin-destination route is represented by a sequence of sectors flown by an aircraft. An alternative route is therefore represented by a sequence of sectors to be flown by the aircraft. The proposed route representation allows the ATFM model to be represented by a collection of subgraphs $G_i = (N_i, A_i)$, where the node set, N_i represents the airports and waypoints overflown by flight i, and the arc set, A_i, connects the nodes for flight i. Each of the decision variables, $x^t_{i,m,n}$, takes value 1 if flight i is planned to arrive at node n at time t through arc (m, n), and takes value 0 otherwise. In addition to minimizing the total ground delay and airborne delay costs, the proposed model also takes into account equity between flights. In [23], integer optimization approaches are used in order to

[1]According to the famous NP $\neq$ P conjecture, a Non-deterministic Polynomial-time hard (NP hard) problem cannot be solved in polynomial time (with respect to the size of the instance). The reader interested in complexity of decision problems and combinatorial optimization problems is referred for example to [77].

allocate ground delays and rerouting options to trajectory flows taking into account airspace sector capacity constraints.

The works described above and presented in [22, 23, 24] all rely on algorithms called branch-and-bound algorithms that implicitly consider all feasible points of the optimization problem to provide optimal solutions to the air traffic management problem. Branch-and-bound is an optimization method that successively partitions the solution space by creating a tree of subproblems, some of which are not further refined, based on associated bounds. Such bounds are key issues of the algorithm, and must be specifically defined for each problem. We refer the reader interested by combinatorial optimization methods to, for example, [77].

In [5], the authors introduce a mixed-integer programming model to minimize traveling time, operating/fuel cost, air/sound pollutions subjected to separation and technical constraints. Their decision variables are the arcs and nodes (in a 3D-mesh network), speeds and departure/arrival times for each flight. The problem is solved by an exact deterministic method but on instances involving not more than 10 flights. Further work focusing on managing each individual flight in a large-scale context can be found, for instance, in [74, 75, 3, 66].

In [74, 75], the peak of airspace congestion is minimized using a route-slot allocation technique. Their problem is modeled as follows. For each flight i, there is a pair of decision variables, (δ_i, r_i), where δ_i is the delay attributed to flight i, and r_i is the alternative route allocated to flight i. These variables are chosen from a set of possible delays and alternative routes associated to each flight i. Each alternative route is represented by a set of waypoints to be over flown by a given aircraft. The set of possible alternative routes is obtained by filtering all possible routes for each origin-destination pair from a week of real flight plans. The objective is to minimize the peak of the controllers' workload (i.e. monitoring and coordination workload), considering flight-connection constraints. This problem is NP hard and is non separable. The optimal alternative routes and slots are simultaneously solved using a stochastic optimization method called genetic algorithm (GA) which is a population-based optimization algorithm inspired by biological evolution [42].

The basic principle of GA relies on the "survival of the fittest" principle [45]. The evolution starts from a randomly-generated population of individual candidate solutions, referred to as *chromosomes*. The chance of reproduction of each chromosome in each generation depends on its *fitness* which depends on the resulting objective function value. The *offspring* (or the new generation) are generated by combining the selected (according to fitness) parent chromosomes via *cross-over*, and/or *mutation*, and/or *cloning* operations. The selection and reproduction process repeats until the maximum number of generations has been attained, or a pre-defined fitness level has been reached by the population.

Their proposed methods are tested on nation-wide scale air traffic over the French airspace. Their method is able to provide a robust alternative flight plan that reduces congestion over the airspace by a factor of 2 [74].

1.2 Trajectory deconfliction methodologies

As mentioned in the previous chapter, in order to accommodate the predicted air traffic demand in the near future, the world's major ATM systems are evolving towards the trajectory-based operation concept. In the framework of this future ATM paradigm, one of the most important ATM efficiency improvements will be obtained through improved aircraft conflict management. Instead of focusing only on satisfying the capacity constraints, during recent years, several researches concentrated on solving directly each individual conflict between aircraft trajectories. This approach considers the air traffic management problem at a fine-grain level by ensuring minimum separation between each aircraft via several trajectory deconfliction methodologies.

In this section, we first discuss different methods to detect conflict between aircraft. Then, we present different strategies and methods that are considered in the literature to solve conflict between aircraft. Finally, we present some recent research works concerning the large-scale trajectory deconfliction problem.

1.2.1 Conflict detection methods

Conflict is a situation where two or more aircraft experience a loss of minimum separation. In order to detect such a conflict, one must predict the future position of each aircraft, using a large-enough time window, based on the current state (position, heading, speed, flight plan, etc.) of the aircraft. Several approaches can be used for state propagation based on the assumptions made on the aircraft states.

One possible approach is to predict the aircraft future position using information on the current state of the aircraft, aircraft dynamic models, and current (or predicted) wind conditions. This approach requires a high computational effort, therefore it is usually used in the cases involving small number of aircraft with short prediction time windows.

Another approach uses flight-plan information to predict the aircraft trajectory. This method is employed by the ATC. It is less accurate than the previous trajectory prediction method. This can be counterbalanced by considering higher degree of uncertainties on the predicted trajectory. Accuracy of the conflict detection depends on the reliability of the predicted aircraft future position. A review of conflict detection and resolution modeling methods is provided, for instance, in [59, 61].

Conflict detection methods can be roughly classified into three categories: nominal, worst-case, and probabilistic conflict detections, according to which assumptions are made on the predicted aircraft trajectory. An example of aircraft state propagation based on the three methods, given in [61], is illustrated in Figure 1.2.

- **The nominal conflict detection** method is straightforward. Aircraft position is projected into the future without considering any uncertainty or deviation of the aircraft from its assigned (nominal) trajectory (Figure 1.2 (a)). Uncertainty can still be accounted for by, for example, introducing a safety buffer, or enlarging the minimum separation requirements. This conflict detection method is used, for instance, in [7, 15, 16, 30, 31].

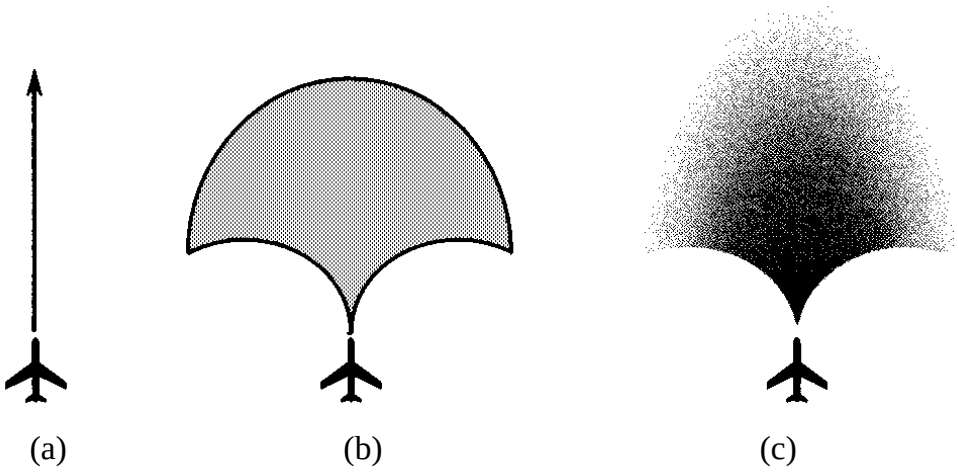

Figure 1.2: Three aircraft state propagation methods: (a) nominal method, (b) worst case method, and (c) probabilistic method from [61].

- **The worst-case conflict detection** assumes that aircraft may not behave as expected, and identifies the conflict as a situation where the separations between the *envelopes* (see Figure 1.2 (b)) of the predicted trajectories do not satisfy the minimum separation requirements. Resolving conflict based on this worst-case assumption is the most robust method. However, it can cause high false-alarm rates by identifying conflicts that are not likely to happen.

- **The probabilistic conflict detection** method is a compromise between nominal and worst-case conflict detections. It computes the probability of conflict between aircraft whose trajectories are described by probability density functions (see Figure 1.2 (c)). The advantage of the probabilistic conflict detection is that different conflict-resolution decisions can be made according to the conflict probability. Probabilistic conflict detection is suitable for assessing the air traffic condition in a large-scale traffic scenario involving a high level of uncertainties, for example in strategic trajectory planning. The probabilistic conflict detection can be carried out in two different approaches. The first one sums up uncertainties to the nominal trajectory, and then computes conflict probability between these trajectories. The second approach constructs the set of all possible trajectories, then weights these trajectories according to their probability of occurring.

1.2.2 Conflict resolution strategies and resolution methods

In this subsection, we present strategies and methods to solve conflicts between aircraft.

Conflict resolution strategies

In a conflict situation involving multiple aircraft, the conflict resolution may be performed based on 1-against-n , pair-wise, or global strategies.

- The **1-against-n** strategy considers one aircraft trajectory after other previously-considered n trajectories according to a given order of priority (e.g. arrival times in the sector). The

aircraft trajectories that are considered first are fixed and become obstacles for the aircraft trajectories that will be considered later. Therefore, the obtained solution depends on the priority order in which the trajectories are considered. This method is effective for a problem involving a small number of aircraft; however, it may not be able to separate all trajectories in some situations.

- In a **pair-wise** strategy, the conflicts between aircraft are solved sequentially considering a pair of aircraft in conflict at a time. This approach is effective; however, it could fail to solve all the conflicts in complex situations.

- The **global** conflict resolution strategy considers the conflicts between all trajectories in the whole traffic situation simultaneously. This approach, designed to manage complex and large-scale traffic situation, is complex and requires more computational effort than the two above-mentioned strategies.

Conflict resolution methods

In order to solve conflicts between aircraft, conflict resolution maneuvers (i.e. horizontal maneuvers, vertical maneuvers, and speed changes) can be computed using, for instance, prescribed, force-field, and optimized methods.

- The **prescribed** method determines conflict resolution maneuvers among a pre-defined set of procedures. The advantage of this method is that the operator (controller, pilot) can be trained to determine the resolution maneuvers so that the response time is decreased. This resolution method is usually used during tactical phases in an open-loop manner; therefore the solution obtained from this method can be too conservative.

- The **force field** method considers each aircraft as a charged particle. The proposed resolution maneuvers is computed based on a relatively simple equation: the electrostatic equation that defines repulsive forces between aircraft. It provides satisfactory solutions in low-density traffic situations. However, it requires a high level of guidance, since the trajectories need to be modified continuously. The most known algorithm is based on navigation function [62]. This method brings a proof for solution. However, the resulting trajectories are not always smooth. Moreover, it does not ensure that aircraft speed stay in a given range.

- The **optimized** method focuses on finding a solution that optimizes a set of pre-defined cost functions. The cost function can include criteria, such as, aircraft separation, fuel consumption, operation costs, time, etc. This method is usually employed in the strategic trajectory deconfliction problem.

1.2.3 Trajectory deconfliction methods

In this subsection, we present the main research works in the literature that address the trajectory deconfliction problem considering large-scale air traffic.

Trajectory deconfliction by genetic algorithms

Aircraft trajectory deconfliction problem that relies on genetic algorithm to solve en-route conflicts between trajectories, taking into account uncertainties of aircraft velocity, is considered in [45]. A 4D trajectory is described by a time sequence of 4D coordinates, sampled with sampling time step of 15 seconds, which have been shown to be small enough to detect every conflict. In the horizontal plane, the initial position of aircraft is represented by a point. Due to the uncertainties, this point becomes a segment, and when there is a change in the speed vector, the aircraft is represented by a 2D convex envelope. In the vertical plane, the aircraft is represented by a cylinder whose height grows with time until the aircraft reaches its requested flight level. Figure 1.3 illustrates the trajectory modeling in the horizontal and vertical planes.

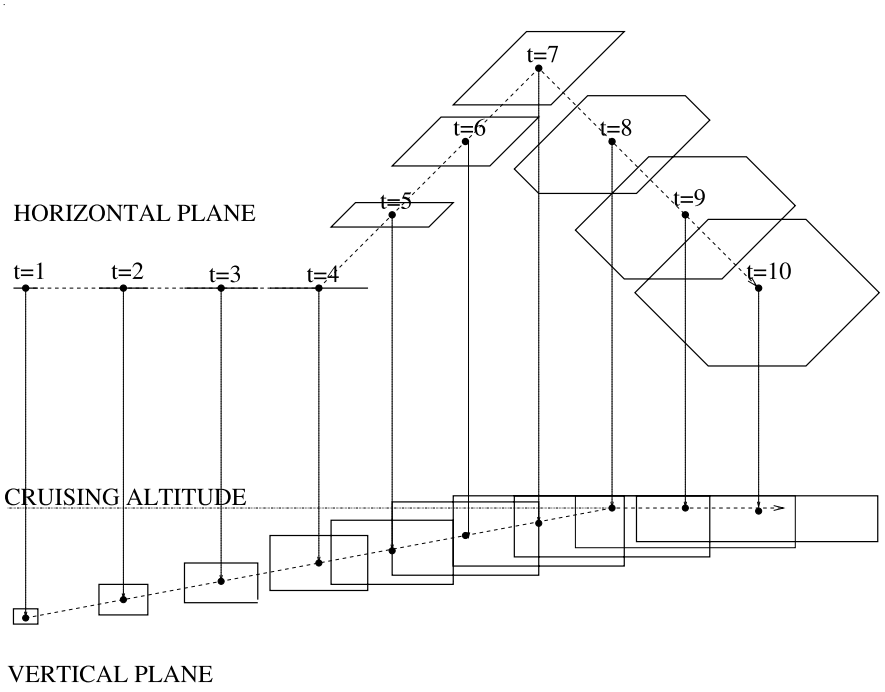

Figure 1.3: Modeling of uncertainty of aircraft trajectory proposed in [45].

The authors propose two conflict-resolution maneuvers: modifying the heading, and modifying the flight level. The maneuver begins at time t_0 and ends at time t_1 as illustrated in Figures 1.4 and 1.5.

The decision variables are the types of maneuver, the starting time t_0 and the ending time t_1. The conflict detection and the conflict resolution are operated on a *sliding time window* T_w, whose length is set between 10 and 15 minutes. The optimization criteria include ensuring all separations between aircraft, minimizing delays, minimizing the number of the maneuvers and the number of aircraft undergoing maneuvers, and minimize the duration of maneuvers.

The solutions are provided by genetic algorithm (GA). The algorithm proposed in [45] is able to solve all conflicts involved in one day of en-route traffic in the French airspace involving 7,540 flights considering different levels of uncertainties (2 % to 30 %) within reasonable computation time (26 - 55 minutes).

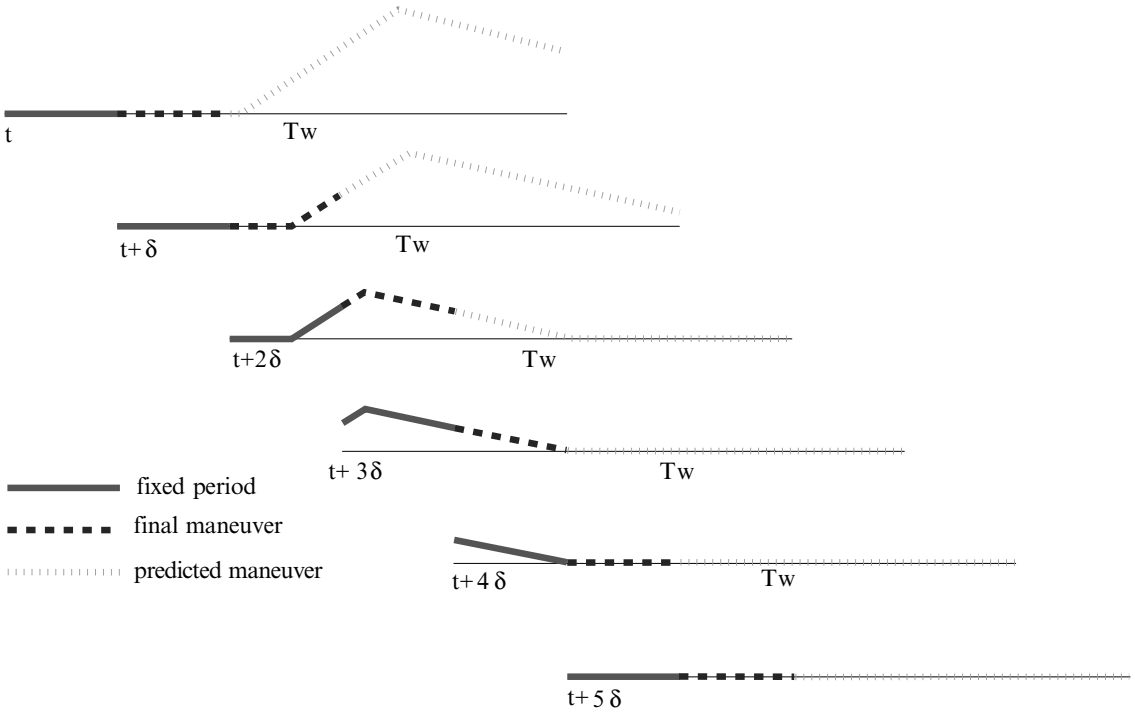

Figure 1.4: Trajectory deconfliction maneuver in the horizontal plane proposed in [45].

Trajectory deconfliction by ground holding and flight level allocation technique

In [15], the authors consider a 4D trajectory deconfliction problem using a ground holding method. The 4D trajectory is defined by a time-sequence of 4D trajectory sample points. The trajectories are sampled with sampling time step of 15 seconds. Potential conflicts between trajectories are detected by an $O(N^2p^2)$ pairwise comparison, where N is the number of trajectories, and p is the total number of sampling points of the N trajectories (see Figure 1.6).

The set of feasible delays that would solve the potential conflicts between trajectories is defined a priori. Uncertainty of aircraft departure times is taken into account by extending the conflict interval (worst-case conflict detection). The optimal conflict-free solutions that minimize the maximal allocated delay is obtained by constraint programming techniques, which rely on two basic concepts: constraint propagation and constructive search. The constraint propagation aims at reducing the search domain of the problem, while the constructive search explores the reduced search domain. The constraint programming solver computes solutions of a given problem by iterating the constraint propagation and the constructive search.

This slot allocation is shown to be powerful for the trajectory deconfliction problem. It is able to deconflict trajectories for a full day of air traffic over the French airspace involving up to 9,500 flights. The proposed trajectory deconfliction method is able to solve all conflicts between trajectories occurring above a given flight level by allocating a single delay to each flight, when uncertainties are not taken into account. However, in presence of take-off time uncertainties, the proposed method must allocate significant delays in order to solve all the conflicts.

In [43], an evolutionary algorithm (EA) is used to solve this ground holding problem.

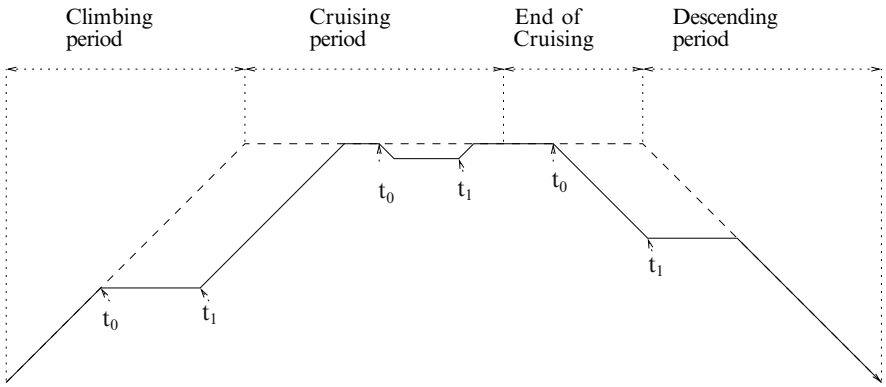

Figure 1.5: Trajectory deconfliction maneuver in the vertical plane proposed in [45].

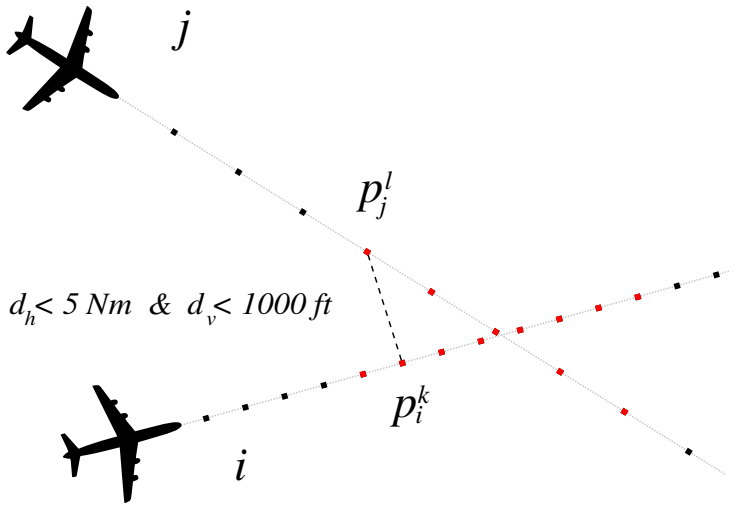

Figure 1.6: Pairwise-comparison conflict-detection technique used in [15].

Since the nation-wide scale air traffic under consideration is non separable, the authors propose to reduce the size of the problem by using a time-sliding window technique: the trajectory deconfliction algorithm only considers aircraft that are scheduled to take off in the next T_w minutes in the future. Numerical results show that EA requires more computation time than constraint programming, but provides better solutions in terms of reduced delays.

To increase the degrees of freedom, the same authors introduce an option to allocate alternative flight levels in [16]. In this work, they propose to reduce the complexity of conflict detection by bounding each trajectory in 2D bounding boxes. Then, the trajectories are probed pairwise; the conflict detection procedure is performed only if the bounding boxes of two trajectory intersect (see Figure 1.7).

In this work, several nearby trajectories are first grouped into flows, then an alternative flight level is allocated to each flow so that two intersecting flows have different flight levels. Then, the remaining conflicts between trajectories belonging to a same flow are subsequently solved by departure-slot allocation technique. The results show advantages of using the flight level allocation technique in terms of reduced delay, in presence of departure time uncertainties. In [7], a flight-level allocation technique is used to address 4D trajectory deconfliction at the

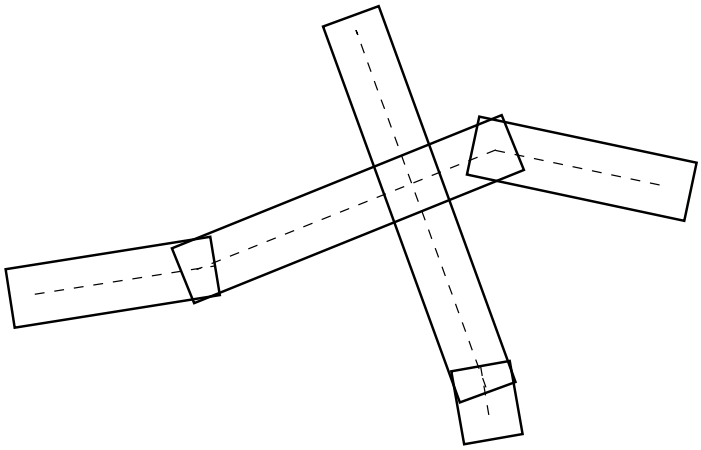

Figure 1.7: Two intersecting trajectories and their bounding boxes [16].

European-continent scale. However, the proposed method yields residual potential conflicts.

Trajectory deconfliction by speed adjustment

Another idea to separate trajectories is based on speed regulations; it is used, for instance, in [29] and [32]. In these works, conflict detection and resolution are performed at two layers with different sampling periods and time windows. Speed regulations introduce additional degree of freedom to the trajectory design. However, it requires numerous extensive and fine-tuned computations, which are not suitable to implement in a large-scale problem.

Trajectory deconfliction by light-propagation algorithm

In [39, 40], a Light Propagation Algorithm (LPA) is introduced to solve potential conflicts between 4D trajectories, to avoid congested area, and to avoid bad-weather areas. The principle of the light-propagation model is similar to those of the force-field conflict resolution method. It mimics the physical propagation of light from one given point towards a destination point. The congested area, bad-weather area, and other aircraft potentially in conflicts are analogous to high refractive-index areas.

In order to find the conflict free path, the wavefront of light in the half space towards the destination is discretized using a discretization angle step $d\theta$, and a discretization time step dt (see Figure 1.8). The discretization angle creates a set of child nodes at each time step. These child nodes determine the region that aircraft has to fly through with velocity v given by $v = \frac{V_{nom}}{I}$, where v_{nom} is the nominal speed of aircraft, and I is the refractive index.

The optimal trajectory (corresponding by the successive child nodes yielding to the shortest-time light ray going to the destination point) that solve conflicts are provided by a Branch-and-Bound (B&B) algorithm. B&B is a method that implicitly enumerates all the feasible solutions of a combinatorial optimization problem, by successively partitioning the solution space and eliminating large subsets of the search space in order to prove that a solution is optimal, without exhaustive search.

The proposed LPA has been tested on a real-world air traffic over the French airspace. The 4D trajectories are iteratively de-conflicted using moving time windows of 21 minutes. For each time window, trajectory segments that are in conflicts are grouped together in a same conflict cluster. Then, each cluster is treated separately. The algorithm is able to solve all the conflicts

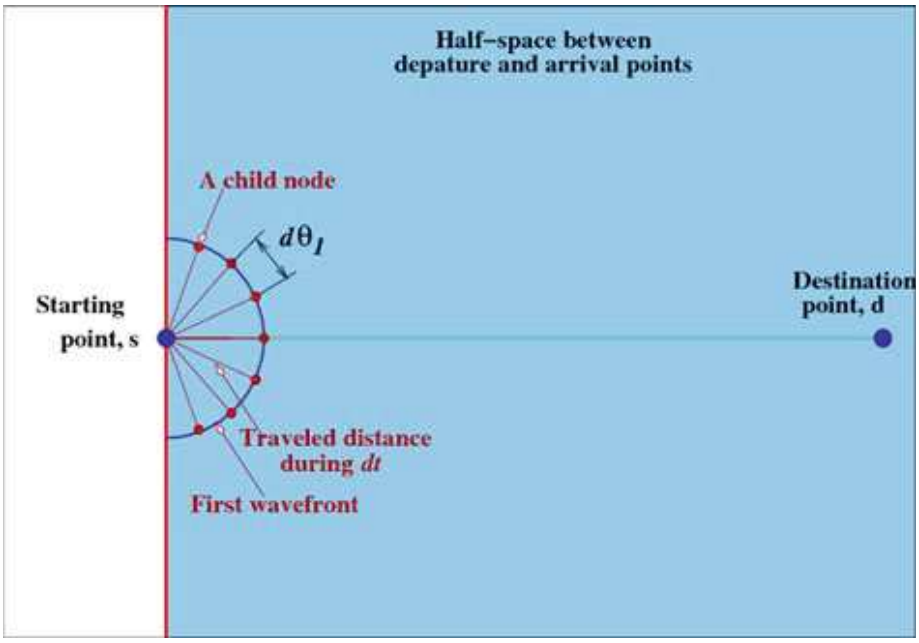

Figure 1.8: Light propagation model proposed in [39].

for the French airspace.

In [40], to improve robustness of aircraft trajectories, uncertainty is modeled as a time segment $[t_{expected} - \delta t_{late}, t_{expected} + \delta t_{early}]$, where $t_{expected}$ is the expected transit time to a given point P, δt_{late} denotes the difference with the latest transit time, and δt_{early} denotes the difference with earliest transit time. The uncertainty increases the difficulty of the problem and reduces the solution space, so that the LPA can remove only 88% of the conflicts.

The remaining conflicts are solved by imposing time constraints called Required Time of Arrival (RTA). The RTA reduces the uncertainty by imposing aircraft to arrive at a given point at a given time with some tolerance. Figure 1.9 illustrates the uncertainty model with an RTA requiring the aircraft to adjust its speed from the time $\frac{2}{3}(RTA - t)$, where t is the predicted starting time.

The introduction of RTA's significantly improves the solution in presence of uncertainty. However, there are still remaining unsolved conflicts.

Real-time conflict-free trajectory optimization

A methodology to optimize and deconflict aircraft trajectories in the horizontal plane, in en-route environment, and in real time is proposed by Matt R. Jardin in [54]. In this work, aircraft trajectories are deconflicted and optimized in the time scale of thirty minutes into the future. The trajectory optimization problem considering a single aircraft is modeled as a cost function minimization with dynamic constraints and constraints on initial and final aircraft states:

$$J_i(x_i) = \int_0^{t_{f_i}} L_i(x_i, t)dt \begin{cases} x_i(0) - x_{0_i} & = 0 \\ x_i(t_{f_i}) - x_{f_i} & = 0 \\ \dot{x} & = f(x_i, t), \end{cases}$$

where $\bar{x}_i$ is the state vector for aircraft i, J_i is the integrated trajectory cost, and the integrated

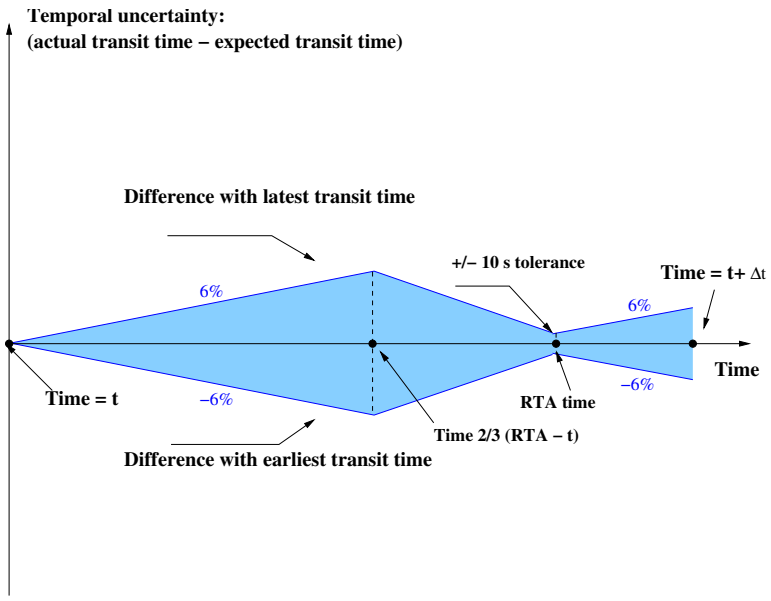

Figure 1.9: Uncertainty model with an RTA point proposed in [40].

objective function, L_i, is defined as the time rate of change of the direct operating costs (e.g. linear combination of fuel and time cost) for aircraft i.

To consider the multi-aircraft trajectory optimization problem, the cost function is written as a sum of N single-aircraft cost functions:

$$J_{tot}(x) = \sum_{i=1}^{N} J_i(x_i),$$

where x is an optimization vector whose i^{th} component is x_i.

Aircraft separation constraints are defined for all time t for all $1 \leq i, j \leq N$ such that $i \neq j$, where $\Delta d_{ij}(x,t)$ is the distance at time t between the aircraft i and aircraft j, and D_{min} is the (given) minimum separation requirement.

The trajectory deconfliction concept proposed in this work focuses on sequentially computing optimal-wind and conflict-free trajectories for each aircraft, considering previously-planned trajectories as obstacles. The wind-optimal route is obtained by using an algorithm called Neighboring Optimal Wind Routing (NOWR). The NOWR algorithm is based on the concept of neighboring feedback control that minimizes objective function by regulating small perturbations around a nominal optimal trajectory. To avoid the computationally inefficient pair-wise conflict search, conflicts between trajectories are detected via a conflict detection algorithm called Conflict Grid (CG).

