Optimization of airport operations : stand allocation, ground routing and runway sequencing

Julien Guepet

UNIVERSITÉ DE GRENOBLE

THÈSE

Pour obtenir le grade de

DOCTEUR DE L'UNIVERSITÉ DE GRENOBLE

Spécialité : **Mathématiques et Informatique**

Arrêté ministérial : 7 août 2006

Présentée par

Julien GUEPET

Thèse dirigée par **Jean-Philippe GAYON**

préparée au sein du **Laboratoire G-SCOP**
et de l'**École Doctorale MSTII**

Optimisation de la gestion des avions dans un aéroport

Affectation aux points de stationnement, routage au sol et ordonnancement à la piste

Thèse soutenue publiquement le **3 décembre 2015**,
devant le jury composé de :

M. Christophe RAPINE
Professeur, Université de Lorraine, Président

M. Christian ARTIGUES
Directeur de recherche, CNRS, LAAS, Rapporteur

M. Dominique FEILLET
Professeur, Ecole des Mines de Saint-Etienne, Rapporteur

M. Rodrigo ACUÑA-AGOST
Head of analysis and research, Amadeus, INR, Co-encadrant de thèse

M. Olivier BRIANT
Maître de conférence, Grenoble INP, Co-encadrant de thèse

M. Jean-Philippe GAYON
Maître de conférence, HDR, Grenoble INP, Directeur de thèse

Table des matières

Remerciements

Je tiens tout d'abord à remercier ceux qui me sont le plus cher, qui m'ont toujours accompagné et qui je l'espère m'accompagneront toujours, mes proches. Mon amour, Alexandra. Ma famille, Christine et Jean-François, Guillaume Emeline et Malo, Emilie Vincent et Baptiste, Yann, Mireille et Jacques, Reine, Christophe. Ma belle-famille, Frédérique et Jean-Yves, Elsa, Corentin, Fabienne et Régis, Chantal, Paul.

Je tiens ensuite à remercier mes encadrants, Olivier, Jean-Philippe et Rodrigo, pour m'avoir accompagné et guidé dans mes travaux, pour leur temps et leur patience, leur intelligence et leur confiance, leur soutien et leur sympathie. Travailler quotidiennement avec des gens aussi brillants et bienveillants fut réel un plaisir et fut très enrichissant, aussi bien humainement que scientifiquement. J'espère que mes travaux sont à la hauteur de la qualité de l'encadrement que j'ai eu la chance d'avoir durant ces trois ans. Je ne serais jamais arrivé au bout de ce chemin sans eux et je souhaite à tout thésard d'être aussi bien encadré que je l'ai été.

Je tiens également à remercier Amadeus, et particulièrement Semi et Rodrigo, pour avoir cru en moi dès le début et m'avoir offert l'opportunité de faire ces travaux, pour m'avoir fait confiance et laissé une totale liberté. Je suis conscient qu'une telle liberté est une chance inouïe, j'ai ainsi eu l'opportunité d'explorer des horizons divers, parfois très éloignés des applications que recherchent souvent les industriels. Je considère que la mixité entre théorie et application est la plus grande richesse de mes travaux. Elle est le fruit de la confiance et de la liberté qui m'a été accordées, merci infiniment.

Je tiens aussi à remercier Géraldine pour son accueil à la tour de contrôle de l'aéroport de Lyon, et Tor et Simon pour leur accueil à l'aéroport de Copenhague, pour leur conseils avisés ainsi que leurs explications.

Je tiens à remercier Nadia, Laurent et Hadrien pour m'avoir donné l'opportunité d'enseigner. Enseigner l'informatique et la recherche opérationnelle sous votre tutelle fut une expérience très enrichissante et très agréable, j'espère qu'une telle opportunité se représentera un jour.

