Optimisation du trafic aérien dans de grands aéroports

Ji Ma

THÈSE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par l'Université Toulouse 3 - Paul Sabatier

Présentée et soutenue par

Ji MA

Le 9 juillet 2019

Optimisation du trafic aérien dans de grands aéroports

Ecole doctorale : **EDMITT - Ecole Doctorale Mathématiques, Informatique et Télécommunications de Toulouse**

Spécialité : **Mathématiques et Applications**

Unité de recherche :
Laboratoire de Recherche ENAC

Thèse dirigée par
Daniel DELAHAYE et Mohammed SBIHI

Jury

M. Abdellah SALHI, Rapporteur
Mme Laetitia JOURDAN, Rapporteure
M. Keiki TAKADAMA, Examinateur
Mme Marie-Pierre Gleizes, Examinatrice
M. Daniel DELAHAYE, Directeur de thèse
M. Mohammed SBIHI, Co-directeur de thèse

Résumé

La croissance du trafic aérien engendre des congestions et des retards des vols, tant dans les aéroports que dans les espaces aériens environnants. En fait, les aéroports sont limités en termes de capacité et représentent les principaux goulots d'étranglement du système de gestion du trafic aérien. Une planification et un contrôle efficaces sont essentiels pour améliorer l'efficacité des opérations aéroportuaires et réduire les retards des vols. Dans des recherches antérieures, plusieurs sous-problèmes liés aux opérations aéroportuaires ont déjà été examinés séparément, tels que le séquencement de pistes, les mouvements au sol, la gestion de l'espace aérien terminal, dite Terminal Maneuvering Area (TMA), etc. Cependant, toutes ces opérations sont étroitement liées et peuvent dépendre les unes des autres. Ceci motive le développement d'approches d'optimisation intégrées pour la gestion du trafic aérien dans les aéroports et dans l'espace aérien environnant. Dans cette thèse, nous proposons une approche d'optimisation à deux niveaux désignés par niveau macroscopique et niveau microscopique. En effet, suivant les horizons de prédiction des différents problèmes, nous considérons d'abord un horizon à long terme avec un réseau abstrait de l'aéroport et de la TMA. Ensuite, nous aborderons le problème à un horizon plus court en considérant un réseau détaillé de l'aéroport.

Dans la première partie de la thèse, nous nous sommes concentrés sur l'optimisation intégrée du problème d'opérations de l'aéroport et du problème de gestion de l'espace aérien terminal à niveau macroscopique. L'aéroport est modélisé comme un réseau abstrait: le terminal, le réseau de taxis et la piste sont considérés comme des ressources spécifiques ayant une capacité maximale définie, et la TMA est modélisée à l'aide d'une structure de réseau d'itinéraires prédéfinis. Ce niveau d'abstraction vise à identifier les situations de congestion aéroportuaire. Nous développons un modèle d'optimisation pour minimiser les retards de vol, résoudre les conflits d'espaces aériens et diminuer les congestions aéroportuaires en contrôlant la vitesse d'arrivée, les heures d'arrivée et de départ et la piste attribuée, tout en respectant diverses contraintes opérationnelles. Une metaheuristique de recuit simulé adaptée, combinée à une approche de décomposition temporelle, est proposée pour résoudre le problème correspondant. Des simulations menées sur des données issues de l'aéroport Paris Charles De-Gaulle montrent des améliorations potentielles en termes de réduction de la congestion et des retards de vol.

La deuxième partie de la thèse porte sur le problème du séquencement de pistes d'aéroport et des mouvements au sol à un niveau microscopique. Dans cette partie, nous représentons l'aéroport (les portes, le réseau de taxi, les pistes) par un graphe détaillé et nous considérons les routes de chaque avion sur la base de ce réseau. Le but est de résoudre tous les conflits entre avions lors du roulage, en leur attribuant des heures de push-back, des vitesses de roulage ainsi que des positions (seuil de piste ou point de croisement) et des temps d'attente. Le modèle d'optimisation est conçu pour réduire la file d'attente des pistes et les retards de vol, ainsi que les temps de roulage tout en considérant les problèmes de sécurité liés aux opérations sur la surface. Une comparaison par rapport aux scénarios de référence indique que l'optimisation donne de bons résultats pour deux grands aéroports: l'aéroport Paris Charles De-Gaulle (CDG) et l'aéroport Charlotte Douglas International

(CLT). Des gains importants en termes de temps de roulage et de réduction de la file d'attente de décollage sont obtenus particulièrement à CLT, qui est plus enclin à la congestion.

La dernière partie de la thèse s'intéresse à un problème se posant à un horizon court terme. Ce problème consiste à séquencer les départs tout en tenant compte des traversées de piste par des arrivées. Dans de nombreux hubs dotés de pistes parallèles, les arrivées doivent traverser la piste de départ pour atteindre le taxiway. Un meilleur séquencement des départs tenant compte des points de passage des arrivées pourrait permettre de réduire les retards des vols. Les contraintes de séparations de piste, les créneaux de vol et la capacité de la file d'attente au seuil de piste sont explicitement prises en compte. Nous présentons deux modèles de programmation linéaire en nombres entiers et un algorithme de recuit simulé. Des études de cas sont conduites pour le doublet de pistes sud à CDG afin de réduire les retards des vols et d'avoir une séquence optimale. De plus, la qualité de la solution et le temps de calcul des différentes approches sont comparés. Les trois méthodes proposées pourraient considérablement améliorer les solutions basées sur la simple règle du premier arrivé, premier servi, et le recuit simulé est le plus efficace pour ce type d'opération.

Abstract

The air traffic growth induces congestion and flight delays both at the airports and in the surrounding airspaces. In fact, the airports are limited in terms of capacity and represent the major bottlenecks in the air traffic management system. Efficient planning and control are critical to enhance the airport operation efficiency and to reduce flight delays. In prior research, several sub-problems associated with airport operations have already been discussed separately, such as runway scheduling, taxiway scheduling, terminal airspace management, etc. However, these operations are closely related and can affect each other. This motivates the development of an integrated optimization approach for managing air traffic at airport and in the surrounding airspace. In this thesis, we suggest a two-level optimization approach which works on both the macroscopic and the microscopic levels. Following the prediction horizon of different problems, we consider first a long term horizon with an abstract network of airport and TMA. Then, we consider a shorter horizon with a detailed network of airport components.

In the first part of the thesis, we focus on the integrated optimization of airport operation problem and terminal airspace management problem at a macroscopic level. The airside is modeled as an abstract network: terminal, taxi network, and runway are seen as specific resources with a defined maximum capacity, and the TMA is modeled by a predefined route network structure. This level of abstraction aims at identifying the airport congestion situations. We develop an optimization model to minimize flight delays, resolve airspace conflicts, and mitigate airport congestions by controlling speed, arrival and departure times, and assigned runway, while keeping various operational constraints. An adapted simulated annealing (SA) metaheuristic combined with a time decomposition approach is proposed to solve the corresponding problem. Computational experiments performed on case studies of Paris Charles De-Gaulle airport show potential improvements on airport congestion mitigation and flight delay reduction.

The second part of the thesis deals with the airport runway and taxiway scheduling problem at a microscopic level. In this part, we represent the airport (gate, taxiway, runway) with a detailed surface node-link network, and we consider individual aircraft trajectories based on this graph. We aim at resolving the ground conflicts among aircraft, assigning the pushback times, the taxi speeds and the positions (runway threshold or holding point) and the holding times. The optimization model is designed to reduce runway queue length and minimize flight delays as well as taxi times with respect to safety concerns in surface traffic operations. A comparison with regard to baseline scenarios of the microscopic optimization benefits is presented for two major airports: Paris Charles De-Gaulle (CDG) airport and Charlotte Douglas International airport (CLT). Important gain in taxi time savings and runway queue length reduction are achieved, particularly at CLT since it is more prone to congestion.

The last part of the thesis focuses on a sub-problem of the microscopic level. It consists in sequencing departures flights incorporating arrival crossings. In many hub airports with parallel runways, arrivals have to cross departure runway to reach the taxiway. A better sequence of departures taking into account arrival crossings could achieve less flight delay. Constraints for minimum

runway separations, flight time window restrictions, and holding queue capacity at runway threshold are explicitly considered. We present two Integer Linear Programming (ILP) models and a SA algorithm. Case studies are conducted for the Southern pair of runways at CDG to reduce flight delays and obtain optimal sequence. Moreover, the solution quality and the computation time of different approaches are compared. The three proposed methods could significantly improve the solutions based on the simple first-come-first-served rule, and the SA is more suitable for implementation.

Acknowledgements

First and foremost, I would like to thank my advisor Daniel Delahaye. From his first TP class during my master study until the end of this thesis, his professional expertise, knowledge and continuous support guided me all the way. I wish to thank him for always believing in my abilities; for several trips to conduct joint research; for discovering his universe of mountain tracking.

I wish equally to thank my co-advisor, Mohammed Sbihi, who taught me vast mathematical knowledge; who is always rigorous, patient, and considerate; who was all the time present whenever I need help in either research part or life part; for his sense of humor; for sharing cherries from his backyard every summer period.

I would like to thank Professor Hamsa Balakrishnan who accommodated me at MIT as a visiting student; who discussed with me every Friday afternoon and guided me in the right direction, and elaborated a joint paper with us.

Many thanks to my jury members for their interest in my work. Thanks to the reporters, Professor Laetitia Jourdan and Professor Abdellah Salhi, for their deep knowledge in optimization theory and insights and expertises on algorithms that I was not aware of. Thanks to the examiners, Professor Pierre-Marie Gleizes and Professor Keiki Takadama, for all the precious remarks that inspired me to improve my future work.

My grateful thanks goes to my university, Civil Aviation University of China (CAUC), and my institute, Sino-European Institute of Aviation Engineering (SIAE) for choosing me to study in France and supporting me financially. Thanks for giving me the opportunity to experience a totally different and amazing culture.

I wish to thank other professors and members in the Z-building: Marcel Mongeau, Sonia Cafieri, Catherine Mancel, Stéphane Puechmorel, Andrija Vidosavljevic, Gilles Baroin, Hélène Weiss, Andrea Hakala, Jean-Baptiste Gotteland, Mathieu Cousy, Georges Mykoniatis.

Many thanks to my labmates in the Optim Group in ENAC for the enjoyable lunch time every day, coffee breaks, card games, picnics and restaurants: Serge Roux, Jun Zhou, Tambet Treimuth, our small funny group; Ying Huo, Xiao Hu, Ruohao Zhang, Shangrong Chen, best Carrefour and KFC partners; Romaric Breil, Vincent Courjault-Radé, Florian Mitjana, Les Trois Mousquetaires; Imen Dhief, Sana Rebbah, Sana Ikli, Man Liang, super girls; Alexandre Tran, Ruixin Wang, Philippe Monmousseau, Xiao Liang, Rui Zhao, Yi Lv, Mingze Sun, Isabelle Santos, Ahmed Khassiba, Olga Rodionova, Mahfoud, Hasan, Evgenii, Souleyman, Ramon, Jérémie, Gabriel, Paveen (whom I apologize to do not mention).

My sincere thanks also goes to friends outside of France: Paolo Scala, Miguel Mujica Mota, Sandeep Badrinath, Ryota Mori, Adriana Andreeva-Mori, Daichi Toratani, Supatcha Chaimatanan, Emmanuel Sunil, Arianit Islami, Daniel González.

Finally, I would like to thank my parents and my boyfriend for their unconditional love and support.

Contents

List of Figures

List of Tables

Abbreviations

ACC	**A**rea **C**ontrol **C**enter
A-CDM	**A**irport-**C**ollaborative **D**ecision **M**aking
AGM	**A**irport **G**round **M**ovement
ALP	**A**ircraft **L**anding **P**roblem
AMAN	**A**rrival **Man**agement
APP	**App**roach Control
ASP	**A**ircraft **S**equencing **P**roblem
ATC	**A**ir **T**raffic **C**ontrol
ATFM	**A**ir **T**raffic **F**low **M**anagement
ATM	**A**ir **T**raffic **M**anagement
ATP	**A**ircraft **T**ake-off **P**roblem
ATS	**A**ir **T**raffic **S**ervice
BADA	**B**ase of **A**ircraft **DA**ta
B&B	**B**ranch and **B**ound
CDA	**C**ontinuous **D**escent **A**pproach
CDG	Charles De-Gaulle Airport
CLT	Charlotte Douglas International Airport
CPS	**C**onstrained **P**osition **S**hift
CTAS	**C**enter-**T**RACON **A**utomation **S**ystem
CTOT	**C**alculated **T**ake-**O**ff **T**ime
CTR	Control zone
DEP	**Dep**arture position
DMAN	**D**eparture **Man**agement
DP	**D**ynamic **P**rogramming
EOBT	**E**stimated **O**ff-**B**lock **T**ime
FAA	**F**ederal **A**viation **A**dministration
FCFS	**F**irst **C**ome **F**irst **S**erved
FIR	**F**light **I**nformation **R**egion
GA	**G**enetic **A**lgorithm
GND	Ground position
IAF	**I**nitial **A**pproach **F**ix
IATA	**I**nternational **A**ir **T**ransport **A**ssociation
ICAO	**I**nternational **C**ivil **A**viation **O**rganization

IF	**I**ntermediate **F**ix
ILS	**I**nstrument **L**anding **S**ystem
IMC	**I**nstrumental **M**eteorological **C**onditions
IP	**I**nteger **P**rogramming
MILP	**M**ixed **I**nteger **L**inear **P**rogramming
NextGen	**Next Gen**eration Air Transportation System
NM	**N**autical **M**ile
PMS	**P**oint **M**erge **S**ystem
RHC	**R**eceding **H**orizon **C**ontrol
RTA	**R**equired **T**ime of **A**rrival
SA	**S**imulated **A**nnealing
SESAR	**S**ingle **E**uropean **S**ky **A**TM **R**esearch
SID	**S**tandard **I**nstrument **D**eparture
STAR	**S**tandard **T**erminal **A**rrival **R**oute
TAM	**T**otal **A**irport **M**anagement
TMA	**T**erminal **M**aneuvering **A**rea
TRACON	**T**erminal **R**adar **A**pproach **Con**trol
TSAT	**T**arget **S**tartup **A**pproval **T**ime
TWR	Tower control

Chapter 1

Introduction

Air transport is a major form of transport throughout the world. In 2018, the total number of passengers traveling by air is expected to be about 4.3 billion, which represents a growth of 4.9% compared to the previous year [1]. The global air traffic will grow at 4.4% rate annually for the next 20 years according to the 2018 edition of Airbus' global market forecast [4]. The order of 5 percent a year worldwide implies a doubling of traffic about every 15 years [5]. Subsequently, the planned increase in airport capacity is not sufficient to meet the traffic demand. As shown in Fig. 1.1, 39 out of the 47 aviation mega cities [2] are approaching full capacity and are already largely congested. Moreover, by 2040, there will be 1.5 million flights more in demand (or 160 million passengers) that can be accommodated in Europe according to [6]. Increasing congestion causes significant delays and operational costs. The average delay per flight in Europe will increase from 12 minutes in 2016 to 20 minutes in 2040 [6].

The continuous increase in air traffic demand and the saturating capacity of the network become a main task for air transport system. Airports and surrounding airspaces are limited in terms of capacity and represent the major bottleneck in the air traffic management system. To mitigate airport

Figure 1.1: Aviation mega cities are schedule-constrained in [1].

[1] Source: IATA (http://airlines.iata.org/news/iata-reveals-2018-financial-forecast)

[2] Aviation Mega City is defined as a city with more than 10,000 daily international passengers who fly more than 2000 nautical miles, according to Airbus.

congestion and increase the efficiency of air transport, one feasible solution may be the construction of new runways and deployment of modern technology. However, it requires long-term planning and implementation, and sometimes it can be limited by geological, economic, environmental, and political factors. Alternatively, efficient planning and use of available resources can be a short-term solution to improve air traffic operations. Therefore, this thesis addresses techniques to optimize airport and surrounding airspace operations.

In this chapter, we first present an overview of the current air traffic management system. Then, we introduce the airport components and the operational control processes. Lastly, objectives, contributions and structure of this thesis are presented.

1.1 Air traffic management system

Air traffic management (ATM) is a system that covers all the activities involved in ensuring the safe and orderly flow of air traffic. It comprises three main services:

- **Airspace management (ASM)** is a service to manage airspace as efficiently as possible in order to satisfy its many users. It concerns both the way airspace is allocated to its various users and the way it is structured in order to provide air traffic control services;
- **Air traffic flow management (ATFM)** is a process that balances air transport network user demands against system capabilities. It regulates the flow of air traffic efficiently in order to decrease or avoid the congestion of control sectors;
- **Air traffic control (ATC)** is a service to ensure aircraft separation and to prevent collisions between aircraft and obstructions in the maneuvering area. It is further divided into three subservices: Area Control Center (ACC) service, Approach Control (APP) service, and Tower Control (TWR) service, according to different flight phases;

Depending on the time horizon, the ATM process is performed in three levels:

- **Strategic level planning.** This phase takes place seven days or more before the day of operations. Capacities that need to be provided in each of air traffic control centers are predicted. This phase also includes avoiding imbalances between capacity and demand for events taking place a week or more in the future;
- **Pre-tactical planning.** This phase takes place one day to six days before the day of operations. The daily plan, the initial network plan, and the measures that will be in force in airspace will be coordinated and known with higher accuracy;
- **Tactical planning.** This phase takes place the day of operations. The daily plan is updated based on the current situation, and aircraft are regulated according to real time traffic demand in order to ensure safety, maximize flight efficiency, and make full use of the available capacity.

The world's major ATM systems are being modernized. The Next Generation of Air Transportation System (NextGen) [7] in the United States and the Single European Sky ATM Research (SESAR) [8] system in Europe are carried out to provide solutions to all actors to improve air safety, reduce air traffic delays, and increase efficiency of air traffic. The research areas of ATM include: high-performing airport operations, advanced air traffic services, optimized ATM network services, and enabling aviation infrastructure etc.

With existing or new technologies, Airport-Collaborative Decision Making (A-CDM) [9] conducted by Eurocontrol is embedded in the ATM operational concept that will improve operational efficiency, predictability and punctuality to the ATM. Based on more accurate and higher quality information, data is shared by all partners network and airport stakeholders to support ATM decisions. Along with the increased predictability, enhanced awareness and more accurate flight information can lead to better quality of subsequent decisions, thus provides more stable traffic flows, more precise calculation of network demand, and less congestion on the ground. A-CDM is fully implemented in 28 airports across Europe. Moreover, Total Airport Management (TAM) [10] concept integrates A-CDM into one holistic architecture to monitor and guide airside and landside operations. It extends the tactical time horizon of A-CDM to pre-tactical and strategic phases. Operational decisions taken by the airport operator or ATC may be made in the full knowledge of airline operational constraints and/or priorities. For practical issues, airport controllers handle the different airport management problems separately and independently with several divisions of responsibility. However, decisions from previous controllers may have impact on next controllers, thus generating a low level of coordination. With optimization techniques, global optimal decisions can be made to solve airport management problems as one and to assist the controllers. These concepts of shared flight information for all stakeholders make it possible to apply decision support tools to optimize the airport operations and consider the integrated optimization of sub-problems in the airport operations. Therefore, the main objective of this thesis is to consider the problem of optimization of air traffic in major airports as a whole. More specifically, optimization will be carried out from the entry in the area near the airport (Terminal Maneuvering Area – TMA) until the exit of this TMA, including the arrival, ground movement, and departure flight phases.

1.2 Components of airport

Airports are divided into *airside* and *landside* areas. Airside refers to the movement area of an airport, adjacent terrain and buildings or portions thereof, access to which is controlled. Airside elements include runways, taxiways, and gates. Landside refers to the area of an airport and buildings to which the non-traveling public has free access.

During different flight phases, aircraft use various components of an airport:

- **Runway:** Runway is a paved land strip on which landing and take-off operations of aircraft take place. Runway are labeled according to the direction they are facing. Therefore, an aircraft using runway 09 would be facing east (90°) while runway 27 (270°) would be facing west. If more than one runway points in the same direction, each runway is identified by appending left (L), center (C) or right (R) to the number. Runway configurations refers to the number and relative orientations of one or more runways on an airfield. Only one runway configuration is active at each time with regard to wind direction.

- **Taxiway:** Taxiway is a path which connects the runway with other areas of an airport (parking areas, hangars, gates, cargo terminals, or other operations areas). The taxiways should permit safe and fluent movements of aircraft to minimize the taxi time and also the fuel consumption.

- **Apron:** Apron is an area which is used as parking slots for aircraft. It can be classified, according to the type of aircraft stands they hold, into passenger building aprons, cargo building aprons, long-term parking aprons, service and hangar aprons, and general aviation aprons [5].

- **Control tower:** Airport traffic control towers are established to provide a safe, orderly and expeditious flow of traffic in the vicinity of an airport. The controller from the control tower

observes all the aircraft through radars and radio communication, and provide pilots informations for landing, take-off and taxiing.

1.3 Airport and TMA control processes

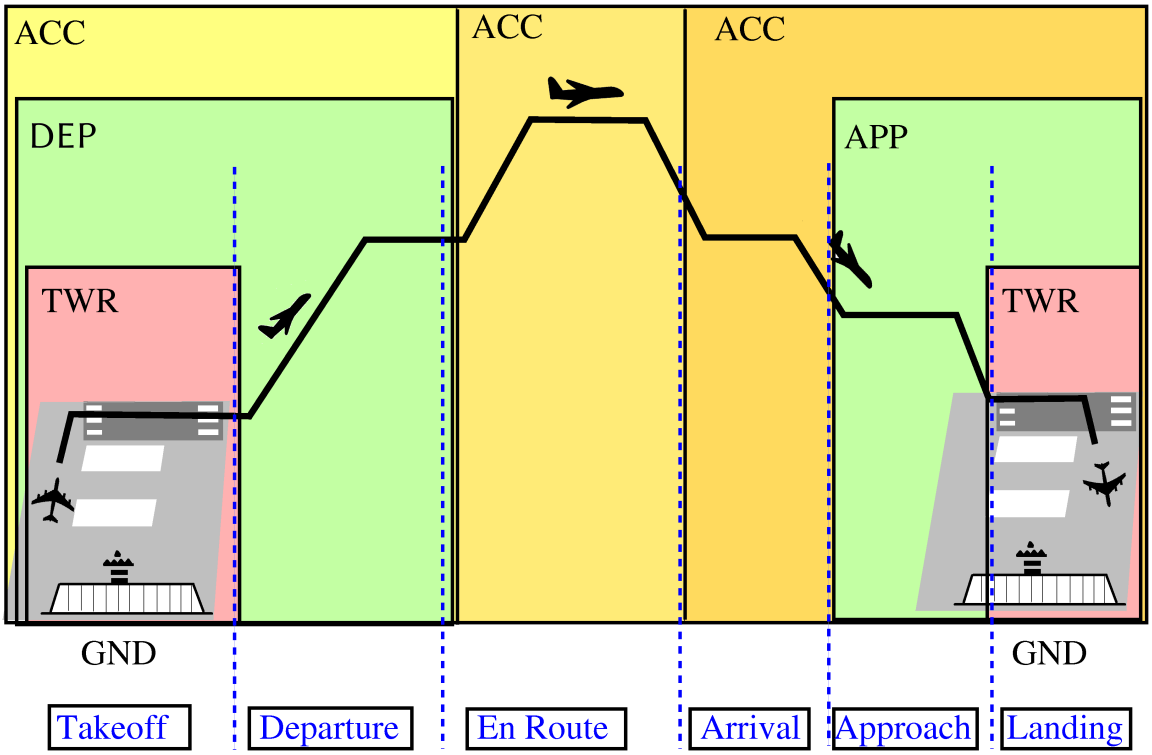


Figure 1.2: Controlled airspaces in different flight phases.

In this section, we first introduce the controlled airspaces in different flight phases. Then, we give a detailed description about the related operational processes considered in this thesis.

1.3.1 Controlled airspaces

As illustrated in Fig. 1.2, there are six flight phases: Take-off, Departure, En Route, Arrival, Approach, and Landing. Different flight phases are operated by different ATC positions. The *ground control* (GND) handles aircraft on the ground (including taxiways and aprons). The *tower control* (TWR) handles aircraft on the active runway(s) and airborne aircraft that are visual with the runway within the airspace around the aerodrome. The *departure control* (DEP) handles departing aircraft after they are airborne and handed off by the TWR until they can be transferred to the next ATC position. The *approach control* (APP) normally handles arriving aircraft near the Initial Approach Fix (IAF) [3] towards the final approach. The approach and departure responsibility airspace are called the TMA. The *area control center* (ACC) normally handles the aircraft while flying en route or climbing or descending from or into an airfield situated in its Flight Information Region (FIR). A FIR is a wide area of airspace in which countries are responsible for the provision of the Air Traffic Services (ATS) [11].

[3] IAF is a fix that marks the beginning of the initial segment and the end of the arrival segment, if applicable.

1.3.2 Arrival and departure processes

This thesis focuses on the airport and terminal airspace operations. In the following section, we will give more detailed descriptions about arrival, ground movement, and departure processes and the different air traffic control positions.

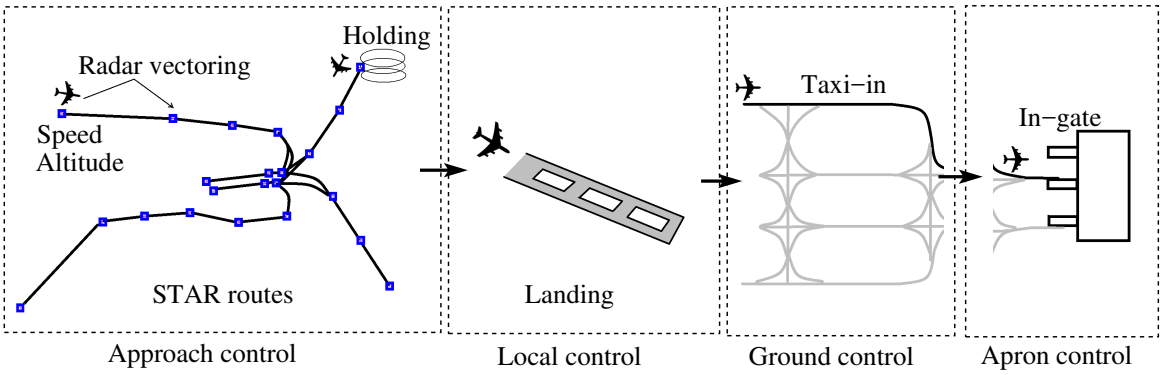


Figure 1.3: Aircraft arrival processes.

Before entering the TMA, the flight should be transferred by the adjacent controller to the *approach controller* when descending to an altitude coordinated among the controllers [12]. The approach controllers use the Standard Terminal Arrival Routes (STAR) that connect the TMA entry points to the runways to handle all the arrival aircraft by managing traffic trajectories and descent paths. They need to merge and sequence aircraft by: first, ensuring traffic safety at any time, i.e. guarantee a minimal vertical separation of 1000 ft or a minimal horizontal separation of 3 NM in the TMA area; secondly, creating an orderly flow of air traffic towards the landing runway. To create the approach sequence while ensuring the safety requirements, the approach controller can provide radar vectoring, assign altitude and speeds, issue a holding clearance over the IAF to handle the traffic. Once establishing the final approach, the flights are transferred to the *local controller* (or tower controller). Local controller is responsible for the runway operations in order to ensure the minimum runway separations caused by wake vortex at all times. If any unsafe conditions are detected, a landing aircraft may be instructed to "go-around" and be re-sequenced into the landing pattern. After touching down on the runway, the aircraft decelerates and exits the runway. In some cases, aircraft may have to wait before crossing an active runway in order to reach the taxiway. Then, the aircraft contacts *ground controller* for taxi-in instructions. The ground controller handles all intermediary taxiing routes to avoid conflicting movements of aircraft, and gives the pilot instructions on reaching the ramp area via the appropriate taxiways. Finally, the pilot contacts the *apron controller* (or ramp controller) who dictates aircraft to move into the designated gate or to wait if the arrival gate is occupied. These processes are summarized in Fig. 1.3.

Departures follow similar processes in the reverse direction. The apron controller dictates aircraft push back time from the gate and gives clearance for taxi up to a spot near the gate. He also coordinates with airline operators in implementing ground delays if necessary. Then, the pilot contacts the ground controller for taxi clearance. The ground controller handles all intermediary taxiing routes to avoid conflicting movements of aircraft, and gives the pilot instructions on reaching the runway holding area via the appropriate taxiways. Next, the pilot contacts the local controller who clears aircraft for take-off ensuring the prescribed runway separation. Then, the departure controller handles departing flights transferred by the local controller following Standard Instrument Departure (SID) procedures that connect runways to the TMA exit points. He is responsible for initial climb up

to the en-route phase and separating departures from arrivals by means of assigning altitudes, climb rates and speeds [13]. These processes are illustrated in Fig. 1.4.

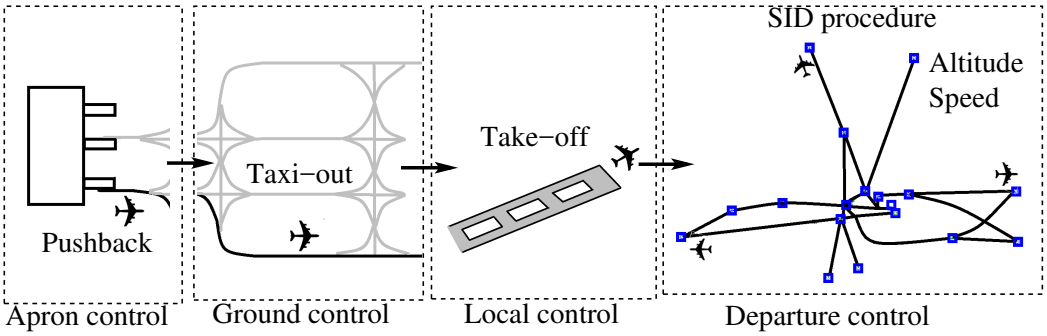
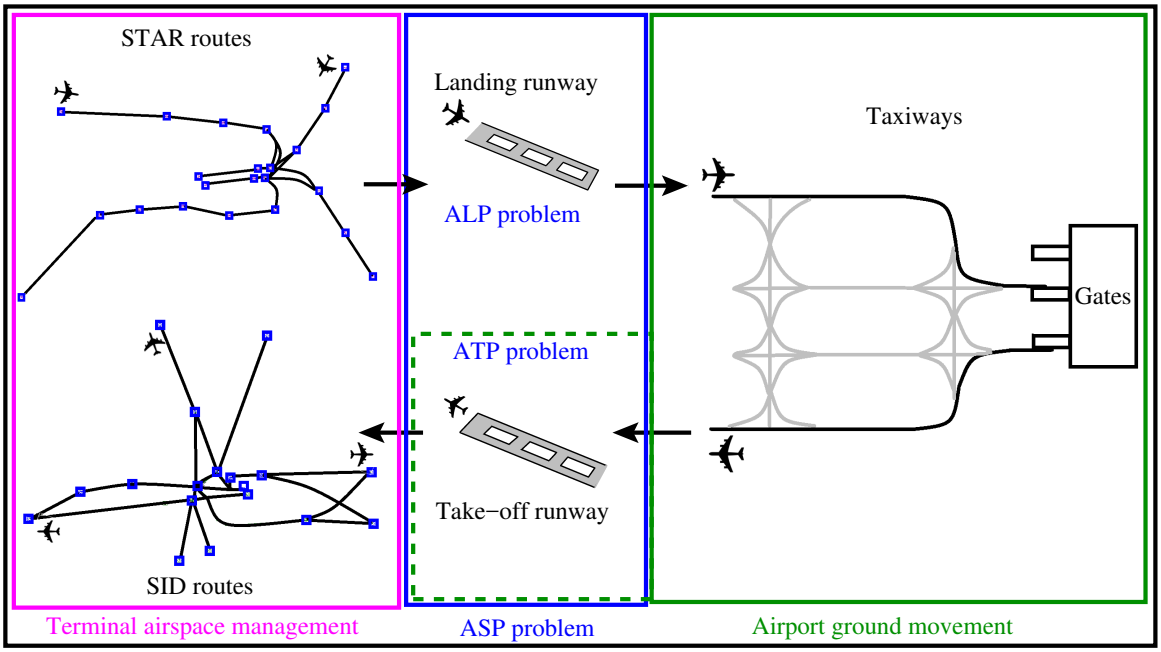


Figure 1.4: Aircraft departure processes.

With many emerging operational concepts and procedures for airport, optimization and decision support tools have been designed to aid controllers to manage the air traffic more efficiently without loss of safety. The coordination of air traffic operations at airport and in the TMA can be possible by using various optimization approaches in different control points, and they can be proposed as potential operational concepts for improving airport operation efficiency and mitigating airport congestions.

1.4 Contributions



Integrated optimization of airport and TMA

Figure 1.5: Air traffic optimization problems in the airport and the surrounded TMA.

Airports as well as the control sectors are limited in terms of capacity and represent the major

bottlenecks in the air traffic management system. As illustrated in Fig. 1.5, several sub-problems related to airport operations have already been discussed in the literature: the terminal airspace management problem which considers the arrival and departure scheduling in the airspace with respect to separation between aircraft and interactions between different traffic flows; the landing/take-off sequencing problem which consists in optimizing the runway usage time of aircraft while ensuring safety issues; the airport ground movement problem which searches for the best taxi path and schedules in order to ensure safe and fluent air traffic operations. The main objective of this thesis is to consider the problem of optimization of air traffic in major airports as a whole. More specifically, optimization will be carried out from the entry in the area near the airport (Terminal Maneuvering Area – TMA) until the exit of this TMA, deciding for each aircraft the following decision variables: arrival time in the TMA, arrival speed, landing time, landing runway, arrival taxi schedule, departure time of the next flight (for the same aircraft), departure taxi schedule, take-off runway.

Due to the complexity of the above-mentioned problem and the different levels of abstraction, we develop a two-level approach to optimizing the air traffic at airport and in its surrounding TMA. The airport models are described by macroscopic level and microscopic level. The level of abstraction for decisions is adapted to the temporal horizon. At the macroscopic level (one hour before landing), the airport components (terminals, taxi network) can be globally modeled as resources with specific capacities (as opposed to individual aircraft or taxiway links). This level of abstraction aims at identifying airport congestion situations. At the microscopic level (a few minutes before landing), we consider individual aircraft trajectories based on an airport surface node-link network model including gates, taxiways, and runways. Airport is seen with more detailed and microscopic decisions like taxi routing and scheduling. Such decisions cannot be taken at the macroscopic level (one hour in advance) due to the remaining uncertainties.

- **Integrated optimization of TMA and airport at the macroscopic level**

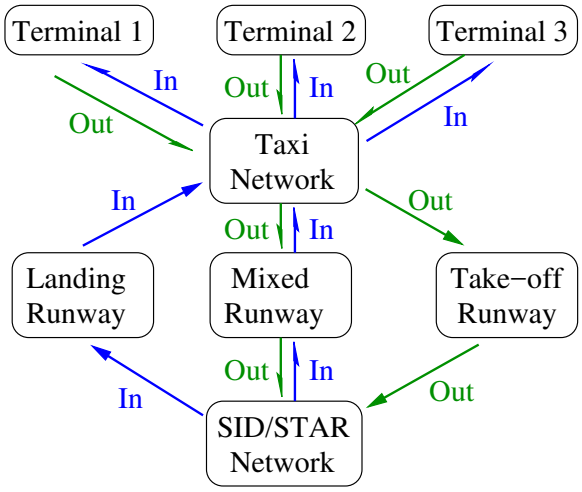


Figure 1.6: Abstract airport network model at the macroscopic level.

The first main contribution of this thesis is to propose an approach for managing congestion at airport and in the surrounding airspace at the macroscopic level. This optimization model is designed to minimize flight delays, resolve airspace conflicts, and mitigate airport congestions by controlling speed, arrival and departure times, and assigned runway, while satisfying various operational constraints. The airspace is modeled using a predefined SID/STAR route structure. The airside is modeled as an abstract network: terminal, taxi network, and

runway were seen as specific resources with a defined maximum capacity, as shown in Fig. 1.6. An adapted simulated annealing metaheuristic method combined with a time decomposition approach is proposed to solve the corresponding problem. Computational experiments performed on case studies of Paris Charles De-Gaulle airport show some potential improvements: First, when the capacity of a certain resource in the airport (terminal, taxi network) is decreased, until a certain threshold, the overload could be mitigated properly by adjusting the aircraft entry time in the TMA and the pushback time. Second, landing and take-off runway assignments in peak hours with imbalanced runway load could reduce flight delays.

- **Airport ground movement and departure runway sequencing at the microscopic level**

  The second main contribution of this thesis to propose an approach for surface operations and runway sequencing problem at the microscopic level. In this part, we represent the airport (gate, taxiway, runway) with a detailed surface node-link network, and we consider individual aircraft trajectories based on this graph. We aim at resolving the ground conflicts among aircraft, assigning the pushback times, the taxi speeds and the positions (runway threshold or holding point) and the holding times. The optimization model is designed to reduce runway queue length and minimize flight delays as well as taxi times with respect to safety concerns in surface traffic operations. A comparison with baseline scenarios of the microscopic optimization benefits is presented for two major airports: Paris Charles De-Gaulle airport (CDG) in Europe and Charlotte Douglas International airport (CLT) in the US. We describe the airport surface operations with an emphasis on their similarities and differences. The two airports have different surface infrastructure characteristics such as holding area, runway configuration, taxiway layout, etc. The two airports handle approximately the same number of aircraft movements. However, the fleet mix at the two airports are significantly different. At CLT, there are lots of interactions between departures and arrivals on the ramp area, we obtain significant taxi-out as well as taxi-in time savings through optimization by gate-holding strategy, the reduction in taxi-in time arise because of better sequencing of runway crossings. Departure runway queue length is controlled and decreased without under-utilizing the runway. Lower benefits are observed at CDG since the airport is relatively less congested.

- **Exact and heuristic algorithms for scheduling aircraft departures**

  In the last part of the thesis, we investigate a sub-problem from the microscopic level, i.e. scheduling aircraft departures incorporating arrival crossings. In many hub airports with parallel runways, arrivals have to cross departure runway to reach the taxiway. A better sequence of departures taking into account arrival crossings could achieve less flight delay. Exact and heuristic algorithms are proposed and compared. First, we present two ILP formulations that model the holding area and the queue for both departures at the runway threshold and arrivals at the holding point. The two ILP models mainly differ in the way the decision variables are defined, one uses time slot based formulation, and the other uses delay-indexed formulation. Then, we apply the SA algorithm and compare the gap of total delay between optimality and heuristic solution and the computation time. Comparison tests are conducted for the Southern pair of runways at CDG and show that the three proposed methods could significantly improve the solutions based on the simple first-come-first-served rule. The delay-indexed formulation is suitable for finding an optimal solution while the time slot based formulation is suitable for finding a feasible solution within a short computation time. When the runway is in high demand, SA is suitable for finding a near-optimal solution in a short computational time that could be used for real-time deployment.

1.5 Thesis structure

The organization of this thesis is as follows. Chapter 2 reviews existing problems in the literature related to airport and surrounded TMA operations. Several sub-problems are discussed in detail, such as the aircraft sequencing problem, aircraft ground movement problem, terminal airspace management problem, etc. Furthermore, the existing integrated works that combine the previously mentioned sub-problems are presented. In Chapter 3, we describe a mathematical framework to deal with the integrated optimization problem of airport and TMA at a macroscopic level. Airport components are considered as specific resources with a certain capacity. The SA algorithm combined with a time decomposition approach is applied to solve such a problem. Computational experiments conducted with the proposed methodology are presented. The preliminary research on merging flows in the TMA by applying the time decomposition approach has been presented at ICRAT [4] conference in 2016 [14]. The results of the integrated optimization problem have been presented at SID [5] conference in 2016 [15], and the analysis of the airport congestion mitigation and the runway assignment have been published in [16]. In Chapter 4, we present a methodology to address the problem of airport ground operations at a microscopic level. As opposed to the macroscopic level, detailed aircraft trajectories based on a node-link network are considered. An optimization approach is proposed to solve the coordinated surface operations problem and runway sequencing problem on network models of hub airport. The preliminary results of the microscopic model have been presented at EIWAC [6] conference in 2017 [17]. A comparative analysis of the proposed approach on two major airports are presented and discussed. In Chapter 5, we compare two exact methods with the previous-mentioned SA algorithm on the problem of departure runway scheduling incorporating arrival crossings. The solution quality as well as computation time are presented based on different traffic scenarios. The results of this chapter have been published in [18]. Finally, we draw conclusions from this thesis and discuss future directions.

[4] ICRAT2016, the 7th International Conference on Research in Air Transportation, Philadelphia, United States

[5] SID2016, the 6th SESAR Innovation Days, Delft, The Netherlands

[6] EIWAC2017, the 5th ENRI international workshop on ATM/CNS, Tokyo, Japan

Chapter 2

Literature review

In this chapter, the research works related to airport and surrounding TMA traffic optimizations are investigated. First, Section 2.1 discusses about the aircraft runway scheduling problem. The aircraft landing problem and the aircraft take-off problem are explained and compared. Section 2.2 presents research on the airport ground movement problem. Then, Section 2.3 addresses the terminal airspace management problem. The integrated air traffic optimization works of TMA and airport are presented in Section 2.4. After that, Section 2.5 gives a short review of optimization problems and resolution methods, with an emphasis on the methods applied in this thesis. Finally, Section 2.6 gives some conclusions.

2.1 Aircraft runway sequencing

The runway is the main bottleneck of airport system, as it is shared by all types of operations (arrivals, departures and crossings) as indicated by field observations [19]. Runway can be used in segregated mode (only landings or only take-offs) or in mixed mode (both landings and take-offs). The mixed mode can achieve higher runway throughput since separation between arrival and departure is shorter, but it is harder for the controllers to manage it. Nevertheless, segregated mode is more often used due to airport layout. Efficient runway operation planning and control are critical to maximize runway throughput and reduce flight delays while satisfying all the system constraints.

One important aspect of runway operations is the aircraft sequencing. It consists in optimizing the runway usage time while ensuring safety by taking into account several operational constraints:

- **Minimum wake turbulence separation** between successive aircraft: The minimum separation requirement is a key factor for deciding the runway capacity. The International Civil Aviation Organization (ICAO) divides aircraft into different wake turbulence weight categories based on the maximum certificated take-off mass. Due to wake vortex, an aircraft is perturbed by the previous one and a minimum separation must be respected to ensure safety. The separation time is dependent on the landing categories of aircraft. For example, a heavy aircraft followed by a light aircraft generates longer separation than a light aircraft followed by a heavy aircraft;

- **Time window constraints** of runway usage for individual flights: the earliest and latest time of arrival limited by aircraft speed, fuel, and controller actions;

- **Precedence constraints:** are pairwise requirements on aircraft that one aircraft must land before another due to priority flights, same jet route followed, etc.

- **Holding point restrictions:** depending on airport layouts, arrivals or departures may wait before using the runway; the queue length of the holding point is restricted, and the order in the holding area should be kept the same.

In general, the first-come-first-served (FCFS) order is the most common technique that controllers use to sequence aircraft. Although this approach is fair enough to maintain the equity of scheduling, FCFS does not consider most useful criterion to alleviate congestion and does not make efficient use of the airport capacity due to its large spacing requirement. Therefore, more efforts have been made to take into account these criteria.

Aircraft Sequencing Problem (ASP) has been studied intensively in the past decades. Bennell *et al.* [20] presented an extensive overview on landing and take-off scheduling problems. Lieder and Stolletz [21] recently summarized different models and solution approaches on the ASP that consider heterogeneous or interdependent runways, featuring articles up to 2015. ASP can be divided into Aircraft Landing Problem (ALP), Aircraft Take-off Problem (ATP), and a combination of both. Numerous physical and operational constraints as well as the layout of runway system must be considered. Various exact and heuristic approaches have been proposed. The solution quality and the computation time of the approach are two important factors. In real situations, controllers can only use algorithms which can quickly find a good solution. Optimal solutions arising from lengthy computation times are of little practical use [20]. Moreover, departures and arrivals are highly coupled processes with complex interactions with regard to runway configurations, weather, mix of aircraft, etc. In the following section, we explain separately the constraints, objective functions, models and solution approaches of the ALP and ATP problem.

2.1.1 Aircraft landing problem

Given a set of arrival flights, the ALP consists in finding the landing sequence which maximizes the efficiency and the throughput of runway system while accommodating various operational constraints. The key constraint for safety is the wake-vortex separation. Other constraints include the earliest and latest time of arrival limited by aircraft speed, fuel, and controller actions; the precedence constraints for which one aircraft must land before another one due to route structure, etc. The objective functions can be maximizing runway throughput, minimizing average flight delay or minimizing flight delay, minimizing weighted sum of delay costs, etc. Different controller decision support tools have been developed and experimented, such as the Center-TRACON Automation System (CTAS) by NASA in the U.S. [33], the MAESTRO Arrival Management (AMAN) owned by Thales group in Europe [34], etc. Different solution approaches have been applied, including Dynamic Programming (DP), Branch and Bound (B&B), Integer Programming (IP), heuristics and metaheuristics. In the following section, we mainly review the most cited works, an extensive overview can be found in [20]. Table 2.1 provides an overview of the related papers on ALP problem. Column 2 shows the main operational constraints. Column 3 lists the objective function. Minimizing the total cost of delay is the most common objective. Column 4 shows the solution approaches, and column 5 gives the problem feature. The ALP problem can be divided into static case (complete information of the set of aircraft are known) and dynamic case (decisions have to be made as time passes and the situation changes).