The basic idea of CG is to store successively the wind-optimal trajectories in a three-dimensional grid (2D space + time). In the deterministic case, trajectories are stored in the CG by setting the value of the corresponding grid cells to one (see Figure 1.10). The size of each grid cell (discretization step) is set according to the minimum separation requirement N_h (typically 5 Nm). If it is found that a grid cell is already occupied, then a conflict is identified

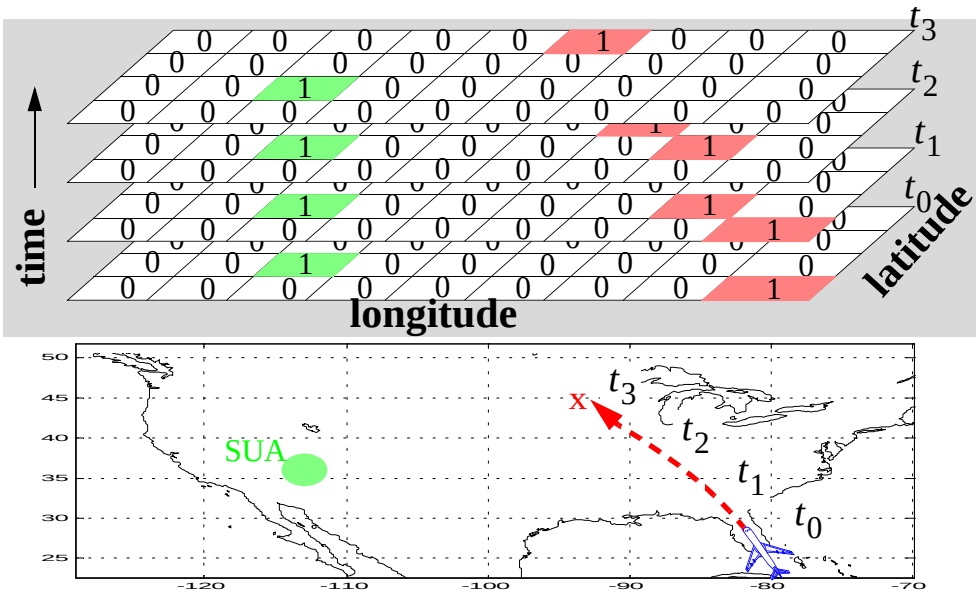

Figure 1.10: The deterministic conflict grid method proposed in [54].

and then conflict-resolution maneuvers are computed.

The conflict grid can also take into account active constraints such as uncertainty on the aircraft trajectories, weather storms, special use airspace, etc. For this stochastic case, the CG stores in each grid cell the probability that at least one such active constraint exists in that grid cell (see Figure 1.11).

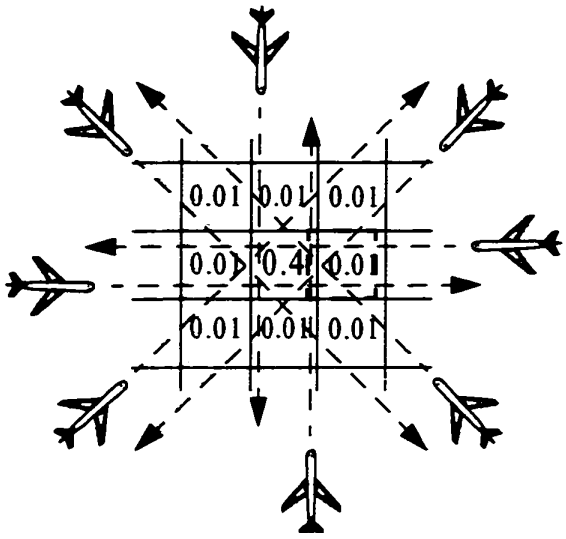

Figure 1.11: The stochastic conflict grid method to compute probability of conflict between multiple aircraft proposed in [54].

Once a conflict is detected, or when conflict probability is larger than a (pre-defined) threshold value, the proposed methodology computes resolution maneuvers by generating a perturbation of the wind-optimal route using a concept of *pseudo-shear*, introducing pseudo wind shear at the nearest (discretized) wind control point (see Figure 1.12). Once the pseudo wind shear is introduced, a new wind optimal route (with respect to the sum of real wind shear and

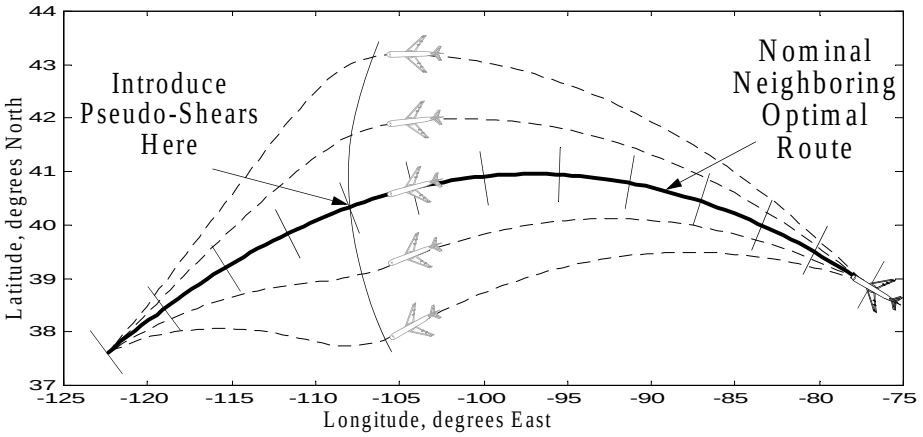

Figure 1.12: Conflict resolution maneuver based on pseudo-shear concept proposed in [54].

pseudo wind shear) is computed. The conflict resolution algorithm iterates until the value of the pseudo-shear that yields conflict free trajectory is found.

The proposed algorithm is able to compute conflict free optimal wind route for one day air traffic over the U.S. considering single flight level. However, the proposed algorithm is limited to the en-route air traffic in the horizontal plane.

Automated conflict detection and resolution

In the free-flight operation context, a methodology to estimate conflict probability between aircraft in presence of trajectory prediction error, and to solve the identified conflict is presented in [48]. The proposed method estimates probability of conflict during a certain period of look-ahead time. The prediction error is modeled as normally distributed with zero mean over ellipses in the horizontal plane, or as ellipsoids in three-dimensional space. Figure 1.13 illustrates the growth of uncertainty ellipses in the along-track direction.

As the prediction errors are modeled as normally distributed, the two error covariances for a pair of aircraft can be combined into a single equivalent error covariance of the relative position of one aircraft with respect to the other. This combined covariance enables us to consider one aircraft as the"stochastic" aircraft, and assign the other aircraft as the "reference" aircraft, as illustrated in Figure 1.14.

The conflict-resolution module aims at modifying the heading of the aircraft in order to maintain the post-resolution conflict probability below a pre-defined threshold conflict probability value, P_{rs}. It is shown that setting the value of P_{rs} in the range of 0.05 - 0.15 yields effective resolution trajectories.

To avoid the time-consuming pair-wise $(N(N-1)/2)$ comparison, where N is the total number of aircraft, the authors propose three methods to limit the computational load in the conflict search algorithm: trajectory-pair pruning, minimizing-separation computation, and time skipping.

The trajectory-pair pruning aims at verifying whether two trajectories are potentially in conflict and in need of further examination, by determining whether they are spatially exclusive in either altitude or horizontal positions.

The minimizing-separation computation method eliminates unnecessary computations based

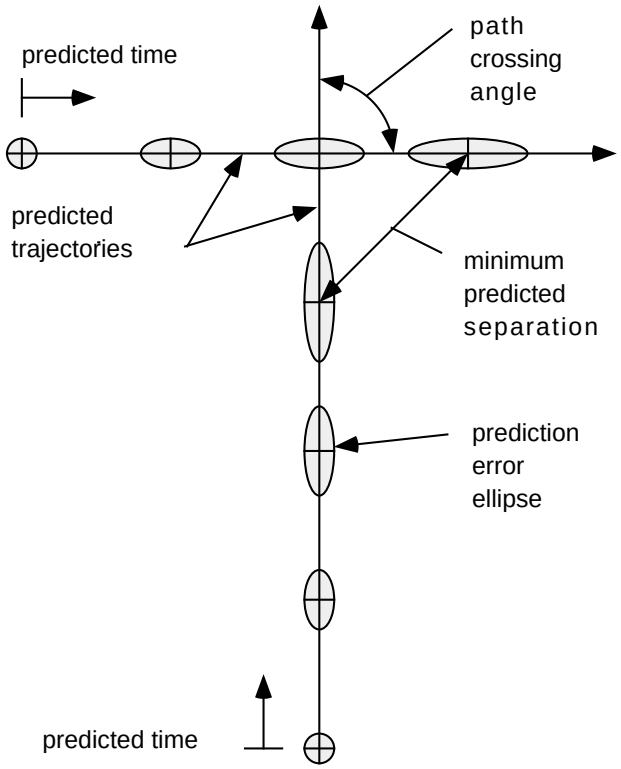

Figure 1.13: Uncertainty of aircraft position in the horizontal plane, modeled as ellipses in [48].

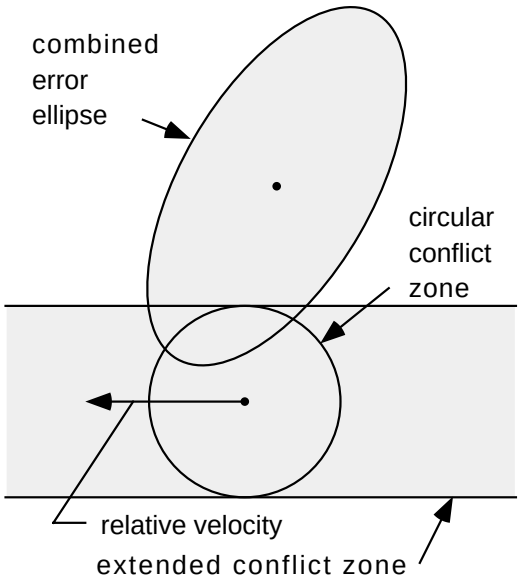

Figure 1.14: Uncertainty of aircraft position in the horizontal plane, modeled as reference aircraft and stochastic aircraft in [48].

on the result of computations already performed (e.g. if the aircraft are already separated in the vertical plane, it is not necessary to compute the horizontal distance).

Finally, the time-skipping method skips the conflict search on the given pair of trajectories when it can surely determine from the two previous time steps that no conflict is possible.

1.3 Conclusions

In this chapter, we reviewed various air traffic decongestion and trajectory deconfliction methodologies from the literature. The simplest strategy to address the air traffic management problem is the ground holding strategy, that focuses on delaying the aircraft on the ground in order to respect the capacity constraints at the destination airport. This strategy is effective in the cases where congestion usually occurs at the airport. However, in Europe, where most congestion is in the airspace sectors, the flow of air traffic between origin and destination airports has to be considered.

In the framework of new ATM paradigm, several researches aim at directly solving conflicts between aircraft. There exists several trajectory deconfliction methodologies relying on, for example, genetic algorithms, speed adjustment, light propagation algorithm, etc. However, none of the proposed methodologies is able to solve globally the trajectory deconfliction problem due to its size and complexity. Most of the algorithms proposed in the literature rely on the moving time window strategy to reduce the size of the problem. This strategy is effective for conflict detection and resolution in tactical phases. However, when high-density traffic is involved, it tends to fail to solve all conflicts.

Numerous optimization methods have been applied to the air traffic management problem. Exact methods such as branch-and-bound, linear programming, and constraint-programming methods can guarantee convergence to a global optimum to the problem. However, for the NP-hard problem we are addressing here, the computation time required to find a global optimum grows exponentially with the size of the problem. Metaheuristic optimization methods have been shown to provide good solutions to the air traffic management problem within reasonable computation time. During recent years, hybrid-metaheuristic optimization methods have been successfully applied on several hard real-world problems, providing good results within much shorter computation time than using exact methods or a single metaheuristic.

Chapter 2

Mathematical model of the strategic trajectory planning methodology

This chapter sets the mathematical framework of the strategic trajectory planning methodology we are proposing. First, the assumptions, simplifications, and given data are presented. Then, the allowed trajectory deconfliction maneuvers are described. Next, a definition of interaction between trajectories is introduced. After that, a mathematical formulation of the strategic trajectory planning problem is proposed, and its complexity is discussed. Finally, a methodology to compute the value of the objective function is presented.

2.1 Problem description, given data, assumption, and simplifications

The strategic trajectory planning problem that we aim to solve in this work considers a set of flight plans for a given day. The objective is to find alternative 4D trajectory for each flight, so as to minimize the total interactions between trajectories. Interaction is, again, the situation occurring in the planning phase when two or more aircraft trajectories compete for the same space at the same period of time.

The given data include:

- An initial set of flight plans. Such flight plans are then used to create N initial (or *nominal*) 4D trajectories by means of a fast time simulation (BADA-Base of Aircraft Data model). These 4D trajectories are labelled with index $i = 1, 2, \ldots, N$;
- The maximum allowed delay departure time shift that can be allocated to each flight i, denoted by δ_d^i;
- The maximum allowed advance departure time shift that can be allocated to each flight i, denoted by δ_a^i;
- The maximum allowed coefficient of route length extension that can be allocated to each flight i, denoted by d_i;

Each initial 4D trajectory, i, is defined by a set of 4D points (x, y, z, t) sampled with a (given) constant sampling time, t_s. The initial 4D trajectory is composed of three parts; the *initial en-route segment* is the shortest possible route between the origin and the destination airports (*great circle path*) and the two (extremity) Terminal Maneuvering Area (TMA) parts of the trajectory, as illustrated in Figure 2.1 and Figure 2.2.

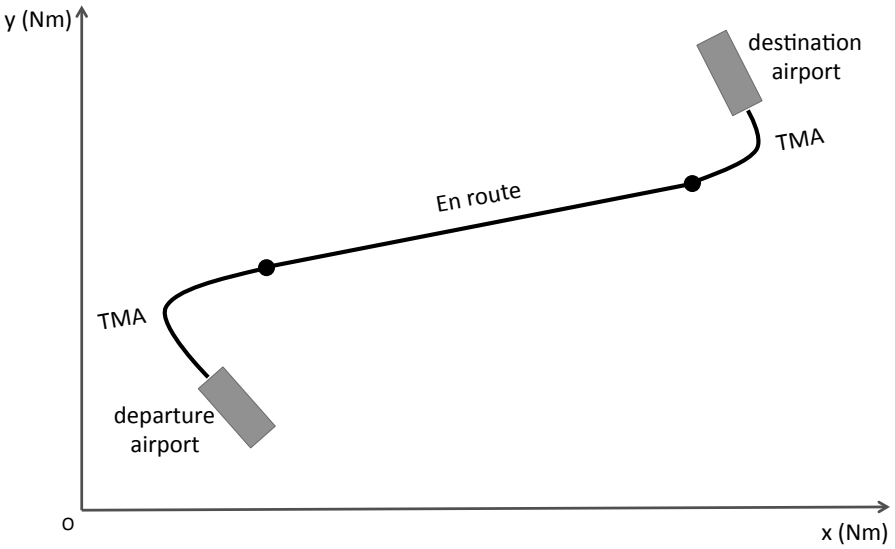

Figure 2.1: A given (initial) trajectory in the horizontal plane consisting of departure, en-route, and two extremity TMA segments.

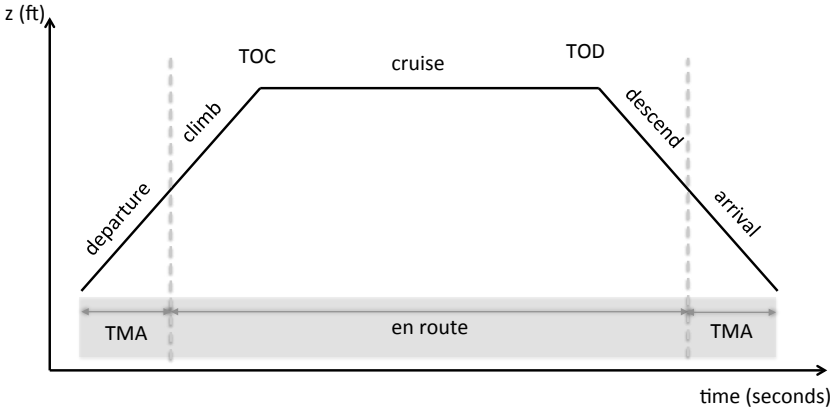

Figure 2.2: A given (initial) trajectory in the vertical plane consisting of departure, en-route, and arrival segments.

In this work, the following assumptions and simplifications are made. The initial (given) velocity profile is assumed to be optimal. To respect the standard departure and arrival procedures in the TMAs, trajectory deconfliction maneuvers in the 3D space domain are limited only to the en-route segment. For TMA areas, only actions in time domain (not in space) will be considered. The airspace is considered as a Euclidean space. Latitudes and longitudes on the

earth surface are projected into (x, y) coordinates. The altitude, in feet, will be represented by the z coordinate. Finally, aircraft are assumed to be able to follow a given 4D trajectory with high precision.

2.2 Trajectory separation maneuvers

In this section, we describe two possible trajectory separation maneuvers, that we are considering in order to generate alternative 4D trajectories that minimize the total interactions:

- shifting the departure time;
- modifying the route (horizontal flight path profile).

In order to minimize the interaction between trajectories, the trajectories must be separated in 3D space and in time. To separate trajectories in the time domain, one can adjust the departure time of each aircraft. However, with increasing traffic volume, large amount of departure-time shift may have to be distributed to each flight in order to separate all trajectories. To limit the amount of time shifts necessary to deconflict all trajectories, the 4D trajectories can also be separated by acting also on the 3D routes.

Since the resolution of this problem is implemented at the strategic level, the interaction-reduction problem can be solved simultaneously on both the spatial and the temporal dimensions. The alternative departure time and the alternative trajectory to be allocated to each flight are modeled as follows.

2.2.1 Alternative departure time

The departure time of each flight, i, can be shifted by a positive (delay) or a negative (advance) time shift denoted by δ_i. The departure time, t_i, of flight i is therefore

$$t_i = t_{i,0} + \delta_i,$$

where $t_{i,0}$ is the initially-planned departure time of flight i. Following common practice in airports, the set of possible values for δ_i will be discrete.

By considering negative (and positive) time shifts, we consider the regular times of departure of aircraft. If δ_i is restricted to be only positive, this corresponds to the regular ground delay program.

2.2.2 Alternative trajectory design

An alternative route should not deviate too much from the nominal route. It should also be computed in a short computation time because of the large-scale applications we wish to solve. To respect the given optimal cruise level, altitude profile, and standard departure and arrival procedures, in this work we concentrate on modifying only the horizontal profile of the en-route segment of a given 4D trajectory. To generate an alternative route, we proposed to modify the (given) initial horizontal flight profile of the en-route segment of a trajectory, by placing a set of virtual waypoints near its initial en-route segment, and then by connecting the successive

waypoints with straight-line segments. Figure 2.3, illustrates a possible alternative horizontal profile for a given trajectory constructed with $M = 2$ virtual waypoints.

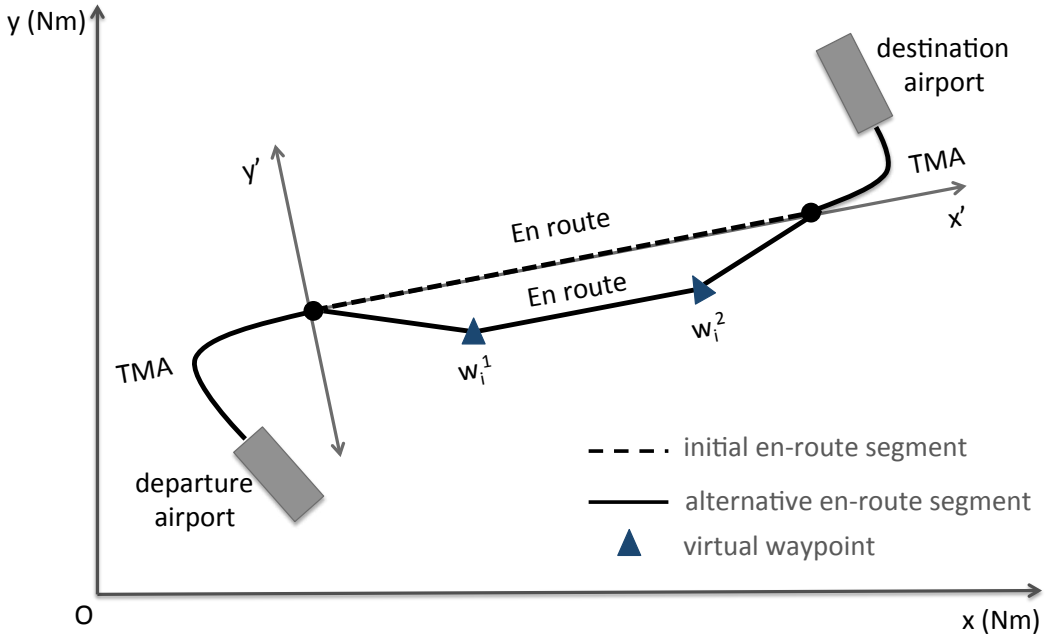

Figure 2.3: An alternative horizontal profile for a given trajectory, i, constructed with $M = 2$ virtual waypoints.

To simplify the presentation of the location of each virtual waypoint, we call *longitudinal axis* (x') the axis that is tangent to the initial en-route segment, and the *lateral axis* (y') is the axis that is perpendicular to the longitudinal axis (in the horizontal plane). Furthermore, we normalize the coordinates of the $x'y'$-reference axes, as illustrated in Figure 2.4, so that the location of virtual waypoints for every trajectory can be represented in the same manner. The position of each waypoint will be defined using these normalized relative $x'y'$-reference axes.

We define, for each flight i, a vector, w_i, of virtual waypoints (our optimization variables) used to control the trajectory shape of flight i:

$$w_i = (w_i^1, w_i^2, \ldots, w_i^M),$$

where M denotes the number of virtual waypoints that the user is allowed to introduce, where $w_i^m = (w_{ix'}^m, w_{iy'}^m)$ is the m^{th} virtual waypoint of trajectory i, and where $w_{ix'}^m$ and $w_{iy'}^m$ are the normalized longitudinal and lateral components of w_i^m respectively. Therefore, the longitudinal component of the m^{th} virtual waypoint, w_i^m, is:

$$w_{ix'} L_{i,0},$$

and the lateral component of the m^{th} virtual waypoint, w_i^m, is:

$$w_{iy'} L_{i,0},$$

where $L_{i,0}$ is the length of the initial planned en-route segment.

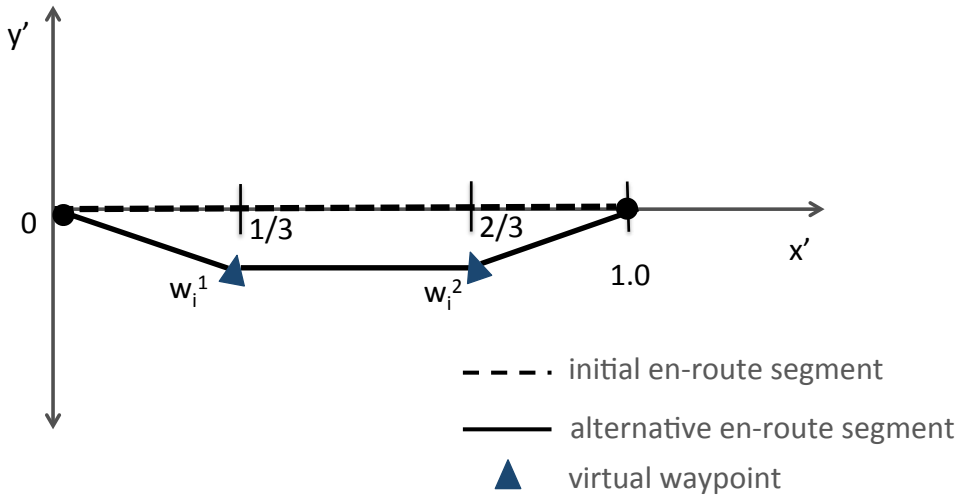

Figure 2.4: An initial and an alternative en-route segment of trajectory, i, on normalized $x'y'$-reference axes, constructed with $M = 2$ virtual waypoints.

Remark that the alternative trajectory will yield an increase in flight duration when compared with the initial trajectory. To compensate this increased flight duration, the altitude profile must be updated to avoid a premature descent. Let T_{ext} be the increased flight duration. In the case of a regional flight whose all flight phases (departure, climb, cruise, descent, and arrival) are executed in the considered airspace area, the altitude profile is updated by extending the cruise phase (constant-level) at the top of descent (TOD) for a duration T_{ext}, as illustrated in Figure 2.5.

On the other hand, for a flight whose origin or destination airports are outside of the current airspace area, the cruise phase can take place inside or outside the current airspace area. This yields six possible configurations of the initial altitude profile, as illustrated in Figure 2.6. Let z_{max} be the maximum altitude that the flight will attain in the current airspace area. In this case, the vertical profile can be updated by extending the flight with a constant altitude z_{max} for a duration T_{ext} (the aim is to preserve the given optimal profile and the same climb / descent slopes).

More precisely, in case 1 and case 3, the TOD takes place in the current airspace area, therefore, the vertical profile is updated by extending the flight with a constant altitude z_{max} for a duration T_{ext}. In case 2 and case 5, only a part of cruise phase takes place in the current airspace area, the vertical profile is updated by extending the flight with a constant cruise altitude for a duration T_{ext}. Finally, in case 4 and case 6, only climb or descent phase takes place in the current airspace area, the vertical profile is updated by extending the flight with a constant cruise altitude z_{max} for a duration T_{ext}.

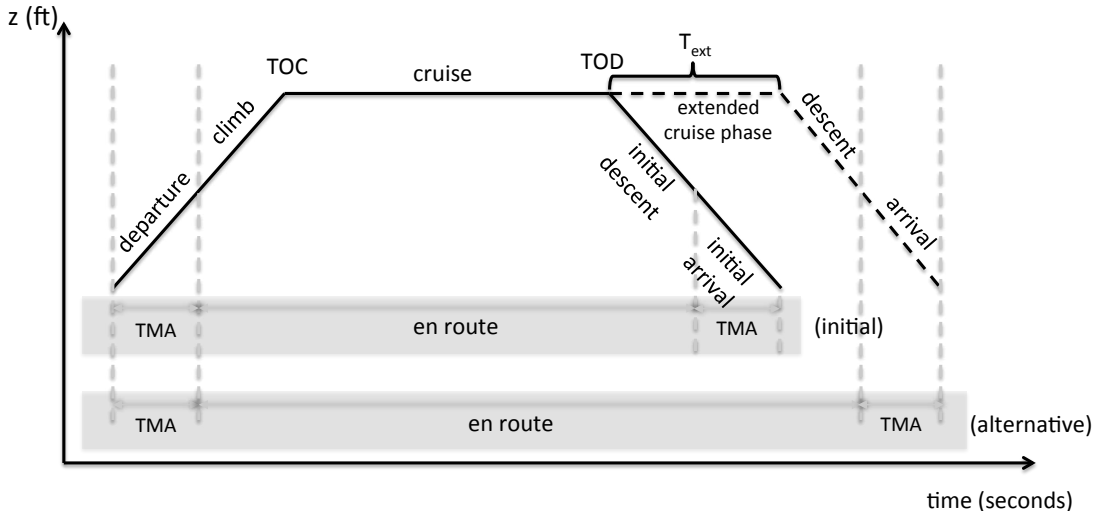

Figure 2.5: Altitude-profile update: extending cruise phase at the top of descent (TOD).

2.3 Optimization formulation

In this section, we present an optimization formulation of the strategic 4D trajectory planning problem. The strategic 4D trajectory planning methodology using route / departure-time allocation can be formulated as an optimization problem aiming at minimizing the interaction between trajectories.

Given data. A problem instance is given by:

- A set of N initial (nominal) discretized 4D trajectories;
- The sampling time step: t_s;
- For each flight i, for $i = 1, \ldots, N$:
  - The initial planned departure time: $t_{i,0}$;
  - The maximum allowed advance departure time shift: $\delta_a^i < 0$;
  - The maximum allowed delay departure time shift: $\delta_d^i > 0$;
  - The length of the initial planned en-route segment: $L_{i,0}$;
  - The maximum allowed route length extension coefficient: $0 \leq d_i \leq 1$;
  - The user-defined parameters controlling the dimensions of the feasible domains for locating the virtual waypoints: a_i and b_i.
  - The number of 4D discretization points on trajectory i: K_i.

Model parameters. Here are user-provided parameters necessary to define our optimization model and the proposed methodology.

- The interpolation sampling time step: t_{interp};
- The number of allowed virtual waypoints: M;

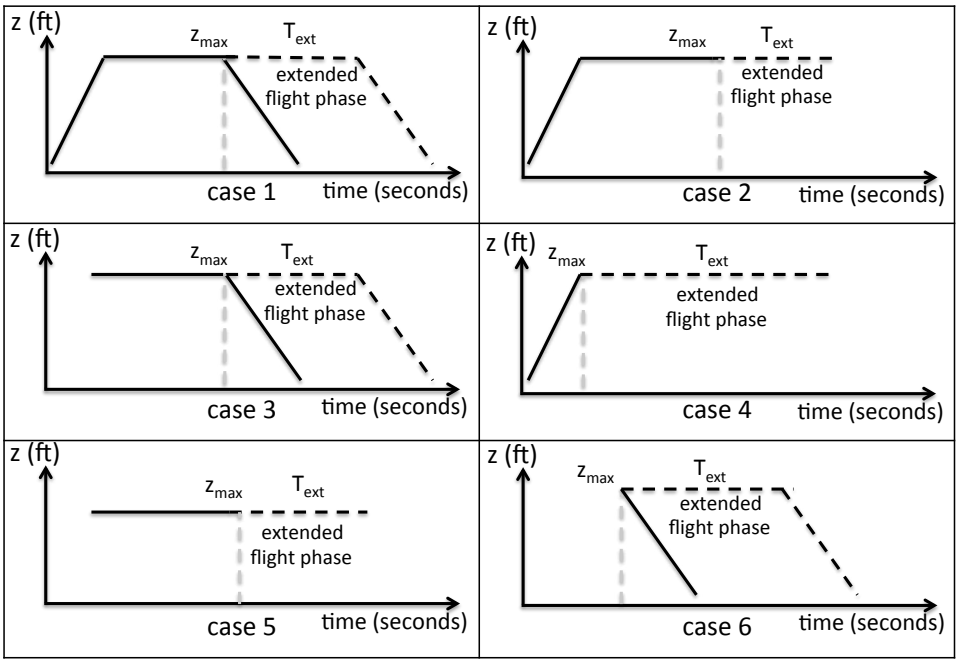

Figure 2.6: Altitude profile update: six possible ways to extend the trajectory at maximum altitude.

- The discretization time step for the possible delay / advance departure-time shift interval: δ_s;
- The constant that defines the number of possible locations of each virtual waypoint: N_w;

The role of these parameters will be made more precise in the sequel.

Decision Variables. As mentioned above, we consider two ways to separate trajectories. In the time domain, one can use a departure-time shift, δ_i, associated to each flight, i. In the 3D space, one can rely on a vector, w_i, of virtual waypoint locations, $w_i := (w_i^1, w_i^2, \ldots, w_i^M)$ associated to each flight, i, where M is the number of virtual waypoints.