Je tiens enfin à remercier tous mes amis et collègues, pour leur soutien et pour tout les bons moments passés ensemble, indispensables à la réussite de l'épreuve qu'est la thèse. Mes amis, Alexandre, Bastien, Clément et Marie, Emilie et Martin. Mes collègues et maintenant amis de G-SCOP, que je ne peux pas tous citer, mais je tiens à remercier particulièrement Boris, Lucile et Geoffrey, Chloé et Gabriel, Lucie et Anne-Laure ma seconde maman. Mes collègues d'Amadeus, mon département (INR) et particulièrement mon équipe. Là encore, une liste exhaustive serait trop longue, mais je tiens à remercier Baptiste, Mourad, Dani, Thierry, Pranav, Alexandre, Valentin, Alejandro et Ezequiel.

Pour finir, je tiens à remercier tous ceux qui m'ont accueilli et hébergé pendant ces trois ans d'itinérance entre Sophia Antipolis et Grenoble.

Chapitre 1

Introduction

En Europe, les aéroports sont de plus en plus congestionnés. Ils sont aujourd'hui responsables de 9,6% du retard des vols et optimiser leur capacité n'a jamais été aussi crucial.

Dans ce chapitre d'introduction, nous présentons le trafic aérien européen et la place de premier ordre qu'occupe aujourd'hui les aéroports. Nous décrivons ensuite les différentes infrastructures qu'un avion utilise entre l'atterrissage et le décollage. Nous verrons que la gestion des avions soulève des problèmes d'optimisation complexes. Cette thèse aborde les problèmes d'affectation aux points de stationnement, de routage au sol et d'ordonnancement à la piste. Nous proposons enfin un résumé des contributions et présentons la structure de ce manuscrit.

Cette thèse est une collaboration avec la société Amadeus, dont le cœur de métier est le développement de solutions pour l'industrie du transport et du tourisme. Amadeus propose aujourd'hui un portefeuille de solutions pour les aéroports. Un effort particulier a été porté sur le réalisme de nos modélisations, ainsi que sur le développement de méthodes suffisamment efficaces pour pouvoir être utilisées en pratique. L'ensemble de nos approches a pu être testé et validé avec des données réelles d'aéroports européens.

1.1 Les aéroports : goulet d'étranglement du trafic aérien européen

Le transport aérien est un pilier de l'économie mondiale moderne. Ce secteur d'activité regroupe 5,1 millions d'emplois dans l'Union Européenne (UE) et représente 2,4% du produit intérieur brut, avec un revenu total de 365 milliards d'euros[1]. En 2013, 9,4 millions de vols ont transporté 842 millions de passagers et 13,4 millions de tonnes de fret et de courrier[2]. Le trafic aérien s'est considérablement accru ces dernières années et il est prévu que cette tendance continue : une hausse annuelle de 2,5% du nombre de vols est prévue jusqu'en 2021 et une augmentation totale de 50% est estimée d'ici à 2035, soit 14,4 millions de vols (Eurocontrol [2013b, 2015]).

Un tel trafic devient de plus en plus difficile à gérer : un retard moyen de 9,7 minutes par vol a été enregistré en 2014, soit un surcoût annuel à l'industrie estimé entre 1 et 2 milliards d'euros (cf. Cook and Tanner [2011]). Seules des infrastructures adaptées ainsi qu'une gestion efficace du trafic peuvent permettre un développement sûr de l'industrie aérienne.

Les aéroports occupent une place de premier ordre dans le trafic aérien, ils servent de passerelle entre la terre et l'espace aérien. Il y a près de 2000 aéroports en Europe, mais seulement 528 couvrent 98% du trafic et les 35 plus importants en regroupent environ la moitié[3]. Les aéroports deviennent un goulet d'étranglement du trafic de plus en plus important. Ils sont aujourd'hui responsables de 9,6% du retard total enregistré, arrivant en troisième position après les compagnies aériennes (31%) et le retard réactionnaire (44%, c'est à dire la propagation du retard). Construire de nouvelles infrastructures aéroportuaires étant très couteux, parfois impossible et n'étant pas une solution à court voire moyen terme, une meilleure gestion de la capacité existante devient cruciale.