Most of the research considered the ALP problem as a static case. Dear [22] first proposed the Constrained Position Shift (CPS) approach that an aircraft can be sequenced (forward or rearward) no more than a pre-specified maximum number of positions from its FCFS position. This approach limits the aircraft reordering flexibility and considers fairness by not causing large deviations. Psaraftis [23] applied the DP algorithm considering the CPS concept and modeled the

Table 2.1: Overview of literature on the ALP problem.

Source	Constraints	Objective function minimize	Solution approach	Feature
Dear [22]	CPS, runway separation	$\sum$ delay	Dynamic Programming (DP)	static
Psaraftis [23]	CPS, runway separation	makespan, $\sum$ delay	DP	static
Balakrishnan and Chandran [24]	CPS, runway separation, time-window, precedence constraints, triangle inequality	makespan, $\sum$ delay, maximum delay	DP	static
Lieder *et al.* [25]	runway separation, time-window	$\sum$ delay	DP with new dominance criterion	static
Beasley *et al.* [26]	runway separation, time-window	$\sum$ delay	MILP solved with CPLEX	static multiple runways
Pinol and Beasley [27]	runway separation, time-window	$\sum$ delay, landing planes as early as possible	Scatter search and bionomic algrithms	static multiple runways
Ernst *et al.* [28]	runway separation, time-window	$\sum$ delay	B&B and heuristic	static
Faye [29]	runway separation, time-window	$\sum$ delay, landing planes as early as possible	dynamic constraint generation algorithm	static multiple runways
Beasley *et al.* [30]	runway separation, time-window	$\sum$ delay	mixed-integer zero-one program	dynamic
Furini *et al.* [31]	runway separation, time-window	$\sum$ delay	MILP solved with CPLEX, tabu search	dynamic
Hu and Chen [32]	runway separation	$\sum$ delay	genetic algorithm	dynamic

problem as a job shop scheduling problem, where runways and aircraft are regarded as machines and jobs. The algorithm was polynomial-time and relied on all aircraft of the same type being identical, thus the time-window constraints were not taken into account. Balakrishnan and Chandran [24] presented DP algorithms for runway scheduling under CPS. Various operational constraints such as time-window restrictions and precedence constraints which had not been modeled by previous approaches were accounted for. Lieder *et al.* [25] developed a new dominance criterion for state space reduction to reduce computation times of DP approach. Other methods were developed and compared, mainly between exact methods (B&B, IP) and heuristic methods. Ernst *et al.* [28] used a B&B method and a problem space search heuristic for both single and multiple-runway problems. Beasley *et al.* [26] first proposed a MILP formulation of the aircraft landing problem in a static case. Pinol and Beasley [27] considered multiple runway ALP and presented two heuristic techniques to solve the problem. Furthermore, Faye [29] proposed a time discretization approach which provides good LP relaxation compared to the model in [26]. Moreover, they proposed a heuristic method based on constraint generation and improved previous results of [27].

Fewer studies have been conducted on the dynamic case. Beasley *et al.* [30] modeled the dynamic case of ALP problem as an application of the displacement problem. A displacement function was defined to quantify the effect of displacing each decision variable from its previous solution value to its new value. Furini *et al.* [31] presented a second MILP formulation and compared it with the previous model in [26], the two formulations mainly differ in the way the decision variables are defined. Additionally, they applied a rolling horizon algorithm to tackle the dynamic case and a tabu search heuristic to achieve short computation times. Hu and Chen [35] introduced the concept of receding horizon control (RHC) to tackle the dynamic aspect of the ALP. RHC is an N-step-ahead on-line optimization strategy. At each time interval, based on current available information, RHC optimizes the particular problem for the next N intervals in the near future, but only the part of solution corresponding to current interval is implemented [35]. This concept reduced computational burden and the solution space. They further integrated the RHC strategy into a genetic algorithm (GA) and investigated airborne delay and computation time [32]. Later, Hu and Paolo [36] extended a binary-representation-based GA integrated with the receding horizon control for the ALP, and showed the benefits compared to the permutation-representation-based GA.

2.1.2 Aircraft take-off problem

ATP have many similarities with ALP in terms of constraints such as wake vortex separation, time window etc. Moreover, ATP needs to consider the limited airside resources such as gates, ramps, taxiways, and holding area resulting in departure queues and congestion on the ground in many large airports. Hence, there is a need to discuss separately the ALP and the ATP. In this section, we first give some operational background of departure management. Then, we review the related articles of ATP.

The Estimated Off-Block Time (EOBT) is the estimated time that an aircraft will be ready after all doors closed, boarding bridge removed, and ready to push back upon reception of clearance from controller. Next, a Target Startup Approval Time (TSAT) is the time provided by controller that an aircraft can expect to receive push back approval taking into account EOBT, and the traffic situation etc. TSAT assignment is important to reduce aircraft waiting time with engine on. If many aircraft arrive at the runway threshold waiting for take-off, a long waiting time of the queue will lead to extra fuel burn. It is more efficient to delay departures before the engines start up so that the traffic arrives at the runway smoothly. Moreover, arrival flights may cross an active departure runway. Controllers delay runway-crossing clearances until a group of aircraft has accumulated at various holding points [19]. However, there is an upper limit on how many aircraft can wait at the same crossing point.

Table 2.2: Overview of literature on the ATP problem.

Source	**Constraints**	**Objective function minimize**	**Solution approach**
Balakrishnan and Chandran [37]	runway separation, time-window, CPS, precedence constraints, runway balancing, accommodate runway crossings	avg. delay makespan	Dynamic Programming (DP)
Monotoya *et al.* [38]	runway separation, time-window, CPS, precedence constraints, accommodate runway crossings	$\sum$ delay makespan	multi-objective DP
Gupta *et al.* [39]	runway separation, time-window, limitation on queue size, FIFO of each queue	$\sum$ delay, makespan, maximum delay	MILP solved with CPLEX
Atkin *et al.* [40]	runway separation, SID separation, CTOT window, holding-point restrictions	weighted sum of the delay, reordering cost, CTOT cost and penalty cost	Tabu search
Anagnostakis and Clarke [41]	runway separation, max number of arrival crossings, max delay before crossing	makespan, $\sum$ delay	Integer Program
Malik and Jung [42]	runway separation, max queue size, FIFO of the queue	makespan, $\sum$ delay	DP, local search

Table 2.2 provides an overview of the related papers on ATP problem. Column 2 shows the main operational constraints. Besides the common constraints of runway separation, time-window similarly to ALP, holding point restrictions, interactions with arrival crossings are considered as well. Column 3 lists the objective function. Minimizing delay and minimizing makespan are the most common objectives. Column 4 shows the solution approaches with both exact and heuristic methods.

Considering the exact approach, Balakrishnan and Chandran [37] introduced a DP algorithm with CPS that set a limit on the number of positions an aircraft can occupy in the sequence for the ATP based on their model for the ALP. They also provide complexity analyses of their algorithms and extend their approach to multiple runways and active runway crossings. Monotoya *et al.* [38] formulate the ATP as a multi-objective optimization problem with respect to the total aircraft delay and the runway throughput and use DP approach to solve it. Gupta *et al.* [39] present a MILP formulation to handle the departure queuing area. A maximum number of aircraft that can occupy a queue is imposed at any time as a constraint. Next work by the same authors [43] presented a MILP formulation for incorporating active runway crossings in departure scheduling. However, the computational performance was poor due to the lack of good lower bounds. Moreover, the MILP formulation only provides optimal solution for each short time frame (e.g. 15 minutes), which is addressed and improved in [44].

Different heuristic approaches have been proposed. Atkin *et al.* [45] applied three different heuristic methods for the ATP, taking into account some specific constraints concerning London Heathrow airport. Further works can be found in [40, 46, 47]. The Calculated Take-Off Time (CTOT) constraints are taken into account in the paper, where the aircraft need to take off within a specified time window which is allocated by the Eurocontrol Central Flow Management Unit for many European aircraft. Anagnostakis and Clarke [41] proposed a two-stage heuristic algorithm to solve departure and runway crossing problem.

Regarding comparative studies between exact and heuristic approaches, Furini *et al.* [31] presented two MILP models and a tabu search heuristic for finding solutions to the ASP in a short computation time. The two MILP formulations were used in a heuristic way (by imposing a time limit). With a time limit of 15 seconds, the tabu search heuristic outperforms the two MILP models. However, the model is applicable for both ALP and ATP, thus some operational constraints for the ATP such as the holding point restriction are not considered. Malik and Jung [42] proposed one exact algorithm and two heuristic algorithms to solve the ATP, and they conclude that heuristics are more suitable than the exact approach to produce good quality solutions in a relatively short computation time.

2.2 Airport ground movement

This section reviews research works related to airport ground movement (AGM). The aim of airport ground operation problem is to search for the best routes and schedules, in order to minimize travel time and to mitigate surface congestion. The airport ground movement problem takes several major constraints into account, including:

- **Minimum taxi separation:** it is required for any two taxiing aircraft to keep a minimum separation and do not conflict with each other. In real operations, this is ensured by pilot visual checks;
- **Taxi path:** aircraft can follow a set of alternate taxi paths validated by the controller to avoid conflicts and forbidden area;

- **Taxi speed:** aircraft follow a certain taxi speed depending either on the type of aircraft or the type of taxiway;
- **Time constraints:** For arrivals, the landing time is usually fixed to start the taxi-in process. For departures, the earliest pushback time is used to either start the taxi-out process, or delay the pushback at the gate.

The objectives and constraints may vary due to the differences between airports and stakeholder aims. Atkin *et al.* [48] provided an overview of the previous research of ground movement, featuring articles up to 2010. Table 2.3 provides an overview of the related papers on AGM problem. Column 2 shows the main operational constraints, the taxi separation, the taxi speed restrictions and the origin and destination timing are the most common constraints. Column 3 lists different routing options. Three possible routing options are most used in the aircraft ground taxi problem: single path, alternate path and free path [49]. In the first case, aircraft follow a predetermined taxi route, which is usually the standard route in the airport. In the second case, several routing options are proposed after applying, for instance, the k-shortest path algorithm [50]. In the last case, any route can be assigned to an aircraft. Column 4 shows the objective functions, and column 5 gives the solution approach. In this section, we divided related works into exact methods and heuristic methods.

2.2.1 Exact methods for AGM

Smeltink *et al.* [51] proposed a MILP formulation to schedule the taxi movements considering standard predefined taxi routes. The objective was to minimize both the total taxi time and the deviation between the desired departure time and the scheduled departure time. They tested flight instances within 30-min time interval (from 23 to 54 aircraft) at Amsterdam Airport Schiphol, the optimal solution cannot be found within 1000 seconds for some instances. Therefore, they developed three different variants of a rolling horizon approach. However, the computation time was still long. Moreover, their methods did not permit holding and rerouting for aircraft. Rathinam *et al.* [60] improved the previous formulation by Smeltink *et al.* [51], and took into account the aircraft speed constraints (i.e., when the lead aircraft travels slower than the aircraft trailing behind), and the runway occupancy time constraint. They minimized the total taxi time using a spacial network and a predefined route for each aircraft. Guépet *et al.* [52] followed the two previous models in [51] and [60] to include alternate paths. They compared four indicators: the average taxi time, the average completion time, the average delay and the on-time performance of the Copenhagen airport. However, their alternate path approach has a limited effect on ground performance but increases a lot the computation time.

Roling and Visser [53] described a MILP formulation to deconflict the taxi plans while minimizing delay and total taxi times. A number of alternative taxi routes can be assigned to each aircraft in addition to the shortest route. However, the results were illustrated in a simple hypothetical airport. Clare and Richard [54] presented a MILP formulation to solve the airport taxiway routing and runway scheduling problem. The objective was to minimize a weighted combination of the makespan, the total taxi time, and the total taxi distance. Their method was implemented in a receding horizon, in order to improve the computation scalability.

Balakrishnan and Jung [55] developed an Integer Programming formulation for modeling taxiway operations at Dallas-Fort Worth airport. They have proposed two control strategies: controlled pushback of departures, and taxi routing for arrivals. The two strategies decreased the taxi time and the time spent waiting in runway crossing queues. However, several operational constraints such as overtaking and collision avoidance at intersections were not taken into account. Lee and Balakrishnan [61] proposed a sequential approach and an integrated approach for airport taxiway and

Table 2.3: Overview of literature on the AGM problem.

Source	**Constraints**	**Routing options**	**Objectives minimize**	**Solution approach**
Smeltink *et al.* [51]	taxi separation taxi speeds restriction origin and destination timing	single path	∑taxi times	MILP solved with CPLEX
Guépet *et al.* [52]	taxi separation taxi speeds restriction origin and destination timing runway sequencing constraint stand blockage constraint	alternate path	∑taxi times+ ∑ delay + ∑completion time+ Nb of flights with long delay	MILP solved with CPLEX
Roling and Visser [53]	taxi separation taxi speeds restriction origin and destination timing	alternate path	∑taxi times+ ∑ holding time	MILP solved with CPLEX
Clare and Richard [54]	taxi separation taxi speeds restriction origin and destination timing runway sequencing constraint	path free	∑taxi times+ ∑ taxi distance+ makespan	MILP solved with CPLEX
Balakrishnan and Jung [55]	taxi separation taxi speeds restriction origin and destination timing runway sequencing constraint	alternate path	∑taxi times+ on-time penalty	Integer Program solved with CPLEX
Pesic *et al.* [56]	taxi separation taxi speeds restriction origin and destination timing	alternate path	mean delay	Genetic Algorithm (GA)
Gotteland *et al.* [57]	taxi separation taxi speeds restriction origin and destination timing runway sequencing constraint	alternate path	mean delay	GA and Brunch&Bound (B&B)
Deau *et al.* [58]	taxi separation taxi speeds restriction origin and destination timing runway sequencing constraint CFMU slot	alternate path	delay targeted runway slots	GA and B&B
Ravizza *et al.* [59]	taxi separation taxi speeds restriction origin and destination timing	path free	∑taxi times	vertex-based label-setting algorithm based on Dijkstra

runway scheduling. The objective is to minimize the runway delay for departure aircraft and the taxi time. The results showed that during peak times, the integrated approach provided the better optimal schedule compared to the sequential approach at the cost of computational performance.

Malik *et al.* [62] proposed a MILP model to provide metering advisories for departure aircraft. The first stage was to give the best runway usage time, while the second stage determined the aircraft release time from the spot to meet the first stage's departure time. Besides considering individual aircraft trajectory optimization for the departure metering problem, Simaiakis and Balakrishnan [63] proposed an aggregate control strategy with a queuing network model of the departure processes to decrease fuel burn and emissions. Based on this pushback rate control strategy, i.e., regulating the rate at which aircraft push back from their gates during airport congestion periods, field tests at Boston airport [64] demonstrated significant fuel burn savings from gate-holds with engines off. While Boston airport primarily experience congestion at the runways, at other airports may occur additional queuing in the ramp area. Badrinath and Balakrishnan [65] modeled the taxi-out process of Charlotte airport by two queues in tandem. Their optimal control policy reduced the taxi-out time and the queue length. Lee *et al.* [66] compared the aggregated queue-based approach and the trajectory-based approach, they found that the trajectory-based approach yields better improvements in terms of average taxi-out time at a relatively uncongested airport, whereas the queue-based control is easy-to-implement without requiring many procedure modifications and advanced technologies.

2.2.2 Heuristic methods for AGM

Pesic *et al.* [56] first applied a Genetic Algorithm (GA) to solve the ground movement problem at Paris Charles De-Gaulle airport. The objective was to minimize the delays and to ensure the separations by modifying the path of aircraft, holding the aircraft at the gate, on taxiways or at the holding point before taking off. Gotteland *et al.* [67] extended the previous work by taking into account speed uncertainty, which is a fixed percentage of the initial defined speed depending on procedures and tuning rate. Furthermore, Gotteland *et al.* [57, 68] compared a sequential B&B algorithm, a global GA, and a hybrid GA and B&B algorithm. The results showed that the B&B solution had higher delay than the two other algorithms, while the hybrid method was the most efficient method. Deau *et al.* [58, 69] extended their previous work by first solving the departure runway sequencing problem using a B&B algorithm. Then, the GA was implemented to solve the ground movement problem by minimizing the taxi delay and meeting the target departure runway slots.

Ravizza *et al.* [59] applied a Quickest Path Problem with Time Windows algorithm to sequentially route aircraft on the airport surface. They further proposed a swap heuristic for modifying the order to reduce the total taxi time. Subsequently, in [70], they combined the previous graph-based routing algorithm and a population adaptive immune algorithm for analyzing the trade-off between taxi time and fuel consumption during taxiing. Furthermore, Chen *et al.* [71, 72] first generated optimal speed profile and embedded it within a multi-objective optimization framework for unimpeded taxiing aircraft, then integrated the previous speed profiles to the routing and scheduling problems.

Benlic *et al.* [73] presented a local search heuristic for the coupled runway sequencing and taxiway routing problems. The work was implemented in a receding horizon framework, and took into account the interactions between arrival and departure aircraft on the airport surface at Manchester airport. The maximum computation time per horizon (40 aircraft in the horizon) is around 95 seconds. Mori [74] applied a tabu search technique to develop a pushback time assignment algorithm in order to reduce take-off delays.

2.3 Terminal airspace management

The air traffic departing from airports follows the Standard Instrument Departure (SID) routes that connect runways to the TMA exit points, while the air traffic arriving to airports follows the Standard Terminal Arrival Routes (STAR) that connect the TMA entry points to the runways. During the transition from the en-route to the terminal airspaces, aircraft arriving from different entry points must be merged and organized into an orderly stream. Controllers give instructions such as speed change, heading change, and flight level change to organize aircraft flows. Arrivals and departures are required to maintain different flight altitudes in order to be spatially segregated. Recently, researchers focused on the terminal routing and scheduling problem in order to improve efficiency of TMA operations.

At a strategic level, SIDs/STARs can be optimized with regard to the total length of the designed routes in order to maximize the efficiency of TMA airspace. Zhou *et al.* [75] proposed a mathematical model of the 3D SIDs/STARs route planning, first designed one single route considering obstacle avoidance using B&B method [76], then it has been extended to multiple routes using SA method [77]. At a tactical level, the aircraft speed on the trajectory need to be determined, conflict detection and resolution between aircraft are taken into account. Zuniga *et al.* [78] proposed a new approach to merge the incoming arrival flows from different routes by mean of speed and path changes using a fish-bone shaped route topology. GA was applied to solve flight conflicts and minimize flight delays. Chida *et al.* [79] extended the previous work by adding the required time of arrival in the TMA as decision variables and compared four route topologies with one or multiple merging points. Xue and Zelinski [80] noticed the fact that in Los Angeles terminal airspace, departures and arrivals have to fly longer-than-necessary distances to resolve potential conflicts, while direct routes with shorter distance exist for arrivals if there is no departure flow, and vice versa. Hence, they proposed three methods: spatial, temporal, and hybrid separations to solve airspace conflicts. Spatial separation means that arrivals and departures keep different flight altitudes at the same fix. Temporal separation utilizes the shared fix by resolving conflicts solely with temporal controls. Hybrid separation is a combination of the two previous method. A non-dominated sorting genetic algorithm [81] was applied and a total of 15 interacting departures and arrivals were tested. Bosson *et al.* [82] presented an alternate method by building a stochastic scheduler based on a machine job-shop scheduling problem formulation to compute schedules for terminal airspace fixes. A multistage stochastic programming approach was used to solve the problem.

Besides the conventional route topologies, the Point Merge System (PMS) proposed by Eurocontrol [83] achieved the aircraft sequence on a point using predefined legs at iso-distance to this point for path shortening or stretching. This novel technology enables aircraft to remain on lateral navigation, and reduce heading instructions thus provides a reduction of workload and communications. Liang [84] defined a multi-level point merge route network at Beijing airport with parallel runways, integrated with a heuristic optimization algorithm to find a conflict-free and less delay trajectory planning. Further development such as the Continuous Descent Approach (CDA) and the runway assignment based on PMS were investigated as well.

2.4 Integrated optimization of TMA and airport

In contrast to the previously mentioned sub-problems related to airport and terminal airspace, few research focuses on the integrated optimization of airport and TMA operations.

Kjenstad *et al.* [85] considered the integrated surface management problem and arrival/departure runway sequencing problem. The airport was represented by a directed graph. A flight route was

a sequence of nodes and arcs. Two types of decision variables were defined: first establishing a route for each aircraft, then determining the precise timings through the route, including take-off and landing times. Flights were assumed to traverse arcs in fixed times and are allowed to wait at nodes. A minimum time separation between flights was defined at every node and arc. The times the flight starts and ends occupy the node/link were calculated. Conflict-free solution existed when the schedule of flights satisfied the condition that distinct flights could not occupy simultaneously the same node/link. Each flight was assigned with a target runway usage time and a departure/arrival runway usage time window. An arrival flight must be assigned to a landing time in the arrival time window. A departure flight was considered as dropped if the take-off did not happen within its departure time window. The objective function to minimize included three terms: the number of dropped departures, the sum of the deviation from the target runway usage time, and the taxi times. The integrated problem was solved in three stages: First, calculating the shortest feasible route for each flight; Then, determining the take-off/landing time by using a zero-one linear program formulation; Lastly, establishing the time in which a flight passes every node and arc of its route with respect to the runway sequence established at previous step, again a linear program based heuristic was used to resolve the third step. The scope of this work focused on the surface management and runway sequencing, the terminal airspace was not taken into account. The real-world constraints such as the layout of the holding area at the runway threshold were not considered. In fact, the holding area is airport-dependent which may limit the number of aircraft waiting at the runway threshold, and it can influence the final flight sequence.

Bosson *et al.* [86] modeled the integrated airspace and surface routing and scheduling problem as a machine job-shop scheduling problem. A set of aircraft are similar to a set of jobs, waypoints (including surface taxi waypoints and terminal airspace waypoints) are similar to machines. The release time was the first arrival fix passage time for arrivals or the estimated pushback time for departures. The due dates corresponded to the in-block time for arrivals and last departure fix passage time for departures. A processing time is defined at each waypoint of the route traveled and depends on the separation time requirements between type-based aircraft pairs. The optimization model had temporal variables to record the aircraft times at waypoint and spatial variables to establish aircraft routes. The model was designed to minimize the sum of total travel times as well as the earliness and tardiness of optimized release times for each flight. Moreover, the uncertainty was considered by using an input set of perturbed due date schedules, and the impact of uncertainty are tackled by minimizing the earliness and tardiness of optimized complete times. This work proposed to solve the integrated problem within one time frame, i.e. the time decisions following the route were simultaneously made from the TMA entry point until the gate at airport for arrivals, and from the gate to the TMA exit point for departures. However, when the arrivals enter the TMA, typically there are still 15-30 minutes left before landing. In this time horizon, decisions for the ground taxi are still far to be taken since there are more uncertainties related to the ground traffic. Hence, we need to make decisions coherently with respect to the prediction horizon of problems.

Frankovich [87] considered both strategic and tactical levels to optimizing the airport traffic flows. The air traffic flow management problem usually focused on traffic flows in a network of airports, while their research paid attention to optimizing traffic flows through an airport. On the strategic level, they proposed a MILP model to select a runway configuration sequence and determine the balance of arrivals and departures. On the tactical level, they assigned flights to runways, determined departure gate-holding durations, and routes flights to their assigned runways. Runway configuration selection is related to airport layouts. In our problem context, for airports with only two runway configurations such as CDG and CLT, the balance of arrivals and departures for different runway configurations is not a critical issue. We consider the problem of airport congestion management with an abstract network and the ground movement problem with a detailed graph

network.

Khadilkar [88] proposed a control strategy for airport and terminal area operations. The airport surface was modeled by a network of major taxiways and intersections, the aircraft entry time into this network was regulated by a control algorithm. The terminal area arrival control was divided into a distributed control in low-density airspace and a centralized control in high-density terminal areas for conflict detection and resolution. Their work modeled the problem at a mesoscopic level with high-fidelity representations of some elements, the detailed graph models and trajectory routing and timing were not considered. As trajectory-based operations are gradually adopted on the airport surface, it is reasonable to consider a detailed node-link airport model and investigate its benefits.

After reviewing the research work related to airport and surrounding TMA traffic optimizations, in the following section we give a brief summary about the optimization problems and methods.

2.5 Optimization problems and methods

Many operational problems such as airport operations illustrated in the previous sections are solved through mathematical optimization, which can be used as a decision support tool facing real problem. In this section, we give a short review of optimization problems and resolution methods, with an emphasis on the methods applied in this thesis.

Optimization consists of finding the best solution among many feasible solutions that satisfy all the constraints in an optimization problem. From the real problem, a mathematical model is built by characterizing the *state space* with the associated *constraints* and the *objective function*. The state space is the set of all possible configurations of a system upon which we may act in order to optimize one (or more) objective(s). The objective function represents the main aim of the model which is either to be minimized or maximized. A single-objective optimization problem involves a single objective function and results in a single solution. A multi-objective optimization considers several conflicting objectives simultaneously. In such a case, a set of alternative solutions called Pareto optimal solutions exist to be chosen by decision makers. In the case of air traffic optimization, the criteria of interest to stakeholders can be generally ranked in order of priority and can be weighted easily within a single one-dimension criterion. The use of multi-objective optimization does not therefore seem indispensable and will not be discussed later, although this field deserves to be considered throughly in future research.

Solution algorithms are developed with regard to the model and provide solutions. The model can be different from reality due to the limitation of optimization algorithms, therefore the solution produced could be rather different from the true solution in the real world [89]. After analyzing the problem and building the mathematical model, one needs to choose the best method or algorithm to use. Depending on the mechanism used to move within the state space, we differentiate between exact approaches and heuristic approaches (see Fig. 2.1).

- **Exact approaches:** are methods that solve a problem exactly and then return the global optimum solution of the problem. In addition, they also provide the mathematical proof that the returned solution is optimal [90]. Classical exact resolution methods include B&B, dynamic programming, linear and integer programming, etc. However, it requires large computation time to solve complex, high dimensional, NP-hard [91] problems.

- **Heuristic approaches:** seeks to produce good-quality but not necessarily optimal solutions in reasonable computation times.

  - **Heuristics:** are basic approximate algorithms that search the solution space to find a

good solution [92]. A good solution means that the optimality is not guaranteed, and how good is the solution compared to the optimum solution is unknown.

– **Metaheuristics:** are higher level concepts for exploring search spaces by using different strategies [92]. The balance between the *exploration* (searching lots of regions of the state space to not get trapped into local optimum) and the *exploitation* (examine carefully if good quality solution exists in a promising region of the state space) is the key point to quickly identify high quality solutions without losing diversification.

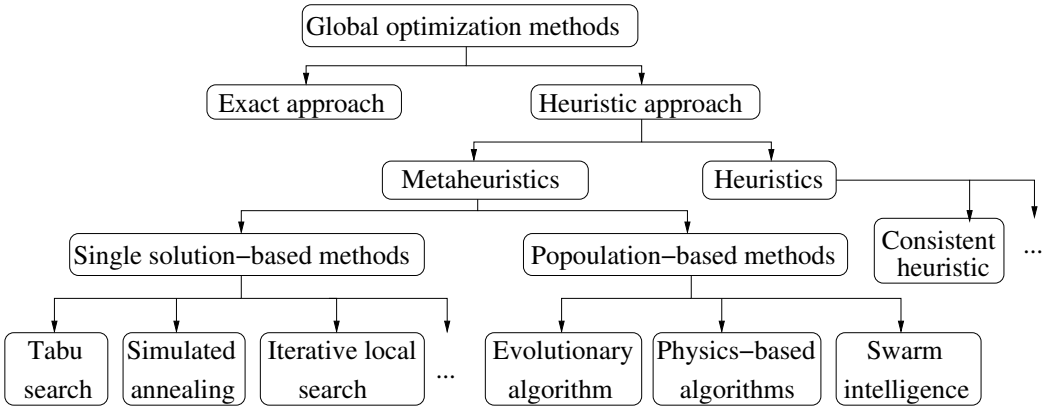


Figure 2.1: Optimization methods classification.

In general, metaheuristics can be roughly categorized into two classes [93]:

- **Single solution-based methods** start with a single initial point in the search space, and move to another point and so forth, describing a trajectory in the search space. Single solution-based methods mainly include the simulated annealing [94], the tabu search [95], the iterated local search [96], etc;
- **Population-based methods** make evolve a set of points in the search space. By evolving a whole population of candidate solutions, the selection processes choose elite solutions and apply different transformation operators. The processes are repeated until reaching the predefined termination criteria. The most studied methods are related to Evolutionary Computation and Swarm Intelligence.

Metaheuristics have been applied to many ATM optimization problems, we refer the reader interested by applications of metaheuristics to ATM problems to the following books [89, 97]. In this thesis, we rely on a simulated annealing algorithm for its simplicity to implement and the applicability to large-dimension state space involving much memory. Moreover, we compare the SA and an exact method in Chapter 5 to illustrate the solution quality and computation time of each approach. In the next section, we give a brief introduction to the classic SA algorithm.

2.5.1 Basics of simulated annealing

In the early 1980s, Kirkpatrick *et al.* [94] introduced the SA metaheuristic in the area of combinatorial optimization. SA is inspired by the annealing process in metallurgy that alters the property of material by controlling the temperature. It can be summarized by the following two steps (see Fig. 2.2):

- Heating the solid to a specific temperature that permits many atomic rearrangements;
- Cooling the solid slowly to reach a state of minimum energy.

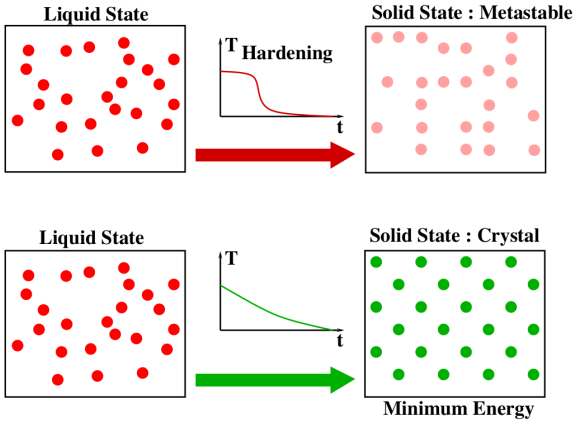


Figure 2.2: When temperature is high, the material is in a liquid state (left). For a hardening process, the material reaches a solid state with non-minimal energy (metastable state; top right). In this case, the structure of the atoms has no symmetry. During a slow annealing process, the material reaches also a solid state but for which atoms are organized with symmetry (crystal; bottom right). [2]

At high temperature, the particles are in a random state. By lowering the temperature slowly, the material is carried to a well ordered solid state of minimal energy. If a sudden cooling occurs, the material is found with metastable structures.

The annealing was modeled and simulated by using the Metropolis algorithm [98]. The algorithm is based on Monte Carlo techniques which consist in generating a sequence of state of the solid. The acceptance probability P_a of a transition from state i to state j is

$$P_a = \begin{cases} 1 & \text{if } E_j < E_i, \\ e^{\frac{E_i - E_j}{k_b T}}, & \text{else.} \end{cases}$$

where E_i and E_j represent the energy of state i and j, T is the temperature of the solid, and k_b is the Boltzmann constant. When the temperature is high, the probability that an atom moves to a state of high energy is close to 1, allowing random movements. When the temperature decreases, the probability becomes lower and lower until the atoms can only move to a state of lower energy.

In the SA algorithm, the Metropolis algorithm is applied to generate a sequence of solutions in the state space S. The set of state space corresponds to the possible states of the solid; the objective function f to be minimized is similar to the energy of the solid. A control parameter, T, is introduced to act as temperature. The user needs to provide for each point of the state space, a neighborhood and a mechanism for generating a solution in this neighborhood. Then, for two points i, j in the state space, the acceptance criterion for accepting the solution j from the current solution i is given by the following probability:

$$P\{\text{accept } j\} = \begin{cases} 1 & \text{if } f(j) < f(i), \\ e^{\frac{f(i) - f(j)}{T}}, & \text{else.} \end{cases}$$

Then, the neighborhood generation principle is similar to the perturbation mechanism of the Metropolis algorithm, and the acceptance criterion represents the Metropolis principle.

A *transition* is defined as the replacement of the current solution by a neighboring solution, it consists of the neighborhood generation and acceptance. Let N be the number of transitions at each time step, the principle of SA can be summarized by Algorithm 1:

Algorithm 1 Simulated Annealing

procedure SIMULATEDANNEALING(x_0, T_0)
  $x_i \leftarrow x_0$
  $y_i \leftarrow f(x_i)$
  $k \leftarrow 0$
  $T_k \leftarrow T_0$
  repeat
    for $i = 0$ to N **do**
      x_j = GenerateNeighbor(x_i);
      CriterionCalculation $y_j = f(x_j)$;
      if accept(y_i,y_j,T_k) **then**
        $x_i = x_j$;
        $y_i = y_j$;
      end if
    end for
    $k \leftarrow k+1$
    T_k = DecreaseTemperature(T_0, k);
  until $T_k \approx 0$
  return x_i
end procedure

procedure ACCEPTTRANSITION(y_i, y_j, T_k)
  if $y_j < y_i$ **then**
    return True;
  end if
  $r \leftarrow$ random([0,1]) ▷ Accept the transition with probability P_i
  $P_i \leftarrow e^{\frac{y_i - y_j}{T_k}}$
  if $r < P_i$ **then**
    return True;
  end if
  return False;
end procedure

We randomly select an initial state x_0. We choose an initial temperature T_0 such that the acceptance probability for a transition that degrades the solution is close to 1. Then, the temperature slowly decreases in the algorithm. The evolution of the temperature enables the modulation between exploration and exploitation in the state space in the process. At the beginning, when the temperature is high, a deteriorated solution is more likely to be accepted, leading to a thorough exploration of the search space by escaping local minimum. As temperature decreases, less degraded solutions are accepted. At the end, no deterioration is accepted when the temperature tends to zero. More fundamental theoretical aspects about SA can be found in [99].

2.5.2 Practical issues of simulated annealing

In order to implement the SA to particular applications, the two following practical issues are discussed in this section: cooling process and evaluation-based simulation.

Cooling schedule

- **Initial temperature** The initial temperature strongly depends on the problem considered. A too large initial temperature cause some waste of calculation time. However, if the initial temperature is chosen too small, the exploration of the search space might be restricted and the quality of the results decreases. Initial temperature can be computed by using the following method:

  1. **Initialize T to be a relative small value, for example, $0.1\overline{f}$ (where $\overline{f}$ is an estimate of the mean of the objective function);**
  2. **Repeat**
  3. k:=0;
  4. **For** $i = 0$ **to** N **do**
     - **Generate a solution j from the solution i;**
     - **If** the solution j satisfies the acceptance criterion of SA **then** k:=k+1;
  5. acceptanceRate = k/N;
  6. T=T*1.1;
  7. **Until** acceptanceRate ≥ 0.8

  There are many other recommendations about the choices of initial temperature, we refer the interested reader to the following books [90, 100].

- **Cooling loop** Usually in SA, the decrease of the temperature at iteration i is described by one of the following classical cooling schedules or laws:

  - **Geometric law:**
    $$T_{i+1} = T_i.\alpha, \quad 0 < \alpha < 1$$
    At each iteration, we get the new temperature by multiplying the former temperature by a predefined coefficient α. The choice of α is critical because if α is too large, the temperature decreases very slowly and the convergence toward the optimum is likely to be too long. However, if α is chosen too small, the temperature decreases fast and the algorithm may be quickly blocked in a local optimum;
  - **Linear Law:**
    $$T_i = T_0 - \beta.i, \quad \beta > 0$$
    where β is a predefined constant value;
  - **Logarithmic law:**
    $$T_i = T_0/log(i)$$
  - **Decrease by tier:** At every k iterations, we multiply the temperature by a coefficient $c(0 < c < 1)$. Here, there are therefore two parameters to be set by the user, which makes the algorithm more difficult to tune.

In this thesis, we have chosen the geometric law, with α obtained following several computational experiments on real data.

- **Stopping criterion** There are several ways to terminate the computation:
  - Upon reaching a predefined maximal number of temperature transitions;
  - Upon reaching a predefined limit for the total running time of the algorithm;
  - When there is no more improvement after a certain (predefined) number of transitions;
  - When there is no more improvement after a certain (predefined) amount of running time.

  In our implementation, the algorithm stops when the final temperature reaches the value $T_0.\varepsilon$, where ε is a predefined coefficient (typically $\varepsilon \simeq 0.001$).

Evaluation-based simulation

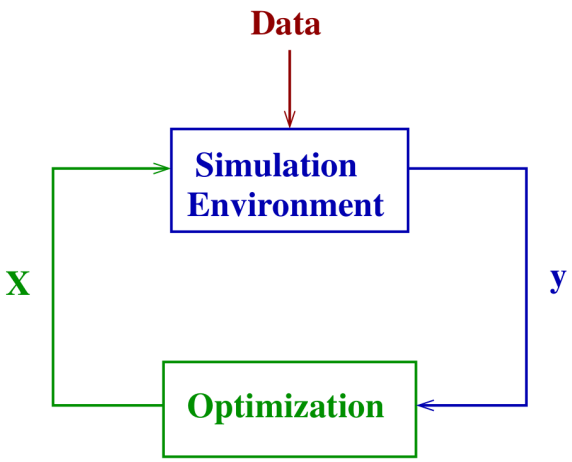
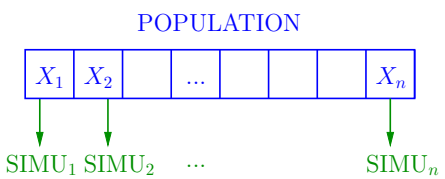
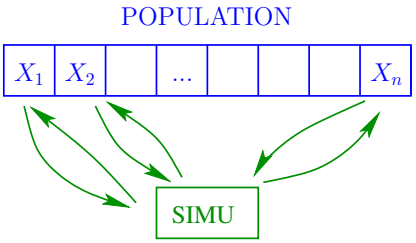


Figure 2.3: Objective-function evaluation based on a simulation process in [2].

As illustrated in Fig. 2.3, in many optimization applications, the optimization algorithm provides a set of state decisions X. Then, the state decisions are transferred to the simulation environment to be evaluated through the simulation process. The computed performance y generated by the simulation is used by the algorithm to decide the next step of the optimization process.

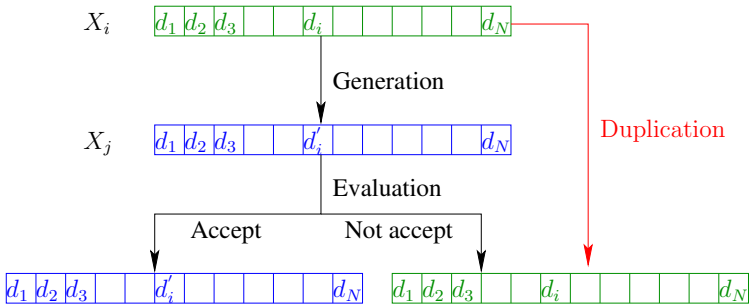


(a) Duplicated simulation environment for each individual.

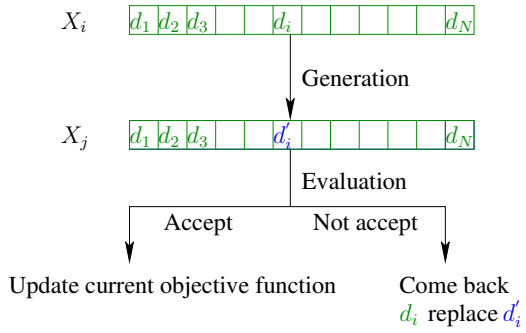


(b) Iterative evaluation of each individual with one simulation environment.

Figure 2.4: Simulation environment for population-based algorithms.



(a) The state space is built with a vector of decisions. The neighborhood change from state X_i to state X_j requires a copy in the memory, which may be inefficient and reduce the performance of SA.



(b) The neighborhood change consists in modifying only one decision d_i in the current state. If the solution X_j is rejected, only the information of d_i needs to be recovered.

Figure 2.5: Neighborhood generation and evaluation process for each transition in SA.

Facing real-life complex systems such as our application context, population-based algorithms may not be adapted to address such problems, mainly when the simulation environment requires huge amount of memory space. As a matter of fact, in the case of a population-based approach, the simulation environment has to be duplicated for each individual of the population of solutions (in Fig. 2.4a), which may require an excessive amount of memory. In order to avoid this drawback, one may think about having only one simulation environment which could be used each time a point in the population has to be evaluated as follows. In order to evaluate one population, one first considers the first individual. Then, the simulation environment is initiated and the simulation associated to the first individual is run. The associated performance is then transferred to the optimization algorithm. After that, the second individual is evaluated, but the simulation environment must be first cleared from the events of the first simulation. The simulation is then run for the second individual, and so on until the last individual of the population is evaluated (see Fig. 2.4b). In this case, the memory space is not an issue anymore, but the evaluation time may be excessive and the overall process too slow, due to the fact that the simulation environment is reset at each evaluation [2].

For each SA transition, a current state X_i and a neighbor state X_j are required for evaluation. From the coding point of view, X_j is generated from a copy of X_i as illustrated in Fig. 2.5a. This may lead to a drastic decrease of the performance of SA facing some state spaces of large dimensions. To avoid this unnecessary duplication, a *come back* operator is considered to modify only one component of the current solution [2]. When the new solution is accepted, then only the current objective function need to be updated; Otherwise, the come back operator is applied to return to the

previous solution without duplication in the memory as shown in Fig. 2.5b.

The come back operator needs to be checked carefully. Only some part of the state decisions need to be updated while others keep their initial values. Incoherent storage and updates would lead to undesired distortion for the searching direction of the algorithm.

2.6 Conclusions

In this chapter, we reviewed several air traffic optimization problems in the airport and the surrounding TMA. First, we reviewed the ASP problem for landing aircraft and take-off aircraft. The ASP problem aims at increasing the efficiency and capacity of the runway system while accommodating various operational constraints such as wake vortex separations, flight time windows, precedence constraints etc. Previous research vary from one single runway to multiple runways with interactions. Some recent works compared between exact methods and heuristic approaches to estimate the gap between optimality and heuristic solution, and showed that the heuristic approaches provided good results within much shorter computation time.

Afterwards, the studies related to airport surface management were presented. Some initial works considered only the taxi routing and scheduling problem, single taxi route or alternate routes were applied with regard to the airport layout and the chosen optimization approach. Subsequently, the integration with departure runway scheduling were studied since these two problems are highly connected. Different algorithms were proposed to calculate the gate release time and the taxi scheduling in order to be more fuel efficient and to reduce taxiway and runway congestion.

During recent years, researchers focused on the terminal airspace management problem, using either conventional SIDs/STARs route structure or some emerging topologies (e.g. PMS), took into account conflict detection and resolution by regulating the aircraft speed, the required time of arrival to minimize the airborne delay and increase the airspace efficiency. Then, research was extended to the integrated air traffic optimization of TMA and airport. Some attempts focused on the integration of landing and take-off runway scheduling and surface managements, further limited researches linked the terminal airspace with the ground movements.

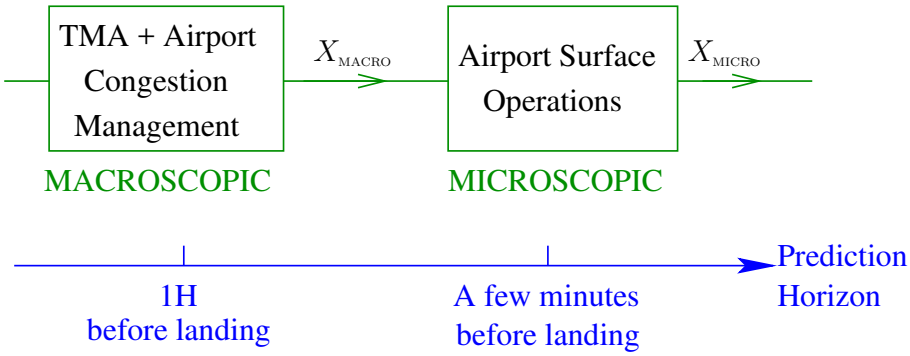


Figure 2.6: Two-level approach considering corresponding prediction horizons.