Let us set the compact vector notation:

$$\boldsymbol{\delta} := (\delta_1, \delta_2, \ldots, \delta_N),$$

and

$$\mathbf{w} := (w_1, w_2, \ldots, w_N).$$

Therefore, the decision variables of our route / departure-time allocation problem can be represented by the vector:

$$u := (\boldsymbol{\delta}, \mathbf{w}).$$

We shall denote by u_i the components of u. It is a vector whose components are related to the modification of the i^{th} trajectory, thereby:

$$u_i = (\delta_i, w_i)$$

Constraints. The above optimization variables must satisfy the following constraints:

- ***Allowed departure time shift***. Departure time of flight i is given by an auxiliary optimization variable, t_i, which is linked to the above decision variables, as follows:

$$t_i = t_{i,0} + \delta_i, \tag{2.1}$$

where $t_{i,0}$ is the initial planned departure time of flight i (given data).

In practical problems, passengers may have to transfer from one flight to another in order to get to their final destination. This generates precedence constraints stipulating that certain flights must arrive at the airport before the departure of others. In addition, each aircraft may be used for several flights on a same day. This raises a constraint of minimum *rotation* time (time required to disembark the passengers, to service the aircraft, and to embark passengers) between flights. Moreover, the number of aircraft departing from and arriving to a given airport at any given time are limited by the airport capacity. These constraints are not taken into account in this work due to the lack of data from airlines and from airports. However, they can easily be handled by, for example, pre-processing the set of feasible time shifts of each flight.

Since it is not reasonable to delay or to advance departure times for too long, the departure time shift, δ_i, is assumed to be limited to lie in the interval

$$[\delta_a^i, \delta_d^i]. \tag{2.2}$$

Common practice in airports conducted us to rely on a discretization of this time interval. Given the (user-defined) time-shift step size δ_s, this yields $N_a^i := \frac{-\delta_a^i}{\delta_s}$ possible advance slots and $N_d^i := \frac{\delta_d^i}{\delta_s}$ possible delay slots of flight i. Therefore, we define the set, Δ_i, of all possible departure time shifts of flight i by

$$\Delta_i := \{-N_a^i.\delta_s, -(N_a^i - 1).\delta_s, \ldots, -\delta_s, 0, \delta_s, \ldots, (N_d^i - 1).\delta_s, N_d^i.\delta_s\}. \tag{2.3}$$

- ***Maximal route length extension***. The alternative trajectory induces route length extension which causes an increase of fuel consumption and flight time. Therefore, it should be limited so that it is acceptable by the airline.

Let $0 \leq d_i \leq 1$ be the maximum allowed route length extension coefficient of flight i (model parameter to be set by the user). To restrain the route length extension, the alternative en-route profile of flight i must satisfy:

$$L_i(w_i) \leq (1 + d_i), \tag{2.4}$$

where $L_i(w_i)$ denotes the normalized length of the alternative en-route profile determined by w_i. The normalized length $L_i(w_i)$ can be straightforwardly computed once the position of the waypoints (contained in the vector w_i) is known as it can be seen from Figure 2.4. Constraint can in fact be implicitly satisfied by restricting the set of possible waypoint

locations (as will be described below).

- ***Allowed waypoint locations.*** To limit the search space, to prevent undesirable sharp turns, and to restrain the route length extension, we bound the possible location of each virtual waypoint. To avoid sharp turns, the longitudinal position of the virtual waypoints should not be too close to each other.

  Let $W_{ix'}^m$ be a set of all possible normalized longitudinal locations of the m^{th} virtual waypoint on trajectory i. For simplicity, for each trajectory i, the normalized longitudinal component, $w_{ix'}^m$, is set to lie in the interval:

$$W_{ix'}^m := \left[\left(\frac{m}{1+M} - b_i\right), \left(\frac{m}{1+M} + b_i\right)\right]. \tag{2.5}$$

  To obtain a regular trajectory, the normalized longitudinal component of two adjacent waypoints must not overlap, i.e.

$$\left(\frac{m}{1+M} + b_i\right) < \left(\frac{m+1}{1+M} - b_i\right) \tag{2.6}$$

  and hence the user should choose b_i so that

$$b_i < \frac{1}{2(M+1)}. \tag{2.7}$$

  Let $W_{iy'}^m$ be a set of all possible normalized lateral locations of the m^{th} virtual waypoint on trajectory i. Similarly, the normalized lateral component, $w_{iy'}^m$, is restricted to lie in the interval:

$$W_{iy'}^m := [-a_i, a_i], \tag{2.8}$$

  where $0 \leq a_i \leq 1$ is a (user-defined) model parameter chosen a priori so as to satisfy (2.4).

  The sets $W_{ix'}^m$ and $W_{iy'}^m$ can be modeled as discrete or continuous sets. In a first approach, we choose to discretize uniformly and symmetrically the interval $[-a_i, a_i]$ into $2N_w + 1$ possible values, for $i = 1, \ldots, N$, where $N_w \in \mathbb{N}^+$ is a (user-defined) model parameter. Therefore, the lateral location $w_{iy'}^m$, of the m^{th} virtual waypoint of flight i can be chosen from the set:

$$W_{iy'}^m := a_i \left\{-1, -\frac{N_w-1}{N_w}, -\frac{N_w-2}{N_w}, \ldots, -\frac{1}{N_w}, 0, \frac{1}{N_w}, \ldots, \frac{N_w-2}{N_w}, \frac{N_w-1}{N_w}, 1\right\} \tag{2.9}$$

  For simplicity, in this first approach, let us assume that the coefficient b_i is set to zero. Therefore, the set of possible normalized longitudinal components, $w_{ix'}^m$, is reduced to the singleton

$$W_{ix'}^m := \left\{\frac{m}{1+M}\right\}. \tag{2.10}$$

  This discretization scheme involves $(2N_w + 1)^M$ possible alternative trajectories for each

flight i. Figure 2.7 illustrates an example of possible alternative horizontal profiles of a flight when $M = 2$, and $N_w = 3$ (therefore 49 possible route choices).

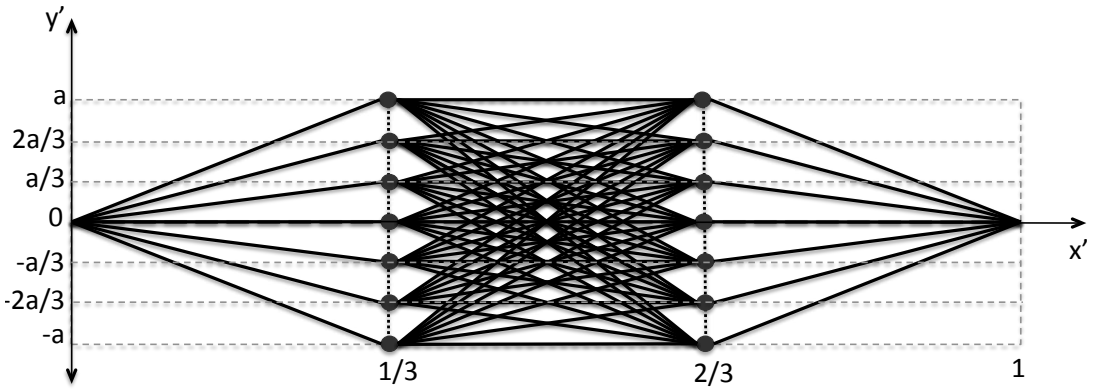

Figure 2.7: All possible horizontal profiles of the en-route segment using discrete virtual waypoints constraint for $M = 2$ virtual waypoints and a discretization step size, $N_w = 3$.

Let us now consider a second approach where we model the sets $W^m_{ix'}$ and $W^m_{iy'}$ as continuous sets. This yields a rectangular shape for the possible locations of the virtual waypoint w^m_i. Figure 2.8 illustrates the possible locations of $M = 2$ virtual waypoints.

Finally, in order to restrain the maximum route extension, the model parameters a_i and b_i must be chosen by the user so that:

$$\max\{L'_i(w_i)|w_i \in W_{ix'} \times W_{iy'}\} \leq (1 + d_i), \tag{2.11}$$

which can be straightforwardly done (for instance, given the value of b_i, one computes the largest possible value for a_i).

Objective function. In strategic trajectory planning, we focus more on separating aircraft trajectories than on solving each conflict locally. Therefore, we voluntarily employ a term *interaction between trajectories*, which is, roughly speaking, a situation which occurs in the planning phase, when more than one trajectory compete for the same space at the same period of time. It is different from the *conflict* situation, which corresponds simply to a violation of the minimum *separation* norm (i.e. 5 Nm horizontally and 1,000 ft vertically). Additional separation conditions, such as time separation, topology of trajectory intersection, distance between trajectories, etc. can also be taken into account in such a concept of interaction.

Here is how we evaluate this interaction-based objective function, for given values of the decision variables $u = (\boldsymbol{\delta}, \mathbf{w})$. One must first discretize each of the N resulting alternative trajectories into a sequence of 4D points: $\{P_{i,k}(u_i)\}_{k=1}^{K_i}$, $i = 1, 2, \ldots, N$. We shall use in the remaining part of this thesis the notation $P_{i,k} = (x_{P_{i,k}}, y_{P_{i,k}}, z_{P_{i,k}}, t_{P_{i,k}})$ to designate the k^{th} 4D point on trajectory i. Each of these points depends only on $u_i = (\delta_i, w_i)$, the i^{th} component of the optimization variable vector u.

For the sake of simplicity, let us assume that minimizing the interaction between trajectories boils down to minimizing the total number of conflicts between aircraft. Consider for example

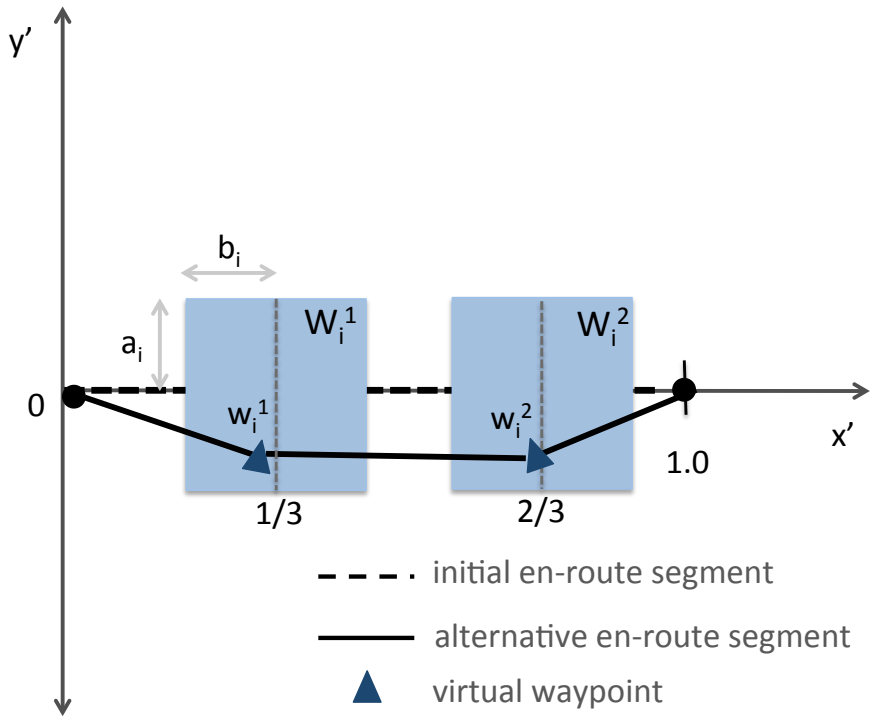

Figure 2.8: Rectangular-shape sets of the possible locations of $M = 2$ virtual waypoints, for trajectory i.

the two trajectories A and B in Figure 2.9. A conflict between $P_{A,k}$ and $P_{B,l}$ occurs when the horizontal distance, $d_h = \sqrt{(x_{P_{A,k}} - x_{P_{B,l}})^2 + (y_{P_{A,k}} - y_{P_{B,l}})^2}$, is less than N_h, the vertical distance, $d_v = |z_{P_{A,k}} - z_{P_{B,l}}|$, is less than N_v, and the difference of the arrival times of both aircraft, $d_t = |t_{P_{A,k}} - t_{P_{B,l}}|$ is zero for some $1 \leq k \leq K_A$ and some $1 \leq l \leq K_B$, where K_A and K_B are the numbers of discretization points on trajectories A and B respectively.

Let us further define an *interaction at a point* $P_{i,k}(u_i)$ to be the sum of all the conflicts associated to point $P_{i,k}(u_i)$; we denote it by $\Phi_{i,k}(u)$. Remark that it depends also on the other trajectories $j \neq i$. More precisely,

$$\Phi_{i,k}(u) := \sum_{\substack{j=1 \\ j \neq i}}^{N} \sum_{l=1}^{K_j} \mathcal{C}(P_{i,k}(u_i), P_{j,l}(u_j)), \tag{2.12}$$

where K_j are the number of sampling points for trajectory j, and

$$\mathcal{C}(P, Q) := \begin{cases} 1 & \text{if point } P \text{ is in conflict with point } Q \\ 0 & \text{otherwise.} \end{cases} \tag{2.13}$$

Figure 2.9 illustrates interaction in the horizontal plane between $N = 3$ trajectories measured at point $P_{B,4}$, yielding $\Phi_{B,4} = 2$.

We define the *interaction, Φ_i, associated to trajectory i*, as:

$$\Phi_i(u) := \sum_{k=1}^{K_i} \Phi_{i,k}(u). \tag{2.14}$$

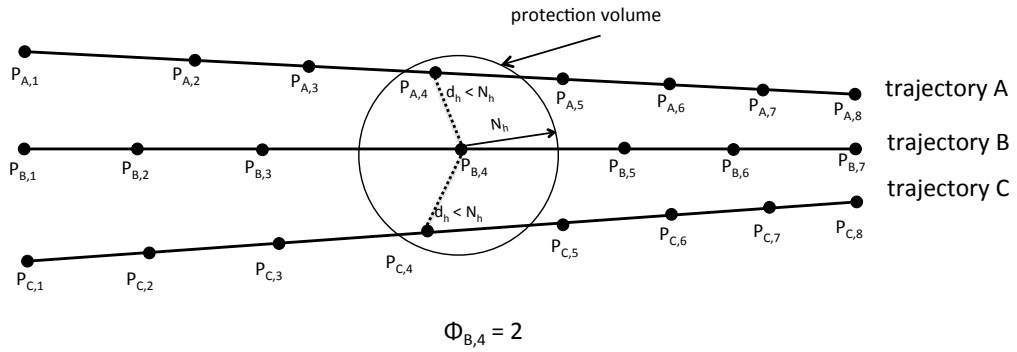

Figure 2.9: Interactions, $\Phi_{B,4}$, at sampling point $P_{B,4}$ of trajectory B.

Finally, *interaction between trajectories*, Φ_{tot}, for a whole traffic situation is simply defined as:

$$\Phi_{tot}(u) := \sum_{i=1}^{N} \Phi_i(u) = \sum_{i=1}^{N} \sum_{k=1}^{K_i} \Phi_{i,k}(u). \tag{2.15}$$

One can observe that the measurement of the interaction between trajectories implicitly take into account the duration of conflict between trajectories. A practical methodology to compute the value of the objective function in a large-scale context is presented in Section 2.4.

The optimization formulation we are about to present involves determining values for the optimization variables w_i and δ_i for each flight $i = 1, 2, \ldots, N$ so as to minimize, $\Phi_{tot}(u)$, the interaction between the N given flights.

In the case where we consider the discrete constraints (2.9) and (2.10) for the possible virtual waypoint locations, the interaction minimization problem can be formulated as a combinatorial optimization problem, as follows:

$$\begin{aligned} &\min_{u=(\boldsymbol{\delta},\mathbf{w})} \Phi_{tot}(u) \\ &\text{subject to} \\ &\quad \delta_i \in \Delta_i, \quad \text{for all } i = 1, \ldots, N \\ &\quad w^m_{ix'} \in W^m_{ix'}, \quad \text{for all } i = 1, \ldots, N, m = 1, \ldots, M \\ &\quad w^m_{iy'} \in W^m_{iy'}, \quad \text{for all } i = 1, \ldots, N, m = 1, \ldots, M, \end{aligned} \tag{P1}$$

where $\Phi_{tot}(u)$ is defined by (2.15), and Δ_i, $W^m_{ix'}$, and $W^m_{iy'}$ are defined by (2.3), (2.10), and (2.9) respectively.

In the case where the sets of possible virtual waypoints are continuous sets, the interaction minimization problem can be formulated as a mixed-integer optimization problem, as follows:

$$\min_{u=(\boldsymbol{\delta},\mathbf{w})} \Phi_{tot}(u)$$

subject to

$$\begin{aligned} &\delta_i \in \Delta_i, \quad \text{for all } i = 1,\dots,N \\ &w^m_{ix'} \in W^m_{ix'}, \quad \text{for all } i = 1,\dots,N, m = 1,\dots,M \\ &w^m_{iy'} \in W^m_{iy'}, \quad \text{for all } i = 1,\dots,N, m = 1,\dots,M, \end{aligned} \tag{P2}$$

where $W^m_{ix'}$, and $W^m_{iy'}$ are defined by (2.5), and (2.8) respectively.

The combinatorial optimization problem (P1) involves the following complexity. The number of possible trajectory solutions for flight i is:

$$(N^i_a + N^i_d + 1).(2N_w + 1)^M.$$

Therefore, the total number of possible solutions is:

$$\prod_{i=1}^{N} (N^i_a + N^i_d + 1).(2N_w + 1)^M,$$

where N^i_a and N^i_d are the possible advance and delay slots of flight i respectively, and N_w are the parameters that define the number of possible values of the longitudinal and lateral components of each waypoint w^m_i, respectively.

If, for instance, for an air traffic involving $N = 10,000$ flights, we let $M = 2$, $N_w = 3$, and we allow $N^i_a = N^i_d = 60$ choices of departure time shifts for each flight, then the cardinality of the feasible space becomes 5.88×10^7. The solution space of problem (P1) is discrete and its size grows exponentially fast. The problem has been shown to be NP hard (relative to the number, N, of flights involved) [24].

The optimization formulation (P2) involves mixed-integer variables introducing also high combinatorics to the search space. Since $W^k_{nx'}$ and $W^k_{ny'} \subseteq \mathbb{R}^2$, we have $\mathbf{w} \in \mathbb{R}^{2NM}$. Moreover, the objective function of both problems (P1) and (P2) is non-separable (each term $\Phi_{i,k}(u)$ does not depend solely on variables w_i and δ_i); it is also affected by neighboring trajectories. The evaluation of the objective function involves a heavy computational burden in practice, as will be seen in the sequel of the thesis, where we consider the continental scale. Besides, the objective function may feature several equivalent global optima plus numerous uninteresting local minima (multimodal). This route / departure-time assignment problem is therefore sufficiently difficult to motivate recourse to a stochastic method for optimization.

2.4 Objective function computation method

This section presents the method used to detect conflicts and to compute interaction between trajectories. A practical data structure that allows to save computation time and memory is also presented.

In order to evaluate the objective function of a candidate solution, $(\mathbf{w}, \boldsymbol{\delta})$, one needs to compute interaction, Φ_{tot}, between the N aircraft trajectories. To avoid the $\frac{N(N-1)}{2}$ time-

consuming pair-wise comparisons, which is prohibitive in our large-scale application context, we propose a grid-based conflict detection scheme as presented in [31]. The idea is to put successively each trajectory in a grid, then check for conflicts only in the surrounding cells of the current trajectory.

First, we define a four dimensional (3D space + time) *grid*. The dimension of the 4D grid must be large enough to include the current airspace. The time dimension of the grid must span enough to include the earliest and the latest flights on a given operational day, taking into account all possible departure time shift options.

This 4D grid is then partitioned into *cells* (see Figure 2.10). The size of each cell in the x, y, and z directions is defined by the minimum separation requirements, N_h and N_v. The size of the cell in the time domain is set according to the discretization step size, t_s. This discretization step size must be sufficiently small so that any conflict occurring between two consecutive sampling time steps can be detected. To detect conflicts, the idea is to store the N trajectories in each corresponding cell in the 4D grid. Then, for each trajectory i, and for each cell (I_x, I_y, I_z, I_t) corresponding to each sampling point $P_{i,k} := (x_{P_{i,k}}, y_{P_{i,k}}, z_{P_{i,k}}, t_{P_{i,k}})$, we simply need to check all the surrounding (adjacent) cells in the x, y, and z directions corresponding to the time period $t_{P_{i,k}}$. If one of these surrounding cells is occupied by another aircraft, for instance j, we then note $j \in (I_x, I_y, I_z, I_t)$, and then the horizontal distance, d_h, and the vertical distance, d_v, between point $P_{i,k}$ and the sample point corresponding to aircraft j are computed. Since the violation of the protection volume can only occur when the points in question are in the same or in adjacent grid cells, the number of points to check is significantly smaller than in a pair-wise comparison method.

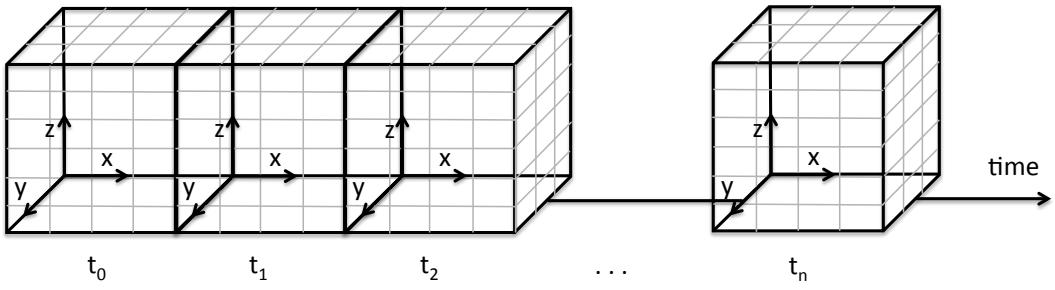

Figure 2.10: Four dimension (space - time) grid.

In order not to underestimate interaction (missing the loss of spatial separation occurring between two successive sampling time steps), trajectories must be discretized with a sufficiently-small sampling time step, t_s, which depends on the maximum possible aircraft horizontal and vertical speeds.

As stated in [16], the worst-case scenario for interaction detection in the horizontal plane occurs when two aircraft follow parallel trajectories that are separated by a distance, D, less than or equal to the horizontal separation norm, N_h, at maximum horizontal speed, $V_{h_{max}}$, with heading in opposite directions. Hence, in the horizontal plane, an undetected interaction can

occur when:

$$t_s > \frac{N_h}{V_{max}} \cos\left(\arcsin\left(\frac{D}{N_h}\right)\right).$$

Figure 2.11 illustrates an undetected violation of the horizontal minimum separation occurring between two successive time steps.

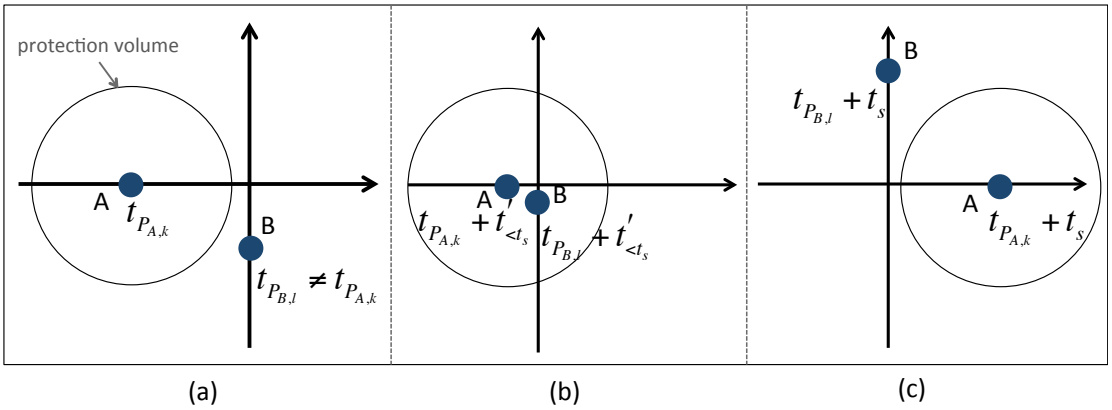

Figure 2.11: Undetected violation of horizontal separation when the sampling step, t_s, is too large. In figures (a) and (c) the horizontal spatial distances between A and B are larger than $N_h = 5$ Nm, therefore the violation of the horizontal minimum separation, which occurs in the meantime (b), cannot be detected.

In the vertical plane, the worst-case scenario occurs when one aircraft is climbing at maximum rate of climb, RoC_{max}, and another is descending at maximum rate of descent, RoD_{max} (see Figure 2.12). Thus, in the vertical plane, in an analogical way as what was done in [16] for the horizontal plane, we can easily show that undetected interaction can occur when:

$$t_s > \frac{N_v}{(RoC_{max} + RoD_{max})}.$$

In order to avoid such undetected conflicts, one can therefore simply choose a sufficiently-small value for the (user-provided) sampling time step, t_s. However, using too small a sampling time step leads to a large number of trajectory sample points, which in turn requires more computation time and memory. Instead, we propose an *inner-loop* algorithm, called `interp`, detecting the violation of minimum separation requirements between two sampling times, t and $t + t_s$, by *interpolating* aircraft positions with a sufficiently small interpolation step size, t_{interp}. This t_{interp} value must be set by the user so as to guarantee that no interaction remains undetected. Then, one checks each pair of these interpolated points. The algorithm stops when a violation of the minimum separation requirements is identified or when all pairs of points have been checked. The inner-loop interpolation algorithm called `interp` is described in Algorithm 2.1.

Recall that $\mathcal{C}(P, Q)$ is defined in (2.13) as 1 when points P and Q are in conflict and zero otherwise. The algorithm to compute the total interaction between the N trajectories, $\Phi_{tot}(\mathrm{u})$, is described in detail in Algorithm 2.2.

In order to optimize the required computation memory, we implement the interaction detec-

Algorithm 2.1 Interp

Require: $P_{i,k}$, $P_{j,l}$ ▷ two trajectory sample points

Discretize, using time step t_{interp}, the trajectory segments $[P_{i,k}, P_{i,k+1}]$ and $[P_{j,l}, P_{j,l+1}]$ as $\{P_\alpha\}_{\alpha=1}^{K}$ and $\{Q_\beta\}_{\beta=1}^{K}$ respectively;
for $k = 0 \to K$ **do** ▷ for each pair of interpolated points
    Initialize $\mathcal{C} := 0$;
    Check conflict, $\mathcal{C} := \mathcal{C}(P_k, Q_k)$;
    if $\mathcal{C} \neq 0$ **then**
        Return $\mathcal{C}$;
        End;
    end if
end for
Return $\mathcal{C}$;

Algorithm 2.2 Interaction computation algorithm

Require: value of the decision variables $u = (\boldsymbol{\delta}, \mathbf{w})$, and the time sequence of 3D grids

Initialize $\Phi_{tot}(u) := 0$;
for i= 1 to N **do** ▷ (for each trajectory i)
    Discretize the alternate trajectory i defined by u_i into a sequence $\{P_{i,k}\}_{k=1}^{K_i}$;
    Initialize $\Phi_i(u) = 0$;
    for $k = 1$ to K_i **do** ▷ (for each point $P_{i,k}$ of trajectory i)
        Initialize $\Phi_{i,k} := 0$;
        Compute the cell I_x, I_y, I_z, I_t corresponding to $P_{i,k}$;
        Compute $\Phi_{i,k}(u)$:
        for $i_x = I_x - 1$ to $I_x + 1$ **do**
            for $i_y = I_y - 1$ to $I_y + 1$ **do**
                for $i_z = I_z - 1$ to $I_z + 1$ **do**
                    if $\exists j \neq i$ such that $j \in (i_x, i_y, i_z, i_t)$ **then**
                        L:= list of all trajectory sample points P_j's in (i_x, i_y, i_z, i_t);
                        for $l = 1$ to length(L) **do**
                            $P := L(l)$;
                            Check conflict, $\mathcal{C} = \mathcal{C}(P_{i,k}, P)$;
                            if $\mathcal{C} = 0$ **then**
                                $\mathcal{C}$:= `interp(P_{i, k}, P)`;
                            end if
                            $\Phi_{i,k} := \Phi_{i,k} + \mathcal{C}$;
                        end for
                    end if
                end for
            end for
        end for
    end for
    $\Phi_i := \Phi_i + \Phi_{i,k}$;
end for
$\Phi_{tot} := \Phi_{tot} + \Phi_i$;
Return Φ_{tot}.

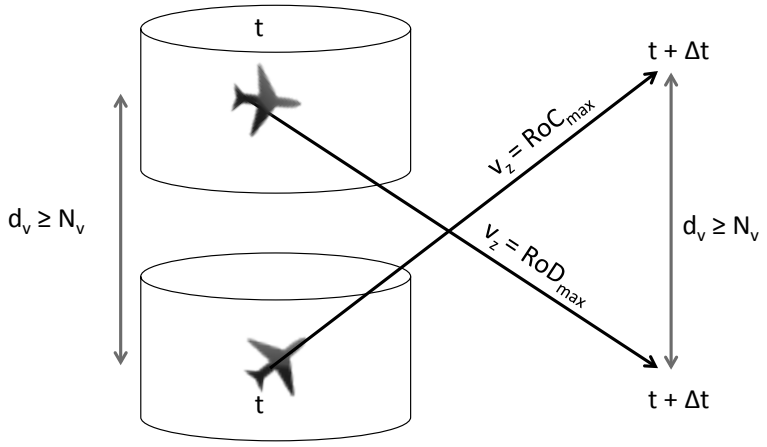

Figure 2.12: Worst-case scenario for interaction detection in the vertical plane between two time steps when one aircraft is descending at maximum rate of descent $(RoD)_{max}$ and another aircraft is climbing at maximum rate of climb $(RoC)_{max}$.

tion scheme using a so-called *hash table* [4], which is a data structure that maps *keys* to values or *entries*. It allows us to store information in an array without pre-defining the size of array in advance. Moreover, the hash table only stores the data once the data is created, therefore empty cells in the array do not occupy memory.

In the optimization process, the computation of the objective function, $\Phi_{tot}(u)$, will be repeated many times, therefore it must be computed as fast as possible. To avoid checking interactions for all N trajectories when some of the trajectories are modified in a new proposed solution, the interaction is updated in a differential manner.

First, the 4D grid is initialized with every cell empty. Then, the N trajectories, corresponding to the initial value of the decision vector, u, are placed in the 4D grid. After that, the *current* interaction, Φ_{i_C}, associated to each trajectory, i, and the current total interaction between trajectories, Φ_{tot_C}, are computed.

Assume now that during the optimization process the decision variables of m flights are to be modified. Let I_{modif} be a list of length m containing the flight indices of all the m flights. To update the value of total interaction, we first remove all the m corresponding trajectories from the 4D grid. Therefore, the interaction associated to each trajectory in I_{modif} has an intermediate value:

$$\Phi_{i,inter}(u) = 0, \quad \forall i \in I_{modif}.$$

Remark that the measurement of interaction is symmetrical. Let $\Phi^{ij}(u)$ denote the *contribution of trajectory i to the interaction associated to trajectory j*. More precisely, we define

$$\Phi^{ij}(u) := \sum_{k=1}^{K_i} \sum_{l=1}^{K_j} \mathcal{C}(P_{i,k}(u_i), P_{j,l}(u_j)),$$

where $\mathcal{C}$ is defined in (2.13), $P_{i,k}$ is the k^{th} sampling point of trajectory i, $P_{j,l}$ is the l^{th} sampling point of trajectory j, and where K_i and K_j are the number of sampling points of trajectory i

and trajectory j respectively. Clearly, we have

$$\Phi^{ij}(u) = \Phi^{ji}(u).$$

Let $\mathcal{N}_i$ be a set of trajectories currently interacting with trajectory i. In other words, $\mathcal{N}_i = \{j : \mathcal{C}(P_{i,k}(u_i), P_{j,l}(u_j)) > 0 \text{ for some indices } k \text{ and } l\}$. Therefore, for each trajectory $i \in I_{modif}$, the interaction associated to each trajectory $j \in \mathcal{N}_i$ has an intermediate value given by:

$$\Phi_{j,inter}(u) = \Phi_j(u) - \Phi^{ij}(u).$$

After that, the *modified* trajectories corresponding to the new decision variable values, $u_i \; \forall i \in I_{modif}$ are placed in the 4D grid. The interaction detection procedure is then performed for all trajectories $i \in I_{modif}$. Then, the interaction, Φ_i, associated to each trajectory $i \in I_{modif}$, is computed.