1. Sources : Commission Européenne (http://ec.europa.eu/transport/modes/air/index_en.htm)
2. Sources : Commission Européenne (http://ec.europa.eu/eurostat/statistics-explained/index.php/Air_transport_statistics)
3. Sources : Eurocontrol (https://www.eurocontrol.int/download/publication/node-field_download-4691-0)

1.2 Le projet *Airport Collaborative Decision Making*

Tandis que l'espace aérien se modernise, notamment grâce au projet SESAR[4], les aéroports ont été identifiés comme un chaînon manquant de la circulation d'informations entre les différents acteurs du réseau, notamment à cause du manque de prédictibilité de l'heure de départ. Le projet *Airport Collaborative Decision Making* (A-CDM, Eurocontrol [2009]) a été créé par le gestionnaire du réseau aérien européen Euroconrol à cet effet. Il a pour but d'améliorer la qualité d'échange d'informations sur les arrivées et les départs avec l'ensemble des acteurs du trafic aérien européen par le biais du centre des opérations du gestionnaire de réseau (*Network Manager Operations Center, NMOC*, anciennement appelé *Central Flow Management Unit, CFMU*). Le cœur de l'A-CDM est un partage transparent d'informations entre chaque acteur d'un aéroport, dans le but d'avoir une vision précise et commune de l'avancement de chaque processus. Il en résulte une estimation de l'heure de fin de préparation d'un vol à la fois plus précise et plus en amont dans le temps, offrant ainsi la possibilité d'anticiper davantage et de prendre des décisions globalement plus appropriées. Ce projet s'appuie sur un suivi minutieux de chaque étape de la rotation d'un avion (cf. Section 1.3). Eurocontrol [2012a] propose une méthodologie et un plan d'implémentation détaillé pour atteindre ces objectifs. Au début de l'année 2015, 15 aéroports ont entièrement fini leur transition vers l'A-CDM.

Il va de soi que le partage de ces informations ne peut se faire que par un système informatique centralisé ou par des systèmes inter-connectés. De tels systèmes rendent possible l'utilisation d'outils d'aide à la décision afin d'optimiser la capacité des aéroports. Dans cette optique, le but de cette thèse est de proposer des modèles mathématiques aussi réalistes que possible, ainsi que des méthodes de résolution efficaces, permettant d'optimiser la gestion de différentes infrastructures utilisées par un avion lors de sa rotation au sol, que nous présentons maintenant.

1.3 La rotation d'un avion au sol

Dans cette partie, nous présentons la rotation d'un avion dans un aéroport, c'est à dire son séjour au sol entre l'atterrissage et le décollage, ainsi que les principales infrastructures qu'il utilise. Nous mentionnons également les différences majeures avec les aéroports américains.

4. Le projet *Single European Sky ATM Research* (SESAR) est un programme visant à moderniser les systèmes de gestion du trafic aérien européen, dans le but de tripler la capacité de l'espace aérien, de réduire de 10% l'impact environnemental, de diviser par 10 le risque d'accident et de diminuer les coûts du contrôle. Le projet a été lancé en 2004 et devrait s'achever à l'horizon 2020. http://www.sesarju.eu/

L'anglais étant la langue officielle du transport aérien, l'appellation anglaise de nombreux termes techniques et propres au domaine aérien est précisée dans cette partie afin de pouvoir situer cette thèse dans un contexte international. Il existe par ailleurs certains mots qui n'ont pas d'équivalent en français.

1.3.1 De l'atterrissage au décollage

Dans les opérations aéroportuaires, le terme de rotation (*turnaround*) se réfère à la période entre l'atterrissage et le décollage d'un avion. Une rotation peut être grossièrement découpée en 5 phases, comme illustré par la Figure 1.1.