The literature on different sub-problems of airport operations is already quite wide. The main limitations of the previous works are the lack of connections between different problems. Although a few works solved jointly the integrated problem, the decisions of each sub-problem were not dealt with properly with respect to the corresponding prediction horizons. In this thesis, we propose a two-level approach to optimizing the air traffic at airport and its surrounding TMA as shown in Fig. 2.6. At the macroscopic level (e.g., one hour before landing), airport is seen like a set of resources

with limited capacities (terminal, taxi network, runway). At the microscopic level (a few minutes before landing), airport is seen with more details and microscopic decisions such as taxi routing and scheduling can be investigated. Such decision can not be taken at the macroscopic level (one hour in advance) due to the remaining uncertainties. This two-level approach aims at solving the integrated airport and TMA operation problem while keeping a good abstraction of different airport components taking into account their prediction horizons.

Chapter 3

Macroscopic optimization of air traffic at airports and in the TMA

Airport operations involve ground movement, runway sequencing and scheduling, terminal airspace management, etc. Segregated researches on these domains have been conducted in the past years and have been proven to improve safety and efficiency. Recently, integrated study of these sub-problems are in trend since they are intimately linked and affected by one another. Integrated optimization can gain potential benefits and target a better synchronization. Progressively, large-scale complex hub airports during peak hours are studied instead of limited toy examples. In general, combining the airside and ground problems and optimize them together can gain more benefit. However, the complexity of the integrated problem grows significantly as well.

In this chapter, we present an integrated air traffic optimization of airport and TMA at the macroscopic level. Section 3.1 describes the problem by introducing the scope and the network model of TMA and airport surface. Section 3.2 presents the mathematical model including the assumptions, input data, decision variables, constraints, and objective functions. A metaheuristic method combined with a time decomposition approach aiming at minimizing the airspace conflicts, airport overload, and total flight delays is presented in Section 3.3. Computational experiments conducted with the proposed methodology are presented in Section 3.4. Conclusions are discussed in Section 3.5.

3.1 Problem description

In terminal airspace, aircraft from different entry points must be merged and sequenced into an orderly stream, follow the Standard Terminal Arrival Routes (STAR), then prepare to land on the runway. After reducing the speed and exiting the runway, aircraft taxi towards the assigned gate. Then, after a certain turnaround duration for disembark, embark and other ground-holding operations, aircraft pushback, taxi out, depart, and follow the designated Standard Instrument Departure (SID) routes.

Based on different levels of fidelity, the airport models are broadly described as microscopic or macroscopic. In microscopic levels, individual aircraft trajectories with detailed information about taxiway routing, gate occupancy are explicitly considered. However, the simulations can be computationally intensive, and such detailed abstraction level is not relevant when decisions have to be taken 30 minutes or 1 hour in advance. As a matter of fact, remaining uncertainties may invalidate such microscopic decision when they are taken too much time ahead. In macroscopic models, the airport components (terminals, taxi network) can be globally modeled as resources with

specific capacities (as opposed to individual aircraft or taxiway links). This level of abstraction aims at identifying the airport congestion situations and facilitating the integration into decision support tools.

Our first step is to consider the terminal and airport integration problem at the macroscopic level, in order to be sufficiently flexible to resolve airspace conflicts (which implicitly represents potential workload for air traffic controllers), to mitigate airport congestion, and to reduce delays. We focus on the pre-tactical off-line planning phase, i.e., we assume the planning time to be several hours, or at least 30 minutes, prior to actual arrival/departure time. The integrated problem is considered in a moving time frame to reduce computational burden and to account for the frozen flights, which were already optimized in the previous time window and are still traveling in the current time window.

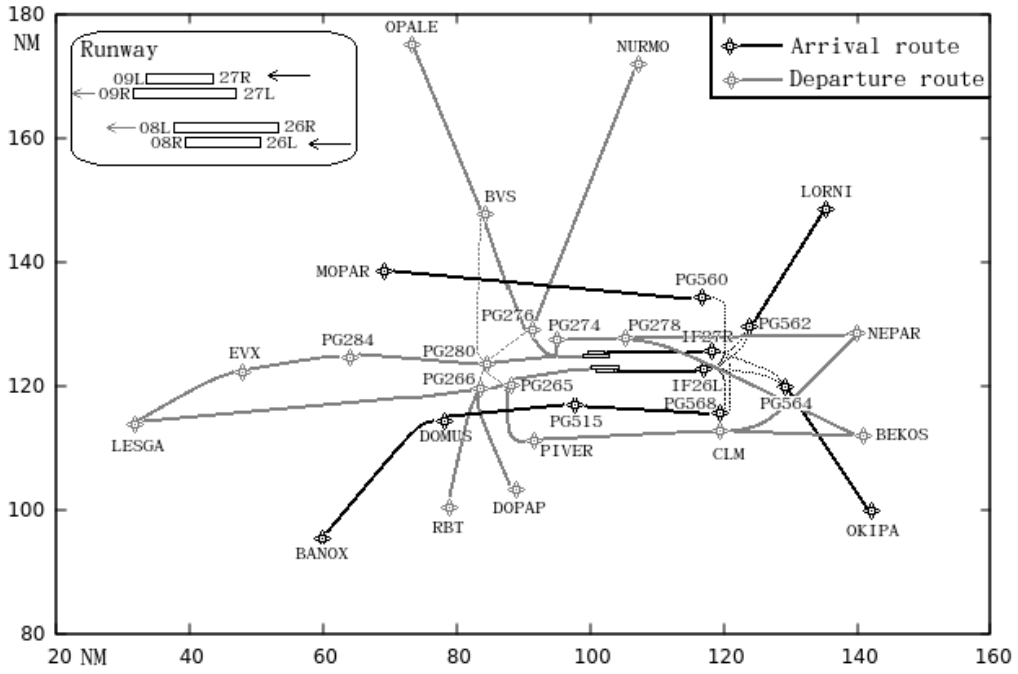


Figure 3.1: Terminal route network of arrivals and departures at CDG in West-flow configuration.

We model the TMA arrival and departure routes by a graph, $\mathcal{G}(\mathcal{N},\mathcal{L})$, in which the aircraft are allowed to fly in the airspace, where $\mathcal{N}$ is the set of nodes and $\mathcal{L}$ is the set of links. Each route is defined by a sequence of nodes and links; the first link starts from an entering point (a so-called Initial Approach Fix (IAF) for arrivals and runway threshold for departures) and the last link ends at the exit point (runway threshold for arrivals and last SID waypoint for departures).

Fig. 3.1 displays an example model of a route network for the Paris Charles De-Gaulle (CDG) airport. CDG is one of the busiest passenger airports in Europe, it is composed of four parallel runways (two for landings and two for take-offs) and three terminals. The West configuration with runways 26L/26R and 27L/27R is illustrated in Fig. 3.1, arrival and departure procedures are respectively represented by black and gray colors. In the arrival procedure, four-corner routes fuse into one to each runway. Each of the starting nodes of these four routes is an IAF. The set of entering points here is $\mathcal{N}_e$={ MOPAR, LORNI, OKIPA, BANOX}. For the departure procedure, one route starts at the runway threshold and ends with one of the SID waypoints in the set $\mathcal{N}_x$={OPALE, NURMO, NEPAR, BEKOS, DOPAP, RBT, LESGA}. The set of routes is denoted as $\mathcal{R} = \{r_e | e \in \mathcal{N}_e \cup \mathcal{N}_x\}$. Each aircraft follows exactly one of these routes corresponding to its entering point and landing

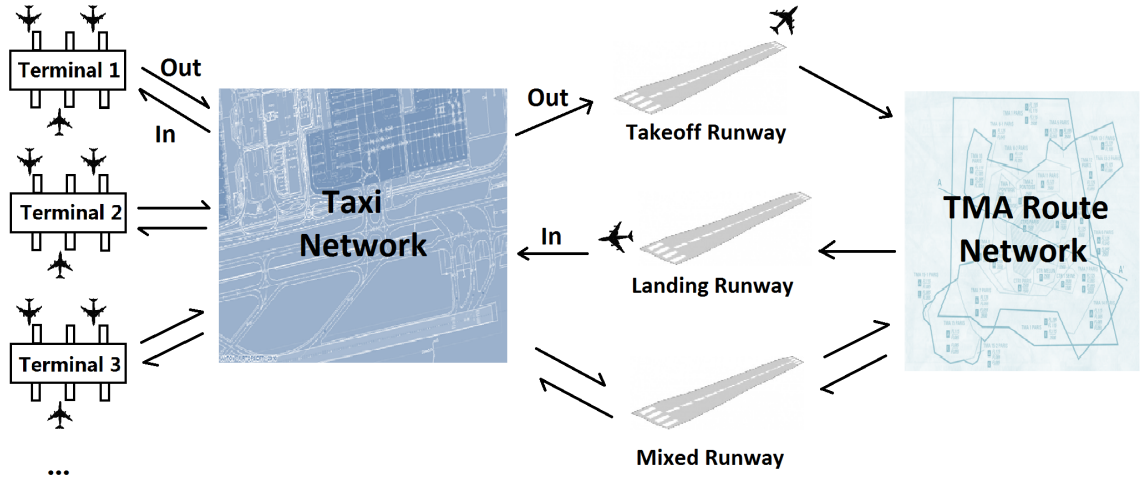


Figure 3.2: Network model of TMA and airport surface. Each component is considered as a resource with a specific capacity.

runway for arrivals, and exit point and take-off runway for departures.

According to real radar data and published routes, departure and arrival trajectories are separated in altitude. The arrival flows from the North-West (South-West) maintain their flight level at about 12000 ft (13000 ft) on the route section overlapping with departure flows between MOPAR and PG560 (DOMUS and PG515), and the departure flows to the North (South) pass below them. In the Eastern part, IF27R keeps a flight level of 5000 ft, IF26L keeps 4000 ft, so that the departures are able to fly above the arrivals.

Different airport components are considered using a network abstraction. Runways and terminals are modeled as resources with a specific capacity. We only take into account the overall capacity of a terminal without considering its individual gates. Taxiway is seen as a network with a capacity of total allowed number of taxi-in and taxi-out aircraft. The network model of TMA and airport surface is illustrated in Fig 3.2.

3.2 Mathematical model

In this section, we describe an integrated air traffic optimization model of TMA and airport. We first give the flight input data. Then, decision variables are defined. Lastly, we clarify constraints and introduce the objective function.

3.2.1 Input data

Assume that we are given a set of flights (or aircraft), $\mathcal{F} = \mathcal{A} \cup \mathcal{AD} \cup \mathcal{D}$, where $\mathcal{A}$ is a set of arrivals, flights that land at the airport and stay until the end of the day; $\mathcal{AD}$ is a set of arrival-departures, flights that arrive at the airport and depart from it after a turnaround duration; $\mathcal{D}$ is a set of departures, flights that are parked at the airport at the beginning of the day and depart later.

For each flight $f \in \mathcal{F}$, the following data is given: wake turbulence category for $f \in \mathcal{F}$, assigned terminal for $f \in \mathcal{F}$, entering waypoint at TMA for $f \in \mathcal{A} \bigcup \mathcal{AD}$, exit waypoint at TMA for $f \in \mathcal{D} \bigcup \mathcal{AD}$, taxi-in duration for $f \in \mathcal{A} \bigcup \mathcal{AD}$, taxi-out duration for $f \in \mathcal{D} \bigcup \mathcal{AD}$, initial landing runway number for $f \in \mathcal{A} \bigcup \mathcal{AD}$ (usually the requested landing runway is linked to the relative position of

the terminal and the landing runway), initial departure runway number for $f \in \mathcal{D} \bigcup \mathcal{AD}$, initial off-block time for $f \in \mathcal{D}$, turnaround duration for $f \in \mathcal{AD}$ and initial exit time at the exit SID waypoint for $f \in \mathcal{D} \bigcup \mathcal{AD}$. Moreover, we know:

- T_f^0: initial RTA (Required Time of Arrival) at the entering waypoint of TMA ($f \in \mathcal{A} \bigcup \mathcal{AD}$);
- V_f^0: initial speed at the entering waypoint of TMA ($f \in \mathcal{A} \bigcup \mathcal{AD}$);
- P_f^0: initial off-block time ($f \in \mathcal{D} \bigcup \mathcal{AD}$), it is the earliest time that an aircraft is ready to depart from its parking position.

Here are the assumptions and simplifications we make for our model:

- Flights are assumed to be able to park at any gates in their assigned terminal;
- We use an average taxi-in and taxi-out duration with regard to terminal and runway for each flight, due to the fact that we do not have information about the gate for the macroscopic model of the airport. Detailed study of airport taxi routings can be found in [17];
- Each aircraft has a constant deceleration or acceleration in the TMA. However, our model can tackle other types of trajectory (real radar data, BADA data) by discretizing the airspace into a space-time grid and detecting conflicts, as done in the work of Chaimatanan *et al.* [101]

3.2.2 Decision variables

The optimization model we are using has five types of decision variables. For arrivals, we have to attribute the entering time in the TMA, the entering speed in the TMA, and the landing runway:

1. Entering time in the TMA for $f \in \mathcal{A} \bigcup \mathcal{AD}$: First, we assume that we are given a maximum delay and a maximum advance, denoted respectively $\Delta T_{\max}$ and $\Delta T_{\min}$, which define the range of possible entering times in the TMA. We therefore define, for each flight $f \in \mathcal{A} \bigcup \mathcal{AD}$, a time-slot decision variable $t_f \in \mathcal{T}_f$, where

$$\mathcal{T}_f = \{T_f^0 + j\Delta T \ | \Delta T_{\min}/\Delta T \leqslant j \leqslant \Delta T_{\max}/\Delta T, \ j \in \mathbb{Z}\},$$

   where ΔT is a discretized time increment, whose value is to be set by the user. In order to shift an aircraft entering time in the TMA, we can either decrease or increase its speed in the en-route phase. In practice, the latter strategy burns more fuel, and may be far less attractive for the airlines. As a consequence, our time slot interval can be asymmetric, with $|\Delta T_{\max}| \geq |\Delta T_{\min}|$. In the following sections, the notation *delay* is used to indicate the time deviation of a flight.

2. Entering speed in the TMA for $f \in \mathcal{A} \bigcup \mathcal{AD}$: $v_f \in \mathcal{V}_f$, where

$$\mathcal{V}_f = \{V_f^{\min} + j\Delta_f^v \mid j \in \mathbb{Z}, |j| \leqslant (V_f^{\max} - V_f^{\min})/\Delta_f^v\},$$

   with Δ_f^v is a (user-defined) speed increment, $V_f^{\min}$ and $V_f^{\max}$ are given as input data corresponding to the minimum and maximum allowable speeds for aircraft f.

3. Landing runway for $f \in \mathcal{A} \bigcup \mathcal{AD}$: r_f^l is the landing runway decision for arrivals. Runway assignment is used to balance the capacity when one runway gets overloaded while another one is still able to accommodate more aircraft. Fig. 3.3 gives an example of how flight delay can be reduced by assigning the landing runways. In Case 1, with a first-come-first-served strategy, a total of 470 seconds is required for all five aircraft to land when all the traffic arrives on southern runway 26L. In Case 2, after optimizing the landing sequence for the same runway, a total of 258 seconds is required. In Case 3, the total landing time is reduced to 120 seconds with the possibility of runway assignment. Runway aircraft assignment enables to increase overall throughput with less delay comparing with the case where aircraft have no options to change their landing runway.

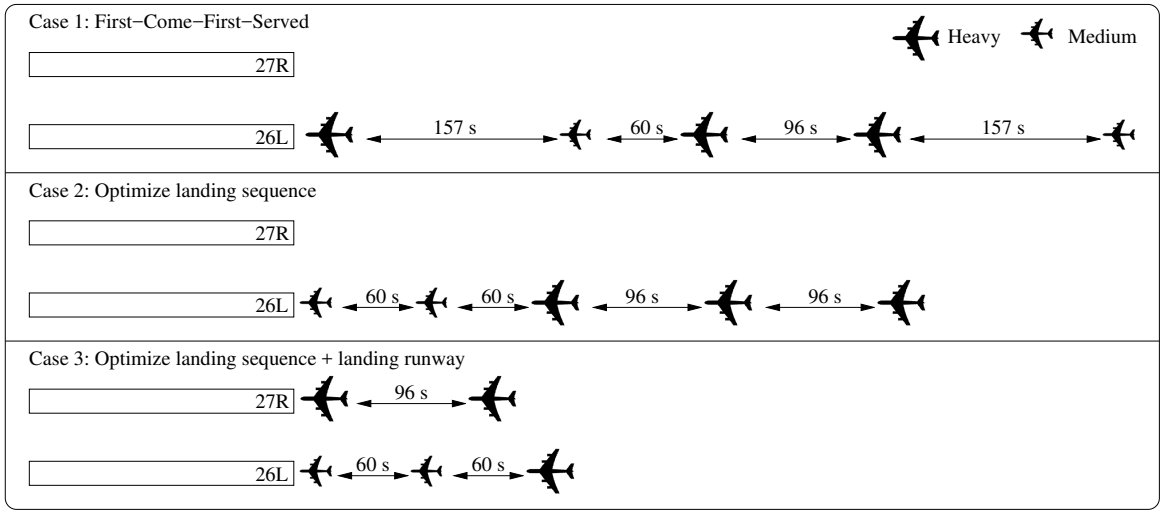


Figure 3.3: Comparison of three landing sequences. In Case 1, first-come-first-served strategy is applied; In Case 2, the landing sequence is optimized with regard to wake turbulence separation requirements; In Case 3, by assigning a landing runway and optimize landing sequence, less delay is achieved.

For departures, we have to decide the departure runway and the pushback time:

4. Departure runway for $f \in \mathcal{D} \bigcup \mathcal{AD}$: r_f^d is the take-off runway decision variable for departures. Similarly, it is possible to yield flights to another take-off runway when the current assigned one is too busy.

5. Pushback time for $f \in \mathcal{D} \bigcup \mathcal{AD}$: $p_f \in \mathcal{P}_f$, where

$$\mathcal{P}_f = \{P_f^0 + j\Delta T \mid 0 \leqslant j \leqslant \Delta T_{\max}^p / \Delta T,\ j \in \mathbb{N}\},$$

where $\Delta T_{\max}^p$ is the maximum pushback delay. In contrast to the entering time decision in the TMA for arrival flights, we can only delay a departure with regard to its earliest initial off-block time.

To summarize, our decision vector is $\mathbf{x} = (\mathbf{t}, \mathbf{v}, \mathbf{l}, \mathbf{d}, \mathbf{p})$, where $\mathbf{t}$ is the entering time vector, $\mathbf{v}$ is the entering speed vector, $\mathbf{l}$ is the landing runway vector, $\mathbf{d}$ is the departure runway vector, and $\mathbf{p}$ is the pushback time vector.

3.2.3 Constraints

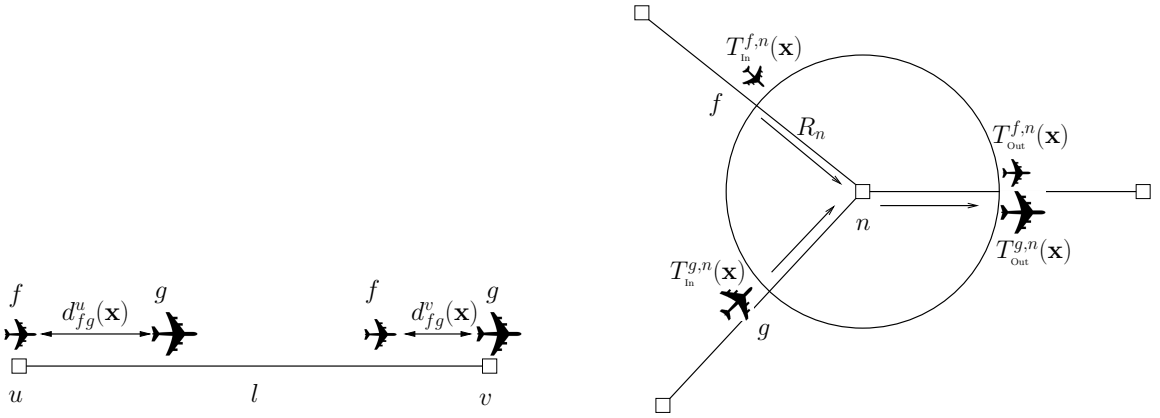
The previous decision variables have to stay in the same ranges which have been already defined. We have two other main constraints: wake turbulence separation, and single-runway separation for arrivals and departures. Before taking into account these separations, we first calculate the passage times at which the aircraft passes each resource (node, link, runway, taxi network, terminal) based on the decision vector $\mathbf{x}$. Let us denote respectively the passage time at the resource m, the entering time to the resource m, and the exit time from the resource m by $T_f^m(\mathbf{x})$, $T_{\text{In}}^{f,m}(\mathbf{x})$, and $T_{\text{Out}}^{f,m}(\mathbf{x})$.

Conflicts detection in the TMA

In this chapter, we make the assumption that the arrival and departure routes are separated in altitude, which corresponds to real-world TMA operations. Therefore, we detect conflicts separately for arrivals and for departures. Considering the above-described TMA route network structure, the TMA separation violation may happen either in the link or in the node:

Table 3.1: Distance-based separation on approach and departure according to aircraft categories (in NM). For example, the minimum distance separation between an heavy aircraft followed by a medium aircraft is 5 NM.

Category		**Trailing Aircraft**		
		Heavy	***Medium***	***Light***
Leading Aircraft	***Heavy***	4	5	6
	Medium	3	3	5
	Light	3	3	3



(a) Link conflict detection at the entry and exit of the link. No overtaking is allowed.

(b) Node conflict detection. At each time, only one aircraft passes the detection zone, i.e., a disk centered at node n with radius R_n.

Figure 3.4: Airspace conflict detection illustration.

- Link conflict: As shown in Fig. 3.4a, for two consecutive flights f, g that are flying through a link $l = (u, v)$, the minimum separation between these two aircraft, s_{fg}, is obtained based on

their respective wake turbulence category as shown in Table 3.1. Then, the actual separation distance of these aircraft at the entry time, $d^u_{fg}(\mathbf{x})$, and at the exit time of link l, $d^v_{fg}(\mathbf{x})$, are computed and compared with s_{fg} to detect potential link conflict.

Let us define, the *link conflict indicator*, $L^l_{fg}(\mathbf{x})$, for aircraft f and g at link l:

$$L^l_{fg}(\mathbf{x}) = \begin{cases} 1, & \text{if } T^u_f(\mathbf{x}) < T^u_g(\mathbf{x}) \text{ and } (d^u_{fg}(\mathbf{x}) < s_{fg} \text{ or } d^v_{fg}(\mathbf{x}) < s_{fg} \\ & \text{or } T^v_f(\mathbf{x}) > T^v_g(\mathbf{x})) \\ 0, & \text{otherwise} \end{cases}$$

where $T^u_f(\mathbf{x})$ is the passage time of flight f at the entry node u of link l, $T^v_f(\mathbf{x})$ is the passage time of flight f at the exit node v of link l.

- Node conflict: If no link conflict is detected, wake-turbulence separation can be guaranteed. However, at the intersection of two successive links, violation of the horizontal separation requirement between any two consecutive aircraft (3 NM in the TMA) may still occur. Therefore, we check that when an aircraft flies over a node, the horizontal separation with other aircraft is maintained. Considering a node n and two aircraft f, g that fly over node n, we consider a disk centered at node n with radius R_n, defined as a *detection zone*. R_n can be simply defined as 3 NM for all nodes. However, it induces some waste of separation and may limit the possibility of finding a good-quality solution. Therefore, we propose a value of R_n that takes into account the worst case of aircraft speeds and intersection angles based on the route map of CDG. Suppose that the following aircraft speed, v_g, is higher than the leading aircraft speed, v_f. They pass consecutively the node n, which is the intersection of two perpendicular links. In our route network, the minimum angle between two consecutive links of one route is larger than 90 degrees. Therefore, the worst case to consider for our node conflict detection is the perpendicular link illustrated on Fig. 3.5. We suppose that the node n is the origin point of our coordinate axes, and that at time 0 flight f arrives at node n. To ensure that aircraft f exits the zone of node n before aircraft g enters, the minimum distance between the two aircraft when the first one is at node n should be $d_{fg}(0) = (1 + v_g/v_f)R_n$. The coordinates of flights f and g as a function of the time t are:

$$\begin{cases} x_f(0) = 0 \\ x_f(t) = v_f t \\ y_f(t) = 0 \end{cases} \tag{3.1}$$

$$\begin{cases} y_g(0) = (1 + v_g/v_f)R_n \\ x_g(t) = 0 \\ y_g(t) = y_g(0) - v_g t \end{cases} \tag{3.2}$$

and the distance between them at time t is:

$$d_{fg}(t) = \sqrt{(x_g(t) - x_f(t))^2 + (y_g(t) - y_f(t))^2} \tag{3.3}$$

After the deviation, the time at which this is minimal is:

$$t_{min} = \frac{(1 + v_g/v_f)v_n}{v_g^2 + v_f^2} \tag{3.4}$$

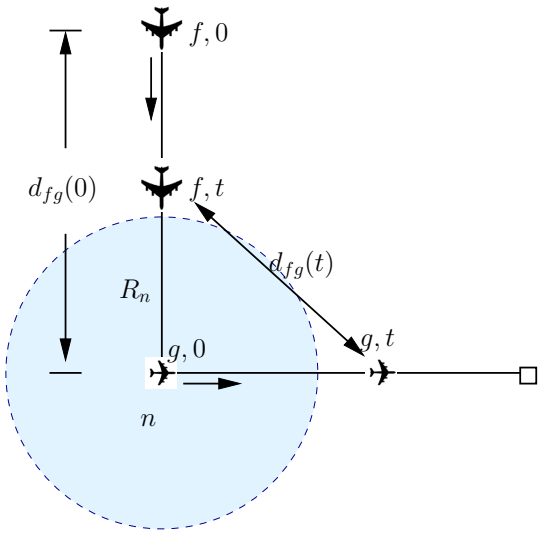


Figure 3.5: Aircraft minimum distance calculation for node conflict.

The corresponding distance, is:

$$d_{fg}(t_{min}) = \frac{(v_g + v_f)R_n}{\sqrt{v_g^2 + v_f^2}}, \tag{3.5}$$

should be larger than 3NM. We consider the extreme case of aircraft speed in our problem: $v_f = 250$kt and $v_g = 430$kt, which corresponds to the minimum and maximum speeds in our data. After calculation, we obtain a radius of $R_n = 2.2$NM.

We must ensure that at every moment only one aircraft passes the detection zone of node. Suppose that aircraft f enters the zone of node n before aircraft g. We denote the entering time to and exit time from this zone for aircraft f (g, respectively) as $T_{\text{In}}^{f,n}(\mathbf{x})$ ($T_{\text{In}}^{g,n}(\mathbf{x})$) and $T_{\text{Out}}^{f,n}(\mathbf{x})$ ($T_{\text{Out}}^{g,n}(\mathbf{x})$). A conflict is detected when $T_{\text{In}}^{g,n}(\mathbf{x}) < T_{\text{Out}}^{f,n}(\mathbf{x})$, which means that aircraft g enters the detection zone before aircraft f exits.

We define the *node conflict indicator* for aircraft f (leading) and g (following) as follows:

$$N_{fg}^{n}(\mathbf{x}) = \begin{cases} 1, & \text{if } T_{\text{In}}^{f,n}(\mathbf{x}) \leq T_{\text{In}}^{g,n}(\mathbf{x}) < T_{\text{Out}}^{f,n}(\mathbf{x}) \\ 0, & \text{otherwise} \end{cases}$$

Runway conflict evaluation

The landing/take-off time difference of any two consecutive aircraft must respect the time separation. The runway separation rules are calculated by incorporating the different flight speeds and their impact on the final approach segment. Here, the separation requirements are shown in Table 3.2, where A refers to Arrival, D refers to Departure, and C refers to Crossing. Due to the runway configuration in CDG, arrivals have to cross departure runways to get to the terminal. We consider that the crossing time of an arrival is 40 seconds.

One runway can be modeled as a specific resource with capacity 1. During high traffic demand periods, the upcoming flights may violate the separation rules and cause runway congestions.

Table 3.2: Single-runway separation requirements according to aircraft categories and to operations (in seconds). A refers to Arrival, D refers to Departure, and C refers to Crossing. H refers to Heavy, M refers to Medium, and L refers to Light. For example, the minimum runway separation between an "A-H" (Arrival-Heavy) and a "D-M" (Departure-Medium) is 60 seconds.

Operation-Category		**Trailing Aircraft**						
		A-H	***A-M***	***A-L***	***D-H***	***D-M***	***D-L***	***C***
Leading Aircraft	***A-H***	96	157	207	60	60	60	-
	A-M	60	69	123	60	60	60	-
	A-L	60	69	82	60	60	60	-
	D-H	60	60	60	96	120	120	60
	D-M	60	60	60	60	60	60	60
	D-L	60	60	60	60	60	60	60
	C	-	-	-	40	40	40	10

Therefore, we note the number of times that the separation is violated and the duration of separation violation for all pairs of aircraft as an indicator for our runway evaluation.

We define the *runway conflict indicator* between two successive aircraft f and g as:

$$P_{fg}(\mathbf{x}) = \begin{cases} 1, & \text{if } 0 \leq T_g^r(\mathbf{x}) - T_f^r(\mathbf{x}) < t_{fg} \\ 0, & \text{otherwise} \end{cases}$$

where $T_f^r(\mathbf{x})$ denotes the time at which aircraft f arrives at the runway threshold, and t_{fg} is the minimum runway separation between flights f and g as shown in Table 3.2.

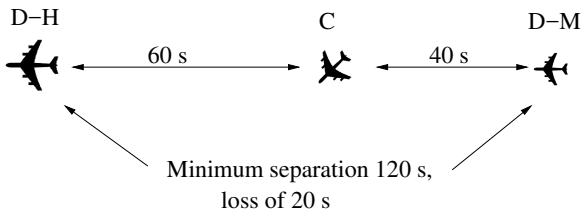


Figure 3.6: Loss of separation in the case "Departure-Heavy (D-H) – Crossing (C)– Departure-Medium (D-M)". The minimum separation between "D-H" and "C" ("C" and "D-M") is 60 (40) seconds respectively. However, the minimum separation between "D-H" and "D-M" is 120 seconds, thus it induces 20 seconds loss of separation if only the required separations between successive aircraft are checked.

One particular case must be considered for the departure runway with a sequence of "Heavy Departure-Crossing-Small/Medium Departure". As shown in Fig. 3.6, each pair of successive flights satisfies to the separation requirement, however, loss of separation occurs between the aircraft "D-

H" and the aircraft "D-M". Therefore, besides detecting the minimum separation between any two successive flights, the loss of triangle inequality as shown in Fig. 3.6 must be detected too.

The total number of conflicts with regard to decision vector $\mathbf{x}$ is defined as follows:

$$A(\mathbf{x}) = \sum_{\substack{f,g \in \mathcal{F} \\ f \neq g}} \left(\sum_{n \in r_f \cap r_g} N_{fg}^n(\mathbf{x}) + \sum_{l \in r_f \cap r_g} L_{fg}^l(\mathbf{x}) + P_{fg}(\mathbf{x}) \right)$$

The TMA separation and the runway separation are ensured by

$$A(\mathbf{x}) = 0$$

3.2.4 Objective function

Our objective function is a weighted sum of the overloads for terminal and for taxi network and flight delays.

- Terminal and taxiway congestion evaluation:

  We have two metrics to measure the terminal congestion. First, the maximum overload number is the maximum value over the period of the difference between the number of aircraft in the terminal and the given terminal capacity. This metric gives us an idea of the time at which severe congestion occurs. However, the maximal overload does not provide sufficient information on the level of congestion. Therefore, another important metric to consider is the average congestion.

  Suppose that we have a discretized time window $\mathcal{T} = \{1, 2, ..., |\mathcal{T}|\}$, let us define the *occupancy indicator* for $t_i \in \mathcal{T}$:

  $$O_m(t_i) = \text{Card}\{f | T_{\text{In}}^{f,m}(\mathbf{x}) \leq t_i \leq T_{\text{Out}}^{f,m}(\mathbf{x})\}$$

  where $T_{\text{In}}^{f,m}(\mathbf{x})$ and $T_{\text{Out}}^{f,m}(\mathbf{x})$ correspond to the entering time and the exit time of resource m (i.e., terminal t or taxi network n). It counts the number of aircraft at time t_i. The overload of resource m at time t_i is then defined as:

  $$G_m(t_i) = \max\{O_m(t_i) - O_m, 0\}$$

  where O_m is the imposed maximum capacity of the resource m.

  The average overload is then defined as $\frac{\sum_{t_i \in \mathcal{T}} G_m(t_i)}{|\mathcal{T}|}$.

  To conclude, the airside capacity overload is expressed as

  $$S(\mathbf{x}) = \frac{\sum_{t_i \in \mathcal{T}} G_t(t_i)}{|\mathcal{T}|} + \max_{t_i \in \mathcal{T}} G_t(t_i) + \frac{\sum_{t_i \in \mathcal{T}} G_n(t_i)}{|\mathcal{T}|} + \max_{t_i \in \mathcal{T}} G_n(t_i)$$

  where $G_t(t_i)$ and $G_n(t_i)$ are respectively the terminal overload and the taxi network overload at time t_i.

  Let us consider a simple example to show how we propose to measure the terminal congestion level. As illustrated in Fig. 3.7, suppose that we have one terminal with three gates (i.e., the capacity $O_t = 3$), and 5 flights turnaround in this terminal during a period of two hours,

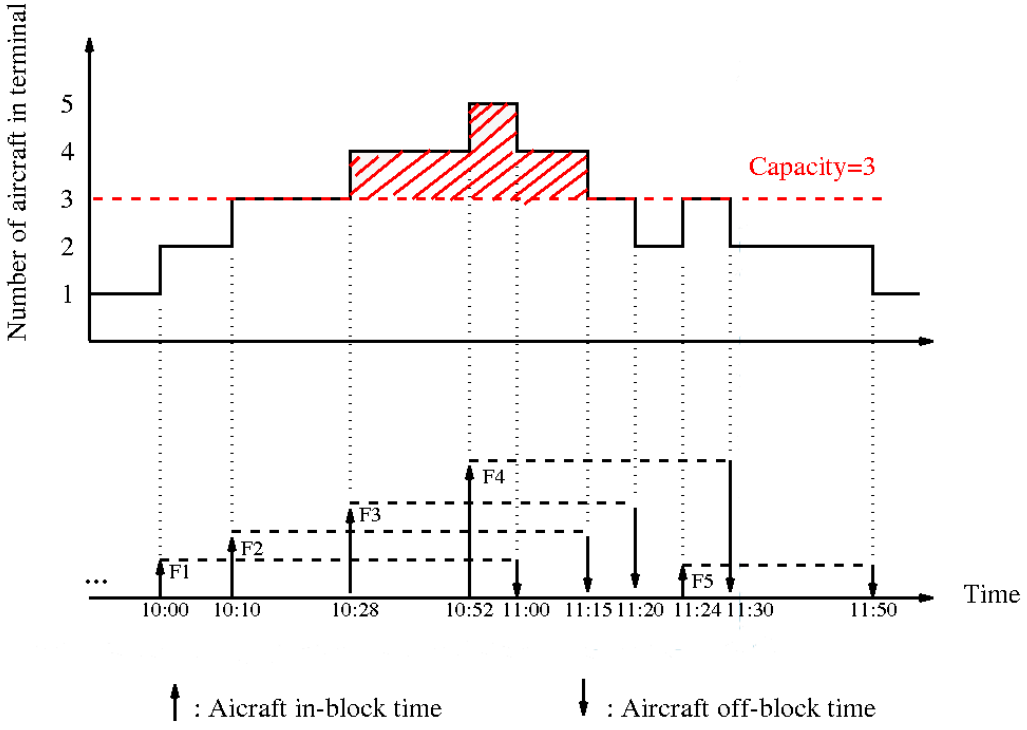


Figure 3.7: Example of terminal congestion evaluation. Five aircraft turnaround in a terminal with a maximum capacity O_t of 3. The congestion time period is shown in red area.

$\mathcal{T} = \{10:00, 10:01, ..., 12:00\}$. The upward (respectively, downward) arrow represents the in-block (off-block) time of one aircraft, linked by a dotted line. We count the cumulated number of aircraft in the terminal as time goes by. Here, the maximum terminal occupancy is 5, therefore the maximum overload $\max_{t_i \in \mathcal{T}} G_t(t_i)$ is 2. We calculate the total overload $\sum_{t_i \in \mathcal{T}} G_t(t_i)$ as well, which is 55 here (the red surface shown in Fig 3.7). The congestion criterion is 2+ 55/120 $\simeq$ 2.458.

- Flight delays: The flight delays $D(\mathbf{x})$ are defined as the total time deviation between the optimized and initial values of RTA and pushback time, $D(\mathbf{x}) = \sum_{f \in \mathcal{F}} (p_f - P_f^0) + \sum_{f \in \mathcal{F}} (t_f - T_f^0)$.

The conflict-avoidance constraint is relaxed into the objective function. Thus, our objective function, to be minimized is therefore a weighted sum of these functions:

$$\gamma_a A(\mathbf{x}) + \gamma_s S(\mathbf{x}) + \gamma_d D(\mathbf{x})$$

where γ_a, γ_s, and γ_d are respectively weighting coefficients for the total number of conflicts in airspace, $A(\mathbf{x})$, the airside capacity overload, $S(\mathbf{x})$, and the flight delays $D(\mathbf{x})$.

3.3 Solution approaches

In this section, we propose a time decomposition approach combined with the SA algorithm to address the integrated terminal airspace management and airport congestion management.

3.3.1 Time sliding-window decomposition approach

The time sliding-window decomposition approach addresses the original problem by decomposition into several sub-problems using a sliding window in order to reduce the problem size and consequently the computational burden. It also enables to optimize decision with fewer uncertainties. This specific approach is generic and can be extended and applied to other real-time operation problems.

Suppose that we are given a total time interval, $[t_{\text{INIT}}, t_{\text{FINAL}}]$, over which we want to optimize. Let us introduce some notations:

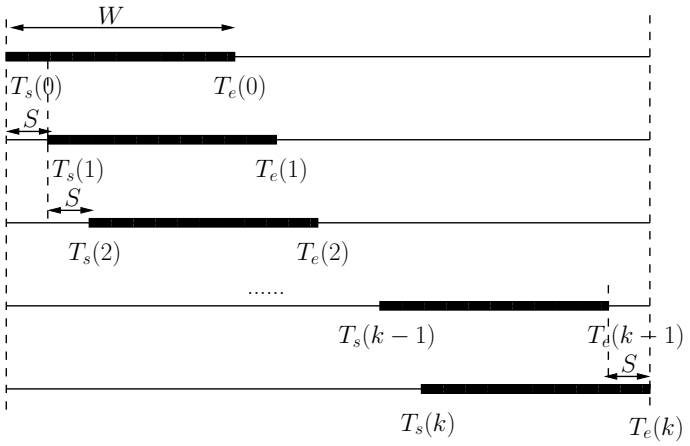


Figure 3.8: Sliding windows from iteration 0 to iteration k with a time length W and a time shift S at each iteration.

- W: the time length of the sliding window;
- S: the time shift of the sliding window at each iteration;
- $T_s(k)$: the starting time of the k^{th} sliding window, $T_s(k) = t_{\text{INIT}} + kS$;
- $T_e(k)$: the ending time of the k^{th} sliding window, $T_e(k) = t_{\text{INIT}} + kS + W$.

Fig. 3.8 illustrates how the operating window slides along the time axis. The first sliding window begins at t_{INIT} and, the optimization algorithm (to be defined later) is applied in the corresponding time interval $[T_s(0), T_e(0)]$. Next, the sliding window is shifted by time S, and the current optimizing interval becomes $[T_s(1), T_e(1)]$. Then, we repeat the process until we reach the k^{th} sliding window such that $T_e(k) \leqslant t_{\text{FINAL}} - S$.

Some parameters are needed to describe the sliding-window approach for each flight $f \in \mathcal{F}$:

- t_s^f: the initial starting time, i.e.,

$$t_s^f = \begin{cases} T_f^0 & \text{if } f \in \mathcal{A} \bigcup \mathcal{AD} \\ P_f^0 & \text{if } f \in \mathcal{D} \end{cases}$$

- $\underline{t_s^f}$: the earliest starting time, i.e.,

$$\underline{t_s^f} = \begin{cases} t_s^f + \Delta T_{\min} & \text{if } f \in \mathcal{A} \bigcup \mathcal{AD} \\ t_s^f & \text{if } f \in \mathcal{D} \end{cases}$$

- $\overline{t_s^f}$: the latest starting time, i.e.,

$$\overline{t_s^f} = \begin{cases} t_s^f + \Delta T_{\max} & \text{if } f \in \mathcal{A} \bigcup \mathcal{AD} \\ t_s^f + \Delta T_{\max}^p & \text{if } f \in \mathcal{D} \end{cases}$$

- t_e^f: the initial ending time, i.e.,
  - For $f \in \mathcal{A}$, it corresponds to the initial in-block time, which is calculated with regard to the initial entry time, the STAR route, the initial entry speed, the average taxi-in duration;
  - For $f \in \mathcal{AD}$, it is the exit time of TMA, calculated with regard to the initial entry time, the STAR route, the initial entry speed, the average taxi-in duration, the turnaround duration, the average taxi-out duration, the take-off time, and the SID route;
  - For $f \in \mathcal{D}$, it is also the exit time of TMA, calculated with regard to the earliest off-block time, the average taxi-out duration, the take-off time, and the SID route.
- $\underline{t_e^f}$: the earliest ending time, i.e.,
  - For $f \in \mathcal{A}$, it corresponds to the earliest in-block time, which is calculated with regard to the earliest entry time in the TMA, the maximum entry speed, STAR route, and the average taxi-in duration;
  - For $f \in \mathcal{AD}$, it is the earliest exit time of TMA, calculated with regard to the STAR route, earliest entry time in the TMA, the maximum entry speed, the average taxi-in duration, the turnaround time, the earliest pushback time, the average taxi-out duration, the take-off time, and the SID route;
  - For $f \in \mathcal{D}$, it is also the earliest exit time of TMA, calculated with regard to the earliest off-block time, the average taxi-out duration, the take-off time, and the SID route.
- $\overline{t_e^f}$: the latest ending time, i.e.,
  - For $f \in \mathcal{A}$, it corresponds to the latest in-block time, which is calculated with regard to the latest entry time in the TMA, the minimum entry speed, the STAR route, and the average taxi-in duration;
  - For $f \in \mathcal{AD}$, it is the latest exit time of TMA, calculated with regard to the STAR route, the latest entry time in the TMA, the minimum entry speed, the average taxi-in duration, the turnaround time, the latest pushback time, the average taxi-out duration, the take-off time, and the SID route;
  - For $f \in \mathcal{D}$, it is also the latest exit time of TMA, calculated with regard to the latest off-block time, the average taxi-out duration, the take-off time, and the SID route.

  Fig. 3.9 gives an overview of the total operations of flight from the entry in the TMA until the exit of this TMA.

Each aircraft is classified with one of the following four statuses, based on the positions of the parameters of flight f relative to the starting and ending times of the current sliding window, k:

- **Completed flight:** $\overline{t_e^f} \leqslant T_s(k)$. The latest ending time for aircraft f, $\overline{t_e^f}$, is lower than the beginning of the k^{th} sliding window, $T_s(k)$, which means that aircraft f has already finished its operation before the start of the k^{th} sliding window;

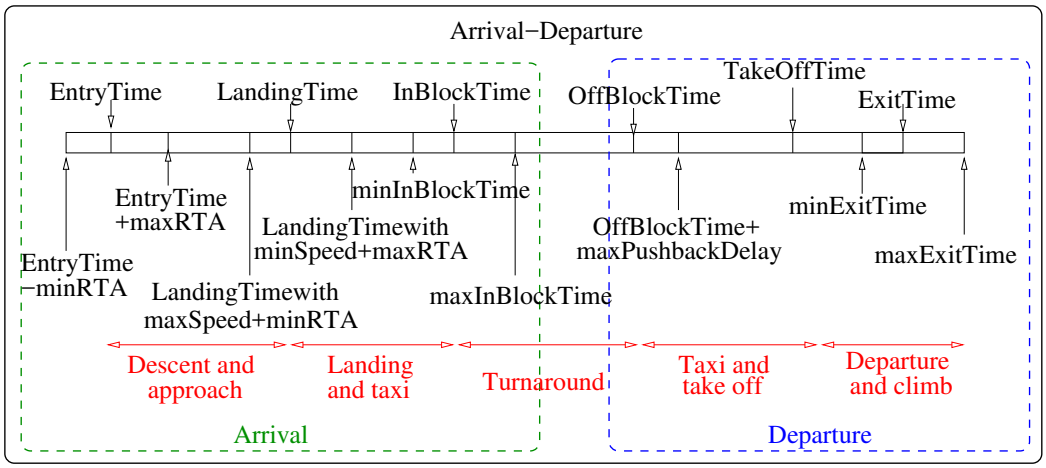


Figure 3.9: Arrival-Departure operations in the TMA. A flight goes through several phases: descent following standard terminal arrival route, land on the runway and taxi to the gate, turnaround, push-back at the gate, taxi between the gate and the runway, take-off and initial climb following standard instrument departure procedure.