Again, the interaction associated to each trajectory, j, interacting with the modified trajectories set is updated as follows:

$$\Phi_j(u) = \Phi_{j,inter}(u) + \Phi^{ij}(u).$$

Finally, the total interaction between trajectories is simply computed as follows:

$$\Phi_{tot}(u) = \sum_{i=1}^{N} \Phi_i(u).$$

This interaction computation method allows us to update the value of objective function when some trajectories are modified in a very short computation time, since we do not need to compute the change of interaction associated to the decisions that are not modified at the current optimization iteration.

2.5 Conclusions

In this chapter, we presented a strategic 4D trajectory planning methodology. First, a definition of interaction between trajectories was given. Then, a route / departure - time allocation technique was proposed. The proposed approach allocates an alternative route (horizontal profile) and/or an alternative departure time to each flight, so as to separate trajectories.

The route / departure time allocations was then formulated as a combinatorial optimization problem, and as a mixed-integer optimization problem depending on the structure of the set of possible virtual waypoint locations. Complexity issues tend to show that a metaheuristic approach should be considered to address the large-space instance of continental traffic.

Then, a computationally efficient method to detect interactions between aircraft trajectories was presented. The proposed procedure relies on a 4D grid to store the information of the trajectory samples. For each cell occupied by an aircraft in the 4D grid, its surrounding cells are checked. If one such surrounding cell is occupied by another aircraft, then the distance between two trajectory samples are measured. In order to guarantee that no interaction occurring

between two time sampling steps is missed, an inner-loop algorithm, that interpolates the aircraft positions and measure distance between these interpolated positions, has been introduced. Finally, practical issues related to the computation of the total interaction between trajectories have been discussed.

Chapter 3

Resolution algorithms for the strategic trajectory planning problem

This chapter proposes two resolution algorithms to solve the strategic trajectory planning problem, formulated under the form of an interaction minimization problem.

First, we present an adaptation of a non-population based metaheuristic algorithm, called simulated annealing, to solve this problem. The proposed simulated annealing algorithm is tested with nation-wide scale and continent-scale en-route air traffic. Numerical results from computational experiments with different setting of the algorithm's parameter values are presented and discussed.

Then, we propose to improve the computational efficiency of the resolution algorithm by hybridizing the simulated annealing with simple local-search algorithms. The hybrid algorithms are tested with nation-wide scale and continent-scale air traffic including this time also the traffic in the terminal maneuvering area (near the airport). Numerical results from computational experiments are presented and discussed.

3.1 Metaheuristic optimization algorithm

To solve the strategic trajectory planning problem formulated under the form of a discrete optimization problem (P1) and a mixed-integer optimization problem (P2), we first rely on a *metaheuristics* optimization method. A metaheuristic is a problem-independent procedure or framework that guides other lower-level heuristic procedures to solve an optimization problem.

Metaheuristics are well known for their ability to find high-quality solutions for large-size and complex problems within reasonable computation times. They are typically inspired by nature or by a physical process, and they can be roughly divided into two different classes [83]:

- **Population-based algorithms.** The population-based algorithms *explore* the search space by evolving a whole population of candidate solutions (diversification). Various selection processes are used in order to choose elite solutions among the population of solutions. Then, these elite solutions are evolved through transformation operators. Se-

lection processes are again applied, and these processes repeat until some pre-defined termination criteria are satisfied. The quality of the solutions obtained from such methods depends mainly on the size of the population. Therefore, these methods are not well adapted for problems that require a lot of computation memory to code the state space. Optimization methods belonging to this class are, for instance, genetic algorithms, ant-colony algorithms, particle swam, etc.

- **Non-population-based algorithms.** These algorithms have typically rather an ability to *intensify* the search in some local regions of the solution space. Their search strategy can be viewed as a *walk* from one solution to another solution in the solution space. Several strategies are proposed in order to prevent the walk from being *trapped* in local minima. Optimization methods belonging to this class are, for example, simulated annealing, local search, tabu search, etc.

In our application context, the evaluation of the objective function value relies on a *black-box* simulation process and no derivatives can be computed through the interaction detection scheme introduced in Section 2.4. For a large-scale problem (i.e. national-scale and continental-scale air traffic contexts), this simulation requires a very large computation memory. In our simulation, one point of the state space may require 2 Go. memory. Therefore, population-based metaheuristic approaches are not suitable for our problem due to the excessive memory requirements intrinsic to population-based optimization algorithms. To solve the problem (P1) and (P2), we first rely on a classical simulated annealing algorithm due to its simplicity to implement.

3.1.1 Simulated annealing (SA) algorithm

Simulated annealing (SA) was separately introduced by S. Kirkpatrick et al. in 1982 [57] and by V. Černý in 1985 [87]. It is well known for its ability to escape from a local minimum (or local trap) by allowing occasional moves that deteriorate the value of the objective function, such deteriorating moves being less and less likely as the number of iterations grows.

Simulated annealing is inspired by the annealing process in metallurgy where the state of a material can be modified by controlling the cooling temperature. The physical annealing process consists in heating up a material to bring it to a high-energy state. Then, it is slowly cooled down until a thermodynamic balance is reached. The temperature is reduced according to a pre-described temperature reduction schedule, until the material reaches a global-minimum energy state and forms a crystallized solid. Decreasing too rapidly the temperature can however yield a non desirable local-minimum energy state.

In the simulated annealing optimization algorithm, the objective function to be minimized is analogical to the energy of the physical problem, while the values of the decision variables of the problem are analogical to the coordinates of the material's particles. A control parameter, T, that decreases as the number of iterations grows, plays the role of the temperature schedule, and a number of iterations, N_I, at each temperature step plays the role of the time duration the material is kept at each temperature stage.

For a physical system, when the system reaches the thermodynamic balance at a given

temperature, T, the energy, E, of its particles is distributed according to the Boltzmann distribution: $e^{\frac{-E}{k_B T}}$, where k_B is the Boltzmann constant.

To simulate this evolution of the physical system towards a thermal equilibrium, the *Metropolis algorithm* [68] is used. The Metropolis algorithm is a Markov Chain Monte Carlo (MCMC) method that generates a sequence of random samples according to pre-defined probability distribution. For a given temperature, T, starting from a current configuration, the state space of the simulated system is subjected to a transformation (e.g. apply a local change to one decision variable). If this transformation improves the objective-function value, then it is accepted. Otherwise, it is accepted with a probability

$$P_{accept} := e^{\frac{\Delta E}{T}}, \tag{3.1}$$

where ΔE is the degradation of the objective-function value (negative for minimization). Repeating this process creates a chain of events (Markov chain), where the next stage only depends on the current state. If this chain is of infinite length, it can be shown that the simulated system will reach the Boltzmann distribution (therefore thermal equilibrium).

Once the equilibrium is reached, the temperature is decreased according to a pre-defined cooling schedule. As the temperature decreases, the probability, P_{accept}, to accept a degrading solution becomes smaller and smaller. Therefore, the system will eventually converge to the nearest local optimum which will expectantly be close to a global optimum. We refer the reader interested by simulated annealing algorithm to the following books [42, 83].

3.1.2 Adaptation of SA for strategic 4D trajectory planning

In order to implement the simulated annealing algorithm to the strategic planning of 4D trajectories, we first define the following parameters.

1. **Neighborhood function.** A neighborhood solution is generated by applying a *neighborhood function* (or transformation operator) that generates a local change to the current solution. This change should be computed rapidly, but should not have large effects on the system; otherwise, the characteristic of the simulated annealing will become those of a pure random search. Here is the neighborhood function we propose.

   To generate a neighborhood solution, first a flight i is randomly chosen. In order not to modify excessively the trajectories that are not involved in any interaction, we set a user-defined threshold value of interaction, denoted Φ_τ, such that the trajectory of a randomly chosen flight i will be modified only if

   $$\Phi_i(u) \geq \Phi_\tau. \tag{3.2}$$

   Otherwise, another trajectory will be randomly chosen until condition (3.2) is satisfied. This process ensures that changes will be first applied on trajectories involved in congestion area.

   Then, for a chosen flight, i, one has to determine whether to modify the location of waypoints, or to modify the departure time, in the next move. A priori, searching for a

solution in the time domain would be better since it does not induce extra fuel consumption. However, empirical tests show that limiting the search to using only that degree of freedom results in prohibitive computational times, and excessive departure time shifts.

Therefore, we introduce a user-defined parameter, $P_w \leq 1$ to control the probability of modifying the value of the i^{th} trajectory waypoint location vector, w_i. The probability to modify rather the departure time is therefore $1 - P_w$. This parameter, P_w, allows the user to set his/her preference on the way to deconflict trajectories.

In order not to generate large changes to the system, only one virtual waypoint, randomly chosen among the M virtual waypoints, $w_i^1, w_i^2, \ldots, w_i^M$, is modified at each time. More precisely, if $r \geq P_w$, where r is a random number generated between 0 and 1, one virtual waypoints, say w_i^m, is randomly chosen, and then, a new location for w_i^m is randomly generated from the set of possible locations, W_i^m. If $r < P_w$, then a new departure time shift is randomly chosen from the set of possible departure time shifts, Δ_i. The neighborhood function we use in this thesis is summarized in Algorithm 3.1.

Algorithm 3.1 Neighborhood function

Require: probability P_w, trajectory i.
1: Generate random number, $r :=$ random(0,1);
2: **if** $r < P_w$ **then**
3: Choose randomly new δ_i from Δ_i;
4: **else**
5: Choose randomly one virtual waypoint w_i^m to be modified.
6: Choose randomly new $w_{ix'}^m$ from $W_{ix'}^m$;
7: Choose randomly new $w_{iy'}^m$ from $W_{iy'}^m$;
8: **end if**

2. **Initial temperature and initial acceptance probabilities.** The initial temperature determines the acceptance probability at the initial temperature. If it is high, the system will accept almost every neighborhood solutions proposed. On the other hand, if it is very low, the system will spend much time evaluating neighborhoods solution that are not likely be accepted, and eventually it will converge to a local minimum.

   Theoretically, the initial temperature, denoted T_0, should be high enough so that the probability of accepting a degrading solution at the end of the first temperature step is close to 1. However, in practice, setting this probability too high may lead to long computation times. This parameter is usually experimentally tuned according to the configuration of the problem being solved. Before starting the optimization process, one observes the evolutions of the system under the transformations generated by the pre-determined neighborhood function. From this observation, one will obtain a rough knowledge about the relation between the objective-function value and the search space (the landscape of the objective function). From this information, one can determine a value for T_0 that is adapted to the problem's configuration.

   To determine the initial temperature and initial acceptance probability, we rely on a practical recommendations given in [42]. They are computed by first generating 100 deteriorating transformations (neighborhood solutions) at random; then by evaluating

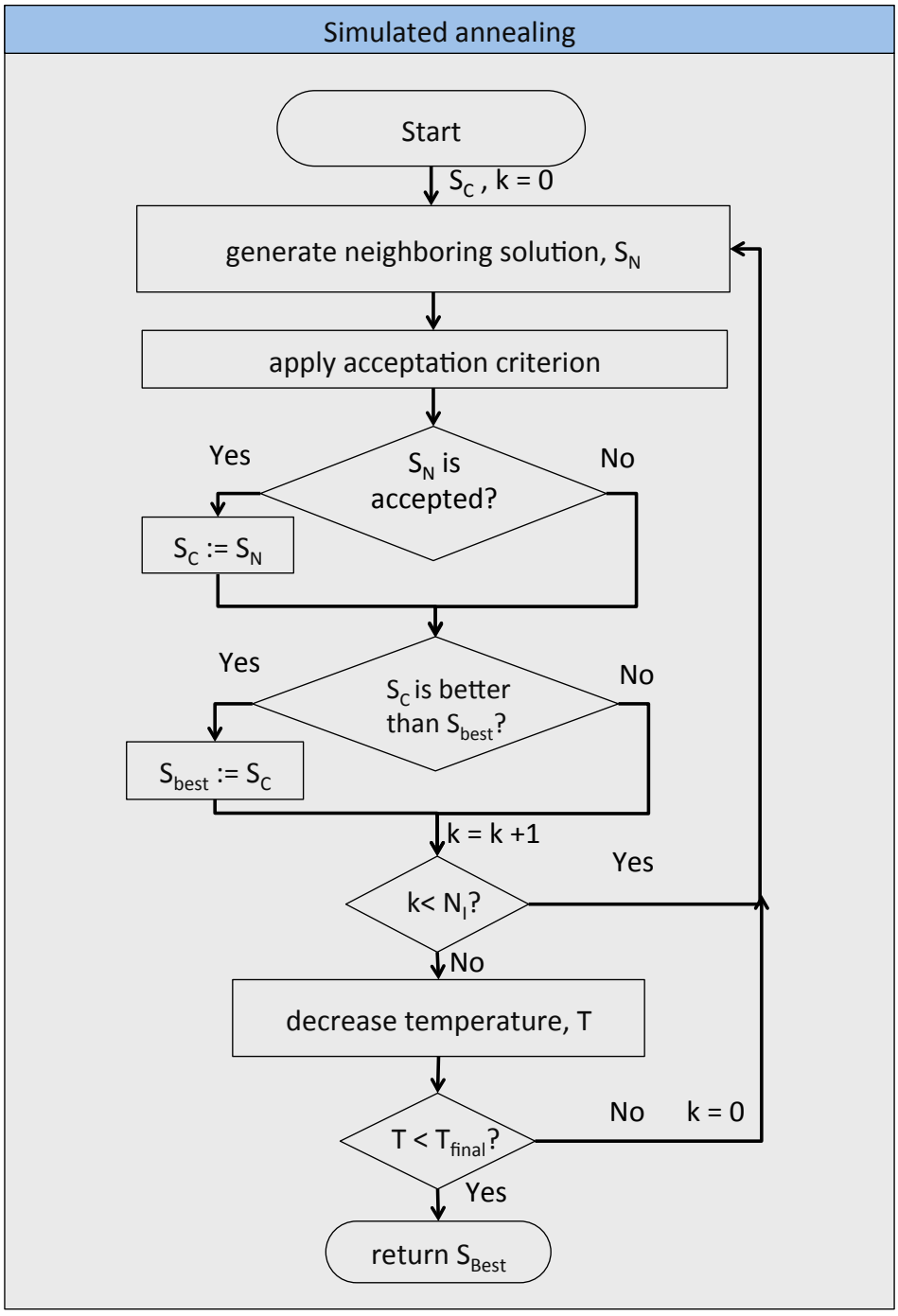

Figure 3.1: Proposed simulated annealing algorithm.

the average variations, ΔE_{avg}, of the objective-function value. The initial temperature, T_0, is then deduced from the relation:

$$\tau_0 = e^{\frac{\Delta E_{avg}}{T_0}},$$

where τ_0 is the initial rate of accepting degrading solutions. The value of τ_0 is empirically set depending of the assumed quality of the initial configuration (initial air traffic situation in our case), for example:

- for poor quality (poor resulting objective-function value) initial configuration: $\tau_0 := 50\%$;
- and for good quality configuration: $\tau_0 := 20\%$.

3. **Cooling schedule.** In the simulated annealing algorithm, the cooling schedule plays an essential role to guide the system towards a good optimum. If the temperature is decreased slowly, the system is more likely to converge to a better solution, but it will require more computation time. On the other hand, decreasing too rapidly the temperature tends to yield undesirable local optima.

   The temperature, T, can be decreased by one of the following laws of decrease:

   - Linear law: $T_i = T_0 - (\alpha.i)$, where $\alpha > 0$ is a pre-defined constant value.
   - Geometrical law: $T_i = \beta.T_{i-1}$, where $0 \leq \beta \leq 1$ is a pre-defined constant value.
   - Logarithmic law. The temperature is decreased by: $T_i = \frac{T_0}{log(i)}$, etc.

   For simplicity, we will decrease the temperature, T, following the geometrical law, where the constant β will be experimentally tuned.

4. **Equilibrium state.** In order to reach an equilibrium, a sufficient number of iterations, denoted N_I, or moves, have to be performed at each temperature step. The value of N_I can be determined in the following manners:

   - Static manner: N_I, is constant in order to ensure that the system reaches an equilibrium at each temperature step. For instance, the temperature will be decreased after $a.N$ iterations, or $b.N$ perturbation accepted, where a and b are user-defined constants, and where N is the number of degrees of freedom of the problem.
   - Adaptive manner. In this case, the number of iterations, N_I, is adapted to the current configuration of the system. In other words, the number of iterations depends on the quality of the solutions found during the iteration process. For instance, one may set N_I by using the best and the worst objective-function values found during the previous (pre-defined) number of iterations to determine the number of iterations to be further performed before reducing the temperature. It is not necessary that the simulated annealing reaches the equilibrium at each temperature step, thus the computation time is decreased.

Again, for simplicity, in our case, the number of iterations, N_I, to be performed at each temperature step will be constant and experimentally set.

5. **Termination criterion.** Theoretically, it is suggested that the SA algorithm stops when the temperature reaches zero. However, this stopping criterion is not utilized in practice, since when the temperature is near zero, the probability of acceptance becomes negligible. Therefore, a better stopping criterion could be to wait that, T_f, is reached, or when a pre-determined number of transitions, or when a pre-defined number of temperature steps without improvement is performed.

   In our case, the simulated annealing algorithm will terminate when the final temperature, T_f, reaches the value $C.T_0$, where $0 \leq C \leq 1$ is a user-defined coefficient.

The overall simulated annealing algorithm we used in this work is summarized in Figure 3.1.

3.1.3 Computational experiments

The simulated annealing algorithm adapted to solve the strategic trajectory planning problem is implemented in Java. It is tested on two problem instances of different sizes. The first problem instance involves national-scale ($\approx$ 8,000 flights) en-route air traffic. The second problem instance considers continent-scale ($\approx$ 29,000 flights) en-route air traffic.

Influences of the optimization formulation chosen (discrete / mixed-integer) on the resolution of the problem is studied. The influence of the user-defined parameters N_w, and Φ_τ on the resolution of the problem is also discussed.

Problem instance 1: National-size en-route air traffic

First, we test the proposed SA algorithm on a national-size en-route air traffic. The given data set corresponds to a full day en-route air-traffic over the French airspace corresponding to the demand of 17^{th} August 2008. It is provided by the Complete Air Traffic Simulator (CATS) [8] and consists of N = 8,836 trajectories using direct route. Figures 3.2 and 3.3 illustrate the initial given trajectory set, sampled with the trajectory discretization time step $t_s = 20$ seconds in the horizontal and the vertical planes respectively (the dense area located at the coordinate point $(0; 5 \times 10^6)$ on Figure 3.2 corresponds to Paris).

To solve this problem instance, we implement our strategic 4D trajectory planning methodology on a Unix platform with a 2.4 GHz processor and 8 GB RAM (personal computer). The initial trajectory set involves Φ_{tot} = 83,044 total interactions between trajectories. Figure 3.4 illustrates the initial interaction, Φ_i, associated to each flight i.

The parameter values chosen to specify the optimization problem are given in Table 3.1. To give an idea of the complexity of the computation of the objective-function value, for this problem instance and using the mixed-integer optimization formulation (P2), when using the sampling time-step value $t_s = 20$ seconds, the initial N trajectories are discretized into 1,387,112 sample 4D points. With regard to the dimension of the search space, when setting $M = 2$, our optimization problem involves for this instance:

- $2MN$ = 35,344 (continuous) virtual waypoints variables (component of the vector $\mathbf{w}$),

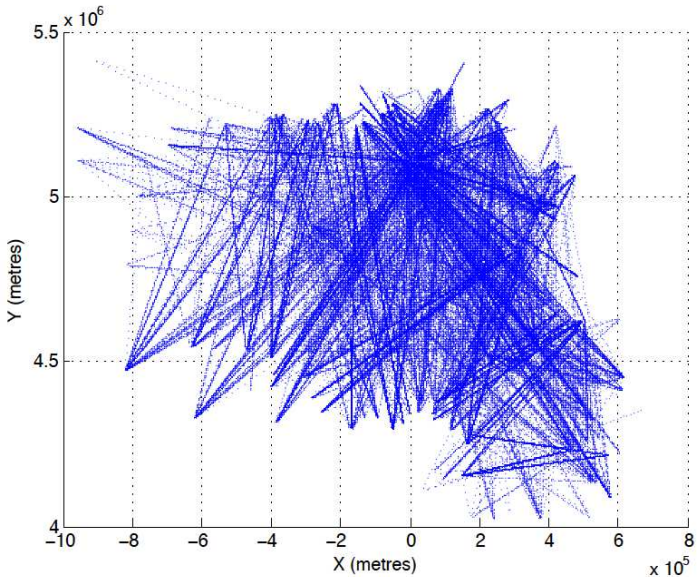

Figure 3.2: The initial given trajectories consisting of a full-day traffic over the French airspace in the horizontal plane.

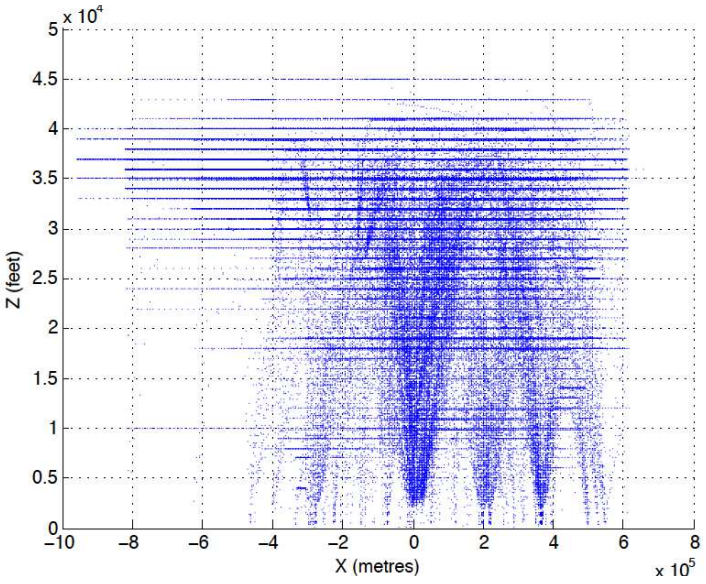

Figure 3.3: The initial given trajectories consisting of a full-day traffic over the French airspace in the vertical plane.

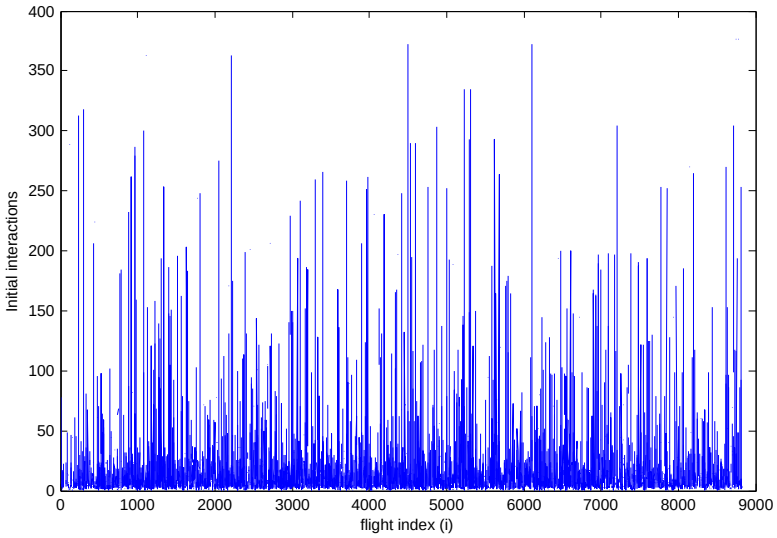

Figure 3.4: Initial interactions associated to each flight index, i, over the French airspace.

parameter	value
Sampling time step, t_s	20 seconds
Maximum departure time shift, $-\delta_a^i = \delta_d^i := \delta$	60 minutes
Discretization time step for possible delays / advance departure-time shifts, δ_s	20 seconds
Maximum allowed route length extension, d_i	0.12 (12 %)
Maximum number of waypoints, M	2

Table 3.1: Chosen (user-defined) parameter values defining the overall methodology applied to problem instance 1 (en-route French air traffic)

- N = 8,836 (discrete) departure-time shift variables (the components of the optimization vector $\boldsymbol{\delta}$), each of which involving $\frac{2\delta}{\delta_s} + 1 = 361$possible values.

The parameters defining the overall resolution methodology are empirically set, and presented in Table 3.2. The algorithm terminates when the final temperature, T_f, is reached or when an interaction-free solution is found.

- **Resolution based on the optimization formulation (P1)** First, we solve the problem instance 1 based on the discrete optimization formulation (P1). The proposed algorithm

parameter	value
Number of iterations at each temperature step, N_I	100
Initial rate of accepting degrading solution, τ_0	0.3
Geometrical temperature reduction coefficient, β	0.99
Final temperature, T_f	$(1/500).T_0$
Inner-loop interpolation sampling time step, t_{interp}	5 seconds
Probability to modify horizontal flight profile, P_w	0.5
Interaction threshold value, Φ_τ	0.5 Φ_{avg}

Table 3.2: Empirically-set (user-defined) parameter values of the resolution methodology.

is implemented with the value of the parameters N_w that defines the number of possible lateral location for each flight i. The influence of the parameter N_w on the resolution of the interaction minimization problem is discussed. For $M = 2$, the longitudinal locations of the virtual waypoints are:

$$w^1_{ix'} = \frac{1}{3},$$

and

$$w^2_{ix'} = \frac{2}{3},$$

for each flight i. The maximum route length extension occurs when

$$w^1_i = (\frac{1}{3},\ a_i),$$

and

$$w^2_i = (\frac{2}{3},\ -a_i),$$

and therefore a_i can be straightforwardly deduced from:

$$(1 + d_i) = 2\sqrt{\left(\frac{1}{3}\right)^2 + (a_i)^2} + \sqrt{\left(\frac{1}{3}\right)^2 + (2a_i)^2},$$

which yields $a_i \approx 0.12$ (for a value $d_i = 0.12$ for the maximal route length extension coefficient).

The set, $W_{iy'}$, of possible lateral locations for each virtual waypoint is therefore given by

$$W^m_{iy'} := 0.12\{-1, -\frac{N_w - 1}{N_w}, -\frac{N_w - 2}{N_w}, \ldots, -\frac{1}{N_w}, 0, \frac{1}{N_w}, \ldots, \frac{N_w - 2}{N_w}, \frac{N_w - 1}{N_w}, 1\},$$

which yields $(2N_w + 1)^2$ possible alternative horizontal profiles.

The simulations with $N_w = 2$, 3, 4, 8 yield 25, 49, 81, 289 possible alternative horizontal flight paths for each flight respectively. The simulations are performed 10 times for each N_w value.

SA yields interaction-free (therefore conflict-free) solution for every flight and for every value of N_w. The average number of iterations performed in order to reach the interaction-free solutions are presented in Figure 3.5, and the corresponding average computation times are reported in Figure 3.6. The number of flights that are subjected to departure-time shifts and route length extensions are compared in Figure 3.7. The average departure-time shifts attributed to each delayed (or advanced) flight are not significantly different when using different values of N_w. The averaged route length extensions applied to each extended flight are compared in Figure 3.8.

SA solves this national-scale en-route air traffic problem instance based on discrete formulation (P1), and yields interaction-free solutions within a significantly short computation time (less than 10 minutes) using a personal computer. The use of high value for the

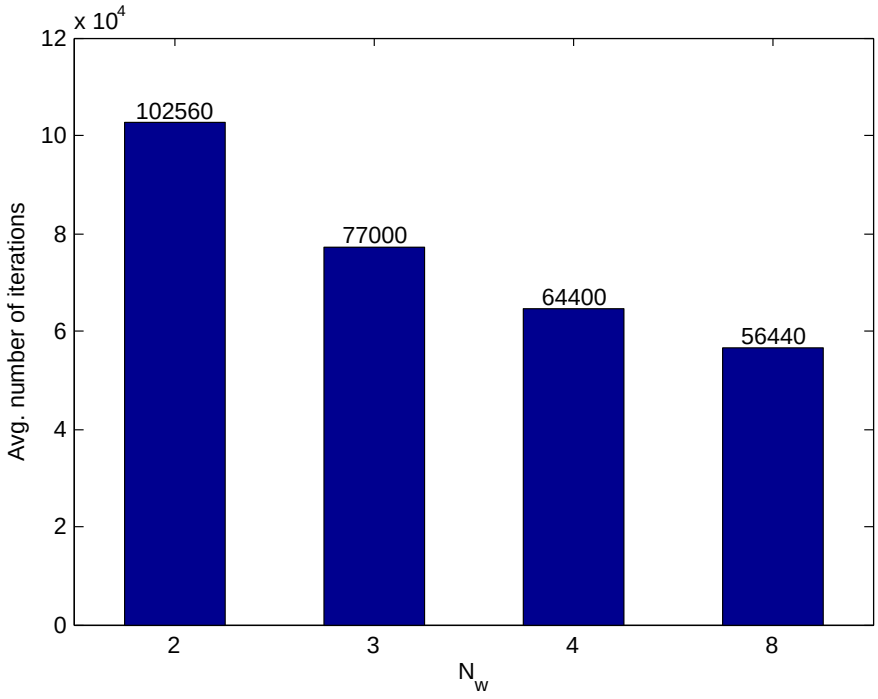

Figure 3.5: Average number of SA iterations to reach an interaction-free solution with different N_w values.

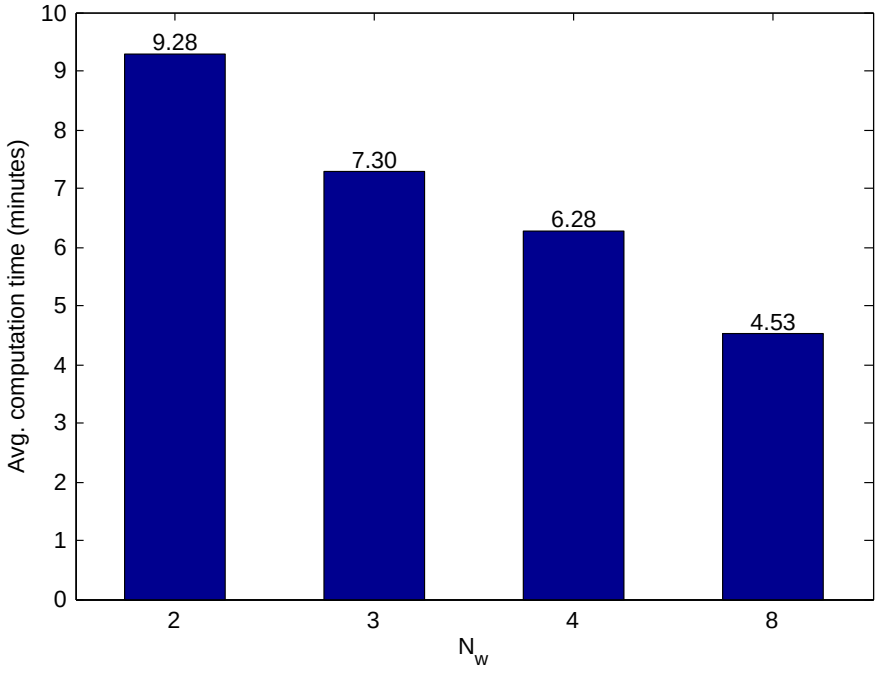

Figure 3.6: Average SA computation time to reach an interaction-free solution with different N_w values.

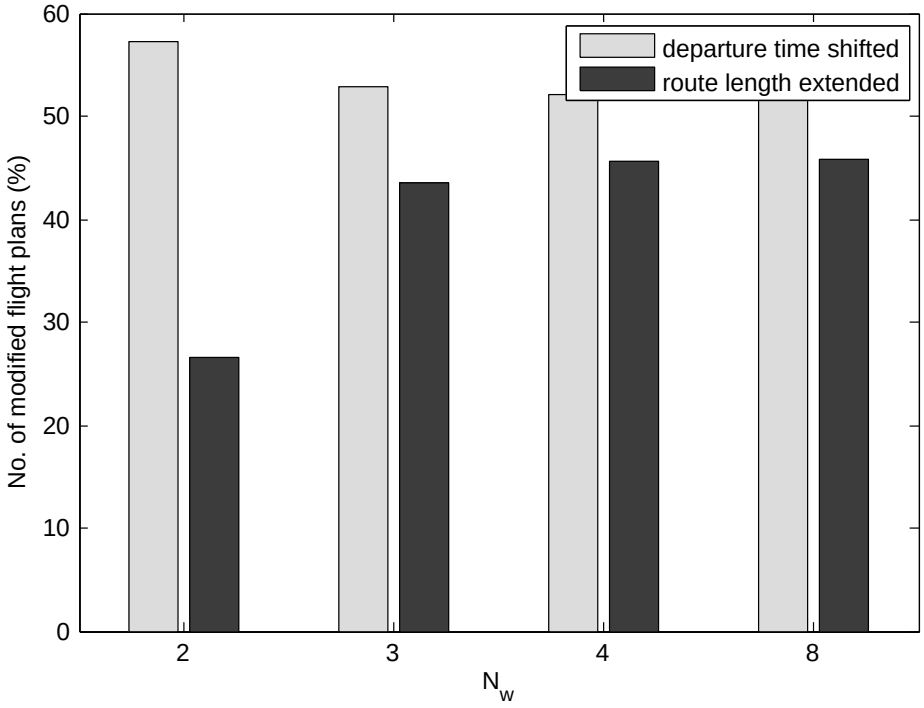

Figure 3.7: Average number of modified flight plans with different N_w values.

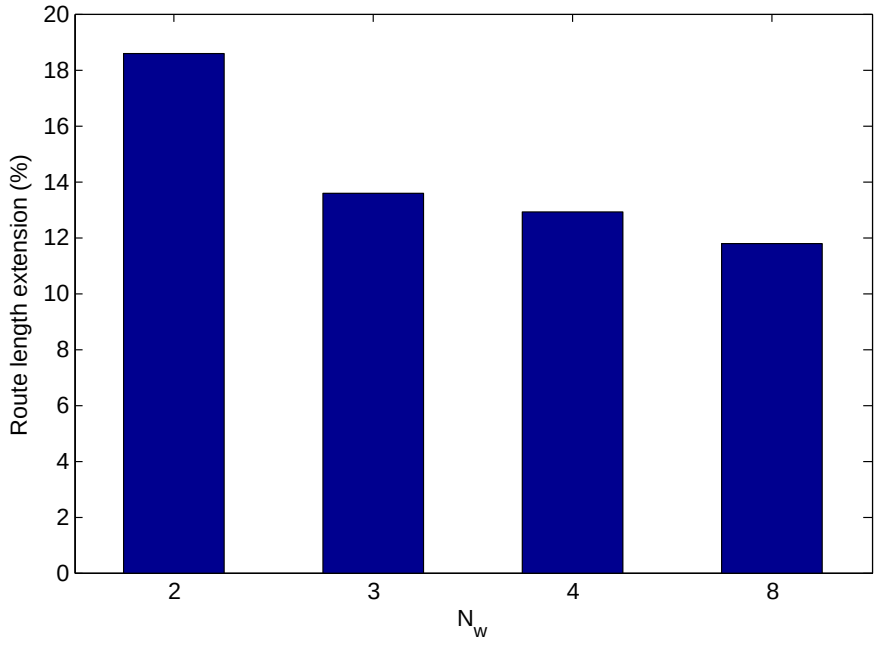

Figure 3.8: Average route length extensions attributed to each extended flight with different N_w values.

parameter N_w increases the richness of the solution space. It allows the algorithm to converge to better solutions (solutions that induce less route length extension) within shorter computation time.

- **Resolution based on optimization formulation (P2)**

  We now implement SA based on the mathematical formulation (P2) with the parameter values as given in Table 3.2.

  We set two different values for the pair of parameters a_i and b_i that define the size of the continuous search space of the possible locations of the virtual waypoints.

  First, the value of parameter b_i that defines the interval of possible longitudinal component of virtual waypoint w_i^m is set to 0, and the value of parameter a_i is set to 0.12 for $i = 1, 2, \ldots, N$, as in the previous case. Then, we empirically set the value of parameter b_i to 1/15. For any flight i, the maximum route extension can occur when:

$$w_i^1 = ((\frac{1}{3} - b_i),\ a_i),$$

  and

$$w_i^2 = ((\frac{2}{3} + b_i),\ -a_i).$$

  Therefore a_i can be straightforwardly deduced from:

$$(1 + d_i) = 2\sqrt{\left(\frac{1}{3} - b_i\right)^2 + (a_i)^2} + \sqrt{\left(\frac{1}{3} + 2b_i\right)^2 + (2a_i)^2},$$

  which yields $a_i \approx 0.126$.

  The simulations are performed 10 times for each possible value of the pair of parameters a_i and b_i. Again, SA yields an interaction-free solution trajectory for every flight. In Table 3.3, we compare the average number of iterations and the average computation time required to solve this problem instance using formulation (P1) with $N_w = 8$, and (P2) with different values of a_i and b_i. This continuous relaxation of the constraint of the virtual waypoint locations and the enlargement of the feasible search domain (case 3 in Table 3.3) reduce significantly the computation time.

Optimization formulation	avg. no. of iterations	avg. computation time (mins)
P1 ($a_i = 0.12$, $N_w = 8$)	56,440	4.53
P2 ($a_i = 0.12$, $b_i = 0.0$)	53,550	5.17
P2 ($a_i = 0.126$,$b_i = 0.067$)	43,480	3.19

Table 3.3: Comparison of the average number of iterations and the average computation time for SA to reach an interaction-free solution using optimization formulations (P1) with $N_w = 8$, and (P2).

  Then, the influence of the threshold value, Φ_τ, that determines the neighboring function on the performance is studied. Simulations with different values of the threshold parameter, Φ_τ, are performed. Let $\Phi_{avg}(u) := \frac{1}{N}\sum_{i=1}^{N}\Phi_i(u)$ denote the average objective-function

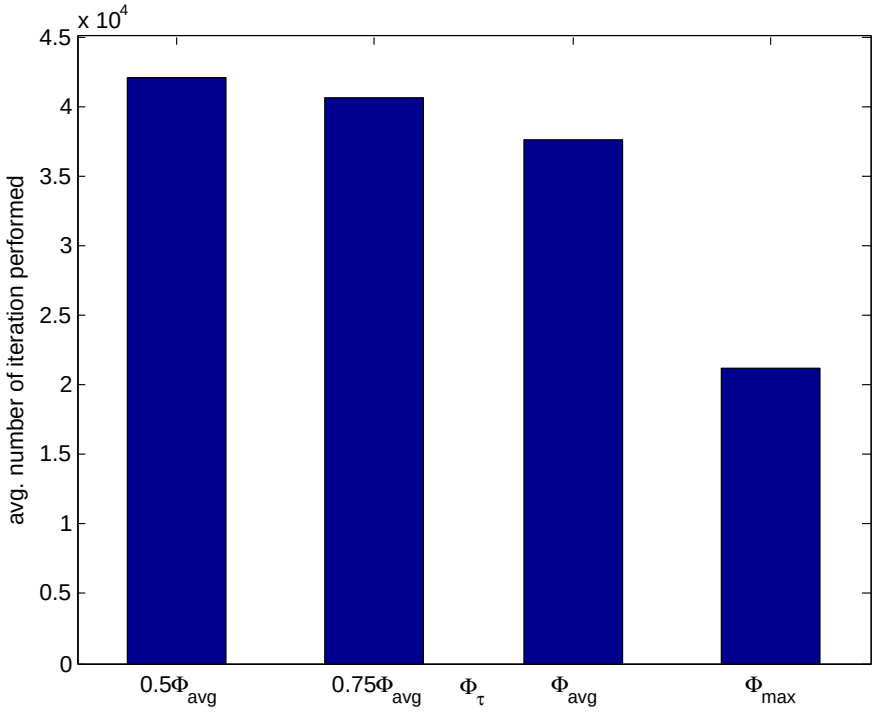

Figure 3.9: Average number of iterations performed to attain an interaction-free solution using different Φ_τ values.

value, and $\Phi_{max} := \max(\Phi_1(u), \ldots, \Phi_N(u))$ denote the maximum objective-function value. We set $\Phi_\tau = 0.5\Phi_{avg}$, $0.75\Phi_{avg}$, Φ_{avg}, and Φ_{max}, respectively. The average number of iterations performed using these values of Φ_τ are compared in Figure 3.9. Figure 3.10 compares number of modified flight plans using different values of Φ_τ.

The computation time required by the SA to attain interaction-free solutions and the number of modified trajectories depend largely on how the neighborhood function selects one solution (u_i) to be modified. When the value of Φ_τ is large, the trajectories that are involved in more interactions are more likely to be chosen. Therefore, the neighborhood function guides the SA towards a more promising region of the solution space, and the algorithm converges faster to interaction-free solutions. On the other hand, when the value of Φ_τ is lower, more candidate flights can be chosen to be modified; therefore the algorithm modifies more trajectories and requires more computation time to converge to interaction-free solutions.

Let us denote, $\Phi_0 := \{\Phi_{1,init}, \Phi_{2,init}, \ldots, \Phi_{N,init}\}$, the vector of the initial interaction computed for each of the N flights. Figure 3.11 and 3.12 illustrate the correlation between Φ_0 and the solution vectors, $\boldsymbol{\delta}$ and $\mathbf{w}$, according to different values of the interaction threshold parameter Φ_τ respectively. The correlation between the indices of the trajectories that are modified (in space / time) and the indices of the trajectories that are involved in the initial interaction directly varies with the value of Φ_τ. The interaction threshold value Φ_τ also influences the number of modified flight plans. As expected, when the neighborhood function targets the trajectories that are involved in more interactions (a large Φ_τ value),

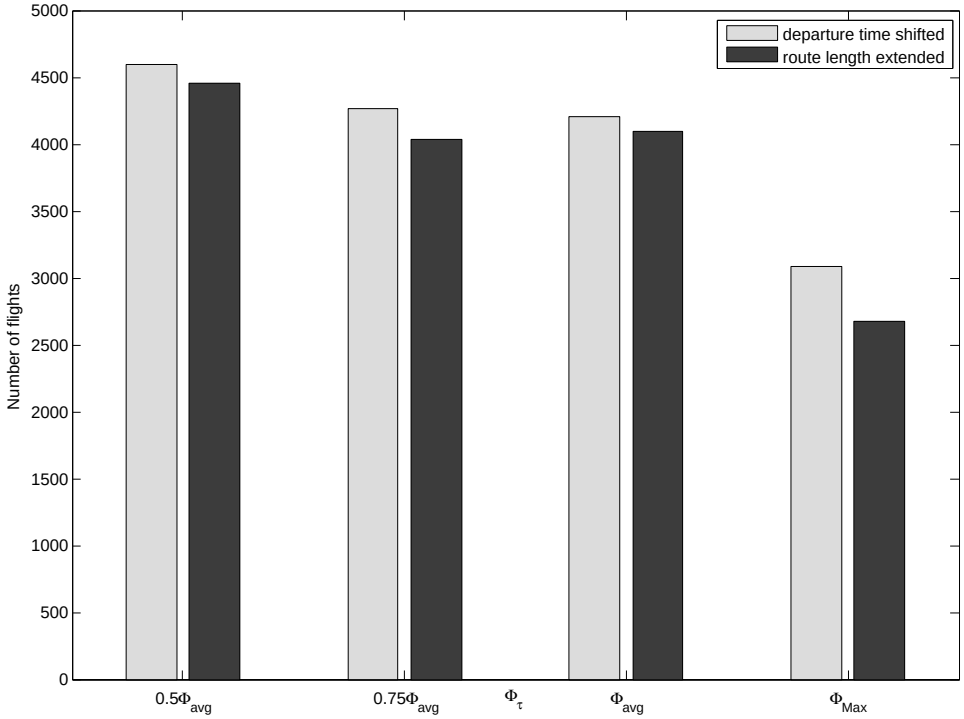

Figure 3.10: Number of modified flight plans using different Φ_τ values.

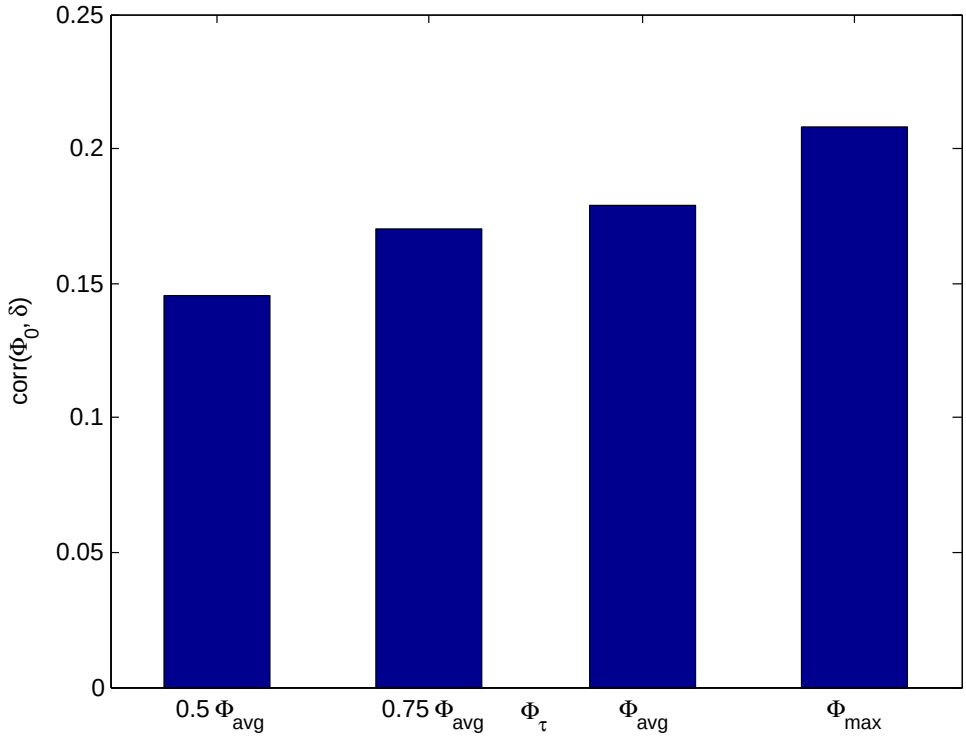

Figure 3.11: Correlation between the solution vector $\boldsymbol{\delta}$ and the vector of initial interaction, Φ_0, associated to each flight with different value of the interaction threshold parameter Φ_τ.

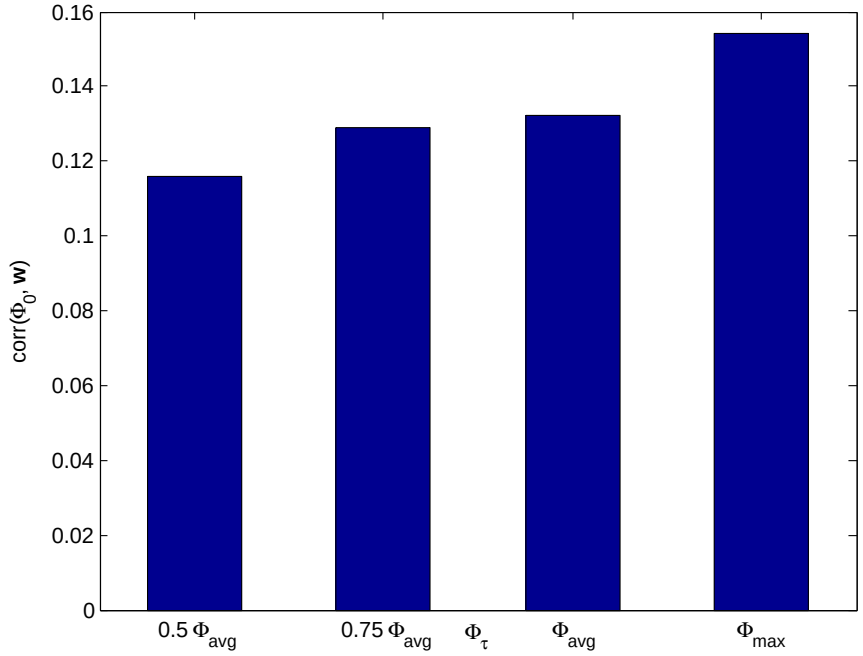

Figure 3.12: Correlation between the solution vector $\mathbf{w}$ and the vector of initial interaction, Φ_0, associated to each flight with different values of interaction threshold parameter Φ_τ.

the number of delayed/advanced flights and the number of extended flights decrease.

Problem instance 2: Continental-size en-route air traffic

Here, the proposed algorithm solves a continent-size en-route air traffic problem. The data set is a full day of air-traffic over the European airspace on July 1, 2011. It consists of 30,695 trajectories simulated with optimal vertical profiles and with direct route. First, we consider only the air traffic in the en-route environment by filtering out the trajectory segments that lies under the altitude of $z = 1{,}000$ feet, (disregarding the whole trajectory when the flight duration of a post-filtering trajectory is shorter than 1 minute). This results in a reduced set of $N = 29{,}852$ en-route trajectories.

We address this problem instance with an AMD Opteron 2 GHz processor with 128 Gb RAM. The user-defined parameter values that specify the problem instance under consideration are the same as given in Table 3.1. To give an idea of the complexity of the computation of the objective function of this problem instance, when using the sampling time-step value $t_s = 20$ seconds, the initial N trajectories are discretized into 7,249,253 sample 4D points. The initial trajectory set involves $\Phi_{tot} = 142{,}144$ total interactions between trajectories. Figure 3.13 illustrates the initial trajectories (blue dots) sampled with a sampling time step, $t_s = 20$ seconds, and the locations where the initial interactions occur (red dots).

With regard to the dimension of the search space, when setting $M = 2$ waypoints, our optimization problem involves, for this instance:

- $2MN = 119{,}408$ (continuous) waypoint variables ($\mathbf{w}$);

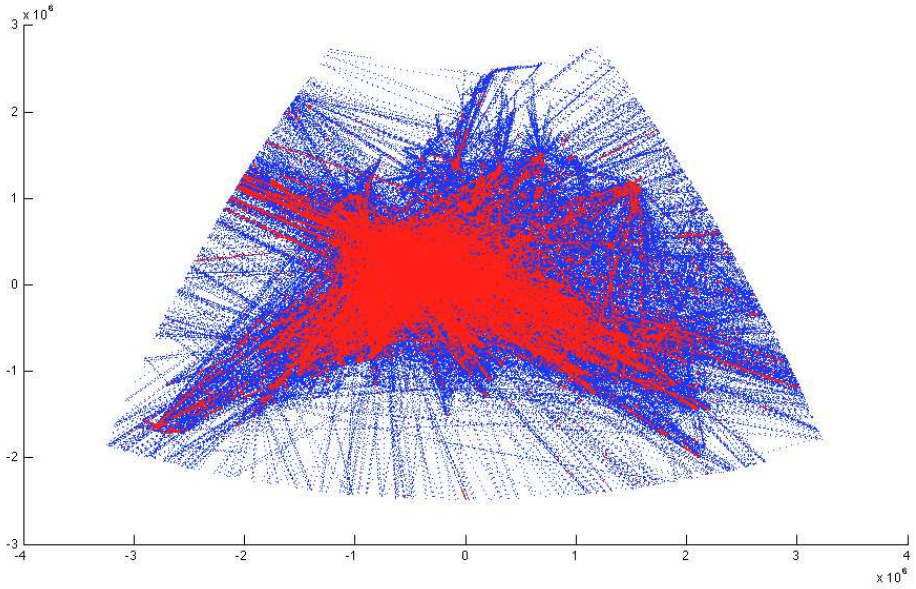

Figure 3.13: Initial trajectory set involving en-route air traffic over the European airspace sampled with $t_s = 20$ seconds with initial location of interactions displayed as red color dots.

Numerical results	value
number of iterations	497,000
avg. computation time (minutes)	76.19
avg. proportion of delayed / advanced flights	71.29%
avg. proportion of extended flights	46.23%
avg. departure time shifts (minutes)	30.14
avg. route length extensions	1.95%

Table 3.4: Numerical results for problem instance 2 solved by SA based on optimization formulation (P2) (averages are computed over 10 runs).

- N = 29,852 (discrete) departure-time shift variables ($\boldsymbol{\delta}$), each of which involving $\frac{2\delta}{\delta_s} + 1$ = 361 possible values.

The parameters defining the overall resolution methodology are the same as those given in Table 3.2, but this time the number of iterations at each temperature step, N_I, is empirically set to 3,500.

This continent-scale problem instance is solved based on the mixed-integer formulation (P2). The SA yields an interaction-free solution for this continent-scale problem instance in a computation time that is still compatible for strategic planning application. Numerical results obtained from the simulation is reported in Table 3.4.

The simulated annealing algorithm yields interaction-free solutions for both national-size and continent-size en-route air traffic. For a national-size problem instance involving 8,836 en-route trajectories, the SA converges to interaction-free solution in a very short computation time (3 to 10 minutes depending on the parameter setting, and on the nature of the constraints (discrete or continuous) describing the location of virtual waypoints). However, when applying on the larger problem instance 2, involving 29,852 en-route trajectories, the required computation time increases significantly. To improve further the computation time, we propose to combine two simple metaheuristic algorithms together. This combination of multiple metaheuristic algorithms is referred to as *hybrid-metaheuristic algorithm*, which will be detailed in the following section.

3.2 Hybrid metaheuristics optimization algorithm

To further improve efficiency of the optimization algorithm, we propose in this section to combine the simulated annealing with another metaheuristics algorithm. This combination of metaheuristic algorithms is referred to as *hybrid metaheuristics*. In recent years, the hybrid metaheuristics have been more and more successful for its capability to solve large and complex real-world problems. Instead of relying on one metaheuristics algorithm, the hybrid metaheuristics allows us to integrate advantages of several metaheuristics together. One of the most used technique to develop a hybrid-metaheuristic algorithm is to combine non-population based with population-based metaheuristics. The population-based metaheuristic is known for its ability to explore the solution space by recombining solutions to generate new ones. Therefore, it can be effectively used to find promising areas of the solution space. The non-population base method, in contrast, focuses on intensify the search in a small neighborhood; therefore, it can be used to drive the search process to the nearest local optimum.

To implement a hybrid metaheuristic, one first has to determine a structure to control the **level of hybridization** of the metaheuristic algorithm. We may classified the level of hybridization into two categories: low-level hybridization and high-level hybridization [28].

- Low-level hybridization addresses an integration of metaheuristics algorithms, where each algorithm has a strong coupling between each other. In this case, the individual components of function in each metaheuristic may be replaced by or exchanged with a function from the other metaheuristic algorithm. For instance, one may use a greedy heuristic as a crossover operator in a genetic algorithm.

- In high-level hybridization, each algorithm retains its own characteristics without direct interactions between the internal functions of the algorithm. Information is exchanged between each self-contained metaheuristic algorithm over a well-defined interface. For instance, one may use the best solution obtained from one metaheuristic algorithm as an initial solution of the other metaheuristic algorithm.

Then, another feature to consider is the **order of carrying out** each metaheuristic algorithm. In general, the algorithms can be executed in a sequential, interleaved or parallel manners.

Finally, to combine effectively several metaheuristics together, one has to determine the **values** of the various user-provided parameters specifying the metaheuristics. These parameters controls, for example, the balance between exploration and exploitation of the solution space. They must be fine-tuned with care as they have a strong impact on the quality of the solution obtained. The interested readers are referred to, for instance, [28, 83], for more details on hybrid-metaheuristics algorithms.

3.2.1 Adaptation of a hybrid-metaheuristic for strategic 4D trajectory planning

As stated earlier in the previous section, population-based metaheuristic are not convenient for the large-scale problem we are considering, due to the computational memory required. Therefore, we propose here to combine the simulated annealing with simple local search algorithms.

Since the simulated annealing has an ability to escape from local minima (or local traps) by accepting degraded solutions from times to times, it can be effectively used to control the *diversification* of the search while the local search will *intensify* the search around a promising region found by the simulated annealing.

Local search

A local search is an algorithm that starts from a given initial solution, and then iteratively replaces the current solution with a better solution in a pre-defined neighborhood. In this work, we rely on a simple *Iterative Improvement* Local Search (IILS) that allows only strict improvement of the objective function value. Given a solution, the IILS generates a neighborhood solution (using a pre-defined neighborhood function), and then moves to this new solution only if it yields an improvement of the objective function value. The algorithm stops when a pre-defined maximum number of iterations, noted N_{Loc}, is reached. The quality of the solution found by the IILS depends on the initial solution, the definition of the neighborhood function, and the neighborhood search strategy.

For simplicity, we rely on the same neighborhood function as that presented in Algorithm 3.1. We introduce two different neighborhood search strategies as follows:

- **Intensifying the search on one Particular Trajectory (PT).** Given a flight i, this state-exploitation step focuses on improving the current solution by applying a local

change from the neighborhood structure only to flight i (only the decision variables u_i are affected).

- **Modifying the Interacting Trajectories (IT).** Given a flight i, this state-exploitation step applies a local change, from the neighborhood structure, to every flight that is currently interacting with flight i. For instance, suppose that trajectory i interacts with trajectories p, q, and r; the changes are then sequentially applied to the decision variables u_p, u_q, and u_r.

Hybrid simulated annealing - local search algorithm

For the sake of simplicity of implementation, the SA and the IILS are hybridized in a self-contained (high-level) manner where each algorithm is sequentially executed. The IILS is integrated as an inner-loop in the SA so that the IILS can be considered as one iteration step of the SA. The initial solution of the IILS is provided by the current solution of the SA. Then, the solution found by the IILS is returned to the SA, where an acceptance condition will be verified. However, the IILS only allows strict improvements of the solution, it will always be accepted by the SA here.

The order of execution of each metaheuristic is given as follows.

- At each iteration of the hybrid algorithm, one flight is randomly chosen among all flights featuring the pre-defined level of interactions, Φ_τ. It is, as before, a pre-defined interaction threshold parameter provided by the user. The hybrid algorithm chooses randomly one flight in the set $\{i \in \{1, 2, \ldots, N\} : \Phi_i \geq \Phi_\tau\}$. Let i denote the selected flight.

- Then, the hybrid algorithm determines whether to perform a classical SA step, or to trigger the IILS, or to perform both algorithms successively. This decision is taken according to a specific (user-defined) probability that depends upon the control temperature, T, and the value of the term, Φ_i, of the objective function corresponding to flight i.

The **probability to carry out SA step**, P_{SA}, is:

$$P_{SA}(T) = P_{SA,min} + (P_{SA,max} - P_{SA,min}) \cdot \frac{T_0 - T}{T_0}, \tag{3.3}$$

where $P_{SA,max}$ and $P_{SA,min}$ are the (user-provided) maximum and minimum allowed probabilities to perform SA.

The **probability of running the IILS** module, P_{Loc}, is given by:

$$P_{Loc}(T) = P_{Loc,min} + (P_{Loc,max} - P_{Loc,min}) \cdot \frac{T_0 - T}{T_0}, \tag{3.4}$$

where, similarly, $P_{Loc,max}$ and $P_{Loc,min}$ are the (user-provided) maximum and minimum probabilities to perform the local search.

Finally, the **probability of carrying out both** SA and the IILS (successively), P_{SL}, is simply:

$$P_{SL}(T) = 1 - (P_{SA}(T) + P_{Loc}(T)) \tag{3.5}$$

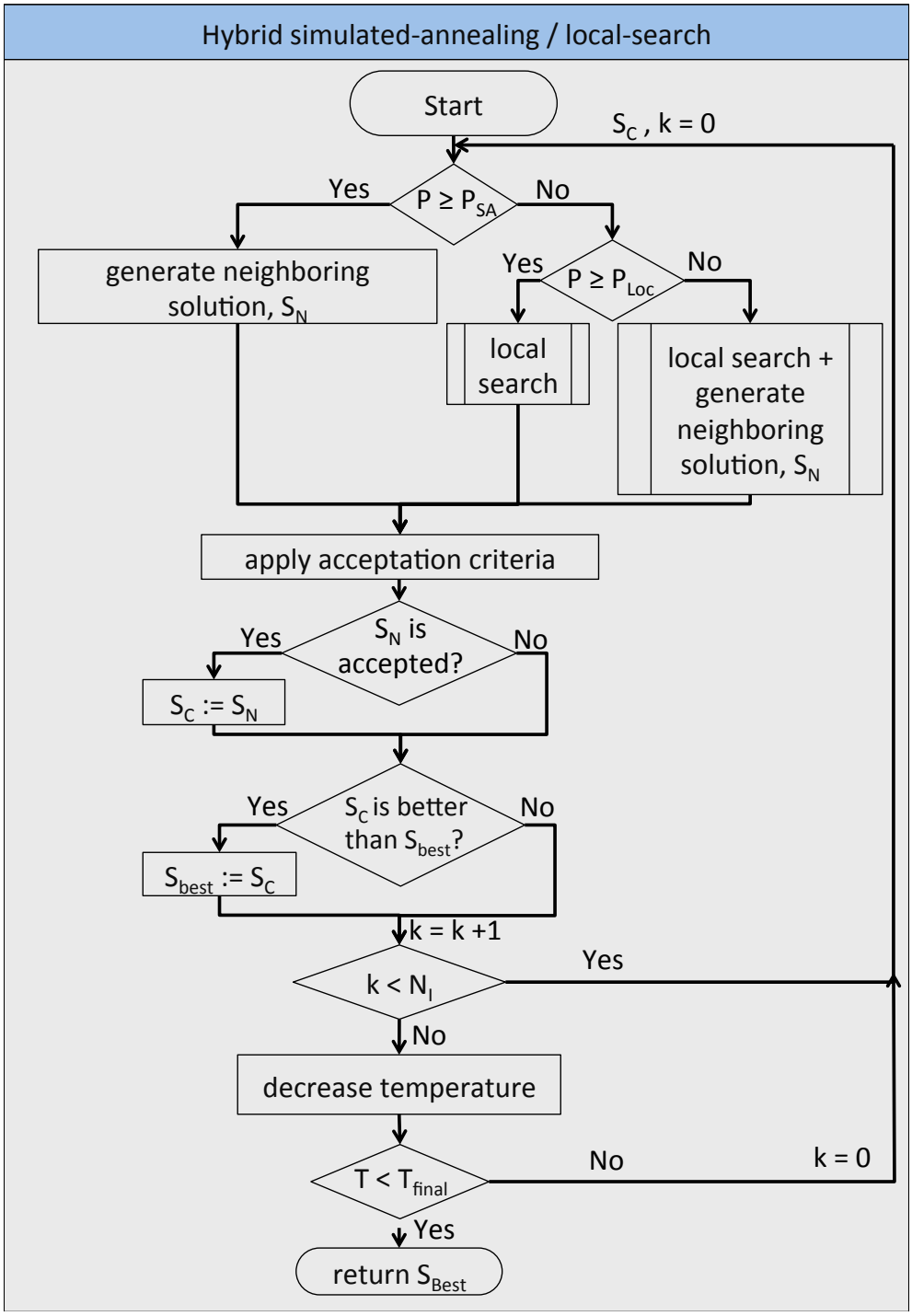

Figure 3.14: Hybrid simulated-annealing / iterative-improvement local search algorithm.