- **On-going flight:** $\underline{t_s^f} \leqslant T_s(k) < \overline{t_e^f}$. The beginning time of the k^{th} sliding window, $T_s(k)$, is between the earliest starting time, $\underline{t_s^f}$, and the latest ending time, $\overline{t_e^f}$, which means that aircraft f has already been assigned, but it may still impact the next aircraft in terms of decision variables;

- **Active flight:** $T_s(k) < \underline{t_s^f} \leqslant \overline{t_s^f} \leqslant T_e(k)$. The time decision interval of flight f is included in the sliding window interval $[T_s(k), T_e(k)]$;

- **Planned flight:** $T_e(k) < \overline{t_s^f}$. The latest starting time, $\overline{t_s^f}$, is larger than the ending time of the k^{th} sliding window, $T_e(k)$, which means that the temporal decision variable interval is not totally included in the time window, so that we could not take decision for aircraft f in this interval. The flight will be considered later.

The status of flight f is updated and changed according to the sliding window being considered. Fig. 3.10 illustrates the four different flight statuses and their positions relative to the sliding window. The different time positions of the aircraft and those of the sliding-window are indicated respectively.

At each step, we take into account the active and on-going aircraft in the sliding window interval to be merged and sequenced. Decisions for the on-going flights have already been made, but these flights still have some influence on the decisions to be made for the active flights. On the other hand, the conflicts involving completed flights have already been resolved and they can not have any impact on the active flights, so they can be cleared out of the decision process and ignored. Then, the optimization window is shifted by the time step S. The aircraft statuses are updated, a new set of flights waiting to be addressed are considered, and the optimization process is repeated, as illustrated in Algorithm 2.

3.3.2 Adaptation of SA to the macroscopic airport optimization problem

The fundamental principles of SA have been introduced in Chapter 1. This section describes the adaptation of SA to the problem of integrated air traffic optimization in airports and TMA.

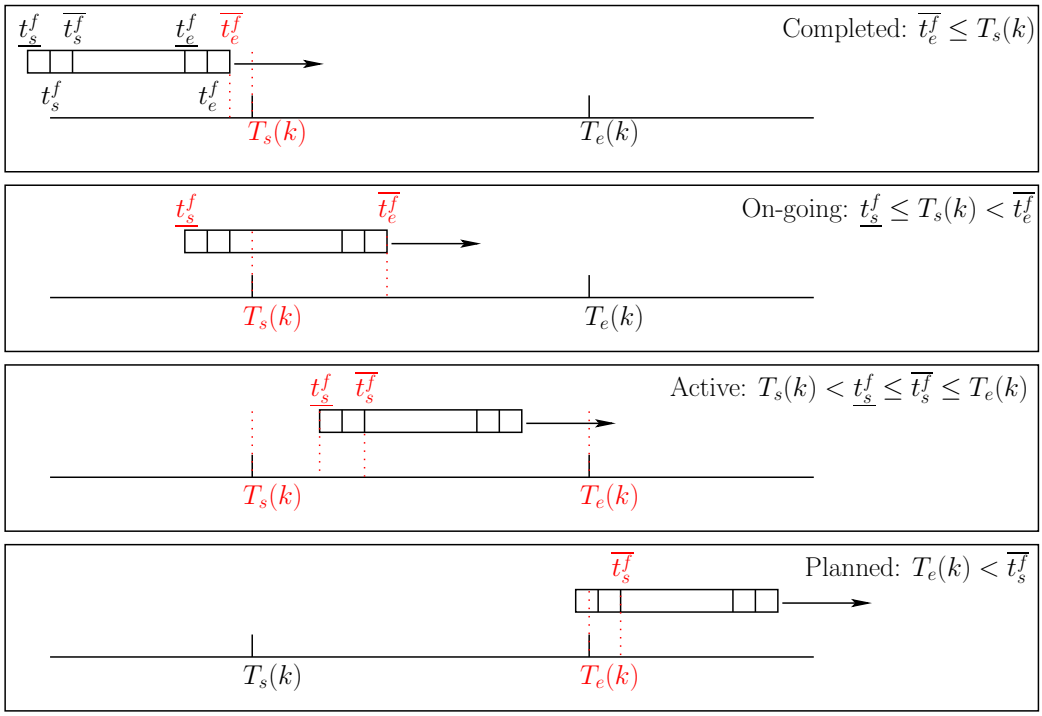


Figure 3.10: Four flights statuses, related to the time position of flight f relative to the current sliding window (k).

Algorithm 2 Sliding-Window Management

procedure SLIDINGWINDOW
  $k \leftarrow 0$;
  $T_s(k) \leftarrow t_{\text{INIT}}$;
  $T_e(k) \leftarrow T_s(k) + W$;
  Determine each flight status relative to sub-window;
  $F_{\text{OPT}} \leftarrow$ Active and on-going flights;
  while $T_e(k) < t_{\text{FINAL}}$ **do**
    if at least one active flight in F_{OPT} **then**
      Subproblem: optimize considering F_{OPT};
    end if
    $T_s(k) \leftarrow T_s(k) + S$;
    $T_e(k) \leftarrow T_e(k) + S$;
    $k \leftarrow k + 1$;
    Update each flight status relative to sub-window;
    Update F_{OPT};
  end while
end procedure

Algorithm 3 Neighborhood function

Require: For each flight f, we record its airspace performance, p_f^a, runway performance, p_f^r, ground performance, p_f^g, the total performance is denoted as $p_f^t = p_f^a + p_f^r + p_f^g$.

The total number of conflicts $P_t = \sum_{f \in \mathcal{F}} p_f^t$;
Generate random number, v = random(0,1);
if $P_t > 0$ **then**
    sum $\leftarrow 0$;
    target $\leftarrow P_t \times v$;
    i $\leftarrow 1$;
    while sum $<$ target **do**
        sum $\leftarrow$ sum+p_i^t;
        $i \leftarrow i+1$;
    end while
else
    Choose randomly one flight i in the flight set;
end if
if $i \in \mathcal{A}$ **then**
    if $p_i^a > 0$ **then**
        Choose with equal probability between the entering time and the entering speed in the TMA, then choose randomly one value between the minimum and the maximum allowed deviation (0 and $\Delta T_{\max}$ for entering time, $V_f^{\min}$ and $V_f^{\max}$ for entering speed);
    else if $p_i^r > 0$ **then**
        Choose with equal probability among the entering time in the TMA,the entering speed in the TMA, and the landing runway;
    else Choose randomly the entering time in $\mathcal{T}_f$;
    end if
else if $i \in \mathcal{D}$ **then**
    if $p_i^g > 0$ **then**
        Choose randomly the pushback time in $\mathcal{P}_f$;
    else
        Choose with equal probability between the pushback time change and the take-off runway change;
    end if
end if

Table 3.3: Empirically-set parameter values of the simulated annealing algorithm with time decomposition approach

Parameter	**Value**
Geometrical temperature reduction coefficient	0.99
Number of iterations at each temperature step	100
Initial rate of accepting degrading solutions	0.15
Final temperature	$10^{-6}T_0$
Time length of the sliding window	2 h
Time shift of the sliding window	0.5 h

The parameters chosen for specifying the resolution algorithm are given in Table 3.3. Moreover, to generate a neighborhood solution, instead of simply choosing randomly a flight f in the active-flight set, we use a method similar to the so-called biased roulette-wheel selection. We note for each aircraft the number of conflicts and the congestion it encounters as its *air* and *ground performance* respectively. Air performance involves link and node conflicts, and ground performance involves runway, taxiway network and terminals congestions. Let us take the example of Fig. 3.7, we note the aircraft which are exposed to overload.

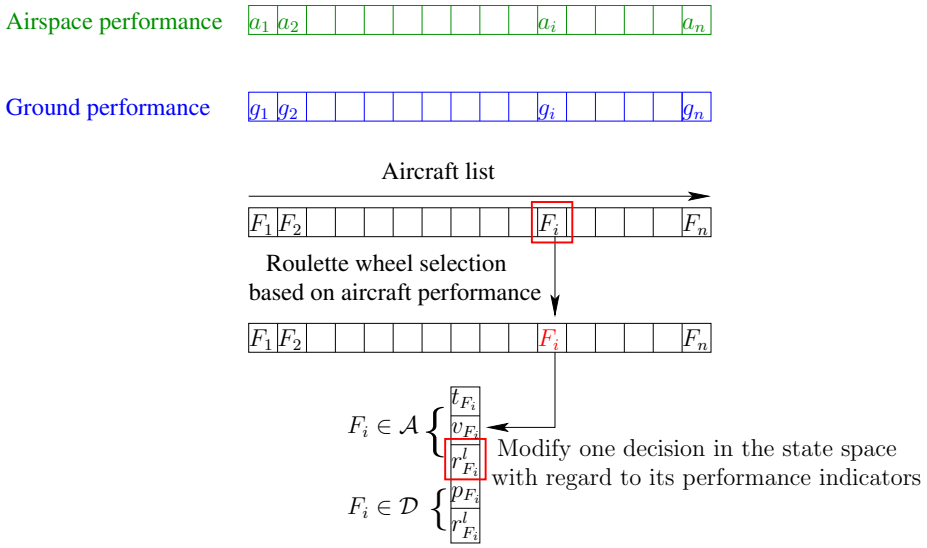


Figure 3.11: Neighborhood generation. We first target one aircraft using a roulette wheel selection based on number of conflicts or aircraft involved in ground congestion. Then, we modify one of its decision variables with regard to the type of aircraft and its performance indicators.

Considering this overload period, it is better to first change the decisions of aircraft which are mostly involved in congestion ($F3$, $F4$) than the ones with less impact ($F5$) in order to mitigate the terminal congestion. In Fig. 3.7, $F1$ and $F2$ are also involved in the congestion period. However, the terminal is still not overloaded when they arrive at the gate. Moreover, our main strategy is to

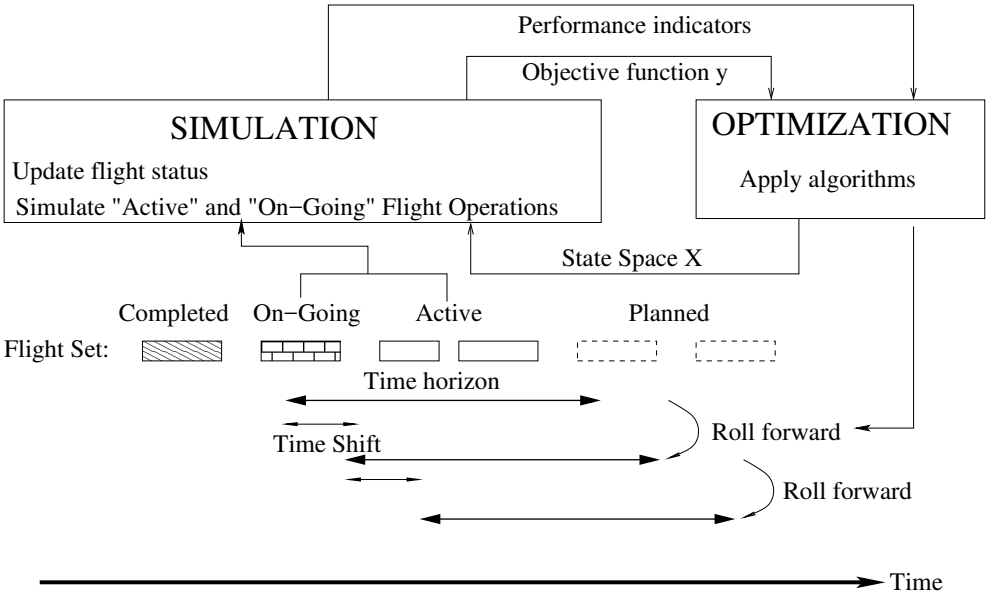


Figure 3.12: Overall simulation-based optimization process.

delay the arrivals to mitigate congestion. Hence, F_3 and F_4 are more critical to be targeted compared to F_1 and F_2. The performance metric can help us to better focus on the fully loaded and congested periods. The fact that our neighborhood definition is based on the air and ground flight performance increases the likelihood that a flight involved in many conflicts, or experiencing severe congestions, will be chosen. As shown in Fig. 3.11, in the neighborhood selection, first, we record different performance indicators for each aircraft. Then, we choose a flight using a roulette wheel selection method based on the conflict performance or congestion performance. Next, we target this flight to decide which decision variable to be changed. Lastly, we randomly choose a value for the related decision variable. To summarize, a detailed description is shown in Algorithm 3.

Fig. 3.12 summarizes the overall optimization process. The simulation process takes the decision proposed by the optimization algorithm and simulates the associated flights in order to produce the objective function and the vector of performances. The objective function and the performance indicators provided by the simulation process guide the optimization module to search for better solution. The time sliding window manager updates flight statuses and puts them into the two previous mentioned modules. The optimization and simulation processes are repeated.

In the next section, we apply the simulated annealing algorithm combined with time decomposition approach to resolve the integrated terminal airspace management problem and airport capacity management problem.

3.4 Results

In this section we present some test problems and analyze the associated results. We have tested our methodology on a four-hour real data case at Paris CDG Airport. Numerical results with different settings of (user-defined) algorithm parameters are presented and discussed. The overall process is run on a 2.50 GHz core i7 CPU, under Linux operating system PC based on Java code.

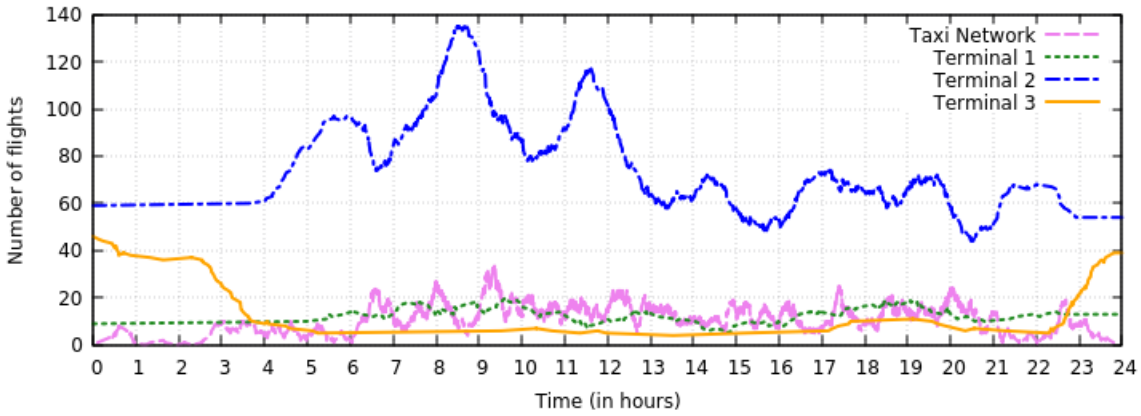


Figure 3.13: Initial terminals and taxi network occupancy on February 18^{th}, 2016. Terminal 2 is the main terminal in CDG and receives much more traffic flows compared to the other two terminals.

Table 3.4: User-defined parameter values specifying the optimization problem

Parameter	**Value**
Discretization time step, ΔT	5 seconds
Discretization speed step, Δ_f^v	$0.01V_f^0$
Maximum delay of RTA at TMA, $\Delta T_{\max}$	30 minutes
Minimum delay of RTA at TMA, $\Delta T_{\min}$	-5 minutes
Maximum pushback delay, $\Delta T_{\max}^p$	15 minutes
Minimum allowable speed, $V_f^{\min}$	$0.9V_f^0$
Maximum allowable speed, $V_f^{\max}$	$1.1V_f^0$
Conflicts weighting coefficient, γ_a	1
Overload weighting coefficient, γ_s	1
Delay weighting coefficient, γ_d	0.001

3.4.1 Real data analysis

A busy winter day on February 18th, 2016 was chosen as our data set. Fig. 3.13 shows the initial terminal and taxi network occupancy over the day, the line colors green, blue, orange, and pink respectively represent Terminal 1, Terminal 2 , Terminal 3 and taxi network occupancy. Terminal 1 consists of a central circular terminal building and seven satellites with boarding gates, thus cannot handle many aircraft and keeps a stable low traffic over the day. Air France operates from Terminal 2, and CDG is the principal hub for Air France (hub airport is used by one airline to concentrate passenger traffic and flight operations at a given airport), thus Terminal 2 is the main terminal of CDG that serves the majority of aircraft. Therefore, we observed much more traffic flows in Terminal 2 than in the other two terminals. Terminal 3 which mainly hosts charter and low-cost airlines, is

mainly composed of hangars for night parking, therefore the departure flights leave the terminal early in the morning and the arrival flights come late at night, forming the curve in orange color shown in Fig. 3.13. Peak hour with a maximum gate occupancy was reached between 8 am and 10 am in Terminal 2. Then, the terminal occupancy decreased sharply, which consecutively leaded to a peak in the taxi network. Here, we extracted the flight data of the most dense time period in the day from 6 am to 10 am as our test case. A total of 332 flights were operated at CDG, including 177 departures and 155 arrivals, 109 flights were arrival-departures. We have in total 67 Heavy and 265 Medium aircraft. The fleet mix ratio in this period is 20% for Heavy aircraft and 80% for Medium aircraft. The parameters chosen for specifying the optimization problem are given in Table 3.4.

We tackle the integrated airport and TMA optimization problem at a macroscopic level, the aim is to show that the proposed algorithm can react in the right direction facing airport capacity reduction. Due to the lack of data, we cannot apply our method to a historic situation. Moreover, directly comparing the optimized results with the historic situation would somehow be difficult, due to the simplifications and assumptions from the model. In this chapter, we build the initial occupancy curve by simulating the process using the initial flight data (initial entering time, initial entering speed, initial pushback time, etc.). We use this curve as our baseline case, and impose a reasonable capacity limit, which is fair to compare with the results from the optimization process.

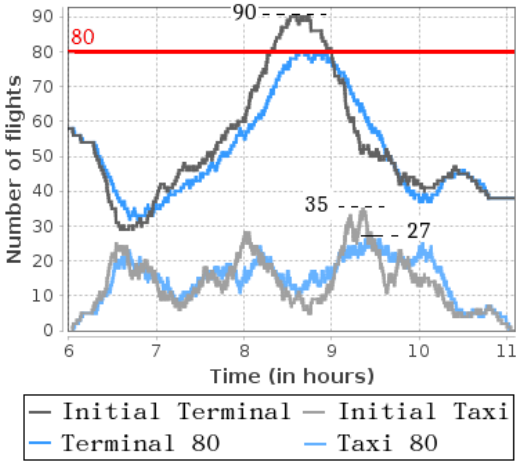
Two major aspects are discussed in the rest of the section: First, different levels of degradations of the terminal capacity and taxi network capacity are imposed to verify the impact on flight delays and on other airport components. Second, we studied the benefits of runway assignment on reducing flight delays in peak hour when two runways are facing imbalanced throughput.

3.4.2 Influence of reduced airport capacity on flight delays

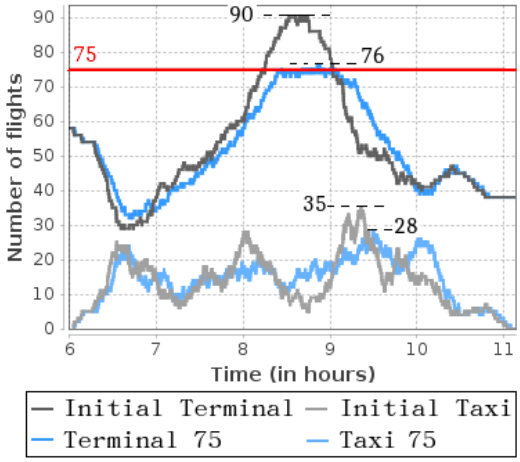
First, we investigated how the different levels of degradation of the terminal occupancy and taxi network occupancy would influence the traffic. Two capacity parameters, the imposed maximum terminal capacity, O_t, and the imposed maximum taxi network capacity, O_n, were defined to investigate the airport congestion problem. In the case of terminal overload, we chose to study the traffic on Terminal 2, because it is much more occupied than the other two terminals and has an important peak hour.

As shown in Fig. 3.14, the dark gray line and the light gray line respectively represented the initial terminal occupancy and the initial taxi network occupancy. The initial maximum gate occupancy for this period is 90. Therefore, we chose O_t=80, 75, and 70 respectively. We first set a threshold of O_t=80. After running the algorithm, the maximum capacity is reduced and kept below the threshold as illustrated in Fig. 3.14a. A decrease of the taxi network occupancy was observed as well. Then, we decreased the capacity to O_t=75, in a short period the traffic exceeded this threshold as shown in Fig. 3.14b (76 instead of 75). When the imposed capacity continued decreasing to O_t=70, for which we encountered a bottleneck and the maximum capacity could not be reduced anymore (78 instead of 70). This was due to the inherent maximum allowed RTA delays that we can change. To make a further test, we set the maximum RTA change, $\Delta T_{\max}$, to be 60 minutes instead of 30 minutes. After launching the algorithm, as shown in Fig. 3.14d, this terminal overload is totally absorbed with the cost of more than 20 aircraft whose time deviations were more than 30 minutes as shown in Fig. 3.15a. We also observed that the RTA distribution shifted to the right as O_t decreases, while pushback delay were not influenced significantly as illustrated in Fig. 3.15b. This is coherent because in order to release terminal congestion, the algorithm has to reduce the arrivals (by delaying RTA decisions) and increase the departures (no pushback delays).

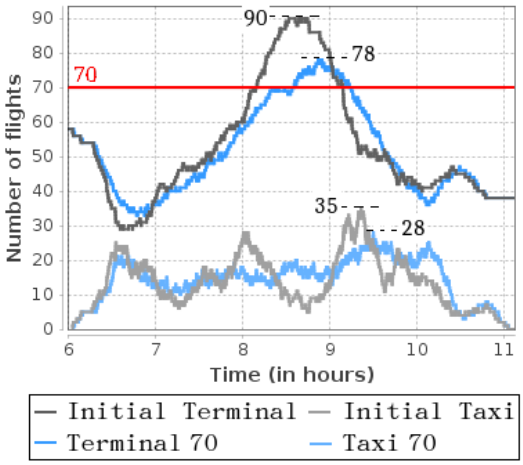
Similarly, the imposed capacity was applied to taxi network. As shown in Fig. 3.16, the initial maximum taxi network occupancy for this period was 35. We set a threshold of O_n= 25, 20 and



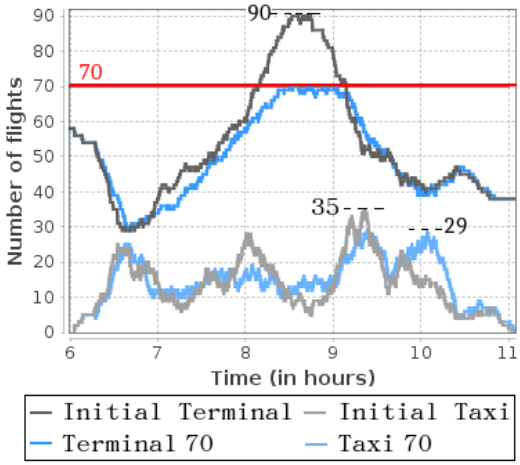
(a) O_t = 80



(b) O_t = 75



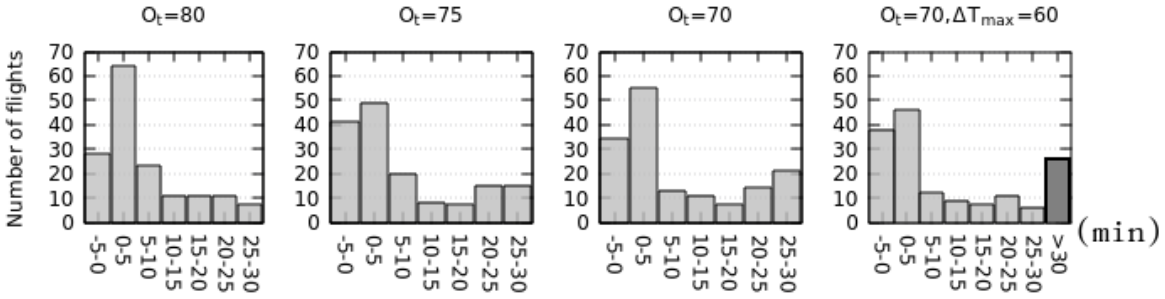
(c) O_t = 70



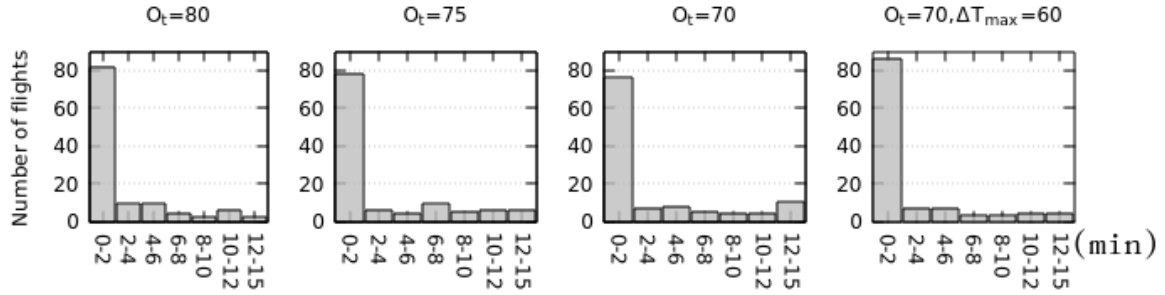
(d) O_t = 70, $\Delta T_{\max}$=60 min

Figure 3.14: Maximum terminal capacity tests, with O_t=80, 75, 70, $\Delta T_{\max}$=30 min, and O_t=70, $\Delta T_{\max}$=60 min respectively. Comparison of initial occupancy and optimized occupancy for terminal and taxi network.

15 to launch three tests separately. In Fig. 3.16, the dark blue line and the light blue line represent the optimized terminal occupancy and taxi network occupancy respectively. In Fig. 3.16a, O_n=25 was easily reached for the whole period after optimization, also a decrease of maximum terminal occupancy was observed, even when we did not put any constraints on O_t. A sharp increase or decrease of gate occupancy would consecutively increase taxi network capacity as well. As our strategy was to delay the aircraft arrival, the curve was shifted to the right compared to the initial occupancy curve. With O_n=20 in Fig. 3.16b, we could see that the traffic overload cannot be absorbed, there was still a maximum taxi network occupancy of 22 around 10 am. With $O_n = 15$ in Fig. 3.16c, the limited flight delays cannot absorb the taxi occupancy either, and the maximum value remained almost the same as with $O_n = 20$. To make a further test, we set the maximum RTA change, $\Delta T_{\max}$, to be 60 minutes instead of 30 minutes. After launching the algorithm, as shown in Fig. 3.16d, this taxi network overload can not be absorbed either, in contrast to the terminal overload. This is because only one congestion period was found in the terminal occupancy, while



(a) RTA delay comparison for arrivals in case O_t=80, 75, 70, $\Delta T_{\max}$=30 min, and O_t=70, $\Delta T_{\max}$=60 min respectively. The distribution shifts to the right when O_t decreases. In the fourth case, the dark gray histogram represents the number of flights whose time deviations are greater than 30 minutes.



(b) Pushback delay comparison for departures in case O_t=80, 75, 70, $\Delta T_{\max}$=30 min, and O_t=70, $\Delta T_{\max}$=60 min respectively.

Figure 3.15: Decisions comparison for different maximum terminal capacities O_t.

taxi network encountered several levels of congestion in different time periods, which were more difficult to mitigate under a certain threshold. When we took a look at the decision changes in Fig. 3.17, the RTA delay and pushback delay distribution shifted to the right when O_n decreased, which indicated that there were more delays for both arrivals and departures. In such case, the algorithm kept aircraft as much as possible at the gate and slowed down arriving aircraft in order to reduce the number of aircraft on the taxi network. Limited capacity of the taxi network caused more flight delays.

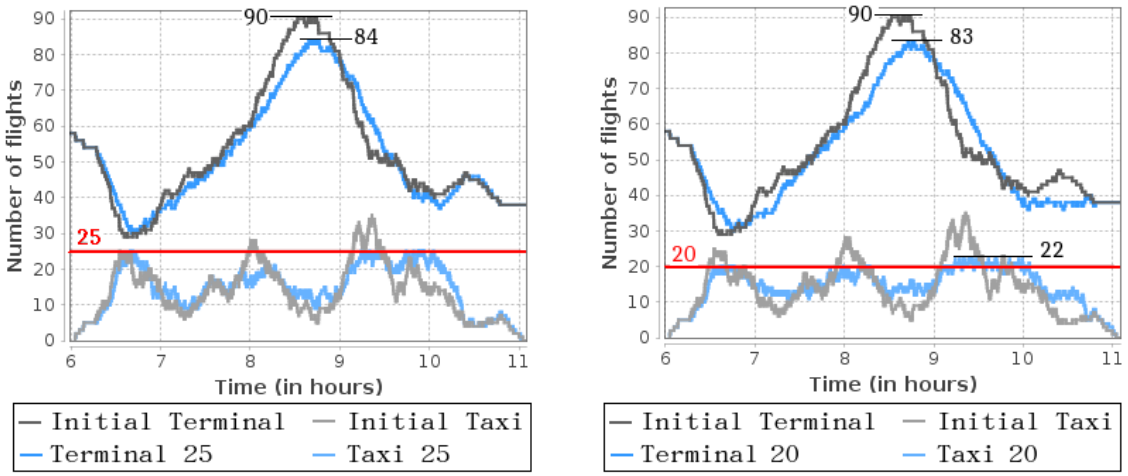
3.4.3 Influence of runway assignment on flight delays

We investigated the benefits of arrival runway assignment and departure runway assignment on reducing flight delays in peak hour when two runways are facing imbalanced throughput.

Paris TMA arrival routes use a four-corner procedure as shown in Fig. 3.1 in Section 3.2. In Table 3.5, southern flows from OKIPA and BANOX mainly use the southern landing runway 26L. Northern flows from MOPAR use more 26L as well, flows from LORNI land more on the Northern runway 27R. Moreover, the flows coming from South sometimes land on the Northern runway, and vice versa. In practice, landing runway changes can be achieved by controllers' tactical vectoring. The departure runway changes are related to a more detailed level of ground operations, i.e., how alternate taxi routes are assigned.

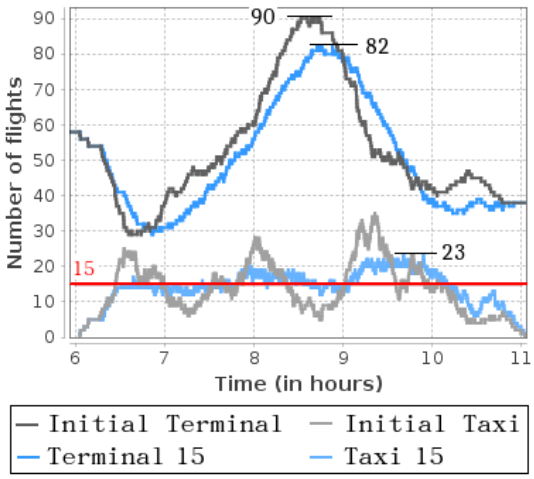
First, we want to investigate how runway changes can bring benefits to reduce flight delays. We set three cases:

- Case 1: Both landing runway and take-off runway are decision variables;

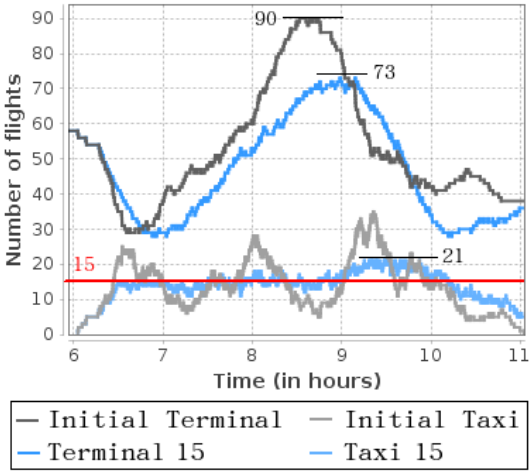


(a) O_n = 25.

(b) O_n = 20



(c) O_n = 15

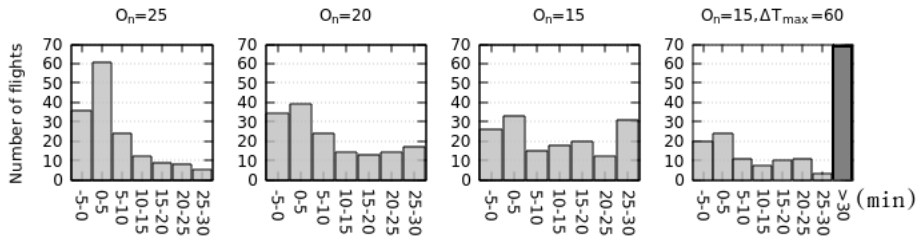


(d) O_n = 15, $\Delta T_{\max}$=60 min

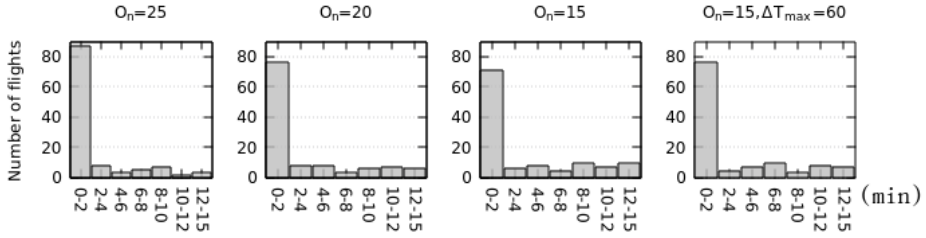
Figure 3.16: Maximum taxi network capacity tests, with O_n=25, 20, 15,$\Delta T_{\max}$=30 min, and O_n=15, $\Delta T_{\max}$=60 min respectively. Comparison of initial occupancy and optimized occupancy for terminal and taxi network.

- Case 2: Take-off runway is a decision variable, landing runway is predefined and fixed;
- Case 3: Landing runway is a decision variable, take-off runway is predefined and fixed.

Fig. 3.18 gave an example of one sliding window optimization evolution; it showed the value of two criteria (number of conflict and total delays) at the end of each temperature step during the cooling process of SA. Solid lines represented the number of conflicts and dashed lines denoted the total delays in minutes. The number of conflicts in Case 1 converged faster than the other two cases. Case 1 and Case 2 reached conflict-free solution almost at the same time, while in Case 3 conflict-free solution can not be found. Seven SID conflicts with in total 160 seconds duration still remained, due to the fact that once the take-off runway was fixed, one can only adjust pushback time to resolve conflicts, thus it is more difficult to find a feasible solution in the given pushback delay period. This result showed that take-off runway assignment did not only balance runway throughput, but may



(a) RTA delay comparison for arrivals in case O_n=25, 20, 15 respectively. The distribution shifts to the right when O_n decreases.



(b) Pushback delay comparison for departures in case O_n=25, 20, 15 respectively. The number of flights whose pushback delay decision change is lower than 2 minutes decreases when O_n decreases.

Figure 3.17: Decisions comparison for different maximum taxi network capacity O_n

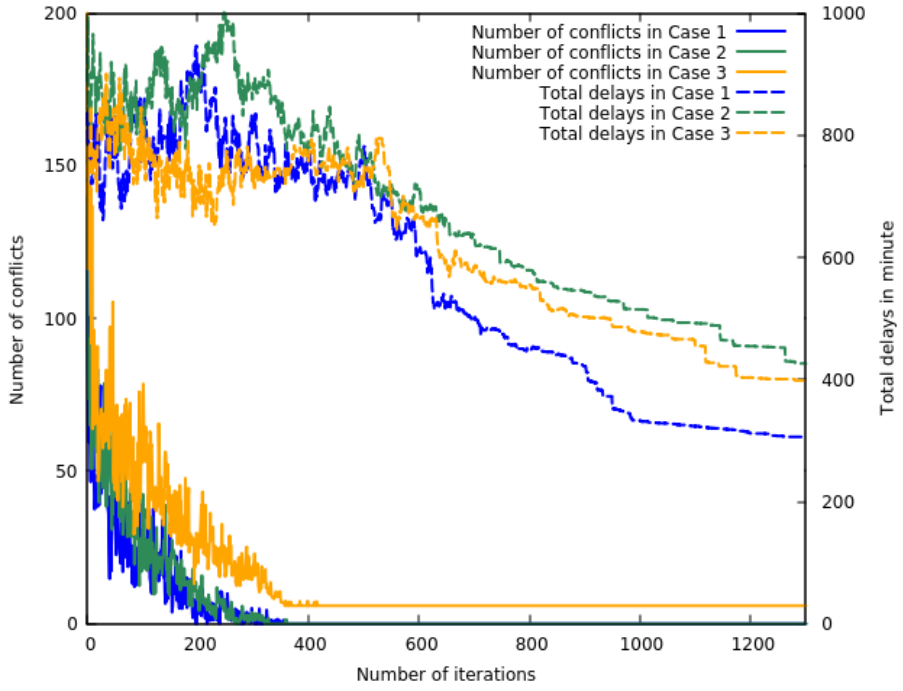
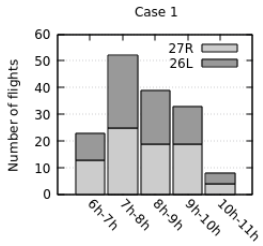


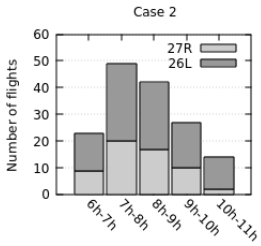
Figure 3.18: Evolution of the two criteria for different runway decisions

also reduce controllers' potential maneuvers in peak hour for the SID airspace. As for total delays, before conflict-free solutions were found, delay criteria, for three cases, stayed at a high level to offer more possibilities for the algorithm to search in the state space to establish first a conflict free solution. Then, the delay criterion started decreasing. We can observe that Case 1 reached the lowest

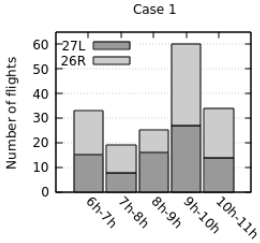
Table 3.5: Landing traffic flow distribution with regard to flight entry point in the TMA and landing runway on February 18^{th}, 2016.

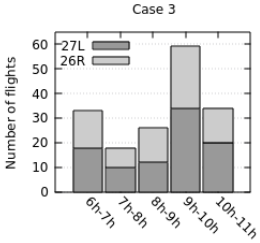
	27R	26L	Total
MOPAR	39 (6%)	77 (13%)	116 (19%)
LORNI	131 (21%)	73 (12%)	204 (33%)
OKIPA	32 (5%)	165 (27%)	197 (32%)
BANOX	15 (3%)	80 (13%)	95 (16%)
Total	217 (35%)	395 (65%)	612 (100%)





(a) Landing throughput for runway 27R and 26L. In Case 1, landing runway is assigned; In Case 2, initial landing runway is used.





(b) Take-off throughput for runway 27L and 26R. In Case 1, take-off runway is assigned; In Case 3, initial take-off runway is used.

Figure 3.19: Landing runways (27R and 26L) throughput comparison for Case 1 and Case 2, and take-off runways (27L and 26R) throughput comparison for Case 1 and Case 3.

delay, while Case 2 remained the highest.

Fig. 3.19 showed the runway throughput for landings and take-offs. The period 7 am–8 am corresponded to the higher landing throughput, then after the turnaround process, the period 9 am–10 am corresponded to the higher departure throughput. We observed a more balanced traffic for each runway in Case 1 without reducing the throughput. Fig. 3.20 showed the distribution of flights RTA and pushback delays. In Case 1, a total of 102 arrival flights (66%) modified their RTA within 5 minutes. While in Case 2 and 3, more RTA delays were required compared to Case 1, only 46% of the flights RTA was less than 5 minutes, and 54% in Case 3. Without the take-off runway changes, much more pushback delays were requested. Pushback delay did not change as significantly as RTA delay, because we had a low demand between 6 am and 9 am, the major departure flows occurred between 9 am and 10 am. Regarding the total RTA delay and pushback delay, as shown in Table 3.6, a decrease of 37 % (from 1425 minutes to 897 minutes) RTA delay was reached for Case 1 compared to Case 2. A decrease of 36 % (from 696 minutes to 443 minutes) pushback delay was reached for Case 1 compared to Case 3.

The average computational time of our optimization algorithm was 13 minutes for a total of 10 sliding windows. As shown in Table 3.7, the average CPU time for each window was less than 2 minutes, which is very promising for tackling such a complicated problem in practice.

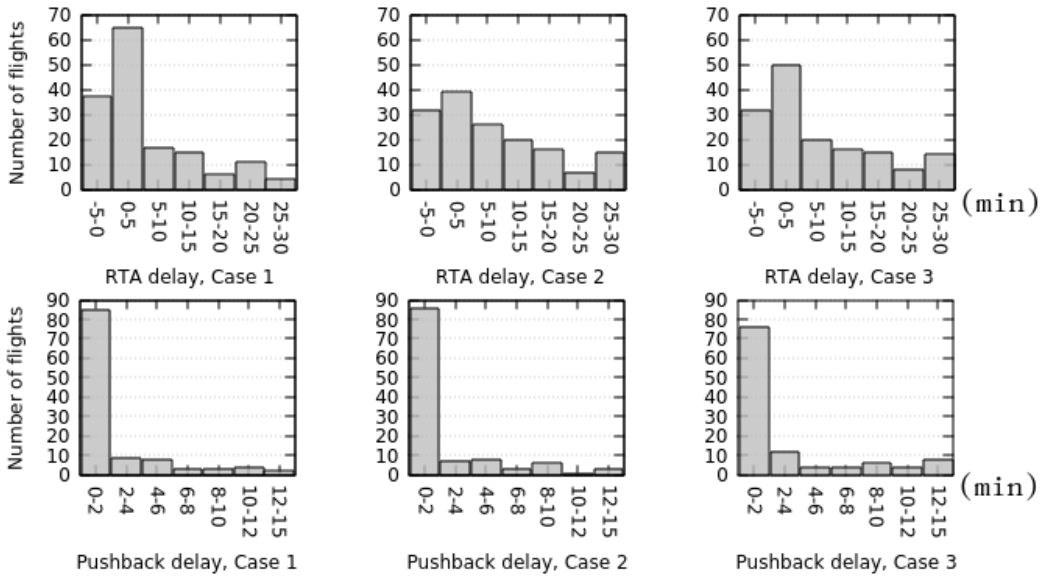


Figure 3.20: RTA and pushback delay decision changes distribution with take-off and landing runway assignment (Case 1), with only landing runway assignment (Case 2), and with only take-off runway assignment (Case 3).

Table 3.6: Total RTA delay and pushback delay comparison.

	Total RTA delay (in minutes)	Total pushback delay (in minutes)
Case 1	897	443
Case 2	1425	501
Case 3	1293	696

3.5 Conclusions

In this chapter, we proposed a model to manage the arrival, surface and departure problems at the macroscopic level to address the connected airport and terminal airspace management problems. The objective was to resolve conflicts in the terminal airspace, to reduce airside capacity overload, and to reduce flight delays. First, we proposed a TMA route network structure and a high level airport abstraction model. Then, a time sliding-window approach combined with a simulated annealing algorithm was applied to solve the problem. The approach was tested in the case of Paris CDG airport and showed some potential benefits: First, reduced terminal capacity until a certain threshold was efficiently mitigated by RTA and pushback time changes. When the imposed capacity was more reduced, the overload could not be mitigated anymore, and the airport could not absorb more demand without imposing delays out of the maximum range. Similarly to terminal occupancy, a decrease of the maximum taxi network capacity could be mitigated by delaying arrivals. Second, landing runway assignment and take-off runway assignment in peak hour with imbalanced runway throughputs could significantly reduce the flight delays. Moreover, the conflicts in the airspace could be resolved also, which may imply that the runway change did not create much more controller's

Table 3.7: Computational time for various problem sizes. The total number of flights are the sum of type “Active” and “On-going”.

Time period	All	Dep.	Arr.	Medium	Heavy	Run time (s)
6:00-8:00	164	101	63	134 (82%)	30 (18%)	51
6:30-8:30	238	137	101	202 (85%)	36 (15%)	85
7:00-9:00	234	123	111	194 (83%)	40 (17%)	88
7:30-9:30	241	119	122	196 (81%)	45 (19%)	93
8:00-10:00	266	127	139	212 (80%)	54 (20%)	103
8:30-10:30	259	121	138	200 (77%)	59 (23%)	104
9:00-11:00	229	106	123	175 (76%)	54 (24%)	85

workload. In future work, the model will be improved to on-line planning taking into account uncertainties. The uncertainty of aircraft arrival time will increase as time passes by, thus a more robust occupancy curve is built and evaluated.

Chapter 4

Microscopic optimization of air traffic at airports

In this chapter, we present a methodology to address the problem of airport ground operations at a microscopic level. In this part, we represent the airport (gate, taxiway, runway) with a detailed surface node-link network, and we consider individual aircraft trajectories based on this graph. We aim at resolving the ground conflicts among aircraft, assigning the pushback times, the taxi speeds and the positions (runway threshold or holding point) and the holding times. The optimization model is designed to reduce runway queue length and minimize flight delays as well as taxi times with respect to safety concerns in surface traffic operations. A comparison of the microscopic optimization benefits with the baseline scenarios is presented for two major airports: Paris Charles De-Gaulle airport (CDG) in Europe and Charlotte Douglas International airport (CLT) in the U.S.

The rest of this chapter is structured as follows. Section 4.1 models the airport ground operation problem. Section 4.2 presents the solution approach. Section 4.3 and Section 4.4 perform tests and analyze respectively the results in the case of CDG and CLT. Section 4.5 gives some discussions of the two airports and we end with some conclusions in Section 4.6.

4.1 Mathematical model

In this section, we describe a trajectory-based optimization model for the airport ground operations problem. We first introduce input data. Next, decision variables are defined. Then, we clarify constraints. Lastly, the objective function is expressed.

4.1.1 Input data

In the trajectory-based approach, the airport surface is represented as a graph network $\mathcal{G} = (\mathcal{N}, \mathcal{L})$. Node $n \in \mathcal{N}$ corresponds to one of the following: runway entry/exit points, holding points, intersections of taxiways or gates. The holding point refers to the runway threshold for departures and runway crossing point for arrivals. Link $l \in \mathcal{L}$ connects two adjacent nodes. Fig. 4.1 gives a simplified example of airport node-link network. For the arrival taxi path, aircraft start taxing from the runway exit point in green color, pass the holding point in red color, follow a set of intermediate nodes in gray color, and finally reaches the gate in blue color. For departure taxi path, aircraft start taxing from the gate, follow a set of intermediate nodes in gray color, wait at the holding point in red color, and end at the runway entry point in green color and take off.

The following notations are defined for our problem formulation:

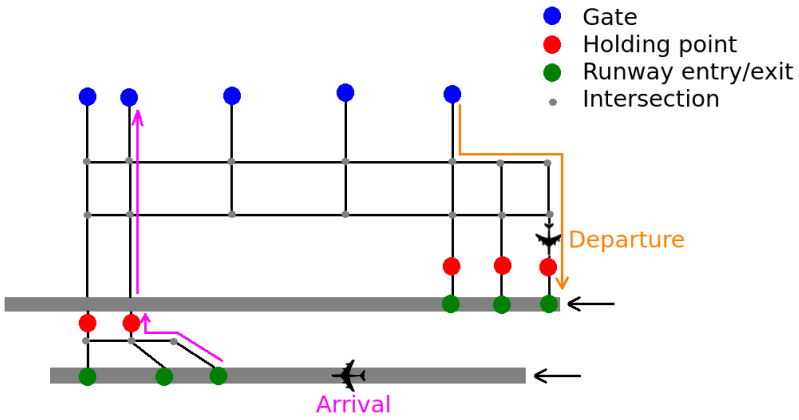


Figure 4.1: Simplified node-link network example.

- $\mathcal{F}$: set of flights, $\mathcal{F} = \mathcal{A} \bigcup \mathcal{D}$;
- $\mathcal{A}$: set of arrivals;
- $\mathcal{D}$: set of departures;
- $\mathcal{R}$: set of landing or take-off runways;
- $\mathcal{H}$: set of holding points at runway threshold;
- $r_f \in \mathcal{R}$: landing or take-off runway for flight f;
- $h_f \in \mathcal{H}$: holding point for flight f;
- γ_f: taxi route followed by flight f containing a set of nodes, including the gate, runway, holding point, and intersection nodes;
- I_f: initial off-block time for departure or landing time for arrival. Note that the landing time of each arrival is predetermined and is not optimized in this study;
- $s_{f,g}$: minimum separation time at runway for two successive flights f and g if f uses the runway before g with respect to their wake turbulence categories. The specific values can be found in Appendix A;
- $V^{\max}$: maximum allowed taxi speed;
- $V^{\min}$: minimum allowed taxi speed;
- N^a : maximum allowed number of holding time slots for arrivals;
- N^d : maximum allowed number of holding time slots for departures;
- N_p : maximum allowed number of pushback delay time slots;
- C^a: maximum capacity at holding point for arrivals;
- C^d: maximum capacity at holding point for departures;

- s: minimum taxi separation;
- Δt: time step;
- Δv: speed increment.

4.1.2 Decision variables

In order to optimize the ground movement, we now consider several potential control points as decisions. For each flight $f \in \mathcal{F}$, the decision variables are defined as follows:

- p_f: Pushback time for departures, discretized into time slots, $p_f \in \{I_f, I_f + \Delta t, I_f + 2.\Delta t, ..., I_f + N_p.\Delta t\}$;
- w_f: Holding time (waiting time at runway threshold for departures and time spent in runway crossing queues for arrivals), discretized into time slots Δt, $w_f \in \{0, \Delta t, 2.\Delta t, ..., N.\Delta t\}$;
- v_f: Taxi speeds, $v_f \in \{V^{\min}, V^{\min} + \Delta v, V^{\min} + 2\Delta v, ..., V^{\max}\}$;

Moreover, the following auxiliary variables are introduced:

- t_f^u: Runway usage time, which is the take-off time for departures and the runway crossing time for arrivals, where

$$t_f^u = \begin{cases} p_f + (\text{total taxi distance})/v_f + w_f & \text{if } f \in \mathcal{D}, \\ I_f + (\text{taxi distance from runway exit to holding point})/v_f + w_f & \text{if } f \in \mathcal{A}; \end{cases}$$

- t_f^c: Completion time for flight f, where

$$t_f^c = \begin{cases} t_f^u & \text{if } f \in \mathcal{D}, \\ t_f^u + (\text{taxi distance from holding point to gate})/v_f & \text{if } f \in \mathcal{A}; \end{cases}$$

To summarize, our decision vector is $\mathbf{x} = (\mathbf{p}, \mathbf{w}, \mathbf{v})$, where $\mathbf{p}$ is the pushback time vector, $\mathbf{w}$ is the holding time vector, and $\mathbf{v}$ is the taxi speed vector.

4.1.3 Constraints

Variables domains

The decision variables of each flight must satisfy the following constraints: the maximum holding time for arrivals and for departures, and the maximum pushback delay are specified by Constraints 4.1, Constraints 4.2, and Constraints 4.3, respectively. Constraints 4.4 define the possible range of taxi speed, when the aircraft is not stopped at holding point.

$$0 \leq w_f \leq N^a.\Delta t, \qquad \forall f \in \mathcal{A}, \tag{4.1}$$

$$0 \leq w_f \leq N^d.\Delta t, \qquad \forall f \in \mathcal{D}, \tag{4.2}$$

$$I_f \leq p_f \leq I_f + N_p.\Delta t, \qquad \forall f \in \mathcal{D}, \tag{4.3}$$

$$V^{\min} \leq v_f \leq V^{\max}, \qquad \forall f \in \mathcal{F}, \tag{4.4}$$

Runway separation

In order to introduce the runway separation constraints, we define the following parameter to represent infeasible assignments of runway usage times. For any two distinct flights $f, g \in \mathcal{F}$ using runway R, we introduce:

$$C^R_{fg}(\mathbf{x}) = \begin{cases} 1 & \text{if } (t^u_g - t^u_f < s_{fg} \text{ or } t^u_f - t^u_g < s_{gf}) \text{ and } r_f = r_g, \\ 0 & \text{otherwise;} \end{cases}$$

which is equal to 1 if f and g use the same runway and the separation is not respected. Then, the minimum runway separation requirement is guaranteed by Constraints 4.5.

$$\sum_{(f,g)\in\mathcal{F}\times\mathcal{F}, f\neq g,} C^R_{fg}(\mathbf{x}) = 0, \tag{4.5}$$

Holding point constraint

For any two distinct flights $f, g \in \mathcal{F}$, we introduce:

$$C^H_{fg}(\mathbf{x}) = \begin{cases} 1 & \text{if } ((t^u_g - w_g < t^u_f - w_f \text{ and } t^u_g > t^u_f) \text{ or } (t^u_f - w_f < t^u_g - w_g \text{ and } t^u_f > t^u_g)) \text{ and } h_f = h_g, \\ 0 & \text{otherwise;} \end{cases}$$

which is equal to 1 if the first-come-first-served order is not respected. Then, the FCFS order is ensured at the holding point by Constraints 4.6,

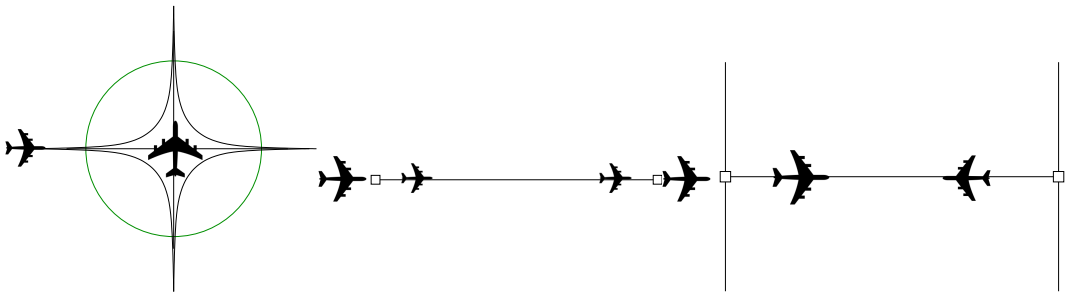
$$\sum_{(f,g)\in\mathcal{F}\times\mathcal{F}, f\neq g,} C^H_{fg}(\mathbf{x}) = 0, \tag{4.6}$$

Moreover, at each holding point, we define a holding capacity indicator for $h \in \mathcal{H}$ at time t as follows:

$$O_{h,t}(\mathbf{x}) = \max\{\text{Card}\{f | h_f = h \text{ and } t^u_f - w_f \leq t \leq t^u_f\} - C, 0\}, \tag{4.7}$$

where $C = C^a$ for arrivals and $C = C^d$ for departures. Let $\mathcal{T} = 1, 2, ..., |\mathcal{T}|$ be the discretized time steps. Then, Constraints 4.8 ensure that the number of aircraft waiting at the holding point does not exceed a maximum specified limit. For arrivals, the holding limit depends on the airport layout. For departures, it is an ATC-defined parameter (runway pressure).

$$O_{h,t}(\mathbf{x}) = 0, \forall h \in \mathcal{H}, \forall t \in \mathcal{T}, \tag{4.8}$$



(a) Node separation (b) Link overtaking (c) Bi-link head-on

Figure 4.2: Taxi separation based on node-link graph.

Taxi separation constraint

Based on the route network structure, ground conflicts may happen only at nodes and on links. In order to ensure the minimum taxi separation, we define three types of conflicts as shown in Fig. 4.2: node conflict, link overtaking conflict, and bi-link head-on conflict. Moreover, we define:

- $C_{\mathcal{N}}(\mathbf{x})$ - the total number of conflicts on nodes;
- $C_{\mathcal{L}}(\mathbf{x})$ - the total number of over-taking conflicts on links;
- $C_{\mathcal{B}}(\mathbf{x})$ - the total number of head-on conflicts on links.

Given the taxi route and chosen values of the decision variables, we can calculate the times at which the aircraft passes its associated nodes and links as shown in Fig. 4.3, that are used to calculate the number of the above-defined three types of conflict. Thus, first we perform the preprocessing to get the passage time on nodes and links as described in Algorithm 4. Then, we use Algorithms 5, 6, and 7 to calculate $C_{\mathcal{N}}(\mathbf{x})$, $C_{\mathcal{L}}(\mathbf{x})$, and $C_{\mathcal{B}}(\mathbf{x})$, respectively.

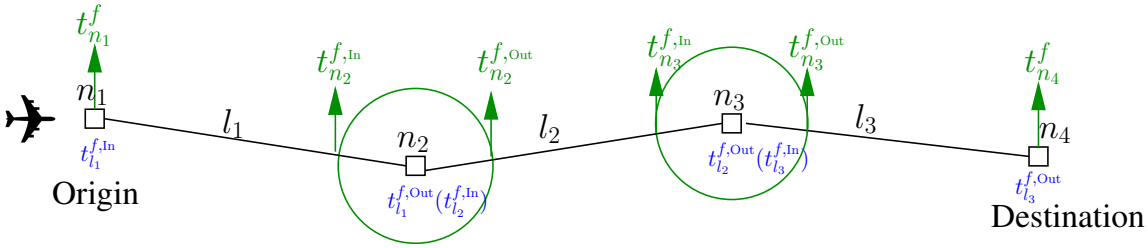


Figure 4.3: Preprocessing for recording flight passage times at nodes and at links.

Node conflict

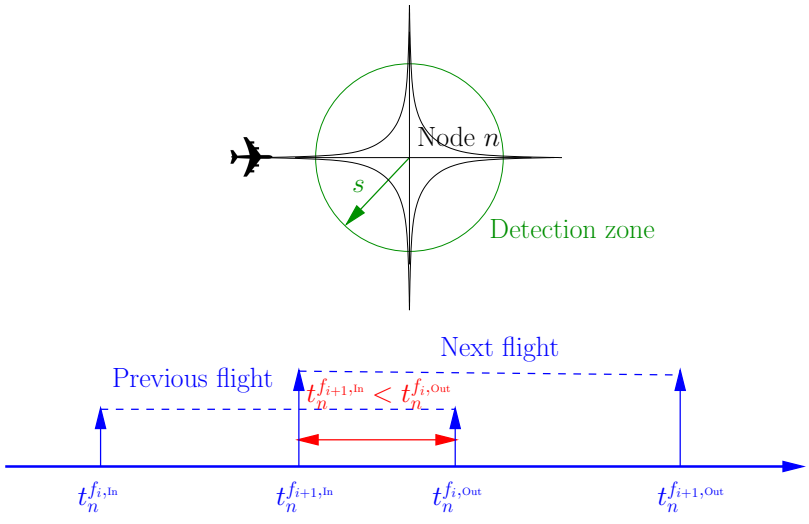


Figure 4.4: Conflict at a node.

The node conflicts are computed as described in Algorithm 5: the passage time of each flight through a node is determined. A conflict is detected if the separation time between two successive

Algorithm 4 Preprocessing for computing the number of ground conflicts

Input: $\mathcal{G} = (\mathcal{N}, \mathcal{L})$ - airport graph;
For each flight f, FP_f is a set of flight paths containing the sequence of nodes $n \in \mathcal{N}$ and links $l \in \mathcal{L}$ passed by flight f.

Output: $\{T_n\}_{n\in\mathcal{N}}$, where T_n records the entry/exit times at which flights pass the detection zone of node n;
$\{T_l^{\text{In}}\}_{l\in\mathcal{L}}$, where T_l^{In} records the times at which flights pass over the entry of l;
$\{T_l^{\text{Out}}\}_{l\in\mathcal{L}}$, where T_l^{Out} records the times at which flights pass over the exit of l;
$\{F_l\}_{l\in\mathcal{L}}$, where F_l records the flights that use the forward direction to pass l;
$\{B_l\}_{l\in\mathcal{L}}$, where B_l records the flights that use the backward direction to pass l.

1: **for** each $n \in \mathcal{N}$ **do** ▷ for each node n of the graph
2: $\quad \text{T}_n := \emptyset$
3: **end for**
4: **for** each $l \in \mathcal{L}$ **do** ▷ for each link l of the graph
5: $\quad \text{T}_l^{\text{In}} := \emptyset$
6: $\quad \text{T}_l^{\text{Out}} := \emptyset$
7: $\quad \text{F}_l := \emptyset$
8: $\quad \text{B}_l := \emptyset$
9: **end for**
10:
11: **for** $f \in \mathcal{F}$ **do**
12: $\quad$ **for** each $n \in \text{FP}_f$ **do**
13: $\quad\quad t_n^{f,\text{In}} := \text{ENTER\_DETECTION\_ZONE}(n)$ ▷ compute the entry time to the detection zone of n
14: $\quad\quad t_n^{f,\text{Out}} := \text{EXIT\_DETECTION\_ZONE}(n)$ ▷ compute the exit time from the detection zone of n
15: $\quad\quad \text{T}_n := \text{T}_n \cup (t_n^{f,\text{In}}, t_n^{f,\text{Out}})$ ▷ record flight f at node n
16: $\quad$ **end for**
17: $\quad$ **for** each $l \in \text{FP}_f$ **do**
18: $\quad\quad t_l^{f,\text{In}} := \text{ENTRY\_TIME}(l)$ ▷ compute the time when f enters l
19: $\quad\quad t_l^{f,\text{Out}} := \text{EXIT\_TIME}(l)$ ▷ compute the time when f exits l
20: $\quad\quad d_l^f := \text{DIRECTION}(l)$ ▷ record the direction
21: $\quad\quad \text{T}_l^{\text{In}} := \text{T}_l^{\text{In}} \cup \{t_l^{f,\text{In}}\}$ ▷ record flight f at link l
22: $\quad\quad \text{T}_l^{\text{Out}} := \text{T}_l^{\text{Out}} \cup \{t_l^{f,\text{Out}}\}$
23: $\quad\quad$ **if** $d_l^f :=$ forward **then**
24: $\quad\quad\quad F_l := F_l \cup f$ ▷ use link in forward direction
25: $\quad\quad$ **else**
26: $\quad\quad\quad B_l := B_l \cup f$ ▷ use link in backward direction
27: $\quad\quad$ **end if**
28: $\quad$ **end for**
29: **end for**

aircraft using the same node is less than the minimum separation time. Considering a node n and two aircraft f_i, f_{i+1} that pass over node n as shown in Fig. 4.4, we consider a disk centered at node n with a radius s, defined as a *detection zone*. s is defined as the safe separation distance on the ground. We must ensure that at every time only one aircraft passes this detection zone. Suppose that aircraft f_i enters the zone of node n before aircraft f_{i+1}. We denote the entering time to and exit time from this zone for aircraft f_i (f_{i+1}, respectively) as $t_n^{f_i,\text{In}}$ ($t_n^{f_{i+1},\text{In}}$) and $t_n^{f_i,\text{Out}}$ ($t_n^{f_i,\text{Out}}$). A node conflict is detected when $t_n^{f_{i+1},\text{In}} < t_n^{f_i,\text{Out}}$, which means that aircraft f_{i+1} enters the detection zone before aircraft f_i exits.

Link overtaking conflict

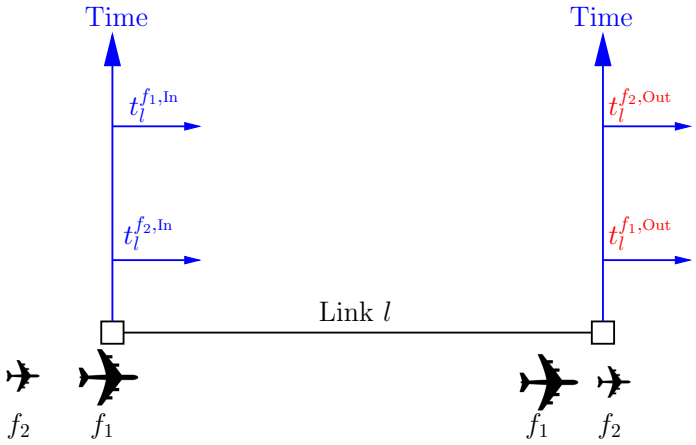


Figure 4.5: Overtaking conflict at a link.

For each link, the entry and exit time of flights passing through the link are compared to check if the order of aircraft that entered the link differs from the exit order. The number of over-taking conflicts is the rank difference between entry and exit order. For example, consider two flights [f_1, f_2] passing link l in the forward direction as shown in Fig. 4.5, the order at the link entry is [f_1, f_2], and the order at the link exit becomes [f_2, f_1]. An over-taking conflict is detected due to the swapped order of f_1 and f_2. Algorithm 6 resumes the procedure.

Bi-link head-on conflict

One link might be occupied by several successive flights in the same direction, but it can never be used by two aircraft from opposite direction at the same time. A head-on conflict occurs when the exit time of an aircraft using a link is earlier than the entry time of another aircraft using the same link heading in the opposite direction. For each link $l = (u, v)$, we define the forward direction as from u to v, and the backward direction as from v to u. Thus, given the taxi path of each flight, we know the flight's traveling direction on the link. Suppose that flight f_1 (f_2) travels in the forward (backward) direction on link l as shown in Fig. 4.6, and flight f_1 enters l earlier than f_2. A head-on conflict is detected when the link exit time of f_1 ($t_l^{f_1,\text{Out}}$) is larger than the link entry time of f_2 ($t_l^{f_2,\text{In}}$). The total number of bi-link head-on conflicts is calculated following Algorithm 7.

The last constraints 4.9 ensure that there is no conflict on the ground.

$$C_{\mathcal{T}}(\mathbf{x}) = 0, \tag{4.9}$$

Algorithm 5 Node conflicts computation.

Input: $\mathcal{N}$ - set of nodes;
$\{T_n\}_{n\in\mathcal{N}}$, where T_n records the times at which flights pass over node n;
Output: $C_{\mathcal{N}}$ - total number of conflicts on nodes.

$C_{\mathcal{N}} := 0$
for each $n \in \mathcal{N}$ **do** ▷ for each node n of the graph
  $m := |T_n|/2$ ▷ number of flights encountered for n
  $[(t_n^{f_1,\text{In}}, t_n^{f_1,\text{Out}}), ..., (t_n^{f_m,\text{In}}, t_n^{f_m,\text{Out}})] := \text{SORT}_{t^{\text{In}}}(T_n)$ ▷ sort flights according to t^{In}
  for $i := 1$ to $m-1$ **do** ▷ for each pair of consecutive flights in the sorted sequence
    if $t_n^{f_i,\text{Out}} > t_n^{f_{i+1},\text{In}}$ **then** ▷ $i+1$ enters the detection zone before i exits
      $C_{\mathcal{N}} := C_{\mathcal{N}} + 1$ ▷ number of conflicts is increased
    end if
  end for
end for

Algorithm 6 Overtaking link conflicts computation.

Input: $\mathcal{L}$ - set of links;
$\{T_l^{\text{In}}\}_{l\in\mathcal{L}}$, where T_l^{In} records the times at which flights pass over the entry of l;
$\{T_l^{\text{Out}}\}_{l\in\mathcal{L}}$, where T_l^{Out} records the times at which flights pass over the exit of l;
$\{F_l\}_{l\in\mathcal{L}}$, where F_l records the flights that use the forward direction to pass l;
$\{B_l\}_{l\in\mathcal{L}}$, where B_l records the flights that use the backward direction to pass l.
Output: $C_{\mathcal{L}}$ - total number of over-taking conflicts on links.

$C_{\mathcal{L}} := 0$
for each $l \in \mathcal{L}$ **do** ▷ for each link l of the graph
  $m_1 := |F_l|$ ▷ number of flights encountered for l in the forward direction
  $[t_l^{f_1,\text{In}}, ..., t_l^{f_{m_1},\text{In}}] := \text{SORT}_{t^{\text{In}}}(T_l^{\text{In}}|_{f\in F_l})$ ▷ sort flights in F_l according to t^{In}
  $[t_l^{g_1,\text{Out}}, ..., t_l^{g_{m_1},\text{Out}}] := \text{SORT}_{t^{\text{Out}}}(T_l^{\text{Out}}|_{g\in F_l)}$ ▷ sort flights in F_l according to t^{Out}
  $m_2 := |B_l|$ ▷ number of flights encountered for l in the backward direction
  $[t_l^{\bar{f}_1,\text{In}}, ..., t_l^{\bar{f}_{m_2},\text{In}}] := \text{SORT}_{t^{\text{In}}}(T_l^{\text{In}}|_{f\in B_l})$ ▷ sort flights in B_l according to t^{In}
  $[t_l^{\bar{g}_1,\text{Out}}, ..., t_l^{\bar{g}_{m_2},\text{Out}}] := \text{SORT}_{t^{\text{Out}}}(T_l^{\text{Out}}|_{g\in B_l)}$ ▷ sort flights in B_l according to t^{Out}
  for $i := 1$ to m_1 **do** ▷ forward direction
    if $f_i \neq g_i$ **then** ▷ entry/exit orders swap
      $C_{\mathcal{L}} := C_{\mathcal{L}} + 1$ ▷ number of conflicts is increased
    end if
  end for
  for $i := 1$ to m_2 **do** ▷ backward direction
    if $\bar{f}_i \neq \bar{g}_i$ **then** ▷ entry/exit orders swap
      $C_{\mathcal{L}} := C_{\mathcal{L}} + 1$ ▷ number of conflicts is increased
    end if
  end for
end for

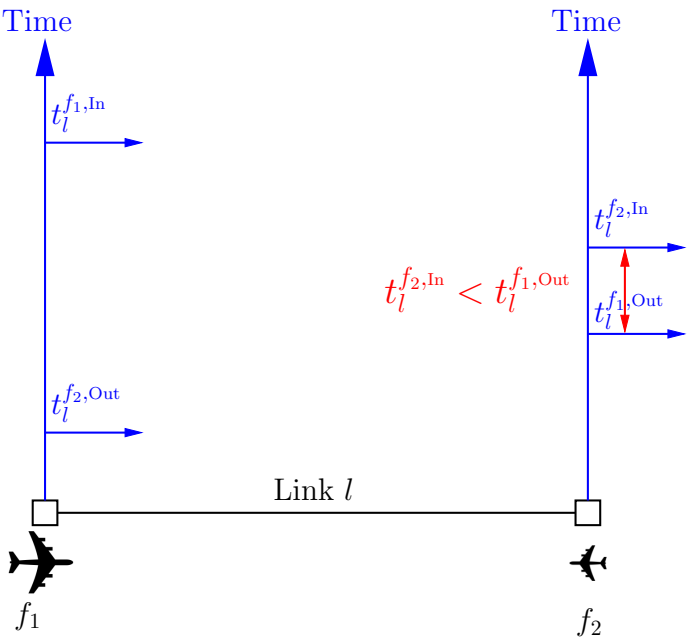


Figure 4.6: Bi-link head-on conflict at a link.

Algorithm 7 Bi-link head-on conflicts computation.

Input: $\mathcal{L}$ - set of links;
$\{T_l^{\text{In}}\}_{l\in\mathcal{L}}$, where T_l^{In} records the times at which flights pass over the entry of l;
$\{T_l^{\text{Out}}\}_{l\in\mathcal{L}}$, where T_l^{Out} records the times at which flights pass over the exit of l;
$\{F_l\}_{l\in\mathcal{L}}$, where F_l records the flights that use the forward direction to pass l;
$\{B_l\}_{l\in\mathcal{L}}$, where B_l records the flights that use the backward direction to pass l.
Output: $C_{\mathcal{B}}$ - total number of bi-link head-on conflicts.

$C_{\mathcal{B}} := 0$
for each $l \in \mathcal{L}$ **do** ▷ for each link l of the graph
  $m := |T_l^{\text{In}}|$ ▷ number of flights encountered for l in both directions (forward and backward)
  $[(t_l^{f_1,\text{In}}, t_l^{f_1,\text{Out}}), \ldots, (t_l^{f_m,\text{In}}, t_l^{f_m,\text{Out}})] := \text{SORT}_{t^{\text{In}}}(T_l^{\text{In}}, T_l^{\text{Out}})$ ▷ sort flights according to t^{In}
  for $i := 1$ to $m-1$ **do** ▷ for each pair of consecutive flights in the sorted sequence
    if $((f_i \in \mathcal{B}_l$ and $f_{i+1} \in \mathcal{F}_l)$ or $(f_{i+1} \in \mathcal{B}_l$ and $f_i \in \mathcal{F}_l))$ and $(t_l^{f_i,\text{Out}} > t_l^{f_{i+1},\text{In}})$ **then**
      $C_{\mathcal{B}} := C_{\mathcal{B}} + 1$ ▷ flights from opposite directions use l at the same time
    end if
  end for
end for

where

$$C_{\mathcal{T}}(\mathbf{x}) = C_{\mathcal{N}}(\mathbf{x}) + C_{\mathcal{L}}(\mathbf{x}) + C_{\mathcal{B}}(\mathbf{x}), \tag{4.10}$$

4.1.4 Objective function

The objective function to minimize is:

$$\alpha\Phi_p(\mathbf{x}) + \beta\Phi_d(\mathbf{x}) + \gamma\Phi_a(\mathbf{x})$$

where α, β and γ are appropriate weighting coefficients,

- $\Phi_p(\mathbf{x}) = \sum_{f\in\mathcal{D}} (p_f - I_f)/|\mathcal{D}|$: Average pushback delay for departures;
- $\Phi_d(\mathbf{x}) = \sum_{f\in\mathcal{D}} (t_f^c - p_f)/|\mathcal{D}|$: Average taxi time for departures;
- $\Phi_a(\mathbf{x}) = \sum_{f\in\mathcal{A}} (t_f^c - I_f)/|\mathcal{A}|$: Average taxi time for arrivals.

Note that the conflict-avoidance constraints are addressed by performing the following relaxation to the objective function,

$$\Phi_c(\mathbf{x}) + \theta(\alpha\Phi_p(\mathbf{x}) + \beta\Phi_d(\mathbf{x}) + \gamma\Phi_a(\mathbf{x}))$$

where $\Phi_c(\mathbf{x}) = C_{\mathcal{T}}(\mathbf{x}) + \sum_{\substack{f,g\in\mathcal{F}\\ f\neq g}} (C_{fg}^H(\mathbf{x}) + C_{fg}^R(\mathbf{x})) + \sum_{h\in\mathcal{H}}\sum_{t\in\mathcal{T}} O_{h,t}(\mathbf{x})$, and θ is a weighting coefficient, set to a small value to make sure that the conflicts are the most important criteria to minimize. Once a conflict-free solution is reached, the system continues to minimize other criteria.

.

4.2 Adaptation of SA to the microscopic airport optimization problem

The optimization problem is solved using an adapted simulated annealing (SA) algorithm. SA is adapted to solve our problem in the following way:

- **Neighborhood function:** generates a local change to the current solution. First, one has to determine which flight's decision variables needs modification. A flight is picked using a so-called biased roulette wheel selection method based on total number of flight conflicts. Flights with higher number of conflicts are more likely to be chosen. Then, we decide which decision variable of this flight is to be changed with regard to the conflict type in which it is involved. Lastly, we choose randomly a discretized value for the related decision variable. Algorithm 8 describes the neighborhood selection process for conflict resolution. Once a conflict-free solution is reached, we change our strategy, to target aircraft for decision changes and try to decrease the flight delays as described in Algorithm 9.
- **Cooling process:** the temperature is decreased via a geometrical law given by $T_i = 0.99T_{i-1}$, where T_i is the temperature at step i. The number of iterations at each temperature step is 100.
- **Stopping criterion:** SA terminates execution either if the pre-defined final temperature is reached (set to be $10^{-9}T_0$, where T_0 is a user-defined initial temperature), or if the solution quality is not improved after 50 temperature steps.

4.3 Case study: Paris CDG airport

We test our methodology on real data from Paris CDG Airport. Numerical results with different settings of (user-defined) algorithm parameters are presented and discussed. This section first describes the ground operations at CDG by highlighting the airport movements and taxi times as well as runway queue length. After that, we present the optimization set-up with some assumptions made in order to simplify the problem. Lastly, we discuss the results consisting of the benefits of gate holding and the departure queue management.

Algorithm 8 Neighborhood function for resolving conflicts.

Require: For each flight f, we record its take-off performance, p_f^d, crossing performance, p_f^c, and ground performance, p_f^g, the total performance is denoted as $p_f^t = p_f^d + p_f^c + p_f^g$.

The total number of conflicts $P_t = \sum_{f \in \mathcal{F}} p_f^t$;
Generate random number, v = random(0,1);
if $P_t > 0$ **then**
    sum $\leftarrow 0$;
    target $\leftarrow P_t \times v$;
    i $\leftarrow 1$;
    while sum $<$ target **do**
        sum $\leftarrow$ sum+p_i^t;
        $i \leftarrow i + 1$;
    end while
else
    Choose randomly one flight i in the flight set;
end if
if $i \in \mathcal{A}$ **then**
    if $p_i^c > 0$ **then** change holding time, choose randomly one value between 0 and the maximum allowed deviation;
    else choose with equal probability between holding time and taxi speed change;
    end if
else if $i \in \mathcal{D}$ **then**
    if $p_i^g > 0$ **then**
        choose with equal probability between pushback time change and taxi speed change;
    else
        choose with equal probability among pushback time change, taxi speed change, and holding time change;
    end if
end if

Algorithm 9 Neighborhood function for minimizing time changes.

Choose one flight f with time decision changes;
if $f \in \mathcal{A}$ **then** choose a new holding time between 0 and current one;
else if $f \in \mathcal{D}$ **then**
    if pushback time changed **then** choose a new pushback time between 0 and current one;
    else if holding time changed **then** choose a new holding time between 0 and current one;
    end if
end if

4.3.1 Ground operations at CDG

In this section, we show some representative metrics related to ground operations at CDG. The evolution of airport movements, the average real taxi times, the aircraft taxi path options, and the physical departure queue lengths, are presented.

Airport movements and layout

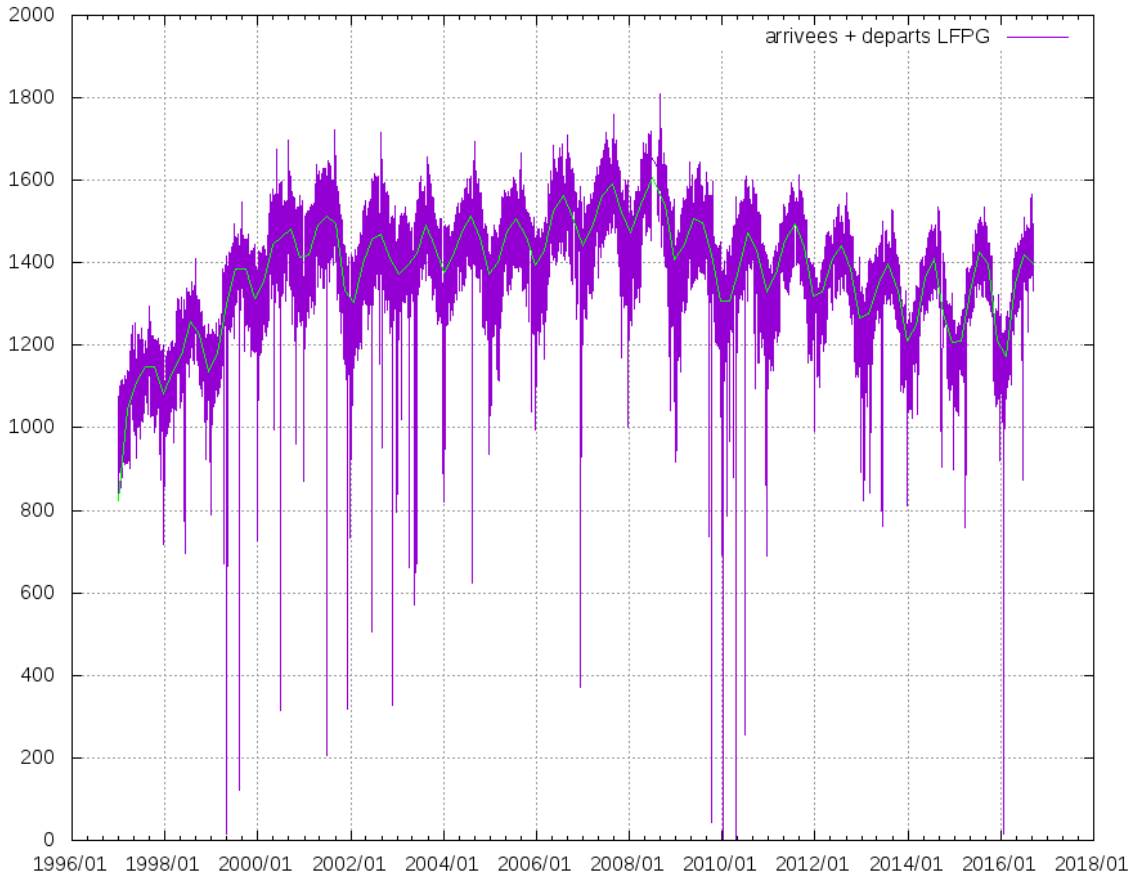
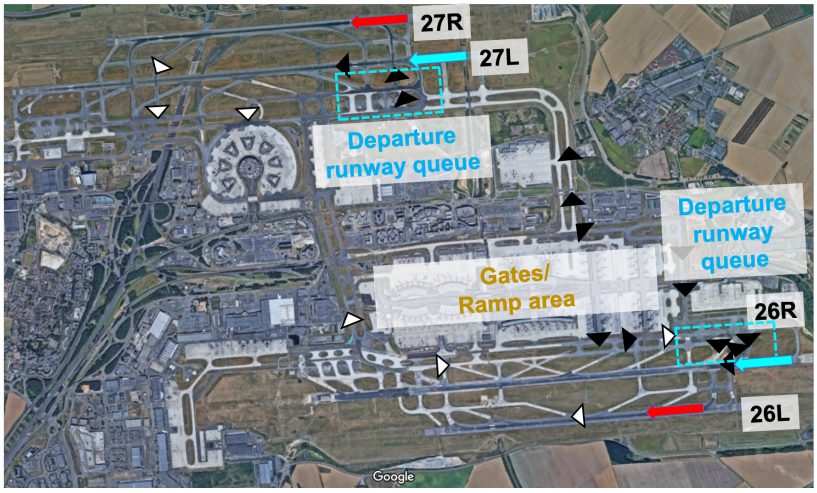


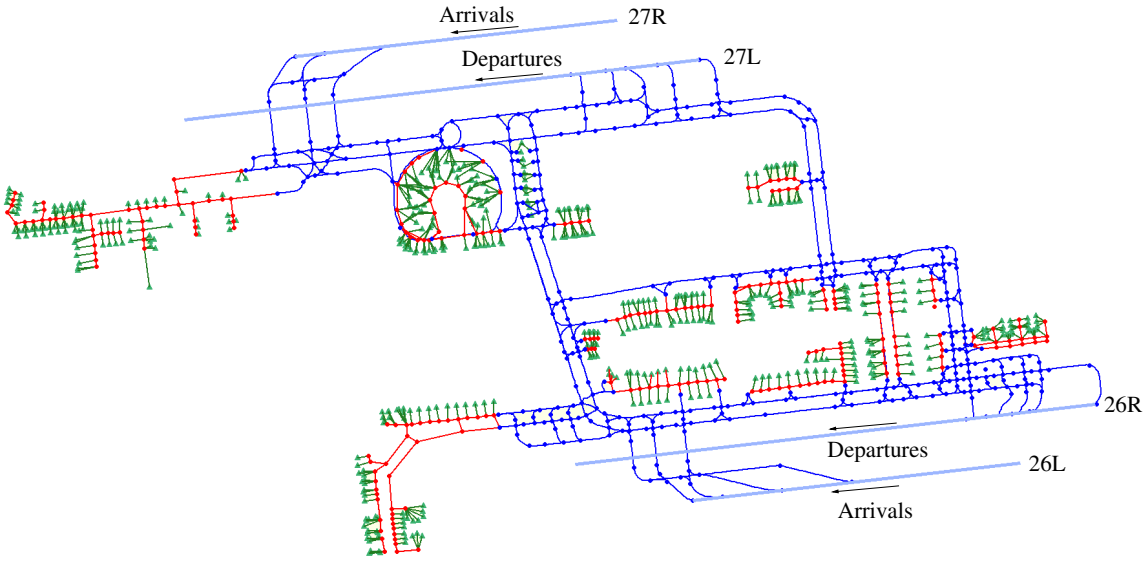
Figure 4.7: Evolution of aircraft movements at CDG from 1996 to 2017. The maximum movements were more than 1600 aircraft per day from 2006 to 2009. The winter period is in lower demand compared to the summer period.

CDG handled around 1,300 flights/day and 66 million passengers in 2016, making it the second busiest airport in Europe and the eleventh busiest airport in the world in terms of aircraft movements [102]. Fig. 4.7 shows the evolution of aircraft movements in CDG from 1996 to 2017. The maximum movements were more than 1600 aircraft per day from 2006 to 2009. In recent years, the winter period (around 1200 movements per day) is in lower demand than the summer period (around 1400 movements per day). The airport has four parallel runways, and operates under two broad runway configurations: West-flow ($26L, 27R|26R, 27L$) and East-flow configuration ($09L, 08R|09R, 08L$). The runway configuration refers to the set of runways used at the airport for an extended period of time. In this chapter, we illustrate the benefits of departure metering with West-flow configuration since it is the predominant runway configuration (75% of the operations in

July-August 2017). Fig. 4.8a shows the airport layout of CDG, along with a snapshot of aircraft positions in West-flow configuration, departing flights represented by black triangles and arriving ones represented by white triangles. The departure and arrival runways are indicated using blue and red arrows, respectively. We can see queues being formed near the departure runways as a result of lower capacity during peak periods of traffic.



(a) Airport layout with a snapshot of traffic movement.

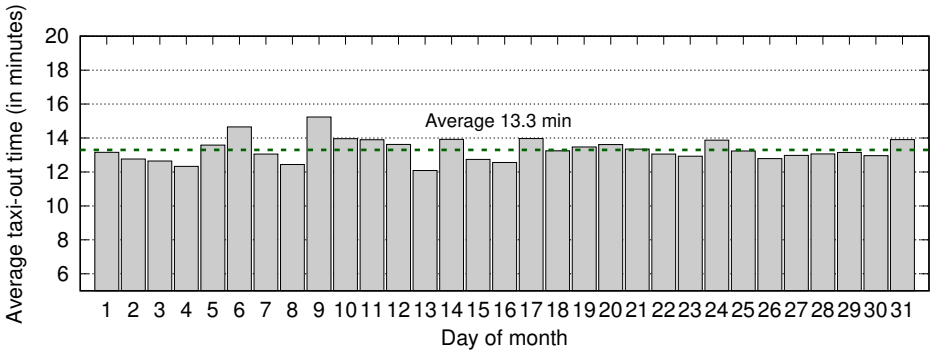


(b) Node-link network.

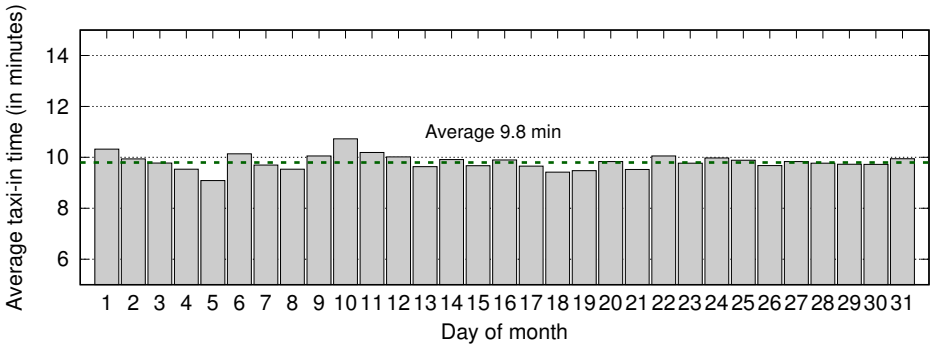
Figure 4.8: CDG in West-flow runway configuration.

Airport taxi times

Fig. 4.9a shows the average daily taxi-out times in July 2017. The monthly average taxi-out time is 13.3 minutes. Moreover, the average taxi-out times to runway 27L and to runway 26R are 14.06 minutes and 12.62 minutes, respectively. This is mainly due to the longer taxi distances from the assigned gates to runway 27L. The days of July 6 and July 9 show a longer average taxi-out time, one reason could be the runway configuration change during the day. On Fig. 4.9b, the monthly average



(a) Average daily taxi-out times in July 2017.



(b) Average daily taxi-in times in July 2017.

Figure 4.9: Average taxi times at CDG in July 2017.

taxi-in time is 9.8 minutes. Similarly, aircraft landing on runway 27R show a longer average taxi-in time compared to aircraft landing on runway 26L.

Airport runway queue analysis

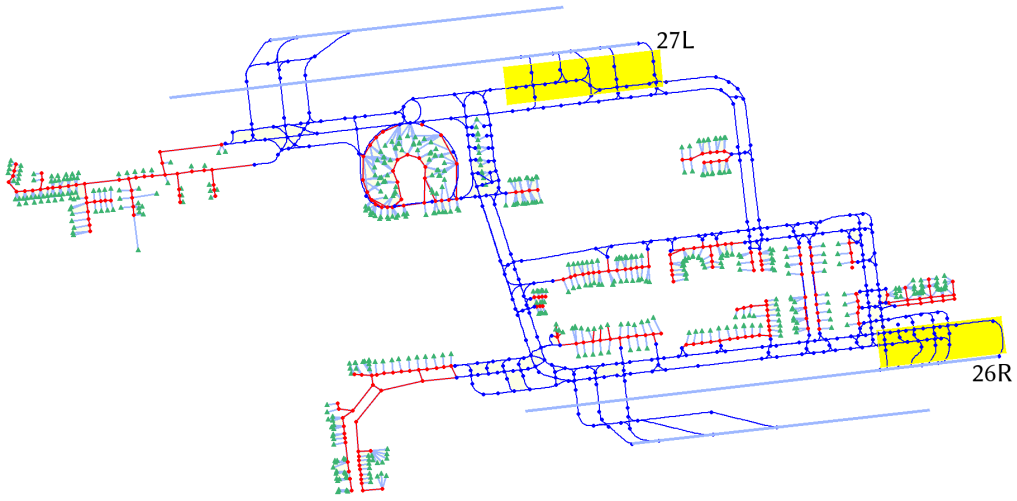


Figure 4.10: Detection of physical departure queues by counting the number of aircraft holding in the yellow area at each time step. The time is discretized every 10 seconds.