A key factor in tuning this hybrid algorithm is to reach a good trade-off between exploration (diversification) and exploitation (intensification) of the solution space, i.e. a compromise between fine convergence towards local minima, and the computation time invested in exploring the whole search space in order not to miss a global optimum.

The proposed hybrid algorithm is detailed in Figure 3.14, where T_{init} and T_{final} are respectively the initial and the final temperatures of the (user-provided) cooling schedule, and where N_I is the maximal number of iterations at each temperature step (also set by the user).

3.2.2 Computational experiments

The proposed hybrid SA / IILS algorithm is implemented in Java and, again, run on an AMD Opteron 2 GHz processor with 128 Gb RAM. First, the hybrid algorithm is tested on the (previously-presented) problem instance 2, involving en-route air traffic over the European airspace. The user-defined parameter values specifying the optimization problem are the same as those given in Table 3.1. Empirical tests lead us to set the user-defined parameters of the hybrid algorithm as given in Table 3.5.

User-defined algorithm parameters	value
Minimum probability to perform SA step, $P_{SA,min}$	0.8
Maximum probability to perform SA step, $P_{SA,max}$	0.9
Minimum probability to perform local search step, $P_{Loc,min}$	0.4
Maximum probability to perform local search step, $P_{Loc,max}$	0.6
Number of iterations at each temperature step, N_I	3,500
Number of iterations of the inner-loop local search step, N_{Loc}	5
Initial rate of accepting degrading solution, τ_0	0.3
Geometrical temperature reduction coefficient, β	0.99
Final temperature, T_f	$(1/500).T_0$

Table 3.5: Empirically set parameters of the hybrid simulated-annealing / iterative-improvement local search algorithm

To investigate the influence of each local search strategy (PT, IT, and PT&IT) on the resolution of the problem, we first perform simulations by using either PT or IT strategy, and then perform simulations using both PT and IT successively (PT&IT).

These three versions of the hybrid algorithm yield interaction-free solutions for every flight. The number of iterations required by each resolution algorithm to reach interaction-free solutions for problem instance 2 and their corresponding computation times are compared in Figure 3.15. One can observe that, the hybrid algorithm converges to interaction-free solutions significantly faster (around 3 times faster) than the classical SA described in Subsection 3.1.2. In addition, the hybrid algorithm converges faster when using both search strategies (PT&IT) successively. Figure 3.16 compares the number of modified flight plans yielded by each resolution algorithm. The number of modified flight plans (delayed / advanced flights and extended flight) is significantly lower when using the hybrid algorithms (PT&IT, PT, and IT) thanks to the more targeting search strategy that intensifies the search on flights that are involved in interactions. The average departure time shifts and the average route length extension of each modified flight are not significantly different when using different local search strategies. This

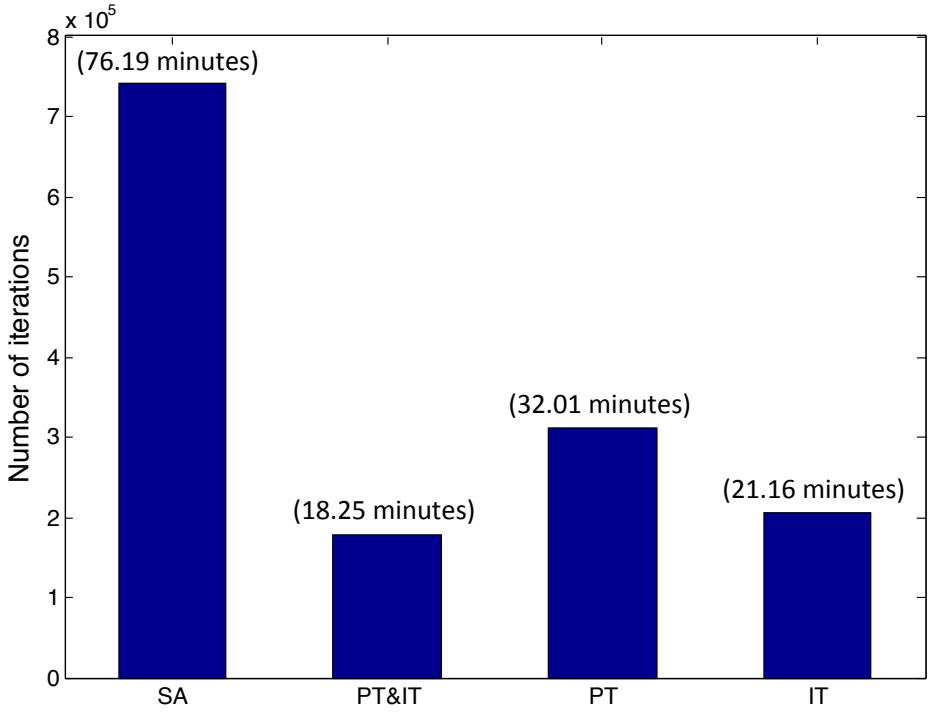

Figure 3.15: Number of iterations that each resolution algorithm performs to reach interaction-free solutions for problem instance 2.

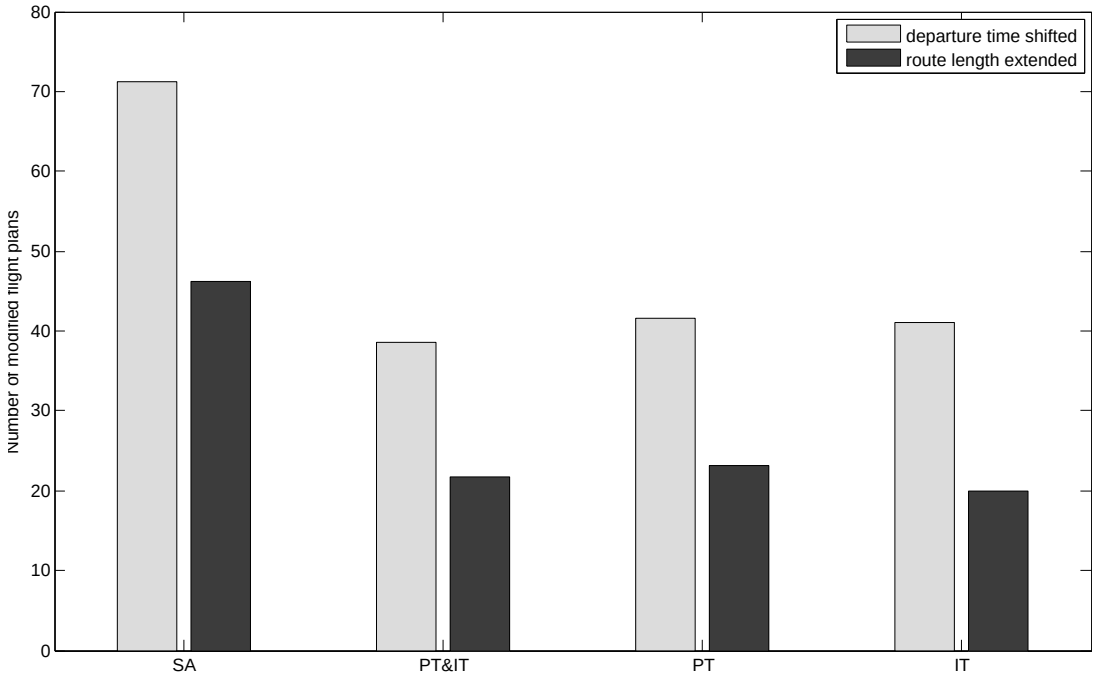

Figure 3.16: Number of modified flight plans yielded by each resolution algorithm.

P_w	residual interactions	no. of iterations	computation time (minutes)
0.00	32	539,000	1,016.7
0.25	0	56,000	51.9
0.50	0	49,000	43.1
0.75	0	52,500	46.8
1.00	1,984	682,500	436.9

Table 3.6: Comparison of the numerical results of problem instance 2 for different values of the parameter P_w.

is due to the fact that our objective function only focuses on minimizing the total interaction between trajectories; route length extension and departure time shift are criteria that are only dealt with as constraints.

To study the influence on the resolution time of the user-defined parameter P_w, that sets the user preference to modify the departure time or to modify the horizontal flight path, simulations are performed with $P_w = 0$, 0.25, 0.5, 0.75, and 1. The maximum allowed departure time shifts are set this time to $-\delta_a^i = \delta_d^i = 120$ minutes. The maximum allowed route length extension is set to $d_i = 0.2$.

Computation time of the hybrid algorithm using different P_w values are reported in Table 3.6. The hybrid algorithm yields interaction-free solutions when P_w is set to 0.25, 0.5, and 0.75. However, by definition of P_w, when P_w is set to zero, the interactions can be solved only by acting in the time domain. When P_w is set to one, the interactions can be solved only by acting in the 3D space domain. In both cases the solution space is thereby significantly reduced. As a consequence, the hybrid algorithm yields residual conflicts and requires long computation times to converge.

Problem instance 3: continent-size air traffic including TMAs

A terminal maneuvering area (TMA, or terminal control area (TCA) in the U.S. and Canada), is a controlled airspace dedicated to approaches or departures. For the sake of simplicity, we shall consider that a given aircraft is in the TMA when its altitude is below 10,000 feet above ground level, and we set the size of the minimum separation in this area, noted $N_{h_{TMA}}$, to 3 Nm.

To respect the departure and approach procedure in the TMA, we do not affect the shape of trajectories in such area. We only modify the shape of trajectories in the en-route phase. Therefore, the TMA-traffic is much more constrained than the en-route segment traffic. Indeed, the interactions occurring in en-route airspace can be separated in space and in time, while the interactions occurring in the TMA can only be separated in the time domain.

This problem instance involves the same traffic as the one of instance 2, with adding traffic in the TMAs. The trajectory set, therefore, consists of $N = 30,695$ trajectories, which yields 235,632 initial interactions. Remark that the total interactions between trajectories considering traffic in the TMA is significantly higher than in the previous case where only en-route air traffic is considered. This is due to the high density of the traffic in TMA. In addition, the interaction-detection module cannot distinguish aircraft using parallel runways from actual interaction. This leads to some false-positive contributions to the interaction. The control parameters of

the optimization algorithm are the same as given in Table 3.5, but this time the number of iterations at each temperature step, N_I is set to 4,500.

The user-defined parameter values used to solve this problem are the same as those given in Table 3.1. However, to solve this more constrained problem instance, the maximum number of waypoints, M is set to = 3. Furthermore, two different values for the maximum departure time shift, $-\delta_a^i$, δ_d^i, and for the maximum route length extension, d_i, are used.

traffic scenario	initial interaction	final interaction	d_i	$-\delta_a^i$, δ_d^i (minutes)	avg. computation time (minutes)	avg. of modified trajectories	avg. of modified departure times
En-route traffic	142,144	0	0.12	60	18.3	21.73 %	38.57%
traffic with TMA	235,632	0	0.12	60	1272.4	63.6%	89.6%
traffic with TMA	235,632	0	0.25	60	756.3	59.7 %	68.3 %
traffic with TMA	235,632	0	0.25	120	478.1	48.3 %	76.1 %

Table 3.7: Numerical results of the strategic planning algorithm taking into account the air traffic in TMA.

The simulation is carried out 10 times. Numerical results are shown in Table 3.7. Again, the proposed algorithm is able to find interaction-free solutions for the given traffic situation involving high-density traffic in the TMA within computation times that are still compatible with strategic planning applications. One observes that the computation time required to obtain the interaction-free solution depends on the size of the solution space. As expected, with the same settings as before ($d_i = 0.12$ and maximum time shift, $-\delta_a^i = \delta_d^i = 60$ minutes), the algorithm requires significantly more computation time for solving the scenario with TMA traffic. However, the required computation time decreases significantly when the solution space is relaxed (i.e. when more candidate solutions are considered; last two lines of Table 3.7).

3.3 Conclusions

This chapter presents resolution algorithms to solve the proposed strategic trajectory planning problem formulated under the form of discrete and mixed-integer optimization problems. The first resolution algorithm relies on a non-population based metaheuristic optimization algorithm called simulated annealing (SA). The SA is implemented and tested with national-size and continental-size en-route air traffic instances involving up to 29,852 flights.

The two mathematical formulations of the strategic trajectory planning problem are compared. First, simulations are performed based on the discrete formulation, using different values for the parameter N_w that controls the total number of possible alternative horizontal profiles. When N_w is increased, the solution space becomes richer. Therefore, it is easier to find interaction-free solutions, and the SA converges faster to the interaction-free solution.

Then, simulations are performed based on the mixed-integer formulation. The SA yields interaction-free solutions in less computation time and induces less route length extension.We studied the influence of the parameter Φ_τ, that controls the neighborhood function, on the computation time required by the simulated annealing. When Φ_τ increases, the neighborhood function targets first the trajectories that are involved in more interactions. Therefore, the SA converges to interaction-free solutions significantly faster when setting $\Phi_\tau = \Phi_{max}$.

To further improve efficiency of the resolution algorithm, we propose to combine the SA

with an iterative-improvement local search algorithm. In this hybrid-metaheuristic algorithm, the SA is responsible for exploring the solution space while the iterative improvement local search intensifies the search around a promising region found by the SA. The hybrid algorithm was tested on the continent-size air traffic instance. It requires significantly less computation time to reach interaction-free solutions than the SA.

Finally, we took into account the air-traffic in the terminal maneuvering area (TMA). The interaction between trajectories in the TMA is more dense and more difficult to solve than in the en-route environment. Therefore, we increased the richness of the solution space by increasing the number of virtual waypoints ($M = 3$). The influence of the size of the solution spaces (maximum allowed departure time shift, maximum allowed route length extension) on the resolution of this problem was studied. As expected, the problem becomes easier to solve when the constraints are relaxed. This nevertheless provides a practical way to address such a difficult problem.

Chapter 4

Extension to the case with uncertainty

In this chapter, we present an extension of the proposed strategic 4D trajectory planning methodology to the case with uncertainties of aircraft positions and arrival times. To consider such uncertainties, we rely on the concept of robust optimization, using two different models of the uncertainty sets.

During flight, aircraft may not be able to comply with its assigned interaction-free 4D trajectory with high precision due to, for instance, wind conditions, passenger delays, etc. Moreover, imposing a hard 4D constraints on the trajectory may results in an increase of fuel consumption and of aircraft engine workload, since the aircraft may have to keep on adjusting its velocity. In order to improve robustness of the strategic trajectory planning, and to relax the 4D constraints, uncertainties of aircraft position and arrival time are taken into account in the trajectory optimization process.

More precisely, we rely on a concept of robust optimization to incorporate the uncertainties into the problem. We propose a robust strategic 4D trajectory planning methodology based on two different uncertainty models: deterministic model, and probabilistic model. The solutions obtained are based on the following assumptions:

- the features of the uncertainties are identical for every aircraft.

- the uncertainty does not grow with time.

The present chapter is organized as follows. Section 4.1 presents a brief introduction to the concept of robust optimization. Section 4.2 proposes a worst-case-oriented robust strategic 4D trajectory planning methodology based on a deterministic-type uncertainty set. Section 4.3 presents a less-conservative robust strategic 4D trajectory planning methodology based on a probabilistic-type uncertainty set. In both Section 4.2 and Section 4.3, the interaction-computation methods considering each type of uncertainty sets are detailed, and numerical results from computational experiments are presented.

4.1 Robust optimization: an introduction

The solutions to an optimization problem can be very sensitive to small changes or perturbations on the data of the system being optimized. Indeed, the data of real-world problems are hardly certain. The uncertainty can result from, for instance:

- Change of environment and operating conditions. These uncertainties can arise from disruptive events linked to, for example, wind conditions, external temperature, pressure, etc.
- Measurement or estimation errors. This type of error can be caused by, for example, limitation of technology and environment conditions that make it impossible to measure exactly the values of the system's parameters. It may also be caused by the approximation errors due to the use of models instead of real physical systems.
- Implementation errors. These errors are caused by the fact that it is not possible to implement a solution exactly as it is computed.

These uncertainties can impact the optimization problem through the objective function or through the constraint set. These uncertainties should be considered, otherwise the quality of solution can be compromised, and/or the constraints of the problem can be violated when implementing the solution.

To handle uncertainty, the concept of *robust optimization* has been developed mainly in the fields of operations research and engineering design [82]. It aims at optimizing the objective function for any realization of some of the data which are known to belong to some sets. A comprehensive survey on robust optimization is provided in [25]. The interested readers are referred to some standard robust optimization literature, for instance, [18, 19, 20, 21].

The concept of robust optimization differs from that of *sensitivity analysis*, which is a post-optimization tool for quantifying the impact of perturbations in the values of the decision variables and in the system data on the optimization cost. In contrast, robust optimization focuses on optimizing the so-called *robust counterpart* of the objective function or satisfying the robust counterpart of the constraints, given the characteristics of the uncertainty. The robust counterpart, or sometimes referred to as *robust regularization* [63], is a function whose value at any given point is the maximum value of the original function in a fixed neighborhood of such a point. In other words, the robust counterpart approach aims to immune the solution of the optimization problem, so that the original constraints are satisfied for any realization of the uncertainties on pre-defined uncertainty sets.

The robustness of the optimization, therefore, lies in the characterization of the uncertainty sets. There are different approaches to characterize and quantify uncertainty. According to [25], there are:

- the deterministic type, which considers every possible realization of the system's parameters in a given bounded set;
- the probabilistic type, which considers probability of the events, represented by probability distributions; and

- the possibilistic type, which considers the possibility or impossibility of the events can be modeled by so-called *fuzzy sets*, whose elements have degrees of membership [89].

Considering the deterministic type, different geometries of uncertainty set have been considered. One of the commonly used geometry for the uncertainty set is the box:

$$\mathcal{U}_\infty = \{\epsilon \mid \|\epsilon\|_\infty \leq \Psi\},$$

where Ψ is a parameter controlling the box's size. Another commonly used and more realistic uncertainty set is ellipsoidal uncertainty set which is motivated by the normal distribution. The ellipsoidal uncertainty set is given by:

$$\mathcal{U}_2 = \left\{\epsilon \mid \|\Sigma^{-1/2}\epsilon\|_2 \leq \rho\right\},$$

where Σ is the covariance matrix of ϵ, and ρ is a parameter controlling the size of the ellipsoid. It is considered, for instance, in [19]. More geometries of uncertainty set are introduced, for example, in [64].

To construct a robust counterpart of a given optimization problem, we shall limit our discussion, without loss of generality, to the case with an uncertain objective function. Note that uncertain constraints can be incorporated by modifying the objective function so that the constraints have no uncertainty. Note also that the robust optimization can be modeled in an opposite manner, by incorporating the uncertainty only to the constraint set (with introduction of new constraints if necessary), and keeping the objective function without uncertainty [21].

Consider the objective function $f(\mathbf{x})$ to be minimized, the uncertainties can be incorporated by considering the robust counterpart function, F_R, given by

$$F_R = \sup_{\epsilon \in \mathcal{U}} f(\mathbf{x}; \epsilon),$$

where $\mathcal{U}$ is a set that defines the uncertainty of the system. We can assume without loss of generality that the uncertainty set, $\mathcal{U}$, has the form $\mathcal{U} = \mathcal{U}_1 \times \mathcal{U}_2 \times \ldots \times \mathcal{U}_n$, where n is the number of uncertainty sets to be considered. This robust counterpart is based on the *Minimax principle* that seeks to immune the solution against the worst-case scenario.

A less conservative approach to define the uncertainty set, based on a probabilistic framework, can also be considered. In this case, it is assumed that the probability distribution of the uncertainty set $\mathcal{U}$ is known, yielding a random objective function, $f(\mathbf{x}; \epsilon)$. In order to optimize such a random objective function, one can, instead, optimize a related deterministic value, for instance, the *expected value*:

$$F_R = \mathbb{E}[f(\mathbf{x}; \epsilon)].$$

This approach is referred to as *Mean value optimization* approach.

There exist other approaches to construct the robust counterpart of a given function, based on different assumptions on the uncertainty sets. The interested readers is referred to, for instance, [25].

The robust counterpart allows one to represent an uncertain problem under a deterministic form, so that it can be solved using the same optimization approaches (with some modifications

if necessary) as for its non-robust counterpart. Modeling a robust optimization problem is to trade-off between the quality of the solution and robustness. Being too pessimistic will yield a solution with higher cost (in terms of an objective function that one minimizes), while being too optimistic will compromise the robustness of the solution obtained.

4.2 Robust strategic 4D trajectory planning based on deterministic-type uncertainty

In this section, we present the strategic 4D trajectory planning problem taking into account the uncertainty of the aircraft position and arrival time, by modeling these uncertainties as deterministic sets. First, we present the uncertainty models that we are considering. Then, an optimization formulation of the robust strategic 4D trajectory planning is introduced. After that, a method to compute the value of the objective function under such uncertainties is detailed. Then, a resolution algorithm to solve this problem is discussed. Finally, numerical results from computational experiments are presented.

4.2.1 Uncertainty model

To consider the uncertainty of aircraft position and arrival time, we characterize the uncertainty sets as follows.

Uncertainty of aircraft position in the horizontal plane

Consider an initial 4D trajectory planning specifying that an aircraft must arrive at a given horizontal point (x, y) at time t. Due to uncertainties, we shall assume that the *real* horizontal position, (x^r, y^r), of the aircraft at time t can be in an area defined by a disk of radius R_h (defined by the user) around (x, y), as illustrated in Figure 4.1. Let $\epsilon_{h_x} = (x^r - x)$ and $\epsilon_{h_y} = (y^r - y)$ denote the uncertainties of aircraft position in the x and the y directions respectively. The vector of uncertainty of aircraft position in the horizontal plane, denoted $\epsilon_h = \{\epsilon_{h_x}, \epsilon_{h_y}\}$, must belong to the set:

$$\mathcal{U}_h := \{\epsilon_h \ : \|\epsilon_h\|_2 \leq R_h\} \tag{4.1}$$

In other words, the possible locations of the aircraft at time t are the elements of the set:

$$\{(x^r, y^r) : (x^r - x)^2 + (y^r - y)^2 \leq R_h^2\}.$$

To ensure horizontal separation of aircraft subjected to such uncertainties, the protection volume has to be enlarged by a radius of R_h as illustrated in Figure 4.1. Thus, the *robust minimum separation in the horizontal plane*, N_h^r, is defined as:

$$N_h^r := N_h + R_h,$$

where N_h is the (usual) minimum horizontal separation of the case without uncertainty.

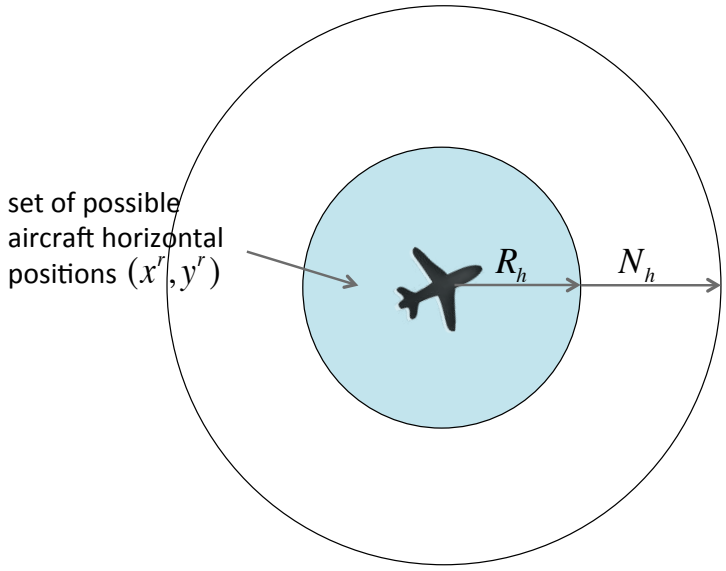

Figure 4.1: Possible aircraft positions in the horizontal plane in presence of deterministic uncertainty.

Uncertainty of aircraft position in the vertical dimension

Aircraft position may be subject to uncertainty in the vertical dimension mainly when the aircraft is not in its cruise phase, e.g. climb, descent; we shall call such flight phase: *non-level flight phase*. We shall assume that during such a non-level flight phase, the real altitude, denoted z^r, of the aircraft at a given time t lies in a bounded interval defined by an uncertainty radius R_v (set by the user) which reduces strongly when the aircraft reaches its requested flight level. The uncertainty of aircraft position in the vertical dimension, noted $\epsilon_v = z^r - z$, must therefore belong to the set:

$$\mathcal{U}_v := \{\epsilon_v \ : \ |\epsilon_v| \leq R_v\}\,. \tag{4.2}$$

In other words, the possible altitudes of the aircraft during non-level flight phase at time t are the elements of the set:

$$\{z^r : z - R_v \leq z^r \leq z + R_v\}.$$

To ensure vertical separation of aircraft subjected to such uncertainties, the vertical separation requirement has to be enlarged by R_v as illustrated in Figure 4.2. Thus, the *robust minimum separation in the vertical dimension*, noted N_v^r, is defined as:

$$N_v^r := N_v + R_v,$$

where N_v is the (usual) minimum vertical separation of the case without uncertainty.

Uncertainty of aircraft arrival time

In addition to the uncertainty in the 3D space domain (see Figure 4.3), aircraft may be subject to uncertainty so that it arrives at a given position with a time error. Let t_ϵ be the *maximum time error* (defined by the user). For simplicity, to implement the interaction

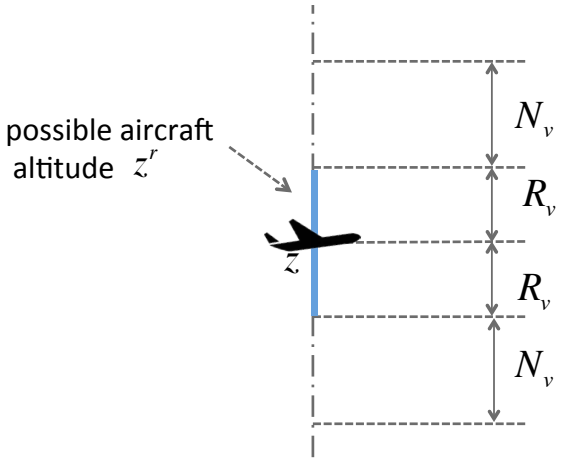

Figure 4.2: Possible aircraft altitude, z^r in presence of deterministic uncertainty in the vertical dimension.

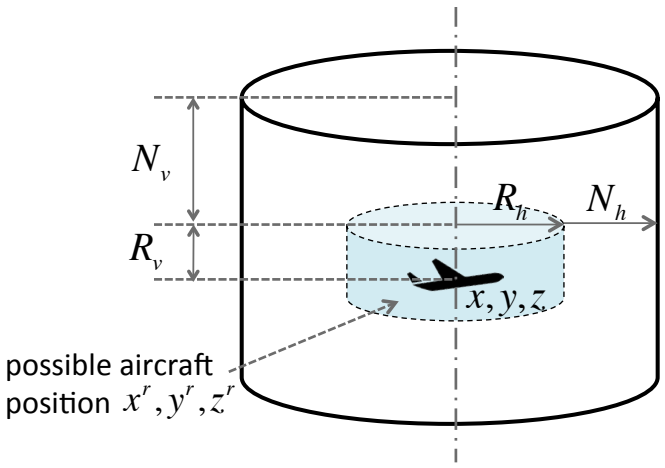

Figure 4.3: Possible aircraft position in the 3D space domain in presence of deterministic uncertainty.

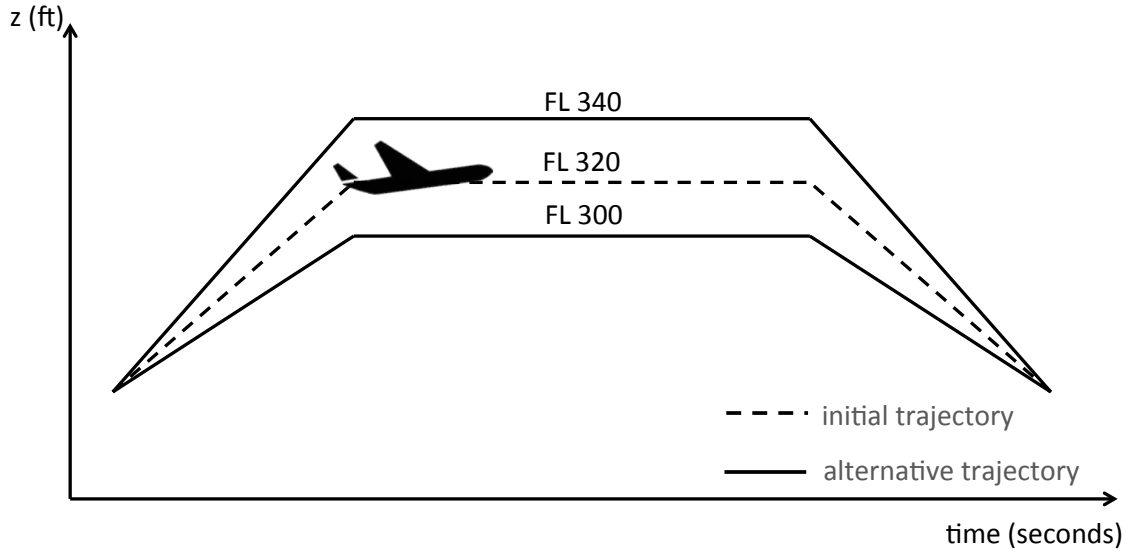

Figure 4.4: Two alternative vertical profiles for a trajectory (two alternative flight levels).

detection scheme, we shall assume that t_ϵ is chosen so that it is a multiple of the discretization time step t_s. The real arrival time, noted t^r, of aircraft at the same trajectory point therefore lies in the time interval:

$$[t - t_\epsilon, t + t_\epsilon].$$

The uncertainty of the arrival time, noted $\epsilon_t = t - t^r$, must therefore belong to the set:

$$\mathcal{U}_t := \{\epsilon_t \ : \ |\epsilon_t| \leq t_\epsilon\}. \tag{4.3}$$

4.2.2 Robust optimization formulation

The uncertainties restrict the solution space of the problem and make it more difficult to solve. Therefore, we introduce an additional degree of freedom to separate aircraft trajectories in the vertical plane, by allowing aircraft to fly at alternative flight levels. The alternative flight level to be allocated to each flight i is modeled as follows.

Alternative flight level. We define another decision variable associated to each flight i: a flight-level shift $l_i \in \mathbb{Z}$. Therefore, the flight level, FL_i, of flight i is given by:

$$FL_i = FL_{i,0} + l_i,$$

where $FL_{i,0}$ is the (given data) initially-planned flight level of flight i. Figure 4.4 shows a trajectory with two alternative flight levels.

Maximum allowed flight-level changes. In order to limit the change of flight levels, the flight level shift is also bounded. The set, ΔFL_i, of all possible flight-level shifts for flight i is:

$$\Delta FL_i := [FL_{i,0} - l_{i,max}, \ldots, FL_{i,0} - 1, 0, FL_{i,0} + 1, \ldots, FL_{i,0} + l_{i,max}], \tag{4.4}$$

where $l_{i,max}$ is the (user-provided) maximum flight level shifts allowed to be allocated to flight i.