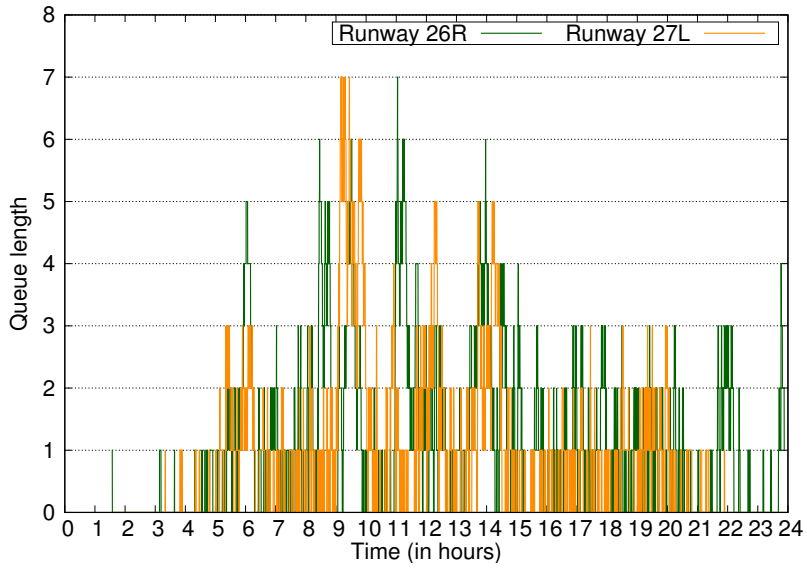
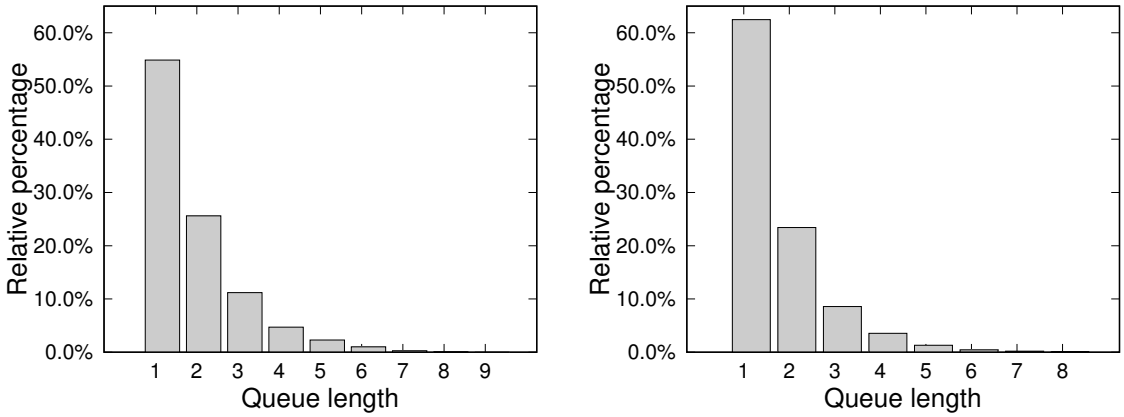


Figure 4.11: Departure runway physical queue length of an illustrative day (July 10^{th}, 2017).



(a) Relative percentages of queue length on runway 26R.

(b) Relative percentages of queue length on runway 27L.

Figure 4.12: Relative percentages of queue length at CDG.

In order to investigate the departure runway queue lengths, we define two yellow rectangle areas to capture the number of aircraft holding inside as illustrated in Fig. 4.10, then one can obtain the queue length profile over the day. Fig. 4.11 illustrates the departure queue length of a representative day, the green curve depicts the queue length at runway 26R, while the orange curve represents the queue length at runway 27L. The maximum queue length was 7 in the period of 8 AM–10 AM, and runway 26R is more utilized than runway 27L. Fig. 4.12 shows the relative percentage of queue length on departure runways. The runway idle time [1] is not taken into account.

[1] Idle time is defined as un productive time on the part of machines caused by management or as a result of factors beyond their control. Runway idle time refers to the time when the runway is not being used but could be.

4.3.2 Optimization set-up

In this section, we clarify the graph model used for optimization, present some assumptions in order to simplify the problem, and specify the values chosen for a number of parameters used in our tests.

The CDG node-link model consists of 1185 nodes and 1441 links that connect adjacent nodes (see Fig. 4.8b), and contains 517 gates including remote stands and cargo area. We made assumptions in order to simplify the problem while keeping some level of reliability.

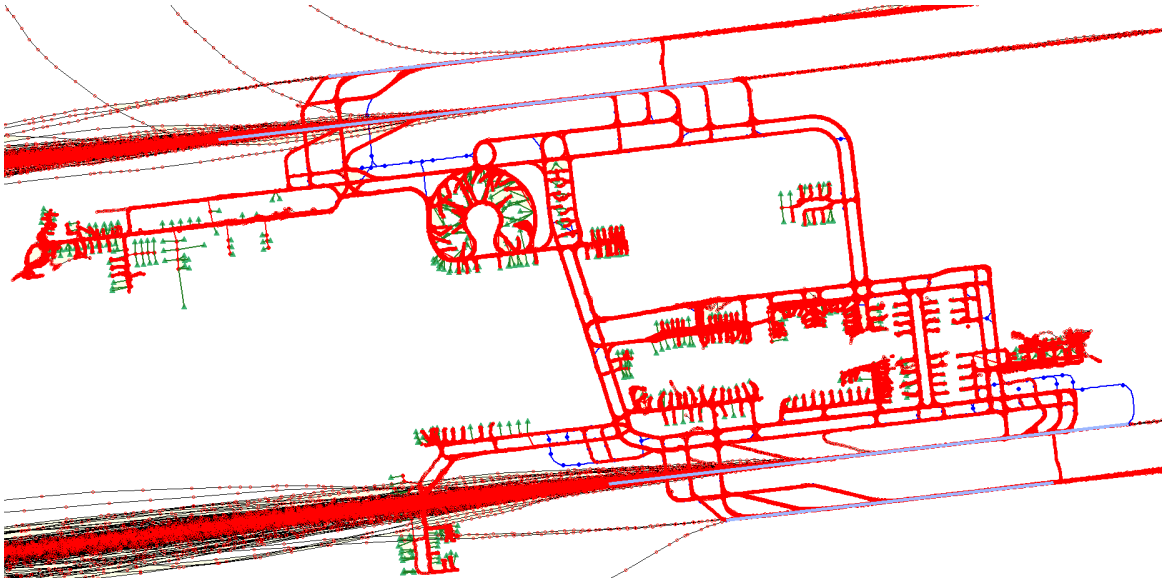


Figure 4.13: Radar tracks coverage on July 11^{th}, 2017. The Western hangars as well as gates in terminal 1 are not fully used.

- Fig. 4.13 illustrates one-day of radar tracks coverage on July 11^{th}, 2017. The hangars in the western side as well as gates in the terminal one are not fully used. Hence, we assume that gates are enough to accommodate all the flights on the current traffic level.
- Aircraft taxi with a constant speed for a given link. For each link, we use the average speed value analyzed with the real data to take into account different taxiway types. The first links connecting gate to its adjacent node in green color in Fig. 4.8b are applied with a pushback speed, the red color links are assumed to use a ramp speed, and the links in blue color are applied with a taxi speed. The maximum speed values are 0.3, 7.0, and 10.0 m/s on gate area, ramp area, and taxiways, respectively. The minimum speed is assumed to be half of the maximum speed in the ramp area and taxiway, and 80% of the maximum taxi speed in the gate area.
- Only West-flow configuration is studied, as West-flow configuration composes 75% of the configuration in the traffic data that we have access to, and East-flow configuration gives similar results.
- Each flight follows one predetermined taxi path consisting of a set of nodes and links.

The minimum separation distance on taxiways is 60 meters and 30 meters on ramp area. In the metering case, the maximum arrival holding time and departure holding time are 5 minutes and

15 minutes. The maximum pushback delay is set to be 10 minutes. We assume that a maximum of 2 arrivals can wait at the holding point, and a maximum of 5 departures can wait at the runway threshold. The weighting coefficients for the objective function are as follows: $\theta = 0.1, \alpha = 1, \beta = 1, \gamma = 0.01$. Time is discretized into 5 sec intervals. Speed is discretized into one percent of the maximum allowed speed.

Next, we illustrate the airport ground optimization results with the benefits of gate holding and the departure queue management.

4.3.3 Benefits of gate holding strategy at CDG

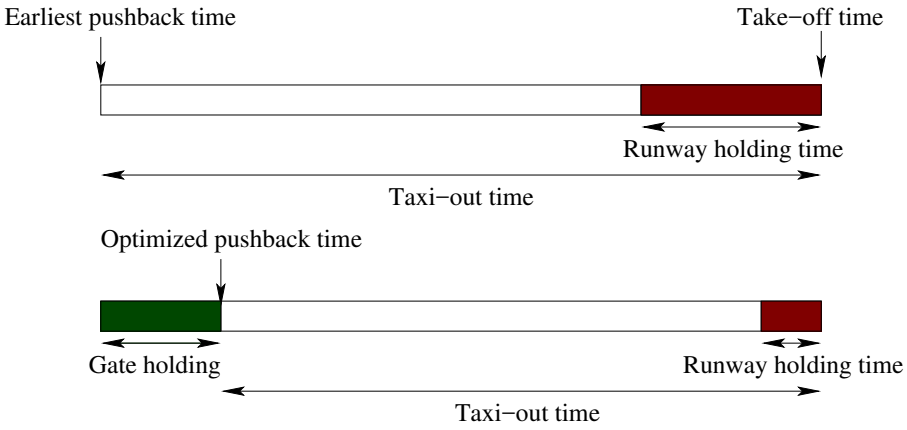


Figure 4.14: Benefits of gate holding by transferring the aircraft waiting time with engine-on to the gate before pushback with engine-off.

As illustrated in Fig. 4.14, the benefits of gate holding is to transfer the aircraft waiting time while taxiing with engine-on to the gate before pushback with engine-off, thus reducing the taxi-out times. In order to make a meaningful comparison, we first set a baseline case by using the earliest pushback times and calculate the arrival and departure taxi times. Taxi speed and holding times at runway threshold are the decision variables. Next, the gate holding strategy is considered. We run the optimization process again by including the pushback time as an additional decision variable.

Table 4.1 lists some metrics of the optimized results. We have tested on traffic data of three days in the 6 AM–10 PM period. We observe an average taxi-out reduction of 1 minute with a cost of 1 minute average gate holding time. Moreover, the taxi-in times keep a similar level without adverse effect from the reduction of taxi-out time. In the last row of Table 2.1, the percentage of flights held at the gate is reported as well.

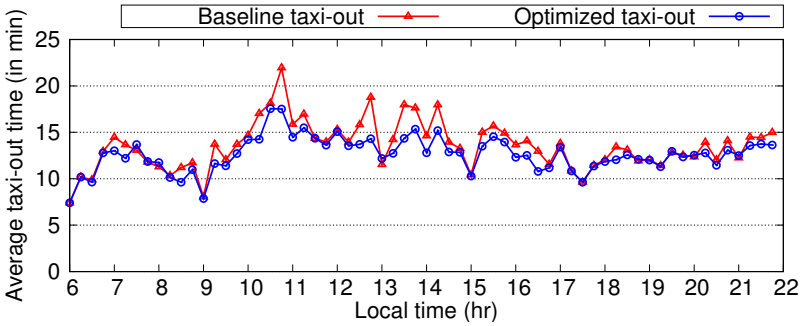
Let us take July 24^{th}, 2017 as an example to detail the taxi times for each time period. Fig. 4.15a shows the average taxi-out times over 15 min time interval for the baseline case (red color) and the optimized case (blue color). We observe that in the peak hours of 10 AM–11 AM and 12 PM–14 PM, the taxi-out time is strongly reduced. Fig. 4.15b illustrates the comparison of taxi-in times. We observe a similar amount of taxi-in times for both cases. The taxi-out time reduction does not really affect the taxi-in times.

4.3.4 Departure queue management at CDG

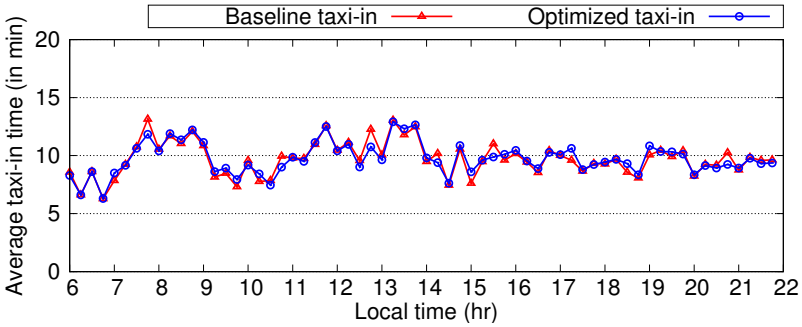
Fig. 4.16a and 4.16b show the comparison of queue length between the baseline case and the optimized case on departure runways 26R and 27L. After optimization, the departure queue length

Table 4.1: Comparison of gate holding results at CDG.

	2017/07/01	2017/07/10	2017/07/24
Time period (local time)		6 AM–10 PM	
Total number of departures	564	655	657
Total number of arrivals	573	612	618
Baseline average taxi-out time (in min)	13.86	14.25	14.22
Optimized average taxi-out time (in min)	12.98	13.22	13.09
Baseline average taxi-in time (in min)	9.96	10.19	10.05
Optimized average taxi-in time (in min)	9.96	10.24	10.08
Average gate holding (in min)	0.94	1.22	1.28
Percentage of flights held at the gate	52 %	62 %	59%

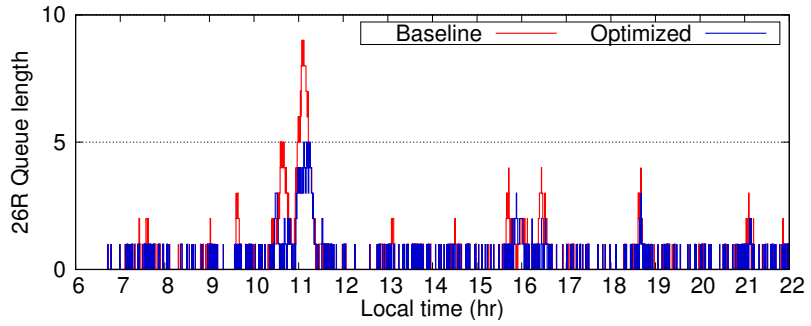


(a) Average taxi-out time

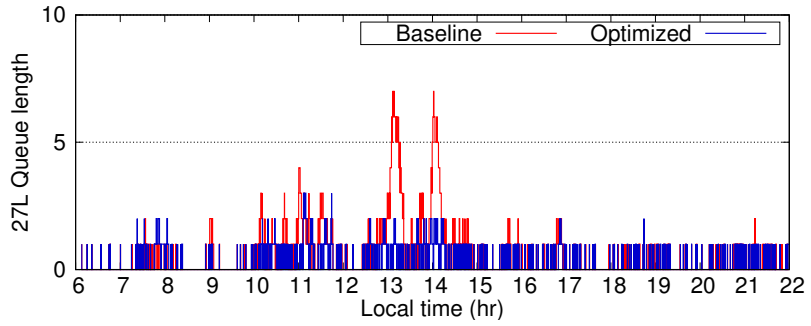


(b) Average taxi-in time

Figure 4.15: Average taxi time comparison at CDG on July 24^{th}, 2017.



(a) Runway 26R



(b) Runway 27L

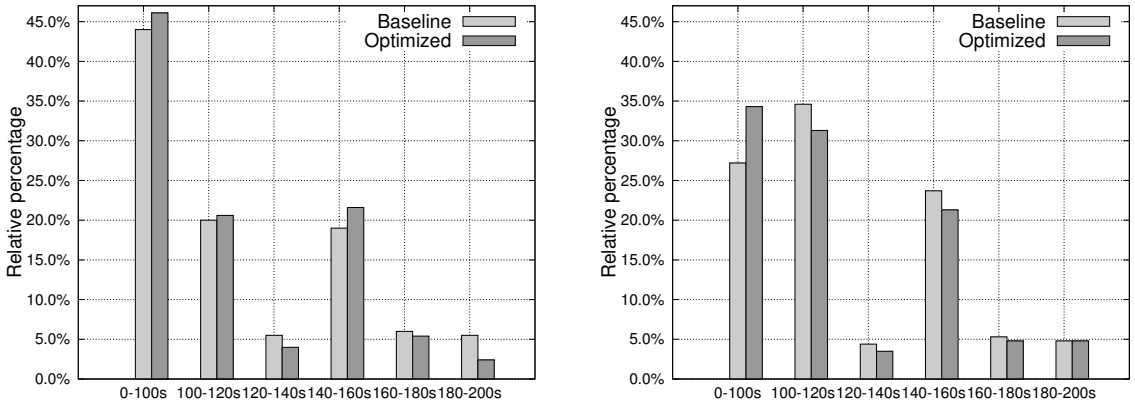
Figure 4.16: Queue management comparison at CDG on July 24^{th}, 2017.

is reduced. Another important issue is the runway throughput. The reduction of queue length should not decrease the runway throughput, i.e., the runway should not be underutilized. Hence, we check the inter-departure spacings for both runways. Fig. 4.17a and Fig. 4.17b illustrate respectively the departure separation reduction for runway 26R and runway 27L of the optimized case compared to the baseline case. We only account the inter-departure separations that are less than 200 sec to focus on the runway fully used periods. There are more aircraft in the horizon of 0-100 s in the optimized case. The average runway separation over 3-day is reduced from 123.5 s to 121.2 s on runway 27L and from 121.2 s to 120.0 s on runway 26R. Thus, our optimization helps to reduce the queue length of departure runways without sacrificing the runway throughput.

4.4 Case study: Charlotte CLT airport

4.4.1 Ground operations at CLT

CLT handled around 1,400 flights/day and 44.4 million passengers in 2016, and is the 7^{th} busiest airport in the world, in terms of aircraft movements [102]. The airport has three parallel runways and one intersecting runway. CLT operates under two broad runway configurations: North-flow (36C, 36L, 36R | 36C, 36R) and South-Flow (18L, 18C, 18R, 23 | 18C, 18L). The North-flow configuration handled about 56% of the traffic in 2016 [103], hence we consider this configuration to illustrate the departure metering benefits at CLT in this chapter. Fig. 4.18a shows the airport layout of CLT. The leftmost runway (36L) is used only for arrivals, whereas, runways 36C and 36R are used under mixed operations. Arrivals and departures are respectively represented by white triangles and black triangles. The landing aircraft from 36L have to cross the departure runway 36C. One can clearly observe queues forming at the runway threshold. An interesting aspect about CLT is that it has



(a) Inter-departure spacings on runway 26R

(b) Inter-departure spacings on runway 27L

Figure 4.17: Inter-departure spacings comparison at CDG on July 24^{th}, 2017.

congestion at multiple areas, we can see queues being formed in the ramp area and near the runway crossing, in addition to the departure runway queue (Fig. 4.18a).

4.4.2 Optimization set-up

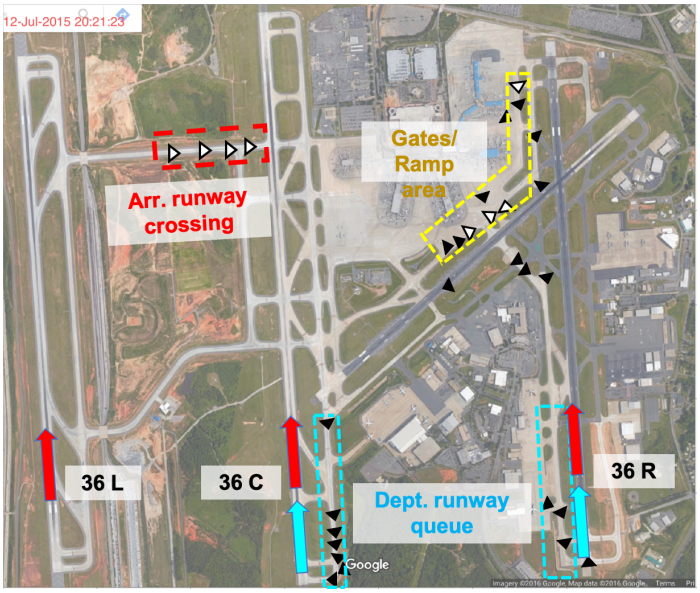
Fig. 4.18b illustrates the node-link model at CLT which consists of 581 nodes and 506 links that connect adjacent nodes. The 5 terminals contain 102 gates. In this section, only one standard taxi route is considered in CLT. Similarly to CDG, we assume that the aircraft taxi with a constant speed with regard to three areas: the pushback speed with green links, the ramp speed with red links, and the taxi speed with blue links. The maximum speed values are 0.15, 6.98, and 9.0 m/s on gate area, ramp area, and taxiways, respectively. The minimum speed is assumed to be half of the maximum speed in the ramp area and taxiway, and 80% of the maximum taxi speed in the gate area. The minimum separation distance is 80 meters in the taxiways and 30 meters in the ramp area. In the metering case, the maximum arrival holding time and the maximum departure holding time are 10 minutes and 15 minutes, respectively. The maximum pushback delay is set to be 20 minutes. The maximum departure queue length and arrival queue length are both 5. The weighting coefficients for the objective function are as follows: $\theta = 0.01, \alpha = 2, \beta = 1, \gamma = 1$. Time is discretized into 5 sec intervals. Speed is discretized into one percent of the maximum allowed speed.

In the following section, we first illustrate the benefits of gate holding on the taxi-out time reduction. Then, we study the mitigation of runway congestions.

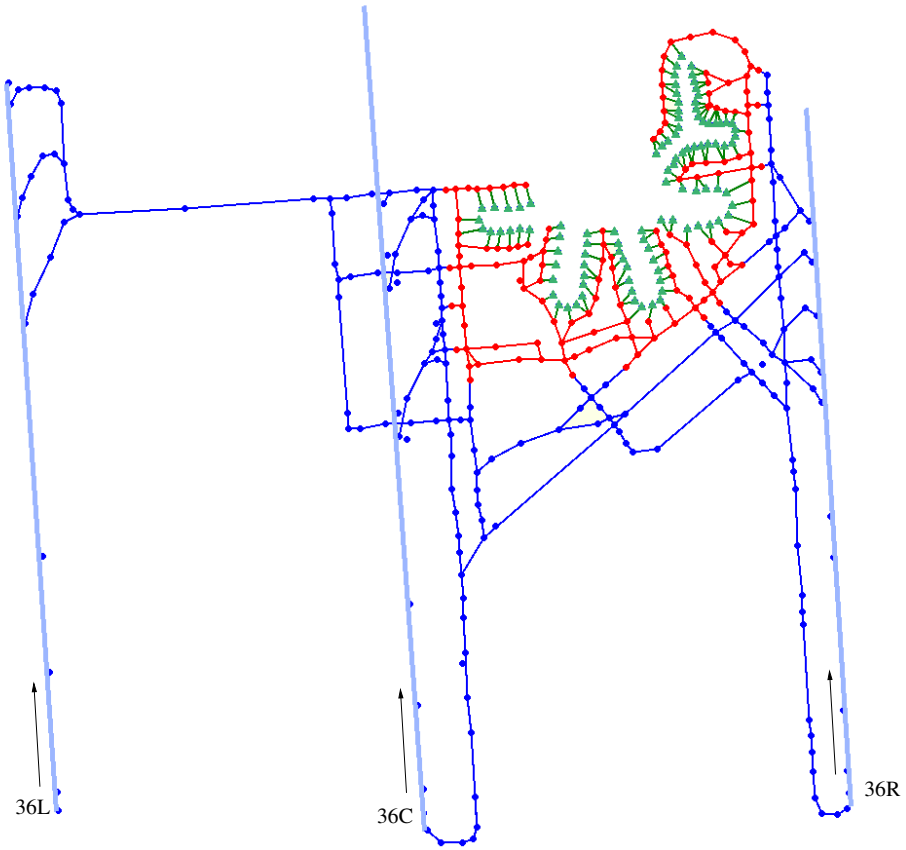
4.4.3 Benefits of gate holding strategy at CLT

Table 4.2 compares the gate holding results. The average taxi-out time over all the 30-min interval decreases from 18.4 minutes to 14.9 minutes with the optimization model, while the average gate holding time of departures is 3 minutes.

Fig. 4.19 shows the taxi-times averaged over 15-min intervals with the optimized pushback time for a typical day at CLT and they are compared with the baseline case. We can see reduced taxi-out as well as taxi-in times in the optimized case, particularly during intervals that have a high baseline value. The reduction in taxi-in time arises because of better sequencing of runway crossings. Note that the reduction in taxi-out time does not really impact the taxi-in time.



(a) Airport layout with a snapshot of traffic.

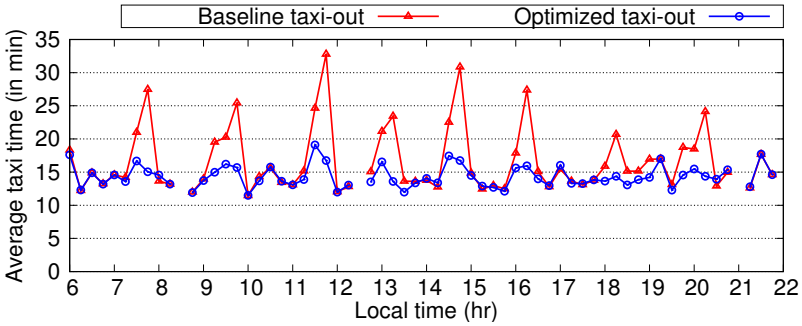


(b) Node-link network.

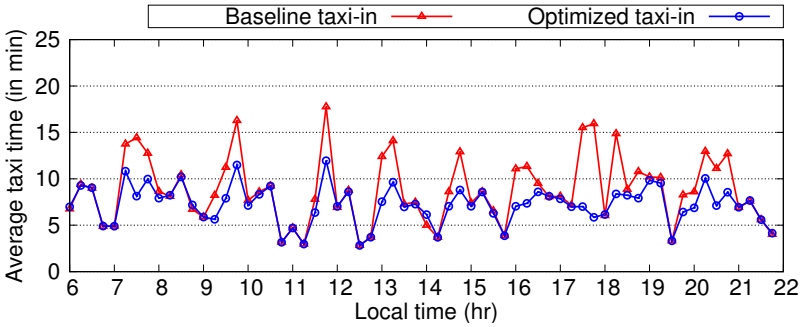
Figure 4.18: CLT in North-flow runway configuration [3].

Table 4.2: Comparison of gate holding results at CLT.

	2015/05/07	2015/07/10	2016/05/06
Time period (local time)		6 AM–10 PM	
Total number of departures	633	627	644
Total number of arrivals	688	689	705
Baseline average taxi-out time (in min)	19.11	18.32	17.92
Optimized average taxi-out time (in min)	15.02	14.93	14.76
Baseline average taxi-in time (in min)	9.60	10.00	9.16
Optimized average taxi-in time (in min)	8.00	8.16	7.57
Average gate holding (in min)	3.09	3.07	2.96
Percentage of flights held at the gate	66%	67 %	64%



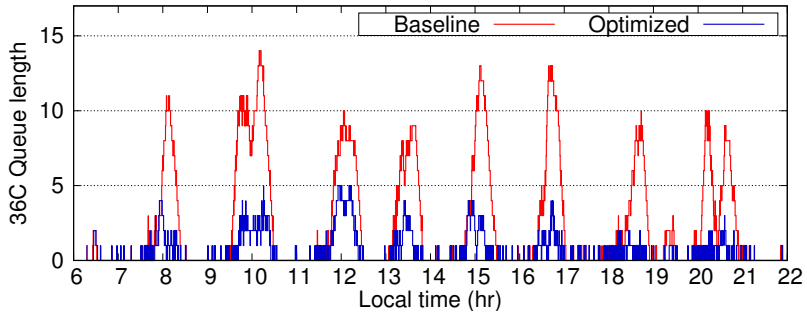
(a) Average taxi-out time



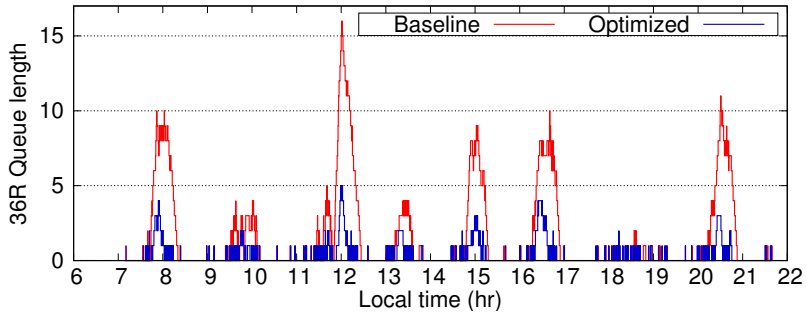
(b) Average taxi-in time

Figure 4.19: Average taxi time comparison at CLT on May 7^{th}, 2015.

4.4.4 Departure queue management at CLT



(a) Runway 36C



(b) Runway 36R

Figure 4.20: Queue management comparison at CLT on May 7^{th}, 2015.

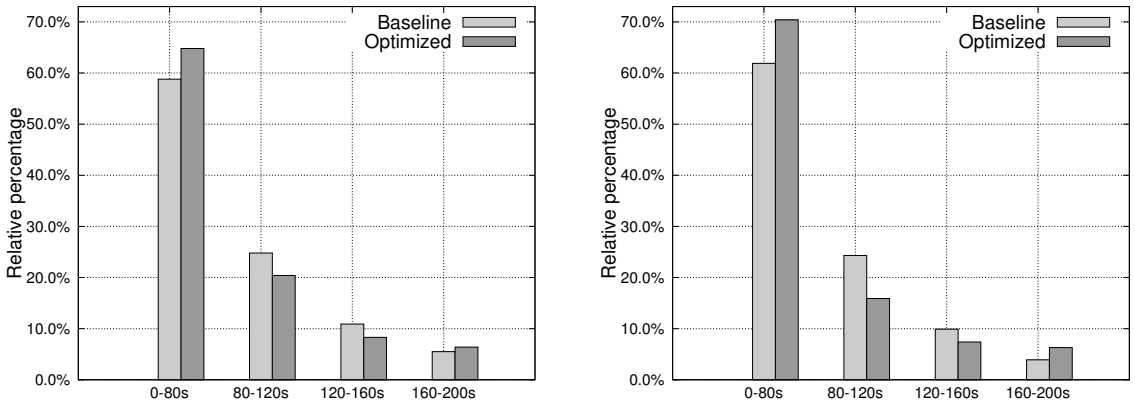
Fig. 4.20 shows the runway queue length over the course of time. The runway congestion occurs as the taxi-out time increases in accordance with the three periods observed in Fig. 4.19a. A maximum of 14 aircraft on runway 36C and 16 on runway 36R wait at the runway threshold before taking off. After optimization, the number is reduced to 5. Fig. 4.21a and Fig. 4.21b illustrate respectively the departure separation reduction for runway 36C and runway 36R of the optimized case compared to the baseline case. There are more aircraft in the horizon of 0-80 s in the optimized case. The average runway separation over 3-day is reduced from 123.5 s to 121.2 s on runway 27L and from 121.2 s to 120.0 s on runway 26R. Thus, the reduction on runway queue length did not affect adversely the runway throughput.

4.5 Discussion

4.5.1 Comparison of airport operations

CDG and CLT are two representative hub airports in Europe and in the U.S.. The two airports handle approximately the same number of aircraft movements. However, the fleet mix at the two airports are significantly different, with CDG handling a larger percentage of aircraft under the 'heavy' category (25%) than CLT (2%). Although both the airports have same number of departure runways, CLT has mixed operations unlike CDG. On the other hand, CDG operates under Instrument Meteorological conditions (IMC) [2] capacity even in good weather conditions, unlike CLT. The result

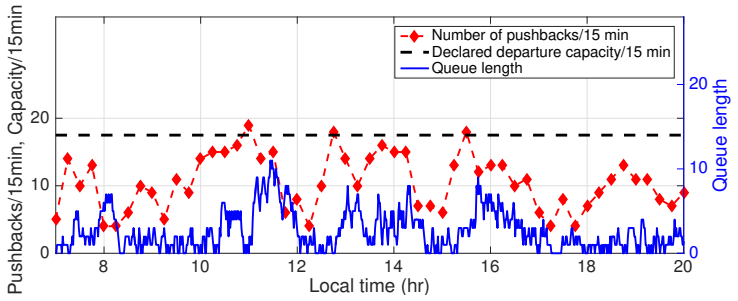
[2]Meteorological conditions expressed in terms of visibility, distance from cloud, and ceiling, less than the minima specified for visual meteorological conditions.



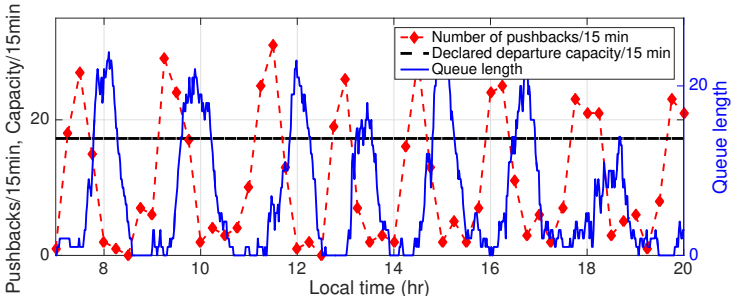
(a) Inter-departure spacings on runway 36C

(b) Inter-departure spacings on runway 36R

Figure 4.21: Inter-departure spacings comparison on May 7^{th}, 2015.



(a) CDG (July 10^{th}, 2017)



(b) CLT (May 7^{th}, 2015)

Figure 4.22: Number of pushbacks (per 15 min), declared departure capacity and queue length for a typical good weather day.

of fleet mix, mixed operations at CLT and IMC capacity at CDG is that the declared departure capacity in good weather condition at both the airports is about the same. Another distinguishing feature is the nature of demand at the two airports. Fig. 4.22 shows the number of pushback (per 15 mins), declared departure capacity and total runway queue length for a typical day at the two airports. The departures are significantly banked at CLT compared to CDG. This results in demand exceeding capacity during peak periods of traffic at CLT, causing the formation of longer queues on the airport surface. One can notice that the demand at CDG rarely exceeds capacity since the European airports are slot constrained unlike US airports. The higher imbalance between demand and capacity at CLT leads to higher taxi-out delays. Hence, we would expect higher benefits with departure metering at CLT than at CDG.

The average taxi-out time in CLT is around 20 minutes, while at CDG the average taxi-out time is around 13 minutes. Congestions occur both at the runway threshold and in the ramp area in CLT due to the shared use of taxiways for arrivals and departures. In CDG, the two main parallel taxiways with preferential direction segregate arrival taxiing and departure taxiing, consequently avoid head-on conflicts between departures and arrivals, congestion only occur at the runway threshold. Considering the departure queue at the runway threshold, CLT reaches twice as much the maximum queue length of CDG (see Fig. 4.20 and Fig. 4.16). The average taxi-in time is similar in both airports, around 10 minutes.

In North-flow configuration at CLT, three parallel runways operate (one for only arrivals, another two for mixed operations). While in CDG, there are four parallel runways, two outer ones for arrivals and 2 inner ones for departures. Arrivals have to cross departure runways to reach terminal both in CDG and in CLT. However, due to different holding area design, CLT can accommodate up to five consecutive arrival aircraft waiting before crossing the departure runway, while CDG can only accommodate a maximum of two arrival holdings, which indicates that arrivals have higher priority than departures.

The fleet mix in CDG is around 75% Large and 25% Heavy, also the holding area for departures is more flexible and can form four or five queues simultaneously, thus there is more potential to increase runway throughput by sequencing departure aircraft. At CLT, there are predominantly Large (or Medium) aircraft operating on the ground, thus reducing taxi-out time is much more interesting than sequencing aircraft since there are no benefits from sequencing with regard to wake turbulence category.

4.5.2 Computing environment

Table 4.3: Computational time for 30-min interval for CDG and for CLT (in min).

Statistic	**CDG**	**CLT**
Max	4	7.8
Mean	1.5	2.3

The overall process is run on a 2.50 GHz core i7 CPU, under Linux operating system PC based on a Java code. The maximum computation time for 30-min intervals is 4 minutes for CDG and 7.8 minutes for CLT. Table 4.3 lists the computation times.

4.6 Conclusion

In this chapter, we have considered the problem of airport ground movement and departure management. First, a mathematical model was presented subject to several important operational constraints: runway separation, taxi separation, restriction on the queue length of holding area, maximum delay before crossing the runway etc. Next, we presented an adaptation of SA algorithm in our problem. After that, computational experiments on two illustrative airport were studied and compared, showing that our approach can reduce efficiently the taxi-out time by gate holding strategy and mitigate departure runway congestions while maintaining runway throughput.

All the presented works in this chapter are based on deterministic models. However, in the current operational environment, various sources of uncertainty in airport operations can influence the system performance. One could build stochastic simulations to evaluate the impact of uncertainty on airport performance. First, the stochastic simulation should comply with reality and simulate the aircraft trajectories on the ground in terms of taxi times and queue lengths. Then, the optimization results could be used as input to the simulation to investigate different performance metrics. Collaborative works on considering several candidate optimization approaches to the airport surface operation problem have been conducted with the Department of Aeronautics and Astronautics at the Massachusetts Institute of Technology. Comparative analysis were conducted by using the trajectory-based optimization presented in this chapter and the algorithms based on queuing network models [65, 104]. Stochastic simulations were used to evaluate the performance of the approaches. A joint paper related to this topic [105] has been presented in the ATM R&D Seminar [3].

[3] 2019 ATM R&D Seminar, the 13th USA/Europe Air Traffic Management Research and Development Seminar, Vienna, Austria, June 17-21, 2019

Chapter 5

Exact and heuristic algorithms for scheduling aircraft departures

The runway is a key airport resource and an efficient runway operation is critical to enhance the airport efficiency and to reduce delays. This chapter addresses a sub-problem from the microscopic level presented in Chapter 4, i.e. the problem of scheduling aircraft departures incorporating arrival crossings. We present two Integer Linear Programming (ILP) models and a simulated annealing (SA) algorithm to solve the same problem such that the gap between optimal solution and heuristic solution as well as the computation time can be compared. Comparison tests are conducted for the Southern side of Paris Charles De-Gaulle Airport. This chapter is organized as follows. In Section 5.1, the problem statement is presented. In Section 5.2, two ILP formulations are proposed in detail, followed by the metaheuristic method in Section 5.3. Finally, computational results and conclusions are summarized in Section 5.4 and Section 5.5 respectively.

5.1 Problem statement

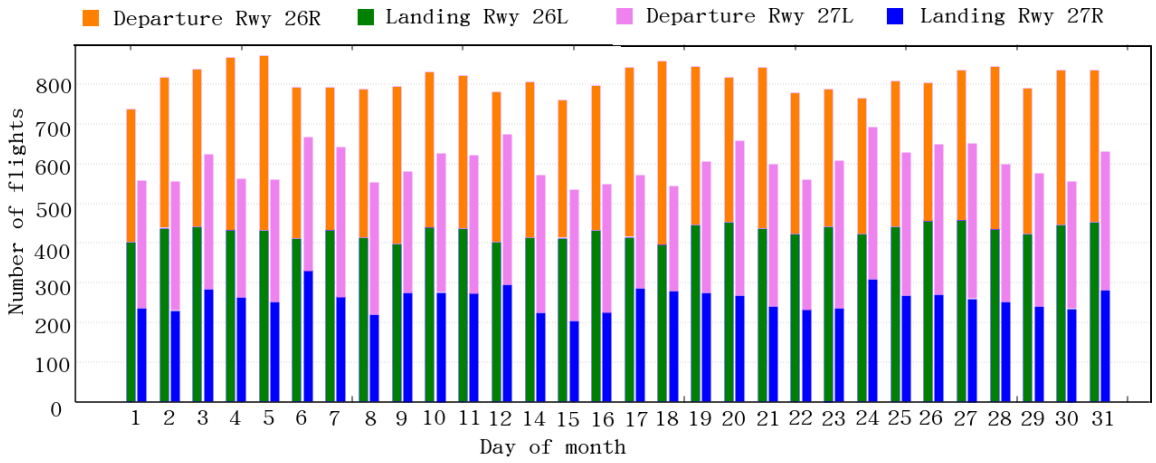


Figure 5.1: Paris CDG airport runway throughput chart in July 2017. Southern pair of runways (26L/26R) is in higher demand compared to northern side (27L/27R). July 13^{th} is excluded due to abnormal data record.

Fig. 5.1 illustrates one month runway throughput of CDG in July 2017. The southern pair of

runways is in higher demand than the northern side due to the fact that it is closer to the main terminal (Terminal 2) of CDG. In this chapter, we focus on the departure scheduling problem incorporating arrival crossings on the southern pair of runways: 26L for arrival and 26R for departure. As shown in Fig. 5.2, there are three main landing runway exits: V2, V3, and V4 and three holding points before crossing: S1, S2, and S3. In practice, the landing runway exit is chosen by the pilot with regard to the aircraft type and how fast it can brake after touching down. Controllers prefer to give priority to arrival aircraft to cross the departure runway without stopping when the departure runway is not fully occupied. When the departure traffic demand is high, controllers indicate arrival aircraft to wait at different holding points and to cross simultaneously. For departures, controllers can hold aircraft at the gate with engines off, or have them waiting at the runway threshold before taking off. In order to ensure safety, the minimum wake turbulence separation between successive aircraft must be respected. Moreover, several operational constraints such as time window of runway usage for individual flights, maximum runway queue capacity, and interactions between arrival crossings and departures need to be considered.

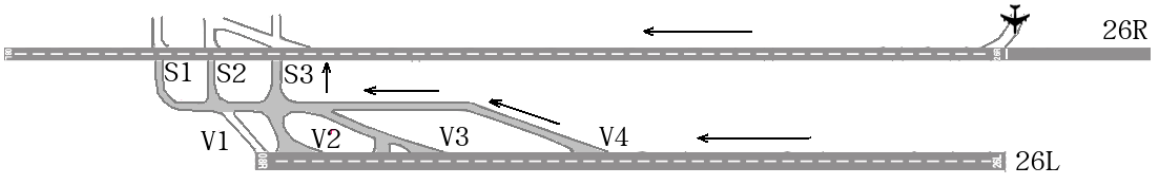


Figure 5.2: CDG Southern side runway layout. Runway 26L is for landing and 26R is for taking off. The arrival aircraft have to cross the departure runway to reach the taxi area. There are three arrival holding points before crossing 26R: S1, S2, and S3, and three main runway exits: V2, V3, and V4.

To be more precise, let $\mathcal{F} = \mathcal{A} \cup \mathcal{D}$ be the set of aircraft (arrivals and departures) that will use the runway during the time horizon $\mathcal{T}$, which is discretized into time intervals $= \{1, \ldots T\}$. Flight taking off/crossing at the time interval $t \in \mathcal{T}$ refers that it occurs at the beginning of time interval t.

We need to define the following notations for our problem formulation:

$\mathcal{F}$ set of flights, $\mathcal{F} = \mathcal{A} \cup \mathcal{D}$,

$\mathcal{A}$ set of arrivals,

$\mathcal{D}$ set of departures,

$\mathcal{E}$ set of landing runway exits, $\mathcal{E} = \{V2, V3, V4\}$,

$\mathcal{H}$ set of holding points before crossing departure runway, $\mathcal{H} = \{S1, S2, S3\}$,

$d_{e,h}$ taxi time from runway exit $e \in \mathcal{E}$ to holding point $h \in \mathcal{H}$, shown in Table 5.1,

C_d departure capacity at the runway threshold,

C_a arrival capacity at holding points,

ρ time slot, which is a multiple of time intervals,

P_i scheduled pushback time for flight $i \in \mathcal{D}$,

X_i taxi time from spot to runway threshold for flight $i \in \mathcal{D}$,

T_i^0 initial runway usage time for flight $i \in \mathcal{D}$, $T_i^0 = P_i + X_i$,

R_i^p maximum pushback delay (in number of slots) for departure flight $i \in \mathcal{D}$,

R_i^w maximum holding time (in number of slots) for departure flight $i \in \mathcal{D}$,

R_i maximum delay (in number of slots) for departure flight $i \in \mathcal{D}$, $R_i = R_i^p + R_i^w$,

$\mathcal{R}_i^p$ set of possible pushback delays for flight $i \in \mathcal{D}$, $\mathcal{R}_i^p = \{1, \ldots, R_i^p\}$,

$\mathcal{R}_i$ set of possible total delays for flight $i \in \mathcal{D}$, $\mathcal{R}_i = \{1, \ldots, R_i\}$,

t_i^T time arriving at the runway threshold for flight $i \in \mathcal{F}$,

T_{ir} possible runway usage time for flight $i \in \mathcal{D}$, $T_{ir} = T_i^0 + r\rho$, $r \in \mathcal{R}_i$,

L_i scheduled landing time for flight $i \in \mathcal{A}$,

O_i runway occupancy time for flight $i \in \mathcal{A}$,

e_i runway exit point for flight $i \in \mathcal{A}$,

T_{ih}^0 initial runway usage time for flight $i \in \mathcal{A}$ if i uses the holding point h, $T_{ih}^0 = L_i + O_i + d_{e_i,h}$,

R_i^a maximum holding time (in number of slots) for flight $i \in \mathcal{A}$,

$\mathcal{R}_i^a$ set of possible holding delays for flight $i \in \mathcal{A}$, $\mathcal{R}_i^a = \{1, \ldots, R_i^a\}$,

T_{ihr} possible runway usage time for flight $i \in \mathcal{A}$ if i uses the holding point h, $T_{ihr} = T_{ih}^0 + r\rho$, $r \in \mathcal{R}_i$,

s_{ij} minimum separation in runway usage time of two successive flights i and j with regard to their wake turbulence categories, as shown in Table 5.2,

M large positive constant.

Our aircraft runway scheduling consists in:

- Choosing a pushback delay d_i, i.e., the difference between the EOBT and the modified pushback time, and a holding duration w_i at the runway threshold for each departure $i \in \mathcal{D}$. With

Table 5.1: Average time from runway exit of 26L to holding point, $T_{e,h}$ (in seconds). The average values are calculated based on one-month real traffic data.

$T_{e,h}$	**S1**	**S2**	**S3**
V2	104	70	59
V3	118	102	75
V4	158	142	115

Table 5.2: Single-runway separation requirements for departures and arrival crossings (in seconds).

Category		**Trailing Aircraft** g			
		Heavy	***Medium***	***Light***	***Crossing***
Leading Aircraft f	***Heavy***	96	120	120	60
	Medium	60	60	60	60
	Light	60	60	60	60
	Crossing	40	40	40	10

these decisions, we know the arrival time at the runway threshold $t_i^T = P_i + d_i + X_i$, and the runway usage time $t_i = t_i^T + w_i$;

- Choosing a holding point h_i and a holding duration w_i for each arrival $i \in \mathcal{A}$. With these decisions, we know the arrival time at holding point $t_i^T = L_i + O_i + d_{e_i,h_i}$, and the runway usage time $t_i = t_i^T + w_i$.

In accordance with the operational practices, pushback delay and holding time are given as multiples of time slot ρ. A runway schedule is feasible if and only if for each pair of distinct flights $(i, j) \in \mathcal{F} \times \mathcal{F}$ with $t_i \leq t_j$, we have

- $t_j - t_i \geq s_{ij}$ and $t_j^T \geq t_i^T$ if $(i, j) \in \mathcal{D} \times \mathcal{D}$, i.e., the minimum wake turbulence separation is respected, and the first aircraft arriving at the holding point departs first;
- $t_j - t_i \geq s_{ij}$ and $t_j^T \geq t_i^T$ if $(i, j) \in \mathcal{A} \times \mathcal{A}$ and $h_i = h_j$, i.e., two aircraft using the same holding point follow the FCFS order, and the minimum separation is respected;
- $t_j - t_i \geq s_{ij}$ if $(i, j) \in \mathcal{A} \times \mathcal{D} \cup \mathcal{D} \times \mathcal{A}$, i.e., the minimum separation is guaranteed between arrival and departure, and vice versa.

Our problem consists in finding a feasible schedule while minimizing the total delay $\sum_{i \in \mathcal{D}} (d_i + w_i) + \sum_{i \in \mathcal{A}} w_i$.

The subproblem of scheduling aircraft take-offs can be seen as a machine scheduling problem with release times, due dates, and sequence-dependent processing times (but zero setup times) [106]. A related problem, the simple total tardiness problem (without sequence dependent setup times), has been shown to be NP-hard [107]. Consequently, our runway scheduling problem is NP-hard too.

5.2 ILP formulations

Two ILP formulations are given in this section. First, we present Model A that uses the number of time slots as decision. This model involves many so called "big-M" constraints that makes the LP relaxation rather weak. In order to reduce this number and to strengthen the formulation, we propose Model B termed as delay-indexed formulation.

5.2.1 Model A: time slot based formulation

The Decision Variables. We define the following decision variables:

$d_i \equiv$ number of pushback delay slots for departure flight $i \in \mathcal{D}$;

$w_i \equiv$ number of holding time slots for flight $i \in \mathcal{F}$;

$t_i \equiv$ runway usage time for flight $i \in \mathcal{F}$;

$t_i^T \equiv$ time arriving at the runway threshold for flight $i \in \mathcal{F}$;

$$\delta_{ij} = \begin{cases} 1, & \text{if } i \text{ uses runway before } j, \\ 0, & \text{otherwise;} \end{cases} \quad \text{for } i \in \mathcal{F}, j \in \mathcal{F}, i \neq j;$$

$$\alpha_{is} = \begin{cases} 1, & \text{if } i \text{ arrives at the runway threshold before time } s, \\ 0, & \text{otherwise;} \end{cases} \quad \text{for } i \in \mathcal{D}, s \in \mathcal{T};$$

$$\beta_{is} = \begin{cases} 1, & \text{if } i \text{ uses runway after time } s, \\ 0, & \text{otherwise;} \end{cases} \quad \text{for } i \in \mathcal{D}, \forall s \in \mathcal{T};$$

$$z_{ih} = \begin{cases} 1, & \text{if } i \text{ crosses runway via holding point } h, \\ 0, & \text{otherwise;} \end{cases} \quad \text{for } i \in \mathcal{A}, h \in \mathcal{H};$$

$$\beta_{ihs} = \begin{cases} 1, & \text{if } i \text{ crosses via holding point } h \text{ after time } s, \\ 0, & \text{otherwise;} \end{cases} \quad \text{for } i \in \mathcal{A}, \forall h \in \mathcal{H}, \forall s \in \mathcal{T}.$$

The Objective Function. Our objective is to minimize the total delay:

$$\text{Min} \sum_{i \in \mathcal{D}} (d_i + w_i) + \sum_{i \in \mathcal{A}} w_i.$$

The Constraints.

$$\text{Min} \sum_{i\in\mathcal{D}}(d_i+w_i)+\sum_{i\in\mathcal{A}}w_i \tag{5.1}$$

$$\sum_{h\in\mathcal{H}} z_{ih}=1, \qquad \forall i\in\mathcal{A}, \tag{5.2}$$

$$t_i^T = T_i^0+\rho d_i, \qquad \forall i\in\mathcal{D}, \tag{5.3}$$

$$t_i^T = \sum_{h\in\mathcal{H}} T_{ih}^0 z_{ih}, \qquad \forall i\in\mathcal{A}, \tag{5.4}$$

$$t_i = t_i^T+\rho w_i, \qquad \forall i\in\mathcal{F}, \tag{5.5}$$

$$\delta_{ij}+\delta_{ji}=1, \qquad \forall i,j\in\mathcal{F}, i<j, \tag{5.6}$$

$$t_j^T - t_i^T \geq -M(1-\delta_{ij}), \qquad \forall i,j\in\mathcal{D}, i\neq j, \tag{5.7}$$

$$t_j - t_i \geq s_{ij} - M(1-\delta_{ij}), \qquad \forall i,j\in\mathcal{D}, i\neq j, \tag{5.8}$$

$$t_j^T - t_i^T \geq -M(3-\delta_{ij}-z_{ih}-z_{jh}), \qquad \forall h\in\mathcal{H}, \forall i,j\in\mathcal{A}, i\neq j, \tag{5.9}$$

$$t_j - t_i \geq s_{ij} - M(3-\delta_{ij}-z_{ih}-z_{jh}), \qquad \forall h\in\mathcal{H}, \forall i,j\in\mathcal{A}, i\neq j, \tag{5.10}$$

$$t_j - t_i \geq s_{ij} - M(1-\delta_{ij}), \qquad \forall (i,j)\in\mathcal{D}\times\mathcal{A}\cup\mathcal{A}\times\mathcal{D}, \tag{5.11}$$

$$\alpha_{is}\leq\alpha_{i(s+1)}, \qquad \forall i\in\mathcal{D}, \forall s\in\mathcal{T}\backslash\{T\}, \tag{5.12}$$

$$\beta_{is}\geq\beta_{i(s+1)}, \qquad \forall i\in\mathcal{D}, \forall s\in\mathcal{T}\backslash\{T\}, \tag{5.13}$$

$$\beta_{ihs}\geq\beta_{ih(s+1)}, \qquad \forall i\in\mathcal{A}, \forall h\in\mathcal{H}, \forall s\in\mathcal{T}\backslash\{T\}, \tag{5.14}$$

$$t_i - \sum_{s\in\mathcal{T}}\beta_{is}=0, \qquad \forall i\in\mathcal{D}, \tag{5.15}$$

$$t_i^T - \sum_{s\in\mathcal{T}}(1-\alpha_{is})=0, \qquad \forall i\in\mathcal{D}, \tag{5.16}$$

$$\sum_{s\in\mathcal{T}}\beta_{ihs}\leq Mz_{ih}, \qquad \forall i\in\mathcal{A}, \forall h\in\mathcal{H}, \tag{5.17}$$

$$t_i - \sum_{h\in\mathcal{H}}\sum_{s\in\mathcal{T}}\beta_{ihs}=0, \qquad \forall i\in\mathcal{A}, \tag{5.18}$$

$$\sum_{i\in\mathcal{D}}(\beta_{is}+\alpha_{is}-1)\leq C_d, \qquad \forall s\in\mathcal{T}, \tag{5.19}$$

$$\sum_{i\in\mathcal{A}|T_{ih}^0\leq s}\beta_{ihs}\leq C_a, \qquad \forall s\in\mathcal{T}, \forall h\in\mathcal{H}, \tag{5.20}$$

$$0\leq d_i\leq R_i^p, \text{ integer}, \qquad \forall i\in\mathcal{D}, \tag{5.21}$$

$$0\leq w_i\leq R_i^w, \text{ integer}, \qquad \forall i\in\mathcal{D}, \tag{5.22}$$

$$0\leq w_i\leq R_i^a, \text{ integer}, \qquad \forall i\in\mathcal{A}, \tag{5.23}$$

$$t_i^T, \text{ integer}, \qquad \forall i\in\mathcal{F}, \tag{5.24}$$

$$t_i, \text{ integer}, \qquad \forall i\in\mathcal{F}, \tag{5.25}$$

$$\delta_{ij}\in\{0,1\}, \qquad \forall i,j\in\mathcal{F}, i\neq j, \tag{5.26}$$

$$\alpha_{is}\in\{0,1\}, \qquad \forall i\in\mathcal{D}, \forall s\in\mathcal{T}, \tag{5.27}$$

$$\beta_{is}\in\{0,1\}, \qquad \forall i\in\mathcal{D}, \forall s\in\mathcal{T}, \tag{5.28}$$

$$z_{ih}\in\{0,1\}, \qquad \forall i\in\mathcal{A}, \forall h\in\mathcal{H}, \tag{5.29}$$

$$\beta_{ihs}\in\{0,1\}, \qquad \forall i\in\mathcal{A}, \forall h\in\mathcal{H}, \forall s\in\mathcal{T}. \tag{5.30}$$

Constraints (5.2) ensure only one holding point for each arrival. Constraints (5.3) link the time arriving at runway threshold and the pushback delay for departures. Constraints (5.4) link the time

arriving at holding point and the holding point decision variables for arrivals. Constraints (5.5) state the runway usage time as a sum of time to reach the holding point and the holding time. Constraints (5.6) guarantee that given any two aircraft, one leads the other. Constraints (5.7) and (5.8) ensure the first-come-first-served (FCFS) order at the runway threshold and for take-off as well as the minimum separation for departures. There are two cases to consider here: if $\delta_{ij} = 1$, i.e., i uses runway before j , then Constraints (5.7) and Constraints (5.8) become $t_j^T \geq t_i^T$ and $t_j - t_i \geq s_{ij}$, ensuring that the FCFS order and the separation are respected. If $\delta_{ij} = 0$, j uses runway before i, then Constraints (5.7) and Constraints (5.8) become $t_j^T - t_i^T \geq -M$ and $t_j - t_i \geq s_{ij} - M$, i.e., $t_j^T - t_i^T$ and $t_j - t_i$ are bigger than some large negative values, thereby ensuring that this constraint is effectively inactive. Similarly, Constraints (5.9) and Constraints (5.10) guarantee that at the same holding point, the crossing order of two landing aircraft keeps the same with the order they reach the holding point, and ensure the minimum separation while crossing. Constraints (5.11) guarantee the minimum separation between arrival and departure (and vice versa). Constraints (5.12), (5.13) and (5.14) ensure the connectivity in time. For example, if flight i arrives at the runway threshold before time s, then for all the time intervals $s' \geq s$, $\alpha_{is'}$ has to have a value of 1. Constraints (5.15) and Constraints (5.16) link the two binary variables to the runway usage time and the arrival time at runway threshold for departures. Constraints (5.17) ensure that $\beta_{ihs} = 0$ if flight i does not use holding point h. Constraints (5.18) is similar to Constraints (5.15), that link the binary variable β_{ihs} to the runway usage time for arrivals. Constraints (5.19) ensure that at every time the number of aircraft waiting at the departure runway threshold will not exceed the departure capacity. The term $\beta_{is} + \alpha_{is} - 1$ is equal to 1 if and only if flight i is waiting at runway threshold during the time interval s. Constraints (5.20) ensure that at every time the number of crossing aircraft waiting at the holding point will not exceed the holding capacity. Notice that, in comparison with the departures, we do not need to introduce variables α_{ihs} for arrivals since the arrival time at runway threshold is known as soon as the holding point is given.

Considering the LP relaxation of the problem, we would like M to be as small as possible, and this can be achieved by tailoring M for each constraint. For example in Constraints (5.7), M can be set to $\max(0, T_i^0 + R_i^p - T_j^0)$. Moreover, as we know the time range where each aircraft can use the runway, we can strengthen the formulation by fixing the value of some variables:

$$\alpha_{is} = \begin{cases} 1, & \text{for } s \in \{T_{iR_i^p}, \ldots, T\}, \\ 0, & \text{for } s \in \{1, \ldots, T_i^0 - 1\}; \end{cases} \quad \forall i \in \mathcal{D};$$

$$\beta_{is} = \begin{cases} 1, & \text{for } s \in \{1, \ldots, T_i^0\}, \\ 0, & \text{for } s \in \{T_{iR_i} + 1, \ldots, T\}; \end{cases} \quad \forall i \in \mathcal{D};$$

$$\beta_{ihs} = 0, \text{ for } s \in \{0, \ldots, T_{ih}^0 - 1\}; \quad \forall i \in \mathcal{A}, h \in \mathcal{H}.$$

5.2.2 Model B: delay-indexed formulation.

The Decision Variables. We define the following binary decision variables:

$$x_{ir} = \begin{cases} 1, & \text{if the take-off time of flight } i \text{ is delayed by } r \text{ slots,} \\ 0, & \text{otherwise;} \end{cases} \text{ for } \quad i \in \mathcal{D}, r \in \mathcal{R}_i$$

$$x_{ihr} = \begin{cases} 1, & \text{if } i \text{ crosses via holding point } h \text{ and is delayed by } r \text{ slots,} \\ 0, & \text{otherwise;} \end{cases} \text{ for } \quad i \in \mathcal{A}, h \in \mathcal{H}, r \in \mathcal{R}_i^a$$

$$y_{ir} = \begin{cases} 1, & \text{if the pushback time of flight } i \text{ is delayed by } r \text{ slots,} \\ 0, & \text{otherwise;} \end{cases} \text{ for } \quad i \in \mathcal{D}, r \in \mathcal{R}_i^p$$

$$\delta_{ij} = \begin{cases} 1, & \text{if } i \text{ uses runway before } j, \\ 0, & \text{otherwise;} \end{cases} \text{ for } \quad i \in \mathcal{F}, j \in \mathcal{F}, i \neq j$$

$$\alpha_{is} = \begin{cases} 1, & \text{if } i \text{ arrives at the runway threshold after time } s, \\ 0, & \text{otherwise;} \end{cases} \quad \text{for} \quad i \in \mathcal{D}, s \in \mathcal{T}$$

$$\beta_{is} = \begin{cases} 1, & \text{if } i \text{ takes off before time } s, \\ 0, & \text{otherwise;} \end{cases} \quad \text{for} \quad i \in \mathcal{D}, s \in \mathcal{T}$$

$$\beta_{ihs} = \begin{cases} 1, & \text{if } i \text{ crosses via holding point } h \text{ before time } s, \\ 0, & \text{otherwise;} \end{cases} \quad \text{for} \quad i \in \mathcal{A}, h \in \mathcal{H}, s \in \mathcal{T}$$

The Objective Function. Our objective is expressed by:

$$\text{Min} \sum_{i \in \mathcal{D}} \sum_{r \in \mathcal{R}_i} rx_{ir} + \sum_{i \in \mathcal{A}} \sum_{h \in \mathcal{H}} \sum_{r \in \mathcal{R}_i^a} rx_{ihr}$$

The Constraints. In order to introduce the constraints, we define the following sets to represent some infeasible assignments pairs of delays (and crossing point for arrivals). Given two distinct flights $i, j \in \mathcal{F}$, we introduce:

$$\mathcal{C}_{ij} = \{(r,q) \in \mathcal{R}_i \times \mathcal{R}_j | T_{jq} - T_{ir} < s_{ij} \text{ and } T_{ir} - T_{jr} < s_{ji}\} \text{ if } (i,j) \in \mathcal{D} \times \mathcal{D}$$

the set of delay assignments which violates the separation requirements between two departure flights, and:

$$\mathcal{C}_{ijh} = \begin{cases} \{(r,q) \in \mathcal{R}_i \times \mathcal{R}_j^a | T_{jhq} - T_{ir} < s_{ij} \text{ and } T_{ir} - T_{jhq} < s_{ji}\} & \text{if } (i,j) \in \mathcal{D} \times \mathcal{A} \\ \{(r,q) \in \mathcal{R}_i^a \times \mathcal{R}_j^a | (T_{ih}^0 - T_{jh}^0)(T_{ihr} - T_{jhq}) < 0 \text{ or } (T_{jhq} - T_{ihr} < s_{ij} \text{ and } T_{ihr} - T_{jhq} < s_{ji})\} & \text{if } (i,j) \in \mathcal{A} \times \mathcal{A} \end{cases}$$

the set of delay assignments which violates the separation requirements between departure and arrival or between two arrivals using the same holding point. The first condition in the set definition expresses the violation of the FCFS order for arrival holding point. Notice that it is not possible to express the FCFS order for departure queue directly in the set definition. It is ensured by Constraints (5.40) and (5.41).

Constraints (5.32) and (5.33) require exactly one time period of runway usage for departure and arrival flights. Constraints (5.34) guarantee that for departures, the arrival time at the runway threshold is always smaller than the take-off time. Constraints (5.35) state the maximum holding time for departures. Constraints (5.36), Constraints (5.37), and Constraints (5.38) ensure the separation criteria between two departures, arrival and departure, and two arrival crossings respectively. Constraints (5.39) guarantee that given any two aircraft, one leads the other. Constraints (5.40) and Constraints (5.41) guarantee the same order waiting at the runway threshold and for take-off for departures. Constraints (5.42–5.44) are the same as Constraints (5.12–5.14) in Model A. Constraints (5.45) and Constraints (5.46) link the binary variables for departures. Constraints (5.47) is similar to Constraints (5.45), that link the binary variables for arrivals. Constraints (5.48–5.49) are the same as Constraints (5.19–5.20) in Model A.

One drawback of this discrete time formulation is a relatively large number of variables (in particular, Constraints (5.36–5.38)). In order to reduce the number of variables, we reformulate Constraints (5.36) and Constraints (5.37) to Constraints (5.57), and Constraints (5.38) to Constraints (5.58),

$$\text{Min} \sum_{i\in\mathcal{D}}\sum_{r\in\mathcal{R}_i} rx_{ir} + \sum_{i\in\mathcal{A}}\sum_{h\in\mathcal{H}}\sum_{r\in\mathcal{R}_i^a} rx_{ihr}, \tag{5.31}$$

$$\sum_{r\in\mathcal{R}_i} x_{ir} = 1, \qquad \forall i \in \mathcal{D}, \tag{5.32}$$

$$\sum_{h\in\mathcal{H}}\sum_{r\in\mathcal{R}_i^a} x_{ihr} = 1, \qquad \forall i \in \mathcal{A}, \tag{5.33}$$

$$\sum_{r\in\mathcal{R}_i} rx_{ir} - \sum_{r\in\mathcal{R}_i^p} ry_{ir} \geq 0, \qquad \forall i \in \mathcal{D}, \tag{5.34}$$

$$\sum_{r\in\mathcal{R}_i} rx_{ir} - \sum_{r\in\mathcal{R}_i^p} ry_{ir} \leq R_i^w, \qquad \forall i \in \mathcal{D}, \tag{5.35}$$

$$x_{ir} + x_{jq} \leq 1, \qquad \forall (r,q) \in \mathcal{C}_{ij}, \forall i,j \in \mathcal{D}, i < j, \tag{5.36}$$

$$x_{ir} + x_{jhq} \leq 1, \qquad \forall (r,q) \in \mathcal{C}_{ijh}, \forall h \in \mathcal{H}, \forall i \in \mathcal{D}, \forall j \in \mathcal{A}, \tag{5.37}$$

$$x_{ihr} + x_{jhq} \leq 1, \qquad \forall (r,q) \in \mathcal{C}_{ijh}, \forall h \in \mathcal{H}, \forall i,j \in \mathcal{A}, i < j, \tag{5.38}$$

$$\delta_{ij} + \delta_{ji} = 1, \qquad \forall i,j \in \mathcal{D}, i < j, \tag{5.39}$$

$$\sum_{r\in\mathcal{R}_j} T_{jr}x_{jr} - \sum_{r\in\mathcal{R}_i} T_{ir}x_{ir} \geq -M\delta_{ij}, \qquad \forall i,j \in \mathcal{D}, i \neq j, \tag{5.40}$$

$$\sum_{r\in\mathcal{R}_j^p} T_{jr}y_{jr} - \sum_{r\in\mathcal{R}_i^p} T_{ir}y_{ir} \geq -M\delta_{ij}, \qquad \forall i,j \in \mathcal{D}, i \neq j, \tag{5.41}$$

$$\alpha_{is} \leq \alpha_{i(s+1)}, \qquad \forall i \in \mathcal{D}, \forall s \in \mathcal{T}\backslash\{T\}, \tag{5.42}$$

$$\beta_{is} \geq \beta_{i(s+1)}, \qquad \forall i \in \mathcal{D}, \forall s \in \mathcal{T}\backslash\{T\}, \tag{5.43}$$

$$\beta_{ihs} \geq \beta_{ih(s+1)}, \qquad \forall i \in \mathcal{A}, \forall h \in \mathcal{H}, \forall s \in \mathcal{T}\backslash\{T\}, \tag{5.44}$$

$$\sum_{r\in\mathcal{R}_i} T_{ir}x_{ir} - \sum_{s\in\mathcal{T}} \beta_{is} = 0, \qquad \forall i \in \mathcal{D}, \tag{5.45}$$

$$\sum_{r\in\mathcal{R}_i^p} T_{ir}y_{ir} - \sum_{s\in\mathcal{T}} (1-\alpha_{is}) = 0, \qquad \forall i \in \mathcal{D}, \tag{5.46}$$

$$\sum_{r\in\mathcal{R}_i^a} T_{ihr}x_{ihr} - \sum_{s\in\mathcal{T}} \beta_{ihs} = 0, \qquad \forall i \in \mathcal{A}, \forall h \in \mathcal{H}, \tag{5.47}$$

$$\sum_{i\in\mathcal{D}} (\beta_{is} + \alpha_{is} - 1) \leq C_d, \qquad \forall s \in \mathcal{T}, \tag{5.48}$$

$$\sum_{i\in\mathcal{A}} \beta_{ihs} \leq C_a, \qquad \forall s \in \mathcal{T}, \forall h \in \mathcal{H}, \tag{5.49}$$

$$x_{ir} \in \{0,1\}, \qquad \forall i \in \mathcal{D}, \forall r \in \mathcal{R}_i, \tag{5.50}$$

$$x_{ihr} \in \{0,1\}, \qquad \forall i \in \mathcal{A}, \forall h \in \mathcal{H}, \forall r \in \mathcal{R}_i^a \tag{5.51}$$

$$y_{ir} \in \{0,1\}, \qquad \forall i \in \mathcal{D}, \forall r \in \mathcal{R}_i^p, \tag{5.52}$$

$$\delta_{ij} \in \{0,1\}, \qquad \forall i,j \in \mathcal{D}, i \neq j, \tag{5.53}$$

$$\alpha_{is} \in \{0,1\}, \qquad \forall i \in \mathcal{D}, \forall s \in \mathcal{T}, \tag{5.54}$$

$$\beta_{is} \in \{0,1\}, \qquad \forall i \in \mathcal{D}, \forall s \in \mathcal{T}, \tag{5.55}$$

$$\beta_{ihs} \in \{0,1\}, \qquad \forall i \in \mathcal{A}, \forall h \in \mathcal{H}, \forall s \in \mathcal{T}, \tag{5.56}$$

$$x_{ir} + \sum_{j\in\mathcal{D}, j\neq i\, q|(r,q)\in\mathcal{C}_{ij}} \sum x_{jq} + \sum_{h\in\mathcal{H}} \sum_{j\in\mathcal{A}\, q|(r,q)\in\mathcal{C}_{ijh}} \sum x_{jhq} \leq 1 + M(1-x_{ir}), \qquad \forall r \in \mathcal{R}_i, \forall i \in \mathcal{D}, \quad (5.57)$$

$$x_{ihr} + \sum_{j\in\mathcal{A}, j\neq i\, q|(r,q)\in\mathcal{C}_{ijh}} \sum x_{jhq} \leq 1 + M(1-x_{ihr}), \qquad \forall r \in \mathcal{R}_i^a, \forall h \in \mathcal{H}, \forall i \in \mathcal{A}. \quad (5.58)$$

There are two cases to consider: if $x_{ir} = 1$, then the take-off time of flight i is delayed by r slots, Constraints (5.57) become $\sum_{j\in\mathcal{D}, j\neq i\, q|(r,q)\in\mathcal{C}_{ij}} \sum x_{jq} + \sum_{h\in\mathcal{H}} \sum_{j\in\mathcal{A}\, q|(r,q)\in\mathcal{C}_{ijh}} \sum x_{jhq} \leq 0$, ensuring that i is separated with respect to all the other flights. If $x_{ir} = 0$, then Constraints (5.57) become $\sum_{j\in\mathcal{D}, j\neq i\, q|(r,q)\in\mathcal{C}_{ij}} \sum x_{jq} + \sum_{h\in\mathcal{H}} \sum_{j\in\mathcal{A}\, q|(r,q)\in\mathcal{C}_{ijh}} \sum x_{jhq} \leq 1 + M$, which inactivates the constraints. The same reasoning applies to Constraints (5.58).

The decision variables of Model A can be linked to the decision variables of Model B using the following equations (5.59–5.62):

$$w_i = \sum_{r\in\mathcal{R}_i} i x_{ir}, \qquad \forall i \in \mathcal{D}, \quad (5.59)$$

$$d_i = \sum_{r\in\mathcal{R}_i^p} i y_{ir}, \qquad \forall i \in \mathcal{D}, \quad (5.60)$$

$$w_i = \sum_{h\in\mathcal{H}} \sum_{r\in\mathcal{R}_i^a} i x_{ihr}, \qquad \forall i \in \mathcal{A}, \quad (5.61)$$

$$z_{ih} = \sum_{r\in\mathcal{R}_i^a} x_{ihr}, \qquad \forall i \in \mathcal{A}, \forall h \in \mathcal{H}, \quad (5.62)$$

It is possible to remove all the "big-M" constraints in Model B by introducing the decision variables x_{irq} instead of x_{ir} and y_{ir}, where

$$x_{irq} = \begin{cases} 1, & \text{if the pushback time of flight } i \text{ is delayed by } r \text{ slots} \\ & \text{and flight } i \text{ holds } q \text{ slots at the runway threshold,} \qquad \text{for} \quad i \in \mathcal{D}, r \in \mathcal{R}_i^p, q \in \mathcal{R}_i^w \\ 0, & \text{otherwise;} \end{cases}$$

However, experimental tests showed that the benefit is counterbalanced by the large increase of the number of variables and constraints.

After describing our two ILP formulations, in the next section we consider also the metaheuristic approach to tackling the aircraft runway scheduling problem.

5.3 Adaptation of SA to the runway sequencing problem

In this section we present a simulated annealing algorithm to solve our aircraft runway scheduling problem. In order to apply SA to our problem, the neighborhood generation is critical and needs to be defined precisely. Our neighborhood choice is similar to a Traveling Salesman Problem with a constrained search space. The algorithm starts with an initial sequence sorted by the earliest runway usage time. First, we randomly choose one flight, and we search for a list of sequence positions that can be used by the current chosen flight without exceeding the maximum delay. Then, we choose one flight in the list and apply three exchange strategies with the current one: *swap*, *inversion*, and

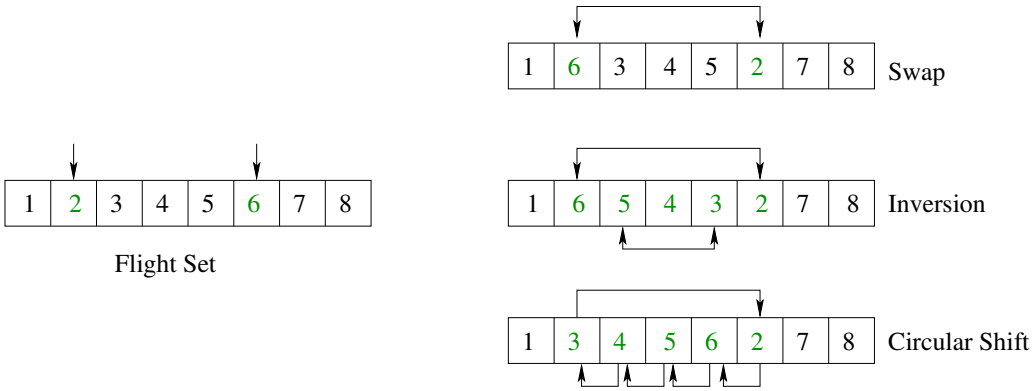


Figure 5.3: Illustration of three neighborhood generation methods: swap, i.e., exchanging the positions of two aircraft; inversion, i.e., inverting the order of aircraft between two positions; circular shift, i.e., moving the first aircraft to the final position, while shifting all other aircraft to the previous position.

circular shift. As shown in Fig. 5.3, swap move exchanges the positions of two aircraft, inversion move inverts the order of aircraft between two positions, and circular shift move changes the first aircraft to the final position, while shifting all other aircraft to the previous position. Note that it is necessary to relax the maximum pushback delay and maximum holding time constraints in order to allow SA to search more freely in the state space. Nevertheless, a penalty coefficient is added if one aircraft cannot meet the maximum delay constraints. At the end of the procedure, SA can always find a solution satisfying the maximum delay requirement. A comparison test with one hour traffic is made to find the most efficient neighborhood generation method. As indicated in Fig. 5.4, the strategy combining circular shift move and swap move can achieve less delay than other strategies. Thus, in the remaining part of the chapter, we choose circular shift move and swap move as our neighborhood generation methods.

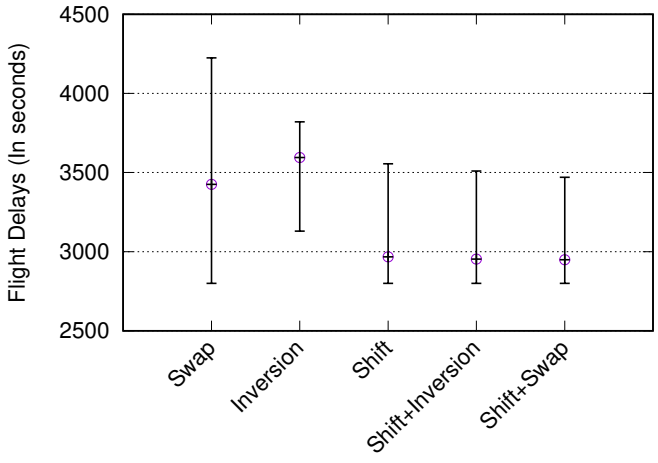


Figure 5.4: Comparison of different neighborhood methods on solution qualities. We run the proposed SA algorithm 10 times with the same input parameters for each method. The error bar represents the minimum delay, the average delay, and the maximum delay respectively.

After the previous step, we obtained a new sequence, and we now have to update the flight schedules: First, we set the earliest runway usage time of the first flight in the sequence as its actual runway usage time. Then, for each aircraft, its runway usage time is set with respect to its earliest available time and the minimum separation with its previous aircraft. Moreover, the best holding point for arrivals is determined. More precisely, we calculate for each holding point the required waiting time for the flight and choose the holding point with the shortest waiting time. Table 5.3 lists some SA parameters related to our problem.

Table 5.3: Empirically-set parameter values of SA.

Parameter	**Value**
Geometrical temperature reduction coefficient, δ	0.95
Number of transitions at each temperature step, N_T	100
Initial temperature, T_0	0.1
Final temperature, T_f	$0.0001T_0$

5.4 Computational results

In this section we present the results of some computational experiments. We test our methodology on both real data cases at Paris CDG Airport and randomly generated data. Numerical results with different settings of (user-defined) algorithm parameters are presented and discussed. The overall process is run on a 2.50 GHz core i7 CPU, PC under Linux operating system, implemented in the Java programming language. The ILP models were solved with GUROBI solver [108] with the default parameters settings. Table 5.4 lists some parameters related to our mathematical formulation. The actual pushback time from historical data is considered as the EOBT in this chapter. In our test cases, all the flights are without CFMU slots due to data unavailability.

Table 5.4: Chosen parameter values specifying the optimization problem.

Parameter	**Value**
Departure capacity at runway threshold, C_d	5
Arrival capacity at holding points, C_a	2
Time slot, ρ	5 seconds
Maximum pushback delay (in number of slots), R_i^p	120
Maximum holding time (in number of slots) for departures, R_i^w	120
Maximum holding time (in number of slots) for arrivals, R_i^a	36

In the current system, controllers schedule aircraft take-offs in a FCFS order with regard to their

earliest arrival time at runway threshold. In this study, we set a FCFS sequence as the baseline based on the earliest arrival time at runway threshold. We assume that all the arrival flights use the closest holding point after they land, i.e., S3. There is no limit on the maximum holding number and the maximum waiting time at the runway threshold. Moreover, a landing aircraft is assumed to cross the departure runway without any holding time. In a real-world case, arrivals do have priority on departures, as departures can be held on the ground. The arrival holding area is limited in terms of capacity and should be vacated as early as possible.

Table 5.5: Example of FCFS sequence and optimized sequence with a random instance of 5 arrivals and 10 departures. H refers to Heavy, M refers to Medium, C refers to Crossing. All the times are in seconds. P_i refers to the scheduled pushback time, X_i refers to taxi time from spot to runway threshold, L_i refers to the scheduled landing time, O_i is the runway occupancy time. t_i^T refers to the time arriving at the runway threshold, t_i refers to the runway usage time. ρw_i refers to the holding duration, and ρd_i refers to the pushback delay. $T_{e,h}$ refers to the taxi time from runway exit e to holding point h.

Data	D01	D02	D03	D04	D05	D06	D07	D08	D09	D10	A01	A02	A03	A04	A05
Category	**H**	M	M	**H**	M	M	M	M	M	M	C	C	C	C	C
P_i	0	10	20	30	40	50	60	70	80	90	–	–	–	–	–
X_i	300	300	300	300	300	300	300	300	300	300	–	–	–	–	–
L_i	–	–	–	–	–	–	–	–	–	–	0	100	200	300	400
O_i	–	–	–	–	–	–	–	–	–	–	60	60	60	60	60
e	–	–	–	–	–	–	–	–	–	–	V4	V4	V4	V4	V4
FCFS	**A01**	**A02**	**D01**	**A03**	**A04**	**D02**	**A05**	**D03**	**D04**	**D05**	**D06**	**D07**	**D08**	**D09**	**D10**
t_i^T	175	275	300	375	475	310	575	320	330	340	350	360	370	380	390
ρw_i	0	0	15	0	0	205	0	295	345	455	505	555	605	655	705
t_i	175	275	315	375	475	515	575	615	675	795	855	915	975	1035	1095
h	S3	S3	–	S3	S3	–	S3	–	–	–	–	–	–	–	–
$T_{e,h}$	115	115	–	115	115	–	115	–	–	–	–	–	–	–	–
Total delay								**4340**							
Optimized	**A01**	**A02**	**D02**	**D08**	**D10**	**D01**	**A03**	**A04**	**A05**	**D09**	**D05**	**D06**	**D07**	**D03**	**D04**
ρd_i	–	–	0	0	30	190	–	–	–	230	330	380	430	535	580
$P_i + \rho d_i$	–	–	10	70	120	190	–	–	–	310	370	430	490	555	610
t_i^T	175	275	310	370	420	490	419	519	575	610	670	730	790	855	910
ρw_i	0	0	5	5	15	5	140	50	0	5	5	5	5	0	5
t_i	175	275	315	375	435	495	559	569	575	615	675	735	795	855	915
h	S3	S3	–	–	–	–	S1	S1	S3	–	–	–	–	–	–
$T_{e,h}$	115	115	–	–	–	–	158	158	115	–	–	–	–	–	–
Total delay								**2950**							

To illustrate how we calculate the FCFS sequence and compare with the optimized sequence, we give an example by generating a random instance consisting of 15 aircraft (5 arrivals and 10 departures). As shown in Table 5.5, given the initial landing time or initial pushback time, we calculate the earliest arrival time at runway threshold, t_i^T. For departures, $t_i^T = P_i + X_i$; For arrivals, $t_i^T = L_i + O_i + T_{e,h}$. We assume that all the arrival flights use the closest holding point after they

land, i.e., S3. Then, we build a FCFS sequence by sorting t_i^T and we obtain the runway usage time t_i with regard to the separation requirements. The optimized sequence is presented in Table 5.5. We observed a decrease of total delay from 4340 s in the baseline case to 2950 s in the optimized case.

Table 5.6: Comparison of heuristics for solving one-hour real traffic on July 11^{th}, 2017. Best results are in bold.

Time Window	Number of flights			Gap (%)				Computational time (s)			
	All	Dep.	Arr.	FCFS	Model A	Model B	SA	Model B Optimum	Model A	Model B	SA
8:00–9:00	54	31	23	69.59	–	–	**2.35**	113.13	10.00	10.00	**3.35**
9:00–10:00	49	23	26	84.43	**3.92**	–	6.34	43.98	10.00	10.00	**3.60**
10:00–11:00	34	16	18	66.16	**0.00**	19.44	0.00	12.31	2.02	10.00	**1.57**
11:00–12:00	55	30	25	82.64	12.08	–	**2.15**	58.35	10.00	10.00	**3.82**
12:00–13:00	40	22	18	84.04	**0.00**	–	4.62	20.15	4.37	10.00	**1.94**
13:00–14:00	25	20	5	33.31	4.89	–	**4.37**	22.47	10.00	10.00	**0.41**
14:00–15:00	46	16	30	79.18	7.32	–	**5.04**	27.51	10.00	10.00	**3.31**
15:00–16:00	41	18	23	67.31	**0.00**	18.68	0.00	13.51	**2.22**	10.00	2.44
16:00–17:00	39	24	15	68.82	**0.00**	–	7.81	26.82	10.00	10.00	**1.52**
17:00–18:00	45	19	26	81.02	10.84	–	**2.06**	29.34	10.00	10.00	**2.94**
18:00–19:00	47	23	24	43.12	10.25	45.27	**0.12**	64.87	10.00	10.00	**3.00**
19:00–20:00	54	32	22	76.23	28.87	–	**5.55**	156.80	10.00	10.00	**3.21**

After this initial test, we chose a heavy traffic summer day, July 11^{th}, 2017 as our test case. An optimization time window of one hour is applied from 8:00 to 20:00; the rest of the day involves only a few flights. As illustrated in Table 5.6, real traffic data provide various mixes of arrivals and departures: three fully loaded periods of 8:00-9:00, 11:00-12:00, and 19:00-20:00 with more than 30 departures per hour per runway, and 14:00-15:00 with 30 arrivals per hour per runway are observed.

Table 5.6 shows the comparison of optimization results for four cases: FCFS, Model A, Model B, and SA. We list the gap, i.e., the percent increase in total delay over the optimal solution. The computation time in seconds is listed in the last four columns. For SA, the gap and the computation time are averaged over 10 times random runs. In order to be fair and practical in the real world application, we set the computational time limit to be 10 seconds for all the algorithms. As shown in Table 5.6, first we can see a significant reduction of delays from Model A and SA compared to FCFS sequence. Model B cannot find a feasible solution in most of the cases within the time limit. Model A can find a near-optimal solution. Nevertheless, it is difficult to prove the optimality due to the poor lower bound when the number of aircraft is large and the traffic scenario is dense. Therefore, we still use Model B to find the optimal solution with a reasonable time from 12 seconds up to 156 seconds, as shown in the first column of computational times in Table 5.6. In contrast, SA can find good quality solutions with most of the gaps less than 10 % within 5 seconds.

In order to test the performance of algorithms facing large instances, we set our time window to be two hours and the algorithm running time limit to be 20 seconds. The number of flights ranges from 65 to 103, which makes it challenging to find a good solution in a short period of time. As shown in Table 5.7, first we observe a large gap between FCFS solution and optimal solution. After optimization, SA can still find a near-optimal solution within the time limit compared to other

Table 5.7: Comparison of heuristics for solving two-hour real traffic on July 11^{th}, 2017. Best results are in bold.

Time Window	Number of flights			Gap (%)				Computation time (s)			
	All	Dep.	Arr.	FCFS	Model A	Model B	SA	Model B Optimum	Model A	Model B	SA
8:00–10:00	103	54	49	84.15	–	–	**3.92**	378.90	20.00	20.00	**15.68**
10:00–12:00	89	46	43	81.51	11.35	–	**0.46**	99.67	20.00	20.00	**10.35**
12:00–14:00	65	42	23	68.74	12.26	–	**2.65**	83.40	20.00	20.00	**3.91**
14:00–16:00	87	34	53	77.13	6.97	–	**4.21**	53.78	20.00	20.00	**11.22**
16:00–18:00	84	43	41	77.56	22.13	–	**3.42**	83.79	20.00	20.00	**8.81**
18:00–20:00	101	55	46	81.88	39.31	–	**5.12**	383.93	20.00	20.00	**13.14**

models. In period 18:00-20:00, the computational time is 384 seconds for Model B, even Model A has a gap of 39 % compared to optimality. This proves that SA is more suitable and more practical for dealing with large instances.

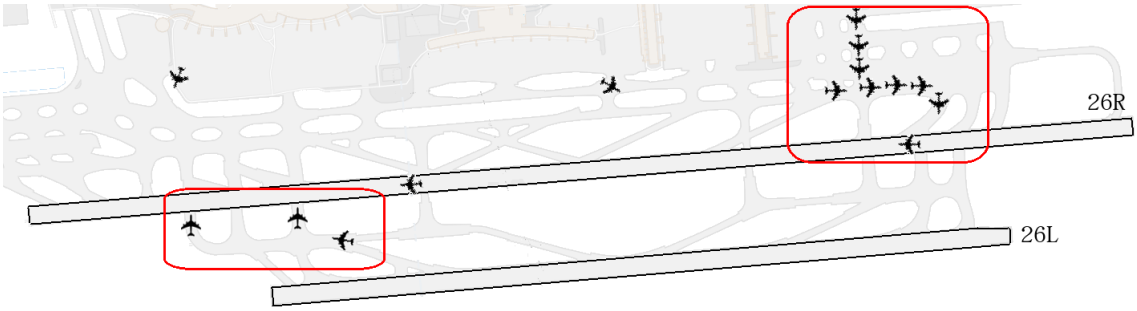
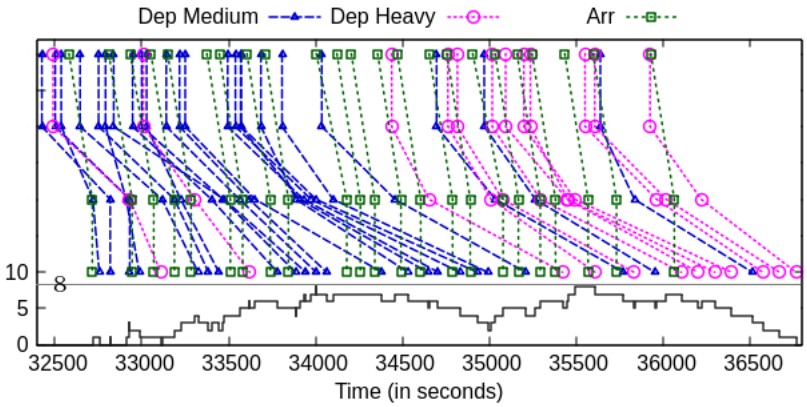


Figure 5.5: Departure queues at 9:30 at runway 26R at CDG Airport on February 18^{th}, 2016. Visualization of actual surface surveillance data.