Decision variables. Recall that we have set $\boldsymbol{\delta} = (\delta_1, \delta_2, \ldots, \delta_N)$ the vector of departure time shift, and $\mathbf{w} = (w_1, w_2, \ldots, w_N)$ the vector of virtual waypoint locations associated to flight $i = 1, \ldots, N$. Let us set another compact vector notation:

$$\mathbf{l} := (l_1, l_2, \ldots, l_N).$$

Therefore, the decision variables of the robust strategic trajectory planning can be represented by the vector:

$$u := (\boldsymbol{\delta}, \mathbf{l}, \mathbf{w}).$$

Objective function. Recall that we have defined the interaction at point $P_{i,k}(u_i)$ to be the sum of all the conflicts associated to point $P_{i,k}(u_i)$:

$$\Phi_{i,k}(u) := \sum_{\substack{j=1 \\ j \neq i}}^{N} \sum_{l=1}^{K_j} \mathcal{C}(P_{i,k}(u_i), P_{j,l}(u_j)),$$

where K_j is the number of sampling points for trajectory j, and

$$\mathcal{C}(P, Q) = \begin{cases} 1 & \text{if point } P \text{ is in conflict with point } Q \\ 0 & \text{otherwise.} \end{cases}$$

Let us now denote ϵ to be the uncertainty of aircraft positions and aircraft arrival times, and let $\mathcal{U} = \mathcal{U}_h \times \mathcal{U}_v \times \mathcal{U}_t$ be the uncertainty set, where $\mathcal{U}_h$, $\mathcal{U}_v$, and $\mathcal{U}_t$ are defined by (4.1), (4.2), and (4.3) respectively. The *robust interaction associated to the point* $P_{i,k}(u_i)$ considering the deterministic-type uncertainty, denoted $\Phi^D_{i,k}(u)$, can be defined as:

$$\Phi^D_{i,k}(u) = \sup_{\epsilon \in \mathcal{U}} \Phi_{i,k}(u, \epsilon) \tag{4.5}$$

The *robust interaction associated to trajectory* i is, therefore, defined as:

$$\Phi^D_i(u) = \sum_{k=1}^{K_i} \Phi^D_{i,k}(u),$$

where K_i is the number of sampled points of trajectory i. Finally, the *robust total interaction between trajectories*, that we are minimizing, is:

$$\Phi^D_{tot}(u) = \sum_{i=1}^{N} \sum_{k=1}^{K_i} \Phi^D_i(u), \tag{4.6}$$

where N is the total number of trajectories.

One wishes, therefore, to determine values for the optimization variables δ_i, l_i, and w_i for each flight $i = 1, 2, \ldots, N$ so as to minimize the total interaction, $\Phi^D_{tot}(u)$, between the N given trajectories.

To summarize, the robust strategic trajectory planning problem, based on deterministic-type uncertainty set, can be represented by an interaction minimization problem formulated as a mixed-integer optimization problem as follows:

$$\begin{aligned} &\min_{u} \Phi^{D}_{tot}(u) \\ &\text{subject to} \\ &\quad \delta_i \in \Delta_i, \quad i = 1, 2, \ldots, N \\ &\quad l_i \in \Delta FL_i, \quad i = 1, 2, \ldots, N \\ &\quad w_i^m \in W^m_{ix'} \times W^m_{iy'}, \quad m = 1, 2, \ldots, M,\ i = 1, 2, \ldots, N, \end{aligned} \tag{P3}$$

where $\Phi^{D}_{tot}(u)$ is defined by (4.6), and Δ_i, ΔFL_i, $W^m_{ix'}$, and $W^m_{iy'}$ are defined by (2.3), (4.4), (2.5), and (2.8) respectively.

4.2.3 Objective function computation

To explain the process to determine the interaction between aircraft trajectories taking into account deterministic-type uncertainties, let us first consider two trajectories A and B illustrated in Figure 4.5, and let P and Q be any pair of sample points on the trajectories A and B respectively. To compute the interaction between these two trajectories, we must check whether the minimum separations, N^r_h and N^r_v is satisfied, between every possible pair of points such as P and Q (pair-wise comparisons). Recall that the real arrival time, t^r_P, of aircraft A at

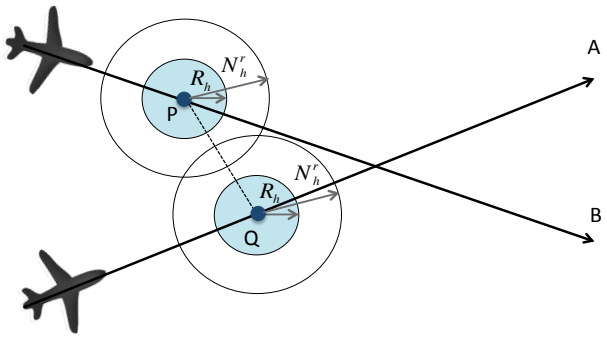

Figure 4.5: Evaluating the interaction between two continuous trajectories A and B in presence of deterministic-type uncertainty.

point P, and the real arrival time, t^r_Q, at point Q are subject to:

$$t^r_P \in [t_P - t_\epsilon, t_P + t_\epsilon],$$

and

$$t^r_Q \in [t_Q - t_\epsilon, t_Q + t_\epsilon],$$

respectively.

A potential conflict between trajectories A and B, taking into account uncertainties, can occur when the three following conditions are satisfied for a certain pair of sample points, P and Q, from each trajectory:

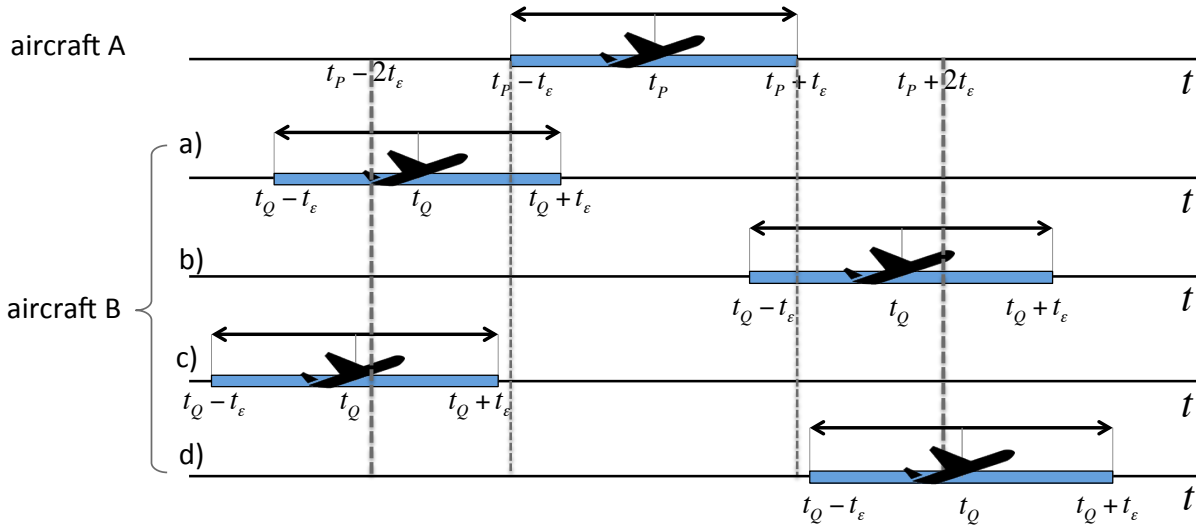

Figure 4.6: Possible scenarios of arrival time of two aircraft to the same 3D space region.

- $d_h := \sqrt{(x_P - x_Q)^2 + (y_P - y_Q)^2} < N_h^r$.
- $d_v := |z_P - z_Q| < N_v^r$.
- $[t_P - t_\epsilon, t_P + t_\epsilon] \cap [t_Q - t_\epsilon, t_Q + t_\epsilon] \neq \emptyset$, i.e. $|t_P - t_Q| \leq 2t_\epsilon$.

When the above conditions are satisfied, we say that *point P is in conflict with point Q taking into account the deterministic-type uncertainty.*

Figure 4.6 illustrates the four possible scenarios of arrival time of two aircraft to the same 3D space region. Remark that a potential conflict between P and Q can occur only in cases a) and case b) where

$$|t_P - t_Q| \leq 2t_\epsilon.$$

Let us define further

$$\mathcal{C}^D(P,Q) = \begin{cases} 1 & \text{if point } P \text{ is in conflict with point } Q \\ & \text{taking into account the deterministic uncertainty} \\ 0 & \text{otherwise.} \end{cases} \tag{4.7}$$

With the above definitions, we can perform implicitly the supremum computation involved in equation (4.5). Indeed, one can straightforwardly check that we have

$$\Phi_{i,k}^D(u) = \sum_{\substack{j=1 \\ j\neq i}}^{N} \sum_{l=1}^{K_j} \mathcal{C}^D(P_{i,k}(u_i), P_{j,l}(u_j)).$$

To implement the interaction detection scheme, in presence of this deterministic-type uncertainty, one simply has to adjust the size of the (3D space) grid cells according to the (user-provided) robust minimum separation N_h^r and N_v^r (*robust grid*). Then, to detect interactions,

for each cell (I_x, I_y, I_z, I_t) corresponding to each sampling point $P_{i,k} := (x_{P_{i,k}}, y_{P_{i,k}}, z_{P_{i,k}}, t_{P_{i,k}})$, one simply needs to check all the surrounding cells (in the robust grid) corresponding to the time period $[t_P - 2t_\epsilon, t_P + 2t_\epsilon]$. The algorithm used to compute the total interaction between N trajectories taking into account the deterministic-type uncertainty is described in detail in Algorithm 4.1.

Algorithm 4.1 Interaction computation algorithm in presence of deterministic-type uncertainty

Require: value of the decision variables $u = (\boldsymbol{\delta}, \mathbf{l}, \mathbf{w})$, and the time sequence of 3D robust grids (taking into account the N_h^r and N_v^r minimum separations.

Initialize $\Phi_{tot}^D(u) := 0$;
for i= 1 to N **do** ▷ (for each trajectory i)
  Discretize the alternate trajectory i defined by u_i into a sequence $\{P_{i,k}\}_{k=1}^{K_i}$;
  Initialize $\Phi_i^D(u) := 0$;
  for $k = 1$ to K_i **do** ▷ (for each point $P_{i,k}$ of trajectory i)
    Initialize $\Phi_{i,k}^D(u) := 0$;
    Compute the cell I_x, I_y, I_z, I_t corresponding to sample point $P_{i,k}$;
    Compute $\Phi_{i,k}^D(u)$:
    for $i_x = I_x - 1$ to $I_x + 1$ **do**
      for $i_y = I_y - 1$ to $I_y + 1$ **do**
        for $i_z = I_z - 1$ to $I_z + 1$ **do**
          for $i_t = I_t - 2\frac{t_\epsilon}{t_s}$ to $I_t + 2\frac{t_\epsilon}{t_s}$ **do**
            if $\exists j \neq i$ such that $j \in (i_x, i_y, i_z, i_t)$ **then**
              L:= list of all trajectory sample point in (i_x, i_y, i_z, i_t);
              for $l = 1$ to length(L) **do**
                $P := L(l)$;
                Check conflict, $\mathcal{C} := \mathcal{C}^D(P_{i,k}, P)$ using (4.7);
                if $\mathcal{C} = 0$ **then**
                  $\mathcal{C}$:=`interp`$(P_{i,k}, P)$;
                end if
                $\Phi_{i,k}^D(u) := \Phi_{i,k}^D(u) + \mathcal{C}$;
              end for
            end if
          end for
        end for
      end for
    end for
    $\Phi_i^D(u) := \Phi_i^D(u) + \Phi_{i,k}^D(u)$;
end for
$\Phi_{tot}^D(u) := \Phi_{tot}^D(u) + \Phi_i^D(u)$;
Return $\Phi_{tot}^D(u)$.

4.2.4 Resolution algorithm

To solve the robust strategic 4D trajectory planning problem, we rely on the hybrid SA / IILS algorithm proposed in Section 3.2. As we have introduced an additional decision variable, l_i, to modify the flight level of any given trajectory i, some modifications to the neighborhood function are made as follows.

Consider a chosen flight i to be modified, we introduce here another user-defined parameter, noted P_l, to control the probability to modify the flight level of flight i. This parameter P_l must satisfy:

$$P_w + P_l \leq 1,$$

where P_w is the previously-defined (user-provided) probability to modify the location of waypoints. Finally, the probability to modify the departure time of flight i is $1 - (P_w + P_l)$. The new neighborhood function considering flight level shifts is summarized in Algorithm 4.2.

Algorithm 4.2 Neighborhood function considering flight level shifts

Require: probabilities P_w, P_l, trajectory i.
Generate random number, $r :=$ random(0,1);
if $r < P_w$ **then**
    Choose randomly one virtual waypoint w_i^m to be modified.
    Choose randomly new $w_{ix'}^m$ from $W_{ix'}^m$;
    Choose randomly new $w_{iy'}^m$ from $W_{iy'}^m$;
else
    if $r < (P_w + P_l)$ **then**
        Choose randomly new flight level shift l_i from ΔFL_i;
    else
        Choose randomly new departure time shift δ_i from Δ_i;
    end if
end if

Two additional local search algorithms to intensify the search on each particular trajectory in different solution spaces are introduced as follows:

- **Intensify the search in the Time Domain (TD).** This local search module intensifies the search by modifying only the departure time of a given trajectory i. The algorithm repeats until a pre-defined number of local-search iterations is performed.
- **Intensify the search in the Flight-level Domain (FD).** This local search module intensifies the search by modifying the flight level of a given trajectory i. If the change of flight level yields an improvement of the objective-function value, the module further intensifies the search on the current flight level by applying a local change from the neighborhood structure to trajectory i (using the PT local search of Subsection (3.2.1). The algorithm repeats until a pre-defined number of local search iterations is performed.

4.2.5 Computational experiments

The robust strategic 4D trajectory planning methodology addressing the deterministic type uncertainty seen before is implemented on an AMD Opteron 2 GHz processor with 128 Gb RAM. It is tested with the national-size and continent-size air traffic previously considered in Chapter 3.

National-size en-route air traffic

First, we test the proposed methodology on the full-day national-size en-route air traffic over the French airspace (Problem instance 1) involving 8,836 trajectories. Simulations are

parameter	value
Sampling time step, t_s	20 seconds
Discretization time step for possible delay / advance departure-time shift, δ_s	20 seconds
Maximum departure time shift, $delta_a^i = \delta_d^i := \delta$	120 minutes
Maximum allowed route length extension coefficient, d_i	0.20
Maximum allowed flight level shifts, $l_{i,max} := l_{max}$	2
Maximum number of virtual waypoints, M	3

Table 4.1: Chosen (user-defined) parameter values specifying the robust optimization problem for the national-size air traffic.

parameter	value
Number of iterations at each temperature step, N_I	200
Initial rate of accepting degrading solutions, τ_0	0.3
Geometrical temperature reduction coefficient, β	0.99
Final temperature, T_f	$(1/500).T_0$
Inner-loop interpolation sampling time step, t_{interp}	5 seconds
Probability to modify horizontal flight profile, P_w	1/3
Probability to modify flight level, P_l	1/3
Threshold value, Φ_τ	0.5 Φ_{avg}

Table 4.2: Empirically-set (user-defined) parameter values of the resolution methodology to solve the national-size air traffic.

performed with different values for the parameters R_h, R_v, and t_ϵ, defining the size of the uncertainty sets.

The parameter values chosen to specify the optimization problem are given in Table 4.1. Simply to give an idea of the complexity of this problem, with regard to the dimension of the search space, remark that our optimization problem involves for this instance:

- $2MN = 53{,}016$ (continuous) virtual waypoint variables (the component of the vector $\mathbf{w}$);
- $N = 8{,}836$ (discrete) departure-time shift variables (the component of the vector $\boldsymbol{\delta}$), each of which involves $\frac{2\delta}{\delta_s} + 1 = 721$ possible values;
- $N = 8{,}836$ (discrete) flight-level shift variables (the component of the vector $\mathbf{l}$), each of which involves $2l_{max} + 1 = 5$ possible values;

for a total of 53,016 continuous variables and $2N = 17{,}672$ discrete variables involving a joint combinatoric of $(\frac{2\delta}{\delta_s} + 1)^N \cdot (2l_{max} + 1)^N = (721 \cdot 5)^{8{,}836}$ possible values.

The parameter values specifying the resolution algorithm are empirically set and given in Table 4.2. The initial and final total interaction between trajectories, the computation time, and the number of iterations performed to solve the problems considering different levels of uncertainty are reported in Table 4.3 (the vertical uncertainty radius, R_v, is used only when aircraft are climbing and descending).

The size of the uncertainty set affects the resolution time and the final total interaction between trajectories. When increasing the time uncertainty, the initial interaction increases significantly (cases 1, 3, 4 and 5), and the algorithm requires more computation time to converge.

case	uncertainty set dimensions	initial Φ^D_{tot}	final Φ^D_{tot}	solved interactions	CPU time (minutes)	no. of iterations
1	$R_h = 0$ Nm. $R_v = 0$ feet. $t_\epsilon = 180$ seconds.	2,282,436	5,934	99.7%	1,093.8	1,083,215
2	$R_h = 1$ Nm. $R_v = 100$ feet. $t_\epsilon = 60$ seconds.	765,448	0	100.0%	101.1	97,400
3	$R_h = 1$ Nm. $R_v = 100$ feet. $t_\epsilon = 120$ seconds.	1,425,384	4,314	99.7%	1,809.0	1,791,000
4	$R_h = 1$ Nm. $R_v = 100$ feet. $t_\epsilon = 240$ seconds.	2,821,706	37,290	98.7 %	2,213.3	2,191,970
5	$R_h = 2$ Nm. $R_v = 100$ feet. $t_\epsilon = 240$ seconds.	5,000,430	110,021	97.9%	2,289.8	2,266,956

Table 4.3: Initial and final total interaction between trajectories for the national-size air traffic, considering different dimensions for the uncertainty set (the vertical uncertainty radius, R_v, is relevant only when aircraft are climbing or descending).

The algorithm reaches an interaction-free solution for the case 2. It solves up to 99.7% of the initial interactions in the remaining cases (1, 3, 4, and 5), within computation times that are still compatible in a strategic planning context (the worst run, case 5, involving less than 38 hours of CPU time).

Continent-size air traffic

To test the proposed methodology on continent-size air traffic, simulations were performed on the en-route traffic scenario as well as on traffic involving the TMAs. However, due to the lack of data, alternative flight levels for these two continent-size instances are not available. To solve these problem instances, we limit the maximum flight level change, $l_{i,max}$, to zero, for all flight i ($l_{max} = 0$). The uncertainty of aircraft position in the TMA is not taken into account ($R_h^{TMA} = 0$, and $R_v^{TMA} = 0$), since during this phase of flight, aircraft are usually required to follow a given path with very high precision.

The user-defined input parameters of the optimization algorithm are all set to the same values as those for the national-size en-route air traffic (Table 4.1), except for l_{max} which is set to zero as explained above. These values are displayed in Table 4.4. Since for this instance there is no flight-level variables, the complexity of the optimization problem (in terms of the dimension of the search space) is reduced. For the problem instance involving en-route air traffic, one has

- $2MN = 179{,}112$ (continuous) virtual waypoint variables (the component of the vector $\mathbf{w}$);
- $N = 29{,}852$ (discrete) departure-time shift variables (the component of the vector $\boldsymbol{\delta}$), each of which involves involving $(\frac{2\delta}{\delta_s} + 1)^N = 721^{29{,}852}$ possible values;

parameter	value
Sampling time step, t_s	20 seconds
Discretization time step for possible delay / advance departure-time shift, δ_s	20 seconds
Maximum departure time shift, $-\delta_a^i = \delta_d^i := \delta$	120 minutes
Maximum allowed route length extension, d_i	0.20
Maximum allowed flight level shifts, $l_{i,max}$	0
Maximum number of waypoints, M	3

Table 4.4: Chosen (user-defined) parameter values specifying the robust optimization problem for continent-size air traffic.

parameter	value
Number of iterations at each temperature step, N_I	4,000
Initial rate of accepting degrading solution, τ_0	0.3
Geometrical temperature reduction coefficient, β	0.99
Final temperature, T_f	$(1/500).T_0$
Inner-loop interpolation sampling time step, t_{interp}	5 seconds
Probability to modify horizontal flight profile, P_w	0.5
Probability to modify flight level, P_l	0.0
Threshold value, Φ_τ	0.5 Φ_{avg}

Table 4.5: Empirically-set (user-defined) parameter values of the resolution methodology to solve the continent-size air traffic instances.

for total of 208,964 decision variables. For the problem instance involving air traffic in the TMAs, one has

- $2MN = 184{,}170$ (continuous) virtual waypoint variables (the component of the vector $\mathbf{w}$);
- $N = 30{,}695$ (discrete) departure-time shift variables (the component of the vector $\boldsymbol{\delta}$), each of which involves involving $(\frac{2\delta}{\delta_s} + 1)^N = 721^{30{,}695}$ possible values;

for total of 214,865 decision variables.

The user-defined parameter values specifying the hybrid SA / IILS algorithm to solve these problem instances are given in Table 4.5. The only difference with those for the national-size instance (Table 4.2) is that the number of iterations at each temperature step is increased from 200 to 4,000, the probability to modify the flight level becomes irrelevant (set to zero), and the probability to modify the horizontal flight profile is consequently increased from 1/3 to 1/2.

The initial and final total interaction between trajectories, and the computation time to solve the problem considering different levels of uncertainty are reported in Table 4.6. Although the trajectories can be separated only by modifying the horizontal flight profile and the departure time of each flight, the resolution algorithm finds an interaction-free solution, taking into account uncertainty of aircraft positions, for both problem instances in cases 2 and 4. When time uncertainty is considered (case 1 and 3), there remains less than 15% of the initial interaction between trajectories. This could be improved by introducing more degrees of freedom to the solution space, e.g. alternative flight levels, or speed regulation in the TMA. The most remaining interactions are located in the TMA. When we took into account only the en-route phase as in the French air traffic case, one can expect much better results. Remember also that in the

case	traffic scenario	uncertainty set dimensions	initial Φ^D_{tot}	final Φ^D_{tot}	solved interactions	CPU time (minutes)	no. of iterations
1	only en-route	$R_h = 3$ Nm. $R_v = 200$ feet. $t_\epsilon = 60$ s.	5,142,632	634,474	87.7 %	2,756.2	2,728,776
2	only en-route	$R_h = 3$ Nm. $R_v = 200$ feet. $t_\epsilon = 0$ s.	430,234	0	100.0 %	347.6	345,528
3	with TMA	$R_h = 0$ Nm. $R_v = 0$ feet. $t_\epsilon = 120$ s.	3,874,402	560,114	85.5 %	2,652.1	2,625,714
4	with TMA	$R_h = 3$ Nm. $R_v = 200$ feet. $t_\epsilon = 0$ s.	487,698	0	100.0 %	578.4	572,648

Table 4.6: Initial and final total interaction between trajectories for the continent-scale air traffic with different dimensions for the uncertainty set.

continental case, there is no alternative choices in the vertical dimension (due to the lack of data).

4.3 Robust strategic 4D trajectory planning based on probabilistic-type uncertainty model

The worst-case-oriented methodology to consider uncertainty presented in the previous section tries to guarantee an interaction-free solution over every possible cases of the given uncertainty set. However, some events corresponding to the points in the uncertainty set have very low probability to occur. Trying to immune the solution against such events could yield unnecessarily costly solutions, and can be interpreted as too conservative for a situation involving high levels of uncertainty as it is the case in strategic planning.

Instead of relying on deterministic sets to define the uncertainties, we assume in this sector that the probability distribution of uncertainty is known. Therefore, instead of trying to guarantee separation between the envelopes of trajectories (the set of all possible trajectories), we introduce a methodology that aims, roughly speaking, at minimizing a weighted overlap of the uncertainty envelopes, where the weights are driven by the uncertainty probability distribution. This will be defined formally in Subsection 4.3.3.

As an aircraft is able to follow a given flight profile with very high accuracy thanks to the flight management system (FMS).[1], we shall consider in this section that the residual uncertainty of aircraft position is more likely to occur in the time domain, assuming that an aircraft is able to follow a given trajectory with high precision in the 3D space domain.

[1]Flight management system (FMS) is an on-board computer system that determines the aircraft exact position and calculates the lateral and horizontal guidance for the aircraft

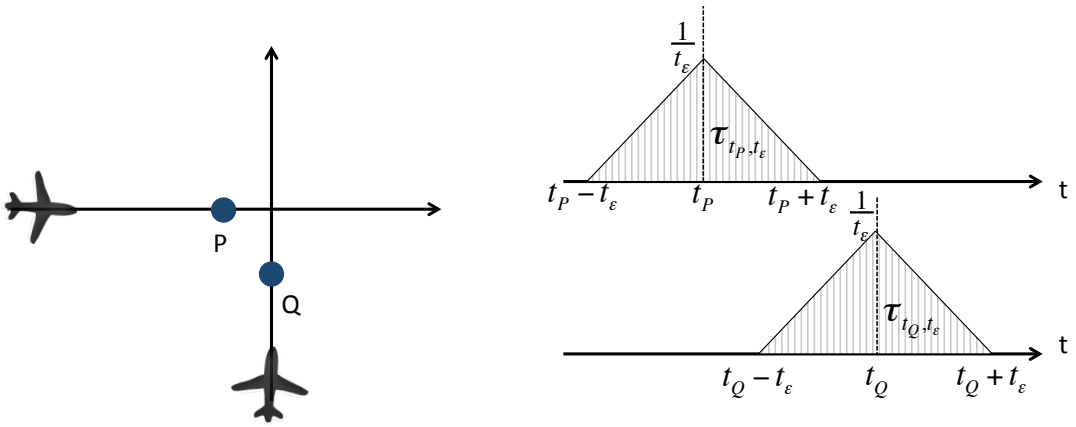

Figure 4.7: Uncertainty of aircraft arrival time, defined by triangular distribution over given time intervals (left: view in the space domain; right: view in the time domain).

4.3.1 Probabilistic uncertainty model

Using the maximum time error, t_ϵ (set by the user), the *predicted* arrival time of an aircraft at a position P under uncertainty lies in the interval:

$$[t_P - t_\epsilon, t_P + t_\epsilon],$$

where t_P is the assigned arrival time to point P. For the purpose of potential conflict detection, we assume here that the predicted aircraft arrival time can be modeled as a random variable with the following triangular distribution defined over the interval $[t_P - t_\epsilon, t_P + t_\epsilon]$. Given the lower limit $t_P - t_\epsilon$, the upper limit $t_P + t_\epsilon$, the *predicted* arrival time, denoted $\hat{t}_P$, to the position P is given by the probability density function:

$$\hat{t}_P(t) = \mathcal{T}_{P,t_\epsilon}(t),$$

where $\mathcal{T}_{P,t_\epsilon}(t)$ denotes the triangular distribution:

$$\mathcal{T}_{P,t_\epsilon}(t) = \begin{cases} 0 & \text{for } t < t_P - t_\epsilon, \\ \frac{(t - t_P + t_\epsilon)}{t_\epsilon^2} & \text{for } t_P - t_\epsilon \leq t \leq t_P, \\ \frac{(-t_P + t_\epsilon - t)}{t_\epsilon^2} & \text{for } t_P < t \leq t_P + t_\epsilon, \\ 0 & \text{for } t_P + t_\epsilon < t. \end{cases} \tag{4.8}$$

Figure 4.7 illustrates the uncertainty of arrival time of two aircraft A and B to the trajectory sample points P and Q respectively defined by a triangular distribution function over the time interval $[t_P - t_\epsilon, t_P + t_\epsilon]$ and $[t_Q - t_\epsilon, t_Q + t_\epsilon]$ respectively.

4.3.2 Optimization formulation

To solve the robust strategic 4D trajectory planning problem based on the above probabilistic-type uncertainty set, we rely on the same trajectory separation methods, the same decision

variables, and the same constraints as presented in Section 4.2. The objective function of this problem can be modeled as follows.

Objective function. Given values of the decision variables, the components of the optimization vector $u = (\boldsymbol{\delta}, \mathbf{l}, \mathbf{w})$, we define a *robust interaction at a point* P based on the probabilistic-type uncertainty, to be the sum of all the probabilistic interaction associated to point P. Given a trajectory i and the k^{th} sample point, $P_{i,k}(u_i)$, we note $\Phi^P_{i,k}(u)$ the robust interaction at point $P_{i,k}(u_i)$ based on probabilistic-type uncertainty. Remark that it depends also on the decision variables related to the other trajectories $j \neq i$. Hence, we have

$$\Phi^P_{i,k}(u) := \sum_{\substack{j=1 \\ j\neq i}}^{N} \sum_{l=1}^{K_j} \mathcal{P}_{t_\epsilon}(P_{i,k}(u_i), P_{j,l}(u_j)),$$

where K_j are the number of sampling points for trajectory j, and where $\mathcal{P}_{t_\epsilon}(P_A, P_B)$ is the probabilistic interaction associated to the sample points P_A and P_B of trajectory A and B respectively. It will be defined precisely in the next Subsection (4.3.3).

In this probabilistic uncertainty context, we redefine the *robust interaction associated with trajectory* i, denoted $\Phi^P_i(u)$, is defined as follows:

$$\Phi^P_i(u) := \sum_{k=1}^{K_i} \Phi^P_{i,k}(u).$$

Finally, the *total interaction between trajectories*, $\Phi^P_{tot}(u)$, for a whole N-aircraft traffic situation is simply defined as:

$$\Phi^P_{tot}(u) = \sum_{i=1}^{N} \Phi^P_i(u) = \sum_{i=1}^{N} \sum_{k=1}^{K_i} \Phi^P_{i,k}(u). \tag{4.9}$$

One wishes to determine values for the optimization variables δ_i, l_i, and w_i for each flight $i = 1, 2, \ldots, N$ so as to minimize the total robust interaction, $\Phi^P_{tot}(u)$, between the N given trajectories.

To summarize, given a value, t_ϵ, of time uncertainty and the triangular probability distribution defined in (4.8), the strategic trajectory planning problem with probabilistic-type uncertainty can be represented by a robust interaction minimization problem formulated as the following mixed-integer optimization problem:

$$\begin{aligned} &\min_u \Phi^P_{tot}(u) \\ &\text{subject to} \\ &\quad \delta_i \in \Delta_i, \quad i = 1, 2, \ldots, N \\ &\quad l_i \in \Delta FL_i, \quad i = 1, 2, \ldots, N \\ &\quad w_i^m \in W^m_{ix'} \times W^m_{iy'}, \quad m = 1, 2, \ldots, M,\ i = 1, 2, \ldots, N, \end{aligned} \tag{P4}$$

where $\Phi^P_{tot}(u)$ is defined by (4.9), and Δ_i, ΔFL_i, $W^m_{ix'}$, and $W^m_{iy'}$ are defined by (2.3), (4.4), (2.5), and (2.8) respectively.

4.3.3 Objective function computation

To explain the process to compute the total robust interaction between trajectories based on probabilistic-type uncertainty, let us consider the trajectories A and B given in Figure 4.7. Let P and Q be any trajectory sample points on trajectories A and B respectively. The predicted arrival time, $\hat{t}_P$, of aircraft A to the given point P, and the *predicted* arrival time, $\hat{t}_Q$, of aircraft B to the given point Q are given by:

$$\hat{t}_P(t) = \mathcal{T}_{t_P,t_\epsilon}(t),$$

and

$$\hat{t}_Q(t) = \mathcal{T}_{t_Q,t_\epsilon}(t).$$

Again, a potential conflict between trajectories A and B occurs when there exists a pair of points, P_A and P_B, from each trajectory such that the three following conditions are satisfied:

- $d_h < N_h$;
- $d_v < N_v$;
- and $[t_P - t_\epsilon, t_P + t_\epsilon] \cap [t_Q - t_\epsilon, t_Q + t_\epsilon] \neq \emptyset$.