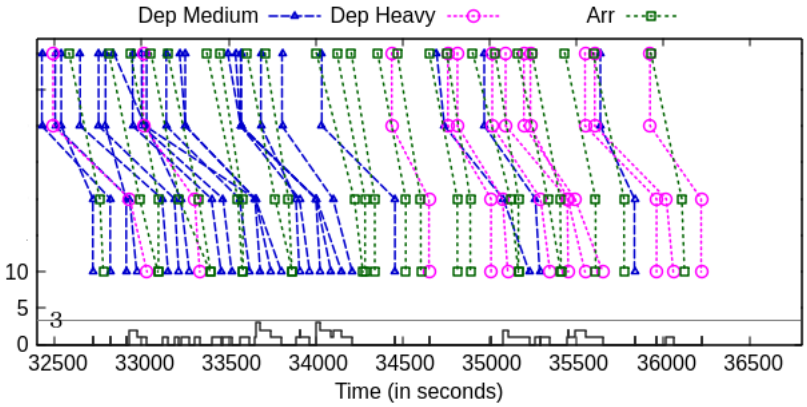
We test another heavy traffic day on February 18^{th}, 2016. A departure queue of 9 aircraft with 3 arrivals waiting at the holding point at 9:30 is illustrated in Fig. 5.5. As shown in Table 5.8, in period 9:00-10:00, there are 58 aircraft with 35 departures and 23 arrivals.

The optimization results are listed in Table 5.8. SA still keeps a similar performance compared to the first test. Model A also shows good performances to find the optimal solution within 10 seconds, except in period 9:00-10:00 with a gap of 41%. Note that period 9:00-10:00 is the period with the highest demand of the day. Below, we analyze in detail the optimization results of time window 9:00-10:00. Fig. 5.6 shows the sequencing results in period 9:00-10:00 for three cases: the FCFS sequence, the optimal sequence with a preference to wait at the holding point, and the optimal sequence with a preference to wait at the gate. One can observe a large departure delay as well as a long departure queue for FCFS case illustrated at the bottom of the Figure 5.6a. While in Fig. 5.6b, much less departure delays are achieved with slight modification of arrival holding time. The departure queue is reduced to 3 aircraft. In Fig. 5.6c, with a preference of holding at the gate, aircraft taxi to the runway threshold and take off smoothly.

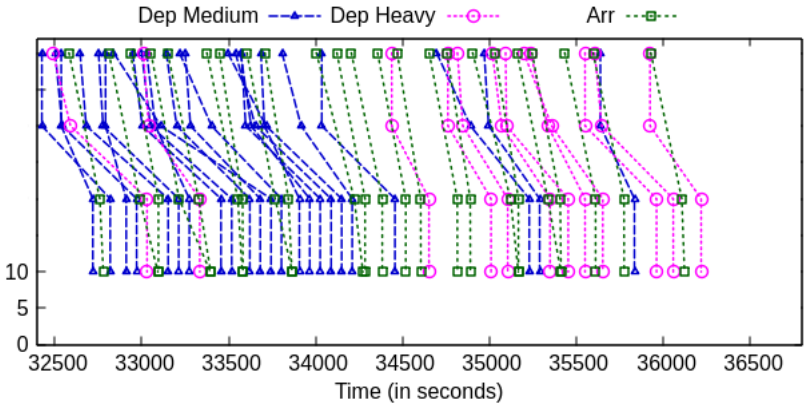
Next, we generate some random data set to check if our methods can be suitable for dense traffic scenarios. The mix of arrivals and departures, and the fleet mix of Medium and Heavy aircraft are



(a) FCFS sequence. Each line represents one flight, the four points from top to bottom for departures indicate respectively initial pushback time, actual pushback time, arriving time at the runway threshold, and take-off time; the three points from top to bottom for arrivals indicate respectively landing time, arriving time at the runway threshold, and crossing time. The departure waiting queue is shown at the bottom of the figure.



(b) Optimal sequence with a preference waiting at the holding point.



(c) Optimal sequence with preference waiting at the gate.

Figure 5.6: Comparison of the FCFS sequence and the optimal sequence with a preference waiting at the holding point, and optimal sequence with preference waiting at the gate.

Table 5.8: Comparison of heuristics for solving one-hour real traffic on February 18^{th}, 2016. Best results are in bold.

Time Window	Number of flights			Gap (%)				Computation time (s)			
	All	Dep.	Arr.	FCFS	Model A	Model B	SA	Model B Optimum	Model A	Model B	SA
6:00–7:00	41	19	22	61.23	11.79	–	**0.64**	86.73	10.00	10.00	**1.51**
7:00–8:00	48	13	35	71.16	**0.00**	53.67	3.81	14.57	5.35	10.00	**2.31**
8:00–9:00	34	17	17	57.49	**0.00**	0.00	0.00	8.20	1.25	8.20	**1.05**
9:00–10:00	58	35	23	85.31	41.09	–	**6.55**	353.74	10.00	10.00	**3.28**
10:00–11:00	48	21	27	79.81	8.46	–	**1.81**	106.17	10.00	10.00	**2.40**
11:00–12:00	42	20	22	26.98	1.57	–	**0.00**	51.01	10.00	10.00	**1.68**
12:00–13:00	42	30	12	33.09	**7.31**	–	8.48	124.70	10.00	10.00	**1.35**
13:00–14:00	37	14	23	89.82	**0.00**	0.00	1.96	6.22	1.36	6.22	**1.33**
14:00–15:00	41	23	18	47.05	**0.00**	69.09	2.54	24.38	6.80	10.00	**1.40**
15:00–16:00	40	20	20	45.77	**0.00**	–	0.99	16.02	5.39	10.00	**1.47**
16:00–17:00	38	10	28	58.86	**0.00**	4.21	0.00	11.51	4.76	10.00	**1.49**
17:00–18:00	46	25	21	84.14	9.09	–	**0.00**	25.20	10.00	10.00	**1.99**
18:00–19:00	45	19	26	69.45	**0.47**	–	0.61	80.24	10.00	10.00	**2.02**
19:00–20:00	46	26	20	73.77	5.26	–	**2.17**	34.19	10.00	10.00	**1.89**
20:00–21:00	41	21	20	74.54	**0.00**	–	2.36	28.71	10.00	10.00	**1.54**

two important factors in departure scheduling. Thus, we generate 15 random high traffic demand scenarios in a time window of 30 minutes. The number of arrivals/departures are set to be 10/20, 15/15, and 15/20. A mix of Heavy aircraft from 0% to 40% is set. Results are shown in Table 5.9: the solution quality decreases when the number of departures increases; SA keeps the shortest computational time; Model A is more suitable for cases with low departure rate.

As conclusion, Model B is suitable for finding an optimal solution with a large computation time while Model A is suitable for finding a feasible solution within a short computation time. In fact, by imposing a time limit, Model A solves most instances and sometimes reaches optimality. However, when the runway is in high demand, Model A is not able to find a good solution. The SA is suitable for finding a near-optimal solution in a short computational time.

5.5 Conclusions

The runway system is a major source of delay in the departure process It is critical to achieve efficient scheduling of aircraft taking into account specific operational constraints. In this chapter, we have developed two ILP models and one metaheuristic algorithm for departure runway scheduling incorporating arrival crossings. Specific constraints such as wake turbulence separations, flight time window restrictions, and holding queue capacity at runway threshold are explicitly considered. Different comparison tests on real and random data were launched and demonstrate that: the time slot based formulation can find an optimal or near-optimal solution quickly, but it is hard to prove optimality facing high demand periods; the delay-indexed formulation can prove optimality in a reasonable time; SA is a good candidate for having a near-optimal solution to reduce flight delays, with

Table 5.9: Comparison of heuristics for solving random data with different fleet mix. A time window of 30 minutes is considered, "10–20–0–100" means 10 arrivals and 20 departures with a fleet mix of 0% Heavy and 100% Medium. Best results are in bold.

Arr–Dep–H%–M%	Gap (%)				Computation time (s)			
	FCFS	Model A	Model B	SA	Model B Optimum	Model A	Model B	SA
10–20–0–100	68.36	15.14	–	**8.01**	97.07	10.00	10.00	**0.58**
10–20–10–90	74.54	27.99	–	**5.99**	80.35	10.00	10.00	**0.57**
10–20–20–80	81.77	**0.00**	–	1.41	31.93	10.00	10.00	**0.56**
10–20–30–70	64.55	28.23	–	**8.80**	601.98	10.00	10.00	**0.59**
10–20–40–60	85.78	3.57	–	**3.57**	36.80	10.00	10.00	**0.57**
15–15–0–100	59.00	**0.00**	–	1.51	25.68	10.00	10.00	**0.63**
15–15–10–90	91.28	**0.00**	–	11.45	13.03	4.06	10.00	**0.70**
15–15–20–80	59.55	19.15	–	**4.78**	146.53	10.00	10.00	**0.67**
15–15–30–70	93.73	**0.00**	–	7.53	15.40	8.59	10.00	**0.73**
15–15-40–60	89.47	**0.00**	–	5.99	13.60	10.00	10.00	**0.74**
15–20–0–100	70.74	40.16	–	**15.54**	52.63	10.00	10.00	**0.96**
15–20–10–90	36.28	15.40	–	**6.21**	86.23	10.00	10.00	**1.01**
15–20–20–80	25.94	5.71	–	**0.32**	154.06	10.00	10.00	**1.02**
15–20–30–70	32.39	**0.64**	–	1.77	48.41	10.00	10.00	**1.11**
15–20–40–60	54.74	22.68	–	**9.03**	2194.23	10.00	10.00	**1.05**

sufficiently small run time compatible with real-time application. The runway sequencing algorithm can be implemented as a decision support tool for controllers in actual operations. However, the benefits of the runway sequencing algorithm may be limited by the uncertainty arising from real operations. For real-time application, flight information needs to be updated frequently and accurately as time passes. For example, the taxi time to the runway threshold can vary with regard to the current aircraft position at the airport after pushback. A proper prediction of the travel time on the surface can help the tactical runway sequencing algorithms to better organize the sequence and to decrease flight delays. Future extensions of the model could develop a robust algorithm facing uncertainties on pushback time and taxi time.

Conclusions and Perspectives

This chapter summarizes the main contribution of this thesis and discusses directions for future research.

Summary of this thesis

Airports and surrounding airspaces are limited in terms of capacity and represent the major bottlenecks of the air traffic management system. In this thesis, we developed a two-level approach to optimizing the air traffic of airport and its surrounding terminal airspace, as shown in Fig. 5.7. The complex interactions and different prediction times of the airport traffic management motivate us to tackle the integrated problem at two levels:

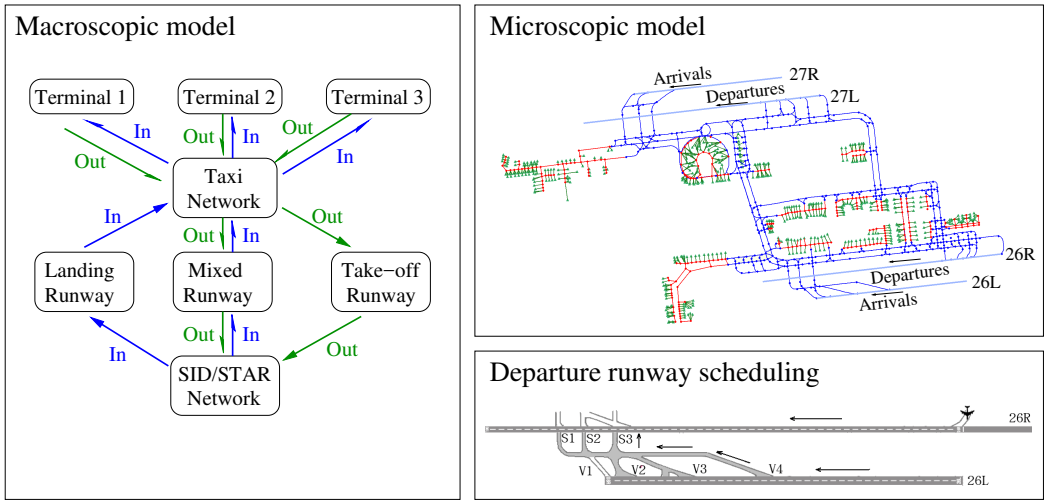


Figure 5.7: Two-level approach to optimizing the integrated air traffic optimization of airport and TMA airspace.

First, we addressed the problems of terminal airspace management and airport congestion management at the macroscopic level through the integrated control of arrivals and departures. Conflict detection and resolution methods were applied to a predefined SID/STAR route structure. The airside was modeled as an abstract network: terminal, taxi network, and runway were seen as specific resources with a defined maximum capacity. This level of abstraction could help better understanding the airport congestion situations. Optimization was carried out from the entry of TMA until the exit of this TMA. Speed, arrival and departure times, and runway assignment were managed to synchronize the air and ground traffic flows. An adapted simulated annealing heuristic combined with a time decomposition approach was proposed to solve the corresponding problem. In a moving

time frame, flights were classified into four statuses based on the current positions relative to the optimization time window.The time sliding window manager updates flight status and puts them into the optimization process. Computational experiments performed on case studies of Paris Charles De-Gaulle airport showed some potential improvements: First, when the capacity of a certain resource in the airport (terminal, taxi network) was decreased, until a certain threshold, the overload could be mitigated properly by adjusting the aircraft entry time in the TMA and the pushback time. Second, landing and take-off runway assignments in peak hours with imbalanced runway load could reduce flight delays.

Note that at the macroscopic level, airport surface was modeled as an abstract network with regard to the long prediction time. At the microscopic level, we considered runway and taxiway schedules with a detailed node-link network model. Accurate aircraft trajectories were calculated with respect to the chosen route and the allowed taxi speed. An optimization model was proposed to solve the coordinated surface operations problem and runway sequencing problem. A comparison of the optimization results with the baseline scenarios was presented for two major airports: Paris Charles De-Gaulle airport (CDG) in Europe and Charlotte Douglas International airport (CLT) in the US. We described the airport surface operations with an emphasis on their similarities and differences. The two airports have different surface infrastructure characteristics such as holding area, runway configuration, taxiway layout, etc. The two airports handle approximately the same number of aircraft movements. However, the fleet mix at the two airports are significantly different. Considering all these differences, higher benefits with departure metering were observed at CLT than at CDG. At CLT, where there are lots of interactions between departures and arrivals on the ramp area, we obtained significant taxi-out as well as taxi-in time savings through optimization by gate-holding strategy. The reduction in taxi-in time arose because of a better sequencing of runway crossings. Departure runway queue length was controlled and decreased without under-utilizing the runway. Lower benefits were observed at CDG since the airport was relatively less congested.

In the last part of the thesis, we investigated a sub-problem from the microscopic level by proposing exact and heuristic approaches to solve it. In many hub airports with parallel runways, arrivals have to cross departure runway to reach the taxiway. A better sequence of departures taking into account arrival crossings could achieve less flight delay. We presented two ILP formulations that modeled the holding area and the queue for both departures at the runway threshold and arrivals at the holding point. The two ILP models mainly differ in the way the decision variables are defined, one used time slot based formulation, and the other one used delay-indexed formulation. Moreover, we applied the SA algorithm and compared the gap of total delay between optimality and heuristic solution and the computation times. Comparison tests were conducted for the Southern side of CDG and showed that the three proposed methods could significantly improve the solutions based on the simple first-come-first-served rule. The delay-indexed formulation was suitable for finding an optimal solution while the time slot based formulation was suitable for finding a feasible solution within a short computation time. When the runway was in high demand, SA was suitable for finding a near-optimal solution in a short computational time that could be used for real-time deployment.

Perspectives

Several research directions can be followed in the future work:

Incorporating uncertainty in the macroscopic model

In real operations, aircraft may not be able to follow its assigned trajectory with high precision due to uncertainties from weather, passenger delay, etc. Moreover, the later the aircraft is involved in the optimization process with regard to the current time, higher is the uncertainty. In order to improve the robustness of the integrated optimization, one could consider the evolutions of the uncertainty during the flight time, and take into account uncertainty of aircraft position and arrival time in the mathematical model. One can smooth the airport occupancy curve considering the current time and future time window. As shown in Fig. 5.8, the red curve represents the deterministic airport occupancy curve, and the blue line indicates the required maximum capacity. Three congestion peaks exceed the maximum capacity and need to be mitigated. However, the priority of these three peaks is not considered in the deterministic model. The one which is close to the current time is more important to be mitigated, the one which is still far from the current time can be mitigated in the future. Therefore, instead of mitigating the congestion of the red curve, one could apply the algorithm to the green curve, which is a filtered version of the red one considering the current time and the evolution of traffic.

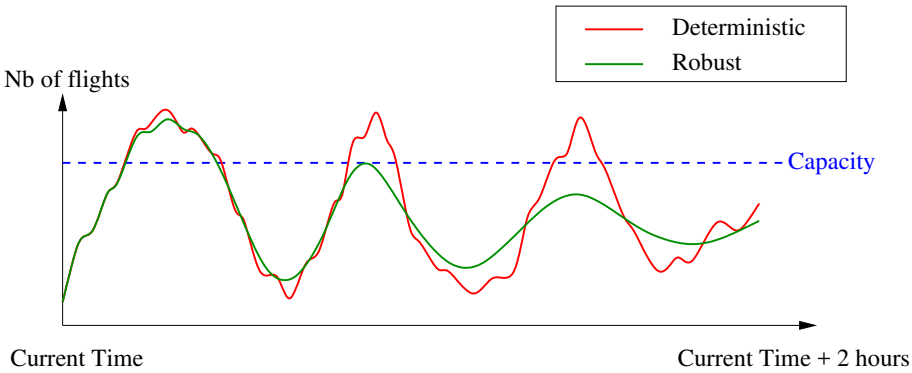


Figure 5.8: Improvement on the deterministic airport occupancy curve by taking into account the position of congestion period with regard to current time.

Extensions of the surface operation problem

Large airports have multiple departure runways. The microscopic model could integrate assigning departures to the proper runway taking into account the taxi distance, the current queue length of each runway, and the SID waypoint after take-off. Moreover, in European airports, the CTOT slots are applied to a certain number of departures to prevent too many aircraft in the air at the same time and place. We shall consider integrating the CTOT slots in the model either as a constraint or as an objective to minimize the number of flights that miss their assigned slots. Airport gate assignment problem could also be integrated in the model since the gate is the origin of departures and the destination of arrivals. Arrivals might be delayed and wait at the ramp area with engines on if their assigned gates are occupied by departures which do not push back yet.

Adaptations to multi-airport systems

In this thesis, we focused on the air traffic optimization at one airport, the same approaches for the macroscopic model could be applied to a multi-airport system. Multiple airport systems exist in

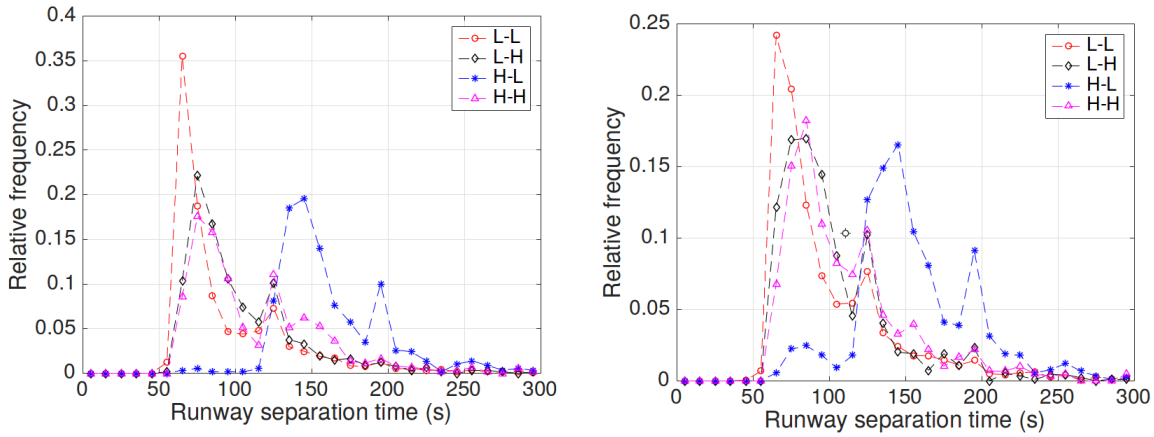
many metropolitan areas, for which several airports share the same terminal airspace. The interactions between the traffic flows from and to different airports make the problem more complicated when considering the terminal airspace management and the airport congestion management. As the complexity of such a problem is higher compared to our current model, additional decision variables and constraints need to be considered as well as extra evaluation metrics.

Multi-objective optimization problem

The design of algorithms for most ATM problems yields a range of multi-objective optimization problems. New formulations could be investigated to handle multi-objective optimization and multiple constraints in the context of airport operation management and terminal airspace management. Decision makers could select their preferred solution from the obtained Pareto optimal set according to different needs of stakeholders.

Appendix A

Departure runway separations



(a) Runway separation time distribution based on wake vortex category on runway 26R at CDG airport. 'L-H' means Large followed by Heavy aircraft.

(b) Runway separation time distribution based on wake vortex category on runway 27L at CDG airport.

Figure A.1: Runway separation time distribution at CDG.

Considering the runway configuration and the IMC/VMC flight conditions of airport, we apply the minimum runway separation based on statical analysis with regard to airport instead of a common minimum separation standard given in Chapter 4. Fig. A.1 illustrates the runway time separation of different wake turbulence categories at CDG. Based on these distributions, we apply the runway separation as shown in Table A.1 for runway 26R and in Table A.2 for runway 27L. Moreover, we assume that the minimum separation for a departure followed by an arrival crossing is 90 seconds, for an arrival crossing followed by a departure is 60 seconds, between two consecutive crossings using a common holding point is 20 seconds.

In the case of CLT, there are predominantly large aircraft operating at the airport. Moreover, CLT operates under VMC in good weather conditions, resulting in smaller runway separations. Thus, we apply the following runway separation standards based on empirical distributions as shown in Table A.3:

Table A.1: Departure runway separation on runway 26R at CDG (in seconds).

Category		**Leading Aircraft**	
		Large	***Heavy***
Trailing Aircraft	***Large***	91	153
	Heavy	101	107

Table A.2: Departure runway separation on runway 27L at CDG (in seconds).

Category		**Leading Aircraft**	
		Large	***Heavy***
Trailing Aircraft	***Large***	95	146
	Heavy	100	106

Table A.3: Departure runway separation at CLT.

Aircraft movements	Time separation (in seconds)
Two take-offs	65
Take-off or Crossing after landing	70
Landing or Crossing after take-off	60
Landing or take-off after crossing	40
Two crossings	20

Bibliography

[1] Airbus, "The market, long and short-term outlook." https://www.airbus.com/content/dam/corporate-topics/.../Global-Market-Forecast.pdf, 2016.

[2] D. Delahaye, S. Chaimatanan, and M. Mongeau, "Simulated annealing: From basics to applications," in *Handbook of Metaheuristics*, pp. 1–35, Springer, 2019.

[3] S. Badrinath, H. Balakrishnan, E. Clemons, and T. Reynolds, "Evaluating the impact of uncertainty on airport surface operations," in *2018 Aviation Technology, Integration, and Operations Conference*, 2018.

[4] Airbus, "Global market forecast 2018-2037." https://www.airbus.com/content/dam/corporate-topics/...day/GMF-2018-2037.pdf, 2018.

[5] A. Odoni and R. De Neufville, *Airport systems: Planning, design, and management*. McGraw-Hill Professional, 2003.

[6] Eurocontrol, "European aviation in 2040, challenges of growth." https://www.eurocontrol.int/sites/default/files/content/documents/official-documents/reports/challenges-of-growth-2018.pdf, 2018. Accessed: 2018-06-19.

[7] F. A. A. (FAA), "Nextgen implementation plan 2016," 2016.

[8] Eurocontrol, "European atm master plan (edition 2015)," 2015.

[9] Eurocontrol, "Airport CDM implementation manual," 2017.

[10] Y. Günther, A. Inard, B. Werther, M. Bonnier, G. Spies, A. Marsden, M. Temme, D. Böhme, R. Lane, and H. Niederstraßer, "Total Airport Management (operational concept and logical architecture)," 2006.

[11] I. V. A. Organization, "Air traffic control position." https://www.ivao.aero/training/.../books/Student_Air_Trafic_Control_Position.pdf.

[12] I. V. A. Organization, "The approach control position." https://www.ivao.aero/training/documentation/books/APC_APP_position.pdf.

[13] I. V. A. Organization, "The departure control position." https://www.ivao.aero/training/documentation/books/APC_DEP_position.pdf.

[14] J. Ma, D. Delahaye, M. Sbihi, and M. Mongeau, "Merging flows in terminal moneuvering area using time decomposition approach," in *ICRAT2016, 7th International Conference on Research in Air Transportation*, 2016.

[15] J. Ma, D. Delahaye, M. Sbihi, and M. Mongeau, "Integrated optimization of terminal manoeuvring area and airport," in *6th SESAR Innovation Days (2016).*, 2016.

[16] J. Ma, D. Delahaye, M. Sbihi, P. Scala, and M. A. M. Mota, "Integrated optimization of terminal maneuvering area and airport at the macroscopic level," *Transportation Research Part C: Emerging Technologies*, vol. 98, pp. 338–357, 2019.

[17] J. Ma, D. Delahaye, M. Sbihi, P. Scala, and M. M. Mota, "A study of tradeoffs in airport coordinated surface operations," in *EIWAC 2017, 5th ENRI international workshop on ATM/CNS*, 2017.

[18] J. Ma, M. Sbihi, and D. Delahaye, "Optimization of departure runway scheduling incorporating arrival crossings," *International Transactions in Operational Research*, 2019. doi: https://doi.org/10.1111/itor.12657.

[19] I. Anagnostakis, *A multi-objective, decomposition-based algorithm design methodology and its application to runaway operations planning*. PhD thesis, Massachusetts Institute of Technology, 2004.

[20] J. A. Bennell, M. Mesgarpour, and C. N. Potts, "Airport runway scheduling," *4OR*, vol. 9, no. 2, p. 115, 2011.

[21] A. Lieder and R. Stolletz, "Scheduling aircraft take-offs and landings on interdependent and heterogeneous runways," *Transportation research part E: logistics and transportation review*, vol. 88, pp. 167–188, 2016.

[22] R. G. Dear, "The dynamic scheduling of aircraft in the near terminal area," tech. rep., Flight Transportation Laboratory, Massachusetts Institute of Technology, 1976.

[23] H. N. Psaraftis, "A dynamic programming approach for sequencing groups of identical jobs," *Operations Research*, vol. 28, no. 6, pp. 1347–1359, 1980.

[24] H. Balakrishnan and B. G. Chandran, "Algorithms for scheduling runway operations under constrained position shifting," *Operations Research*, vol. 58, no. 6, pp. 1650–1665, 2010.

[25] A. Lieder, D. Briskorn, and R. Stolletz, "A dynamic programming approach for the aircraft landing problem with aircraft classes," *European Journal of Operational Research*, vol. 243, no. 1, pp. 61–69, 2015.

[26] J. E. Beasley, M. Krishnamoorthy, Y. M. Sharaiha, and D. Abramson, "Scheduling aircraft landings—the static case," *Transportation Science*, vol. 34, no. 2, pp. 180–197, 2000.

[27] H. Pinol and J. E. Beasley, "Scatter search and bionomic algorithms for the aircraft landing problem," *European Journal of Operational Research*, vol. 171, no. 2, pp. 439–462, 2006.

[28] A. T. Ernst, M. Krishnamoorthy, and R. H. Storer, "Heuristic and exact algorithms for scheduling aircraft landings," *Networks: An International Journal*, vol. 34, no. 3, pp. 229–241, 1999.

[29] A. Faye, "Solving the aircraft landing problem with time discretization approach," *European Journal of Operational Research*, vol. 242, no. 3, pp. 1028–1038, 2015.

[30] J. E. Beasley, M. Krishnamoorthy, Y. M. Sharaiha, and D. Abramson, "Displacement problem and dynamically scheduling aircraft landings," *Journal of the Operational Research Society*, vol. 55, no. 1, pp. 54–64, 2004.

[31] F. Furini, M. P. Kidd, C. A. Persiani, and P. Toth, "Improved rolling horizon approaches to the aircraft sequencing problem," *Journal of Scheduling*, vol. 18, no. 5, pp. 435–447, 2015.

[32] X.-B. Hu and W.-H. Chen, "Genetic algorithm based on receding horizon control for arrival sequencing and scheduling," *Engineering Applications of Artificial Intelligence*, vol. 18, no. 5, pp. 633–642, 2005.

[33] H. Erzberger, T. J. Davis, and S. Green, "Design of center-TRACON automation system," 1993.

[34] V. Kapp and M. Hripane, "Improving TMA sequencing process: Innovative integration of AMAN constraints in controllers environment," in *Digital Avionics Systems Conference, 2008. DASC 2008. IEEE/AIAA 27th*, IEEE, 2008.

[35] X.-B. Hu and W.-H. Chen, "Receding horizon control for aircraft arrival sequencing and scheduling," *IEEE Transactions on Intelligent Transportation Systems*, vol. 6, no. 2, pp. 189–197, 2005.

[36] X.-B. Hu and E. Di Paolo, "Binary-representation-based genetic algorithm for aircraft arrival sequencing and scheduling," *IEEE Transactions on Intelligent Transportation Systems*, vol. 9, no. 2, pp. 301–310, 2008.

[37] H. Balakrishnan and B. Chandran, "Efficient and equitable departure scheduling in real-time: new approaches to old problems," in *7th USA-Europe Air Traffic Management Research and Development Seminar*, pp. 02–05, 2007.

[38] J. Montoya, S. Rathinam, and Z. Wood, "Multiobjective departure runway scheduling using dynamic programming," *IEEE Transactions on Intelligent Transportation Systems*, vol. 15, no. 1, pp. 399–413, 2014.

[39] G. Gupta, W. Malik, and Y. Jung, "A mixed integer linear program for airport departure scheduling," in *9th AIAA Aviation Technology, Integration, and Operations Conference (ATIO) and Aircraft Noise and Emissions Reduction Symposium (ANERS)*, p. 6933, 2009.

[40] J. A. Atkin, E. K. Burke, J. S. Greenwood, and D. Reeson, "Hybrid metaheuristics to aid runway scheduling at london heathrow airport," *Transportation Science*, vol. 41, no. 1, pp. 90–106, 2007.

[41] I. Anagnostakis and J.-P. Clarke, "Runway operations planning: a two-stage solution methodology," in *Proceedings of the 36th Annual Hawaii International Conference on System Sciences*, IEEE, 2003.

[42] W. Malik and Y. C. Jung, "Exact and heuristic algorithms for runway scheduling," in *16th AIAA Aviation Technology, Integration, and Operations Conference*, p. 4072, 2016.

[43] G. Gupta, W. Malik, and Y. Jung, "Incorporating active runway crossings in airport departure scheduling," in *Proceedings of the AIAA Guidance, Navigation and Control Conference*, p. 7695, 2010.

[44] S. W. Bae, J. Park, and J.-P. Clarke, "Modified mixed integer linear program for airport departure scheduling," in *Proceedings of the AIAA Guidance, Navigation and Control Conference*, p. 4885, 2013.

[45] J. A. Atkin, E. K. Burke, J. S. Greenwood, and D. Reeson, "A metaheuristic approach to aircraft departure scheduling at london heathrow airport," in *Computer-aided Systems in Public Transport*, pp. 235–252, Springer, 2008.

[46] J. A. Atkin, E. K. Burke, J. S. Greenwood, and D. Reeson, "On-line decision support for take-off runway scheduling with uncertain taxi times at london heathrow airport," *Journal of Scheduling*, vol. 11, no. 5, p. 323, 2008.

[47] J. A. Atkin, E. K. Burke, J. S. Greenwood, and D. Reeson, "An examination of take-off scheduling constraints at london heathrow airport," *Public Transport*, vol. 1, no. 3, p. 169, 2009.

[48] J. A. Atkin, E. K. Burke, and S. Ravizza, "The airport ground movement problem: Past and current research and future directions," in *Proceedings of the 4th International Conference on Research in Air Transportation (ICRAT), Budapest, Hungary*, pp. 131–138, 2010.

[49] J. Guépet, *Optimization of airport operations: stand allocation, ground routing and runway sequencing*. PhD thesis, Université Grenoble Alpes, 2015.

[50] E. Q. Martins and M. M. Pascoal, "A new implementation of yen's ranking loopless paths algorithm," *4OR: A Quarterly Journal of Operations Research*, vol. 1, no. 2, pp. 121–133, 2003.

[51] J. Smeltink and M. Soomer, "An optimization model for airport taxi scheduling," in *Proceedings of the INFORMS Annual Meeting, Denver, USA*, 2004.

[52] J. Guépet, O. Briant, J.-P. Gayon, and R. Acuna-Agost, "The aircraft ground routing problem: Analysis of industry punctuality indicators in a sustainable perspective," *European Journal of Operational Research*, vol. 248, no. 3, pp. 827–839, 2016.

[53] P. C. Roling and H. G. Visser, "Optimal airport surface traffic planning using mixed-integer linear programming," *International Journal of Aerospace Engineering*, vol. 2008, no. 1, p. 1, 2008.

[54] G. Clare and A. G. Richards, "Optimization of taxiway routing and runway scheduling," *IEEE Transactions on Intelligent Transportation Systems*, vol. 12, no. 4, pp. 1000–1013, 2011.

[55] H. Balakrishnan and Y. Jung, "A framework for coordinated surface operations planning at dallas-fort worth international airport," in *Proceedings of the AIAA Guidance, Navigation and Control Conference*, p. 6553, 2007.

[56] B. Pesic, N. Durand, and J.-M. Alliot, "Aircraft ground traffic optimisation using a genetic algorithm," in *Proceedings of the 3rd Annual Conference on Genetic and Evolutionary Computation*, pp. 1397–1404, Morgan Kaufmann Publishers Inc., 2001.

[57] J.-B. Gotteland and N. Durand, "Genetic algorithms applied to airport ground traffic optimization," in *Proceedings of the Congress on Evolutionary Computation, Canberra, Australia*, 2003.

[58] R. Deau, J.-B. Gotteland, and N. Durand, "Airport surface management and runways scheduling," in *8th USA/Europe Air Traffic Management Research and Development Seminar*, 2009.

[59] S. Ravizza, J. A. Atkin, and E. K. Burke, "A more realistic approach for airport ground movement optimisation with stand holding," *Journal of Scheduling*, vol. 17, no. 5, pp. 507–520, 2014.

[60] S. Rathinam, J. Montoya, and Y. Jung, "An optimization model for reducing aircraft taxi times at the dallas fort worth international airport," in *26th International Congress of the Aeronautical Sciences (ICAS)*, pp. 14–19, 2008.

[61] H. Lee and H. Balakrishnan, "A comparison of two optimization approaches for airport taxiway and runway scheduling," in *Digital Avionics Systems Conference (DASC), 2012 IEEE/AIAA 31st*, 2012.

[62] W. Malik, G. Gupta, and Y. Jung, "Managing departure aircraft release for efficient airport surface operations," in *Proceedings of the AIAA Guidance, Navigation and Control Conference*, p. 7696, 2010.

[63] I. Simaiakis and H. Balakrishnan, "Queuing models of airport departure processes for emissions reduction," in *Proceedings of the AIAA Guidance, Navigation and Control Conference*, p. 5650, 2009.

[64] I. Simaiakis, H. Khadilkar, H. Balakrishnan, T. G. Reynolds, and R. J. Hansman, "Demonstration of reduced airport congestion through pushback rate control," *Transportation Research Part A: Policy and Practice*, vol. 66, pp. 251–267, 2014.

[65] S. Badrinath and H. Balakrishnan, "Control of a non-stationary tandem queue model of the airport surface," in *American Control Conference (ACC), 2017*, pp. 655–661, IEEE, 2017.

[66] H. Lee, I. Simaiakis, and H. Balakrishnan, "A comparison of aircraft trajectory-based and aggregate queue-based control of airport taxi processes," in *Digital Avionics Systems Conference (DASC), 2010 IEEE/AIAA 29th*, IEEE, 2010.

[67] J.-B. Gotteland, N. Durand, J.-M. Alliot, and E. Page, "Aircraft ground traffic optimization," in *ATM 2001, 4th USA/Europe Air Traffic Management Research and Development Seminar*, 2001.

[68] J.-B. Gotteland, *Optimisation du trafic au sol sur les grands aéroports*. PhD thesis, Institut National Polytechnique de Toulouse, 2004.

[69] R. Deau, *Optimisation des séquences de pistes et des mouvements au sol sur les grands aéroports*. PhD thesis, Institut National Polytechnique de Toulouse, 2010.

[70] S. Ravizza, J. Chen, J. A. Atkin, E. K. Burke, and P. Stewart, "The trade-off between taxi time and fuel consumption in airport ground movement," *Public Transport*, vol. 5, no. 1-2, pp. 25–40, 2013.

[71] J. Chen, M. Weiszer, P. Stewart, and M. Shabani, "Toward a more realistic, cost-effective, and greener ground movement through active routing—part i: Optimal speed profile generation," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 5, pp. 1196–1209, 2016.

[72] J. Chen, M. Weiszer, G. Locatelli, S. Ravizza, J. A. Atkin, P. Stewart, and E. K. Burke, "Toward a more realistic, cost-effective, and greener ground movement through active routing: A multiobjective shortest path approach," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 12, pp. 3524–3540, 2016.

[73] U. Benlic, A. E. Brownlee, and E. K. Burke, "Heuristic search for the coupled runway sequencing and taxiway routing problem," *Transportation Research Part C: Emerging Technologies*, vol. 71, pp. 333–355, 2016.

[74] R. Mori, "Development of a pushback time assignment algorithm considering uncertainty," *Journal of Air Transportation*, pp. 1–10, 2017.

[75] J. Zhou, *Optimal Design of SIDs/STARs in Terminal Maneuvering Area*. PhD thesis, Universite Toulouse 3 Paul Sabatier (UT3 Paul Sabatier), 2017.

[76] J. Zhou, S. Cafieri, D. Delahaye, and M. Sbihi, "Optimizing the design of a route in terminal maneuvering area using branch and bound," in *Air Traffic Management and Systems II*, pp. 171–184, Springer, 2017.

[77] J. Zhou, S. Cafieri, D. Delahaye, and M. Sbihi, "Optimization-based design of departure and arrival routes in terminal maneuvering area," *Journal of Guidance, Control, and Dynamics*, vol. 40, no. 11, pp. 2889–2904, 2017.

[78] C. Zuniga, D. Delahaye, and M. A. Piera, "Integrating and sequencing flows in terminal maneuvering area by evolutionary algorithms," in *Digital Avionics Systems Conference (DASC), 2011 IEEE/AIAA 30th*, IEEE, 2011.

[79] H. Chida, C. Zuniga, and D. Delahaye, "Topology design for integrating and sequencing flows in terminal maneuvering area," *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering*, vol. 230, no. 9, pp. 1705–1720, 2016.

[80] M. Xue and S. Zelinski, "Optimal integration of departures and arrivals in terminal airspace," *Journal of Guidance, Control, and Dynamics*, vol. 37, no. 1, pp. 207–213, 2013.

[81] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan, "A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: Nsga-ii," in *International conference on parallel problem solving from nature*, pp. 849–858, Springer, 2000.

[82] C. Bosson, M. Xue, and S. Zelinski, "Optimizing integrated terminal airspace operations under uncertainty," in *Digital Avionics Systems Conference (DASC), 2014 IEEE/AIAA 33rd*, IEEE, 2014.

[83] L. Boursier, B. Favennec, E. Hoffman, A. Trzmiel, F. Vergne, and K. Zeghal, "Merging arrival flows without heading instructions," in *7th USA/Europe Air Traffic Management Research and Development Seminar*, 2007.

[84] M. Liang, *Aircraft Route Network Optimization in Terminal Maneuvering Area*. PhD thesis, Université Paul Sabatier (Toulouse 3), 2018.

[85] D. Kjenstad, C. Mannino, T. E. Nordlander, P. Schittekat, and M. Smedsrud, "Optimizing AMAN-SMAN-DMAN at hamburg and arlanda airport," *3rd SESAR Innovation Days*, 2013.

[86] C. Bosson, M. Xue, and S. Zelinski, "Optimizing integrated arrival, departure and surface operations under uncertainty," in *11th USA/Europe ATM R&D Seminar (ATM2015), Lisbon, Portugal*, 2015.

[87] M. J. Frankovich, *Air traffic flow management at airports: A unified optimization approach*. PhD thesis, Massachusetts Institute of Technology, 2012.

[88] H. H. D. Khadilkar, *Networked control of aircraft operations at airports and in terminal areas*. PhD thesis, Massachusetts Institute of Technology, 2013.

[89] D. Delahaye and S. Puechmorel, *Modeling and optimization of air traffic*. John Wiley & Sons, 2013.

[90] J. Schneider and S. Kirkpatrick, *Stochastic optimization*. Springer Science & Business Media, 2007.

[91] M. R. Garey and D. S. Johnson, *Computers and intractability*, vol. 29. wh freeman New York, 2002.

[92] L. Bianchi, M. Dorigo, L. M. Gambardella, and W. J. Gutjahr, "A survey on metaheuristics for stochastic combinatorial optimization," *Natural Computing*, vol. 8, no. 2, pp. 239–287, 2009.

[93] C. Blum and A. Roli, "Metaheuristics in combinatorial optimization: Overview and conceptual comparison," *ACM computing surveys (CSUR)*, vol. 35, no. 3, pp. 268–308, 2003.

[94] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, no. 4598, pp. 671–680, 1983.

[95] F. Glover and M. Laguna, "Tabu search," in *Handbook of combinatorial optimization*, pp. 2093–2229, Springer, 1998.

[96] H. R. Lourenço, O. C. Martin, and T. Stützle, "Iterated local search," in *Handbook of metaheuristics*, pp. 320–353, Springer, 2003.

[97] N. Durand, D. Gianazza, J.-B. Gotteland, and J.-M. Alliot, *Metaheuristics for Air Traffic Management*. John Wiley & Sons, 2015.

[98] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller, "Equation of state calculations by fast computing machines," *The Journal of Chemical Physics*, vol. 21, no. 6, pp. 1087–1092, 1953.

[99] P. J. Van Laarhoven and E. H. Aarts, "Simulated annealing," in *Simulated annealing: Theory and applications*, pp. 7–15, Springer, 1987.

[100] J. Dréo, A. Pétrowski, P. Siarry, and E. Taillard, *Metaheuristics for hard optimization: methods and case studies*. Springer Science & Business Media, 2006.

[101] S. Chaimatanan, D. Delahaye, and M. Mongeau, "A hybrid metaheuristic optimization algorithm for strategic planning of 4d aircraft trajectories at the continental scale," *IEEE Computational Intelligence Magazine*, vol. 9, no. 4, pp. 46–61, 2014.

[102] A. C. International, "2016 aircraft movements." https://aci.aero/data-centre/annual-traffic-data/aircraft-movements/2016-final-summary/.

[103] Federal Aviation Administration, "Aviation System Performance Metrics (ASPM)." http://aspm.faa.gov/, 2018. Retrieved Oct 18, 2018.

[104] S. Badrinath, M. Z. Li, and H. Balakrishnan, "Integrated surface–airspace model of airport departures," *Journal of Guidance, Control, and Dynamics*, pp. 1–15, 2018.

[105] S. Badrinath, H. Balakrishnan, J. Ma, and D. Delahaye, "A comparative analysis of departure metering at paris (cdg) and charlotte (clt) airports," in *13th USA/Europe ATM R&D Seminar (ATM2019), Lisbon, Portugal*, 2019.

[106] I. A. Chaudhry and A. A. Khan, "A research survey: review of flexible job shop scheduling techniques," *International Transactions in Operational Research*, vol. 23, no. 3, pp. 551–591, 2016.

[107] J. Du and J. Y.-T. Leung, "Minimizing total tardiness on one machine is NP-hard," *Mathematics of Operations Research*, vol. 15, no. 3, pp. 483–495, 1990.

[108] L. Gurobi Optimization, "Gurobi optimizer reference manual," 2018.