The probabilistic interaction, denoted $\mathcal{P}_{t_\epsilon}(P, Q)$, associated to the trajectory sample points P and Q is formally defined as follows:

$$\mathcal{P}_{t_\epsilon}(P, Q) := \int_{I_{PQt_\epsilon}} \hat{t}_P(t)\hat{t}_Q(t)dt, \tag{4.10}$$

where I_{PQt_ϵ} denotes the time interval $[t_P - t_\epsilon, t_P + t_\epsilon] \cap [t_Q - t_\epsilon, t_Q + t_\epsilon]$. Remark that when this intersection is the empty set, the integral in (4.10) reduced to zero. The interaction detection algorithm taking into account probabilistic uncertainty of aircraft arrival time is detailed in Algorithm 4.3.

4.3.4 Computational experiments

To test the proposed robust strategic 4D trajectory planning methodology based on the probabilistic-type uncertainty model, we rely on the hybrid SA / IILS algorithm with modifications presented in Subsection (4.2.4). The proposed methodology is implemented on an AMD Opteron 2 GHz processor with 128 Gb RAM. It is tested again with the national-size and continent-size air traffic presented in Chapter 3.

National-size en-route air traffic

The parameter values that specify the problem under consideration are the same as those given in Table 4.1. The parameters of the hybrid simulated-annealing / local-search are the same as those presented in Table 4.2.The simulations are performed considering successively aircraft maximum time uncertainty, t_ϵ, of 1 up to 4 minutes, respectively.

The initial and final interaction between trajectories and the required computation time are reported in Table 4.7. Remark that the initial total interactions between trajectories are

Algorithm 4.3 Interaction computation algorithm in presence of probabilistic-type uncertainty

Require: value of the decision variables $u = (\boldsymbol{\delta}, \mathbf{l}, \mathbf{w})$

Initialize $\Phi^P_{tot}(u) := 0$;
for i= 1 to N **do** ▷ (for each trajectory i)
  Discretize the alternate trajectory i defined by u_i into a sequence $\{P_{i,k}\}_{k=1}^{K_i}$;
  Initialize $\Phi^P_i(u) = 0$;
  for $k = 1$ to K_i **do** ▷ (for each point $P_{i,k}$ of trajectory i)
    Initialize $\Phi^P_{i,k}(u) := 0$;
    Compute the cell I_x, I_y, I_z, I_t corresponding to $P_{i,k}$;
    Compute $\Phi^P_{i,k}(u)$:
    for $i_x = I_x - 1$ to $I_x + 1$ **do**
      for $i_y = I_y - 1$ to $I_y + 1$ **do**
        for $i_z = I_z - 1$ to $I_z + 1$ **do**
          for $i_t = I_t - 2\frac{t_\epsilon}{t_s}$ to $I_t + 2\frac{t_\epsilon}{t_s}$ **do**
            if $\exists j \neq i$ such that $j \in (i_x, i_y, i_z, i_t)$ **then**
              L:= list of all trajectory sample point P_j in (i_x, i_y, i_z, i_t);
              for $l = 1$ to length(L) **do**
                $P := L(l)$;
                compute probabilistic interaction, $\mathcal{P} := \mathcal{P}_{t_\epsilon}(P_{i,k}, P)$ using (4.10);
                if $\mathcal{P} = 0$ **then**
                  $\mathcal{P}$:= `interp(P_{i, k}, P)`;
                end if
                $\Phi^P_{i,k}(u) := \Phi^P_{i,k}(u) + \mathcal{P}$;
              end for
            end if
          end for
        end for
      end for
    end for
  end for
  $\Phi^P_i(u) := \Phi^P_i(u) + \Phi^P_{i,k}(u)$;
end for
$\Phi^P_{tot}(u) := \Phi^P_{tot}(u) + \Phi^P_i(u)$;
Return $\Phi^P_{tot}(u)$.

significantly smaller than those of the worst-case-oriented approach. This is not surprising, since in the later (deterministic) case one counts one interaction when in the former (probabilistic) case there is even only a tiny positive probability of conflict.

The proposed strategic trajectory planning methodology is able to find interaction-free trajectory planning for all cases. When considering higher level of time uncertainty (4 minutes), the solution space becomes more constrained and therefore the algorithm requires more computation time to converge.

t_ϵ (seconds)	initial Φ^P_{tot}	final Φ^P_{tot}	solved interactions	CPU time (minutes)	No. of iterations
60	217,441.37	0.0	100.0 %	116.07	114,970
90	274,953.55	0.0	100.0 %	175.4	173,736
120	383,967.60	915.04	99.8 %	586.3	1,031,730
240	718,374.42	1,547.13	99.8 %	1,052.4	1,041984

Table 4.7: Numerical results for the national-size air traffic considering four different levels of aircraft maximum time uncertainty (1 to 4 minutes).

Continent-size air traffic

Now we test the algorithm with the continent-size air traffic considering en-route as well as the air traffic in the TMA. The parameter values that specify the problem under consideration are, here again, the same as those given in Table 4.4. The parameters of the hybrid SA / IILS are the same as those given in Table 4.5, with the number of iterations at each temperature step, N_I empirically set to 2,000, more than for the above, smaller, national-size instance (N_I =200).

case	traffic scenario	t_ϵ (seconds)	initial Φ^P_{tot}	final Φ^P_{tot}	solved interaction	CPU time (minutes)	no. of iterations
1	en-route	60	529,555.5	12,550.0	97.6 %	1,341.7	1,328,152
2	en-route	120	1,079,738.4	40,706.2	96.2 %	2,254.2	2,231,881
3	with TMA	60	1,128,282.7	85,185.3	92.5 %	2,340.7	2,317,065
4	with TMA	120	2,344,753.2	147,250.5	93.7%	2,453.8	2,429,264

Table 4.8: Numerical results for the continent-size instances, with and without TMA traffic, considering two different levels of time uncertainty.

The initial and final interactions between trajectories, and computation time to solve the problem are reported in Table 4.8. Recall again that, as in the case without uncertainty, alternative flight levels for these two (with and without TMA traffic) continent-size instances are not available. Therefore, due to this lack of data, these problem instances can be separated only by modifying the horizontal flight profile and by modifying the departure time of aircraft ($l_{i,max}$ is therefore set to zero for all flight i, and the only decision variables are the components of the optimization vectors $\mathbf{w}$ and $\boldsymbol{\delta}$). Nevertheless, there still remains less than 7% of the initial interactions taking into account the probabilistic-type time uncertainty.

4.4 Conclusions

In this chapter, we have presented a methodology to consider uncertainty of aircraft position and arrival time in the strategic 4D trajectory planning process. The proposed approach relies on the concept of robust optimization to incorporate uncertainties into the trajectory optimization process. This problem is more difficult to solve, since the uncertainty decreases substantially the size of the search space. Therefore, additional degrees of freedom to modify the flight level of each flight have been introduced in order to able to propose a resolution algorithm that is compatible with the operational context of large-scale problem.

First, the uncertainties have been modeled with deterministic sets. The algorithm therefore tries to minimize the interaction between trajectories considering all possible scenario implicitly described by the uncertainty sets (worst-case approach). The algorithm developed for the case without uncertainty was adopted via a modification of the way the objective function is evaluated. The modified algorithm was tested on national-size and continent-size air traffic. It was able to find interaction-free solutions for some uncertainty set sizes. There remains less than 15 % of the initial interactions when the size of the uncertainty set is larger.

Since such a worst-case uncertainty model can be far too conservative in practice, probabilistic-type uncertainty sets were then considered to represent aircraft arrival time, assuming that an aircraft is able to follow the 3D spatial constraints of its trajectory with high precision. The resulting algorithm was first tested on a national-size scenario. It is able to separate all trajectories considering time uncertainty intervals ($2t_\epsilon$) of up to 3 minutes, and solve 99.8 % of interactions considering time uncertainty intervals ($2t_\epsilon$) of up to 8 minutes. Then, it was tested on the continent-size air traffic scenario considering en-route traffic, as well as traffic in the TMA. The algorithm is able to separate near 97 % of the initial interaction between trajectories for en-route traffic, and near 94% when including TMA traffic considering time uncertainty interval ($2t_\epsilon$) of 4 minutes.

The level of uncertainty to be considered is a trade-off between the desired robustness of the solution obtained and the associated trajectory modifications costs, to be decided by the user. Considering too important uncertainty in strategic planning will, indeed, results in a lost of capacity, since large portions of airspace have to be cleared for a given aircraft for a long period of time. Instead, the user can consider lower uncertainty levels, and iteratively solve the remaining interactions during pre-tactical and tactical phases.

When the traffic in the TMA is included, result may be improved by adding speed control variable in the state space to reduce interaction in such area which is not the case in the above-presented algorithm (in all benchmark with TMA, the remaining interaction are mainly located in such area).

Conclusions and perspectives

To accommodate the continuously growing air traffic demand, the world's major air traffic management (ATM) systems are being transformed towards trajectory-based operations, which focus on managing aircraft trajectories in order to improve the ATM efficiency and to maximize the use of the available airspace. One of the key factors to improve the ATM capacity is the new conflict management concept that aims at transferring the tactical conflict detection and resolution tasks to the strategic trajectory planning phase. Instead of the more conventional approach that tries to satisfy given airspace capacity constraints, the new conflict management concept focuses on increasing the capacity by managing conflicts between aircraft trajectories so that the air traffic will become easier to manage. Therefore the controller will be able to accommodate more flights in a given airspace.

Contributions

In this thesis, we have contributed to the domain of air traffic management research in the framework of future ATM paradigm. More precisely, we have proposed and developed the following model, algorithms and overall methodology:

Mathematical model for strategic trajectory planning methodology

We have introduced a concept of interaction to quantify the situation where pairs of aircraft samples trajectory points violate the separation norms so that minimizing interaction boils down to minimizing conflicts between trajectories.

We have introduced a mathematical model to separate all aircraft trajectories for large-scale traffic scenarios by allocating alternative routes, alternative departure times, and alternative flight levels to flights in conflicts, yielding to a discrete optimization problem and to a mixed-integer optimization problem.

4D grid-based interaction-computation algorithm

We have developed a computationally-efficient algorithm to detect and compute interaction between N aircraft trajectories that can handle large-scale applications. To avoid using the $\frac{N(N-1)}{2}$ time-consuming pair-wise comparisons to detect conflicts between any pair of trajectory sample points, we have developed a detection scheme exploiting a four-dimension (3D space + time) grid that limits the number of sampling trajectory points to be checked.

This algorithm is very flexible and easy to extend to larger airspace areas, to other separation constraints, or to complex uncertainty models. Furthermore, due to its short computation time, the application of the proposed detection algorithm is not only limited to the strategic planning phase, it can also be applied, for instance, in a tactical planning context involving a smaller number of trajectories.

Metaheuristic algorithms to solve the interaction-minimization problems

To solve the above-described optimization problems for large-scale and complex strategic 4D trajectory planning problems, we relied on metaheuristic algorithms.

First, we adapted a simulated annealing algorithm, a classical non-population based metaheuristic algorithm, to address the problem. The proposed implementation was successfully tested on national-size and continent-size air traffic scenarios involving more than 30,000 trajectories. However, the simulated annealing requires more than one hour of computation time to converge to zero-interaction solutions.

To improve the efficiency of the resolution algorithm, we have developed a hybrid-metaheuristic algorithm, combining the simulated annealing with two simple local search modules. This hybrid algorithm was successfully implemented and tested on both the national and continent size air traffic, and converged to interaction-free solutions up to 3 times faster than the pure simulated annealing.

Overall methodology for strategic trajectory planning

The overall methodology we introduced first simulates trajectories using the BADA (Base of Aircraft Data) model, and proposes modified 4D trajectories thanks to the above optimization and the efficient interaction-computation algorithm. As a result, one can handle large-scale air traffic within reasonable computation time.

Methodology for robust strategic 4D trajectory planning

To improve robustness of the resulting interaction-free 4D trajectories, we have introduced a method to consider uncertainty of aircraft position and arrival time, based on the concept of robust optimization. Two different uncertainty models were used to represent the uncertainty sets.

The first one modeled the uncertainty with deterministic sets. Our overall methodology was adapted to take into account such a worst-case oriented uncertainty model. It was successfully implemented and tested. However, it appears that it is too conservative to try to guarantee interaction-free trajectories against every possible uncertainty scenario in the strategic planning when one deals with high levels of uncertainty. Indeed, preliminary numerical simulations tend to show that using too large uncertainty sets can result in a decrease of capacity.

We have considered a less conservative uncertainty model, representing uncertainty of aircraft arrival time using probabilistic sets. This yields a probabilistic adaptation of our concept of interaction. The proposed methodology was again adapted, implemented and tested, yielding interaction-free solutions for time uncertainty intervals up to 4 minutes wide.

Perspectives

Further research could follow the following recommendations:

Speed adjustment in the TMAs

The high density air traffic in the TMAs was managed in this thesis by acting only in the time domain in order to respect the standard departure and arrival procedures. To improve the interaction management in this area, one could consider increasing the number of degrees of freedom in the search space by allowing also speed adjustment (new decision variables in the optimization formulations) in the TMAs.

Minimizing flight-plan modifications

The objective of the proposed methodology was to minimize the total interaction between trajectories, while the cost of modifying the flight plans were considered implicitly in the optimization constraints. Our methodology can find zero-interaction solutions; it would be relevant to attempt at choosing, among all such solutions, the ones that involve the lowest cost, including for instance the costs associated to the modification of flight plans (e.g. delay cost, extra fuel consumption). To extend the work presented in this thesis, one can concentrate on reducing the total costs associated to the modifications of the initial flight plans; thereby reducing the number of departure time shifts, the total trajectory length extension, and the number of flight-level shifts.

This could be achieved, for example, by relying on the concept of "dynamical" constraints, where the optimization constraints may change over the solving time. The idea is to start from a small, restricted search space and, if necessary, gradually increase the size of the search space as a function of the simulated-annealing temperature steps that have been performed. Another idea is to rely on the concept of multi-objective optimization [46]. Due to the computational memory requirement, one could simply consider scalarizing the objectives into a single one.

Improving uncertainty models

In this thesis, preliminary steps to improve robustness of the resulting trajectories were conducted by taking into account uncertainty of aircraft position and arrival time into the strategic trajectory planning process. To improve further robustness of the solutions, one could consider using more realistic uncertainty models, for instance, considering evolutions of the uncertainty sets during the flight time.

References

[1] E. Aarts and J. K. Lenstra. *Local search in combinatorial optimization*. Princeton University Press, 1997.

[2] A. Agustín, A. Alonso-Ayuso, L.F. Escudero, and C. Pizarro. Mathematical optimization models for air traffic flow management: A review. In *A. Bui, I. Tseveendorkj (Eds.), Combinatorial Optimization in Practice*, volume 8, pages 141–184. Hermann Informatique, Studia Informatica Universalis, 2010.

[3] A. Agustín, A. Alonso-Ayuso, L.F. Escudero, and C. Pizarro. On air traffic flow management with rerouting. Part I: Deterministic case. *European Journal of Operational Research*, 219(1):156–166, May 2012.

[4] A.V. Aho, J.E. Hopcroft, and J.D. Ullman. *Data Structures and Algorithms*. Addison-Wesley, 1983.

[5] A. Akgunduz, B. Jaumard, and G. Moeini. Non-time indexed modeling for en-route flight planning with speed-fuel consumption trade-off. In *ISIATM 2013, the 2nd International Conference on Interdisciplinary Science for Innovative Air Traffic Management*, Toulouse (France), 2013.

[6] S. Alam, K. Shafi, H. A. Abbass, and M. Barlow. An ensemble approach for conflict detection in free flight by data mining. *Transportation Research Part C: Emerging Technologies*, 17(3):298–317, 2009.

[7] C. Allignol, N. Barnier, and A. Gondran. Optimized vertical separation in Europe. In *DASC 2012, the 31st IEEE/AIAA Digital Avionics Systems Conference*, pages 4B3–1–4B3–10, October 2012.

[8] J.-M. Alliot, J.-F. Bosc, N. Durand, and L. Maugis. Cats: A complete air traffic simulator. In *DASC 1997, the 16th AIAA/IEEE Digital Avionics Systems Conference*, volume 2, pages 8.2–30–8.2–37 vol.2, October 1997.

[9] A. Alonso-Ayuso, L.F. Escudero, and F.J. Martín-Campo. Collision avoidance in air traffic management : A mixed-integer linear optimization approach. *IEEE Transactions on Intelligent Transportation Systems*, 12(1):47–57, 2011.

[10] G. Andreatta, L. Brunetta, and G. Guastalla. The flow management problem: recent computational algorithms. *Control Engineering Practice*, 6(6):727–733, 1998.

[11] G. Andreatta, A. R. Odoni, and O. Richetta. Models for the ground holding problem. In L. Bianco and A.R. Odoni, editors, *Large Scale Computation and Information Processing in Air Traffic Control*, Transportation Analysis, pages 125–168. Springer, 1993.

[12] G. Andreatta and G. Romanin-Jacur. Aircraft flow management under congestion. *Transportation Science*, 21(4):249–253, 1987.

[13] Anonymous. Worldwide market forecast for commercial air transport 2010-2029. Technical Report YGR-5067, Marketing, Japan Aircraft Development Corporation, 2010.

[14] M. A. Arostegui, S. N. Kadipasaoglu, and B. M. Khumawala. An empirical comparison of tabu search, simulated annealing, and genetic algorithms for facilities location problems. *International Journal of Production Economics*, 103(2):742–754, October 2006.

[15] N. Barnier and C. Allignol. 4D - trajectory deconfliction through departure time adjustment. In *ATM 2009, the 8th USA/Europe Air Traffic Management Research and Development Seminar*, Napa (California), 2009.

[16] N. Barnier and C. Allignol. Combining flight level allocation with ground holding to optimize 4D-deconfliction. In *ATM 2011, the 9th USA/Europe Air Traffic Management Research and Development Seminar*, Berlin (Germany), 2011.

[17] N. Barnier and P. Brisset. Graph coloring for air traffic flow management. *Annals of Operations Research*, 130(1-4):163–178, 2004.

[18] A. Ben-Tal, L. El Ghaoui, and A. Nemirovski. *Robust optimization*. Princeton University Press, 2009.

[19] A. Ben-Tal and A. Nemirovski. Robust convex optimization. *Mathematics of Operations Research*, 23:769–805, 1998.

[20] A. Ben-Tal and A. Nemirovski. Robust solutions of uncertain linear programs. *Operations Research Letters*, 25:1–13, 1999.

[21] D. Bertsimas, D.B. Brown, and C. Caramanis. Theory and applications of robust optimization. *SIAM Review*, 53(3):464–501, 2011.

[22] D. Bertsimas, G. Lulli, and A. Odoni. The air traffic flow management problem: An integer optimization approach. In *Integer Programming and Combinatorial Optimization*, volume 5035 of *Lecture Notes in Computer Science*, pages 34–46. Springer, 2008.

[23] D. Bertsimas, G. Lulli, and A. Odoni. An integer optimization approach to large-scale air traffic flow management. *Operations Research*, 59(2):211–227, 2011.

[24] D. Bertsimas and S. Patterson. The air traffic flow management problem with en-route capacities. *Operations Research*, 46(3):406–422, 1998.

[25] H.-G. Beyer and B. Sendhoff. Robust optimization - A comprehensive survey. *Computer Methods in Applied Mechanics and Engineering*, 196(33-34):3190 – 3218, 2007.

[26] C. E. Bichot. Metaheuristics versus spectral and multilevel methods applied on an air traffic control problem. In Alexandre Dolgui, editor, *12th IFAC Symposium on Information Control Problems in Manufacturing*, pages 493–498, May 2006.

[27] C. Blum, M. Aguilera, A. Roli, and M. Sampels. *Hybrid Metaheuristics: An Emerging Approach to Optimization*. Springer, 1st edition, 2008.

[28] C. Blum and A. Roli. Hybrid metaheuristics: An introduction. In C. Blum, M. J. B. Aguilera, A. Roli, and M. Sampels, editors, *Hybrid Metaheuristics*, volume 114 of *Studies in Computational Intelligence*, pages 1–30. Springer, 2008.

[29] S. Cafieri and N. Durand. Aircraft deconfliction with speed regulation: new models from mixed-integer optimization. *Journal of Global Optimization*, 58(4):613–629, April 2014.

[30] S. Chaimatanan, D. Delahaye, and M. Mongeau. A methodology for strategic planning of aircraft trajectories using simulated annealing. In *ISIATM 2012, the 1st International Conference on Interdisciplinary Science for Innovative Air Traffic Management*, Florida, 2012. `http://hal-enac.archives-ouvertes.fr/hal-00912772/PDF/Chaimatanan/ISIATM2012.pdf`.

[31] S. Chaimatanan, D. Delahaye, and M. Mongeau. Strategic deconfliction of aircraft trajectories. In *ISIATM 2013, the 2nd International Conference on Interdisciplinary Science for Innovative Air Traffic Management*, Toulouse (France), 2013. `http://hal-enac.archives-ouvertes.fr/hal-00868450/PDF/isiatm2013/submission/123.pdf`.

[32] S. Constans, B. Fontaine, and R. Fondacci. Minimizing potential conflict quantity with speed control. In *The 4th Eurocontrol Innovative Research Workshop And Exhibition*, pages 265–274, Bretigny-sur-Orge (France), 2005.

[33] D. Delahaye, C. Peyronne, M. Mongeau, and S. Puechmorel. Aircraft conflict resolution by genetic algorithm and B-spline approximation. In *EIWAC 2010, the 2nd ENRI International Workshop on ATM/CNS*, pages 71–78, Tokyo (Japan), 2010.

[34] D. Delahaye and S. Puechmorel. 3D airspace sectoring by evolutionary computation: Real-world applications. In *Proceedings of the 8th annual conference on Genetic and evolutionary computation (GECCO 2006)*, pages 1637–1644, New York, 2006.

[35] D. Delahaye and S. Puechmorel. 3D airspace design by evolutionary computation. In *DASC 2008, the 27th IEEE/AIAA Digital Avionics Systems Conference*, pages 3.B.6–1–3.B.6–13, October 2008.

[36] D. Delahaye and S. Puechmorel. Air traffic complexity based on dynamical systems. In *The 49th IEEE Conference on Decision and Control (CDC)*, pages 2069–2074, 2010.

[37] D. Delahaye and S. Puechmorel. *Modeling and Optimization of Air Traffic*. Wiley-ISTE, 2013.

[38] D. Delahaye, S. Puechmorel, P. Tsiotras, and E. Feron. Mathematical models for aircraft trajectory design: A survey. In *Air Traffic Management and Systems*, pages 205–247. Springer, 2014.

[39] N. Dougui, D. Delahaye, and M. Mongeau. A new method for generating optimal conflict free 4D trajectory. In *The 4th International conference on research in air transportation*, pages 185–191, Budapest (Hungary), 2010.

[40] N. Dougui, D. Delahaye, S. Puechmorel, and M. Mongeau. A light-propagation model for aircraft trajectory planning. *Journal of Global Optimization*, 56(3):873–895, 2013.

[41] G. Dowek and C. Munoz. Conflict detection and resolution for 1,2,...,n aircraft. In *The 7th AIAA Technology, Integration and Operations Conference*, Belfast (Ireland), 2007.

[42] J. Dreo, A. Petrowski, P. Siarry, and E. Taillard. *Metaheuristics for hard optimization*. Springer, 2006.

[43] N. Durand, C. Allignol, and N. Barnier. A ground holding model for aircraft deconfliction. In *DASC 2010, th 29th IEEE/AIAA Digital Avionics Systems Conference*, pages 2.D.3–1–2.D.3–10, October 2010.

[44] N. Durand and J.-M. Alliot. Ant colony optimization for air traffic conflict resolution. In *ATM 2009, the 8th USA/Europe Air Traffic Management Research and Development Seminar*, Napa (California), 2009.

[45] N. Durand and J. B. Gotteland. Genetic algorithms applied to air traffic management. In *Metaheuristics for Hard Optimization*, pages 277–306. Springer, 2006.

[46] M. Ehrgott. *Multicriteria optimization*. Springer, 2005.

[47] E. Elbeltagi, T. Hegazy, and D. Grierson. Comparison among five evolutionary-based optimization algorithms. *Advanced Engineering Informatics*, 19(1):43–53, 2005.

[48] H. Erzberger and R. A. Paielli. Conflict prediction and resolution in the presence of prediction error. In *The 1st USA/Europe Air Traffic Management R&D Seminar*, Saclay (France), June 1997.

[49] D. Gianazza and N. Durand. Separating air traffic flows by allocating 3D-trajectories. In *DASC 2004, the 23rd IEEE/AIAA Digital Avionics Systems Conference*, volume 1, pages 2.D.4–21–13 Vol.1, 2004.

[50] M. Guanglei and Q. Fei. Flight conflict resolution for civil aviation based on ant colony optimization. In *The 5th International Symposium on Computational Intelligence and Design (ISCID)*, volume 1, pages 239–241, October 2012.

[51] R. Hoffman and M. O. Ball. A comparison of formulations for the single-airport ground-holding problem with banking constraints. *Operations Research*, 48(4):578–590, 2000.

[52] I. Hwang and C. E. Seah. Intent-based probabilistic conflict detection for the next generation air transportation system. *Proceedings of the IEEE*, 96(12):2040–2059, 2008.

[53] M. R. Jardin. Real-time conflict-free trajectory optimization. In *ATM 2003, the 5th USA/Europe Air Traffic Management Research and Development Seminar*, Budapest (Hungary), 2003.

[54] M. R. Jardin. *Towards Real Time En Route Air Traffic Control Optimization*. PhD thesis, Standford University, 2003.

[55] A Joulia and C. Le Tallec. Aircraft 4D contract based operation: The 4DCO-GC project. In *28th International Congress of the Aeronautical Sciences*, Brisbane, Australia, 2012.

[56] J. Kennedy and R. C Eberhart. *Swarm Intelligence*. K. Morgan, 2001.

[57] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220:671–680, 1983.

[58] J. Knopman and J.S. Aude. Parallel simulated annealing: An adaptive approach. In *Parallel and Distributed Processing Symposium*, page 522. IEEE International, 1997.

[59] J. K. Kuchar and L. C. Yang. Survey of conflict detection and resolution modeling methods. *AIAA Guidance, Navigation, and Control conference*, pages 1388–1397, August 1997.

[60] J. K. Kuchar and L. C. Yang. Conflict detection and resolution, air traffic control, alerting systems, warning systems. *IEEE Transactions on Intelligent Transportation Systems*, 1(4):179–189, 2000.

[61] J. K. Kuchar and L. C. Yang. A review of conflict detection and resolution modeling methods. *IEEE Transactions on Intelligent Transportation Systems*, 1(4):179–189, 2000.

[62] S. M. LaValle. *Planning algorithms*. Cambridge University Press, 2006.

[63] A. S. Lewis. Robust regularization. Technical report, School of ORIE, Cornell University, Ithaca (New York), 2002.

[64] Z. Li and C. A. Floudas. Robust counterpart optimization: Uncertainty sets, formulations and probabilistic guarantees. In *Proceedings of the 6th conference on Foundations of Computer-Aided Process Operations*, Savannah (Georgia), January 2012.

[65] J. Lohn and J. Rios. A comparison of optimization approaches for nationwide traffic flow management. In *AIAA Guidance, Navigation and Control Conference*, Chicago (Illinois), 2009.

[66] G. Lulli and A. R. Odoni. The European air traffic flow management problem. *Transportation Science*, 41(4):431–443, 2007.

[67] P. M., J. Hu, J. Lygeros, and S. Sastry. A probabilistic approach to aircraft conflict detection. *IEEE Transactions on Intelligent Transportation System*, 1(4):199–220, 2007.

[68] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller. Equation of state calculations by fast computing machines. *Journal of Chemical Physics*, 21:1087–1092, 1953.

[69] Z. Michalewicz. *Genetic Algorithms + Data Structures = Evolution Programs*. Springer-Verlag, London (UK), 3rd edition, 1996.

[70] X. Min. Airspace sector redesign based on Voronoi diagrams. *Journal of Aerospace Computing Information and Communication*, 6:624–634, 2009.

[71] M.S. Nolan. *Fundamentals of Air Traffic Control*. Cengage learning, 5th edition, 2011.

[72] A. R. Odoni. The flow management problem in air traffic control. In *Odoni, A.R., Bianco, L., Szego, G. (eds.), Flow Control of Congested Networks*, pages 269–288. Springer, 1987.

[73] A. R. Odoni. Issues in air traffic flow management. In Heinz Winter and Hans-Gustav Nüßer, editors, *Advanced Technologies for Air Traffic Flow Management*, volume 198 of *Lecture Notes in Control and Information Sciences*, pages 43–63. Springer, 1994.

[74] S. Oussedik. *Application de l'Evolution artificielle aux problèmes de congestion du trafic aérien*. PhD thesis, École Polytechnique, Palaiseau (France), 2000.

[75] S. Oussedik and D. Delahaye. Reduction of air traffic congestion by genetic algorithms. In *Parallel Problem Solving from Nature — PPSN V*, volume 1498 of *Lecture Notes in Computer Science*, pages 855–864. Springer, 1998.

[76] M. Ozgur and A. Cavcar. 0-1 integer programming model for procedural separation of aircraft by ground holding in ATFM. *Aerospace Science and Technology*, 1:1–8, january 2014.

[77] C. H. Papadimitriou and K. Steiglitz. *Combinatorial Optimization: Algorithms and Complexity*. Prentice-Hall, 1982.

[78] T. Prevot, S. Shelden, and J. Mercer. ATM concept integrating trajectory-orientation and airborne separation assistance in the presence of time-based traffic flow management. In *DASC 2003, the 22nd Digital Avionics Systems Conference*, 2003.

[79] O. Richetta. Optimal algorithms and a remarkably efficient heuristic for the ground-holding problem in air traffic control. *Operations Research*, 43(5):pp. 758–770, 1995.

[80] J. Rios and J. Lohn. A comparison of optimization approaches for nationwide traffic flow management. In *AIAA Guidance, Navigation, and Control Conference*, Chicago (Illinois), August 2009.

[81] H.D. Sherali. and J.M. Hill. Configuration of airspace sectors for balancing air traffic controller workload. *Annals of Operations Research*, 203:1–29, 2011.

[82] G. Taguchi. Quality engineering through design optimization. *Kraus International Publications*, 1984.

[83] E.-G. Talbi. *Metaheuristics: From Design to Implementation*. Wiley, 2009.

[84] M. Terrab and A. R. Odoni. Strategic flow management for air traffic control. *Operations Research*, 41(1):138–152, 1993.

[85] J. Tian and H. Xu. Optimizing arrival flight delay scheduling based on simulated annealing algorithm. *Physics Procedia*, 33(0):348 – 353, 2012. 2012 International Conference on Medical Physics and Biomedical Engineering (ICMPBE2012).

[86] H. H. Toebben, C. Le Tallec, A. Joulia, J. Speidel, and C. Edinger. Innovative future air transport system: Simulation of a fully automated ATS. In *26th International Congress of the Aeronautical Sciences*, Anchorage, Alaska, 2008.

[87] V. Černý. Thermodynamical approach to the traveling salesman problem: An efficient simulation algorithm. *Journal of Optimization Theory and Applications*, 45(1):41–51, 1985.

[88] P. B. Vranas, J. Bertsimas, and A. R. Odoni. The multi-airport problem in air traffic control. *Operations Research*, 42(2):249–261, 1994.

[89] L.A. Zadeh. Fuzzy sets. *Information and Control*, 8(3):338 – 353, 1965.

[90] Z.-H. Zhan, J. Zhang, and Y. Li. An efficient ant colony system based on receding horizon control for the aircraft arrival sequencing and scheduling problem. *IEEE Transactions on Intelligent Transportation Systems*, 11:399–412, 2010.