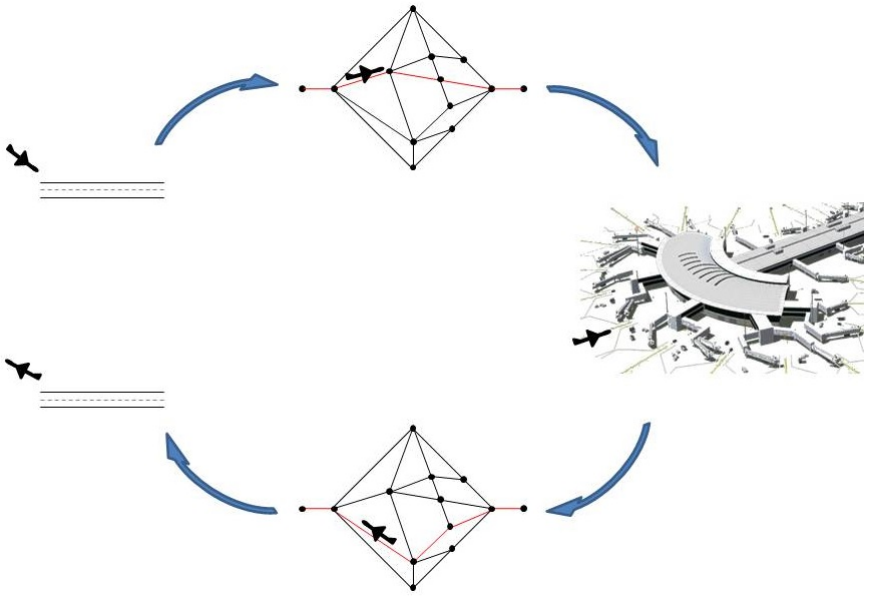


FIGURE 1.1 – La rotation d'un avion au sol

1. Atterrissage (*landing*) : l'avion entre en phase d'approche et se pose sur une piste d'atterrissage (*runaway*).
2. Roulage (*taxi-in*) : après avoir atterri, l'avion roule jusqu'à son point de stationnement (*stand*) à travers un réseau de routes appelées *taxiways*.
3. Opérations au point de stationnement (*ground handling*) : une fois que l'avion est garé, la phase d'opérations au point de stationnement commence et l'avion est préparé pour son prochain vol. Cette phase inclut le débarquement et l'embarquement des passagers, des bagages et de la nourriture, le nettoyage de l'intérieur de l'appareil, le ravitaillement en kérosène, l'approvisionnement en eau saine et autres opérations de maintenance.

4. Roulage (*taxi-out*) : une fois que l'avion est prêt et autorisé à quitter son point de stationnement, il est repoussé (*push back*) et roule vers une piste pour le décollage.
5. Décollage : l'avion décolle et quitte l'espace aérien de l'aéroport.

1.3.2 Les infrastructures

Lors de sa rotation, un avion utilise de nombreuses infrastructures qui sont gérées par divers acteurs.

Pistes de décollage et atterrissage (*runways*)

Les pistes de décollage et atterrissage font le lien entre l'espace aérien et le sol. Leur longueur est souvent comprise entre 4 et 5 kilomètres. Il arrive qu'elles soient plus courtes auquel cas certains appareils ne peuvent pas les utiliser. Elles sont délimitées par deux lignes latérales continues et une ligne centrale discontinue sur laquelle le train de roues avant roule lors du décollage et de l'atterrissage. Une même piste peut être utilisée dans deux directions (depuis chaque extrémité) mais pour des raisons de stabilité aérodynamique, les avions décollent ou atterrissent toujours face au vent. Ainsi, une seule direction, appelée configuration, est utilisée à la fois. Une configuration est nommée par les deux premiers chiffres du degré d'angle qu'elle fait avec le nord dans le sens anti-horaire (par exemple 27 correspond à un angle de 270°). Il arrive fréquemment que deux ou trois pistes soient parallèles dans un même aéroport, elles sont alors distinguées par les lettres L(eft), C(enter) ou R(ight) (cf. Figure 1.2).

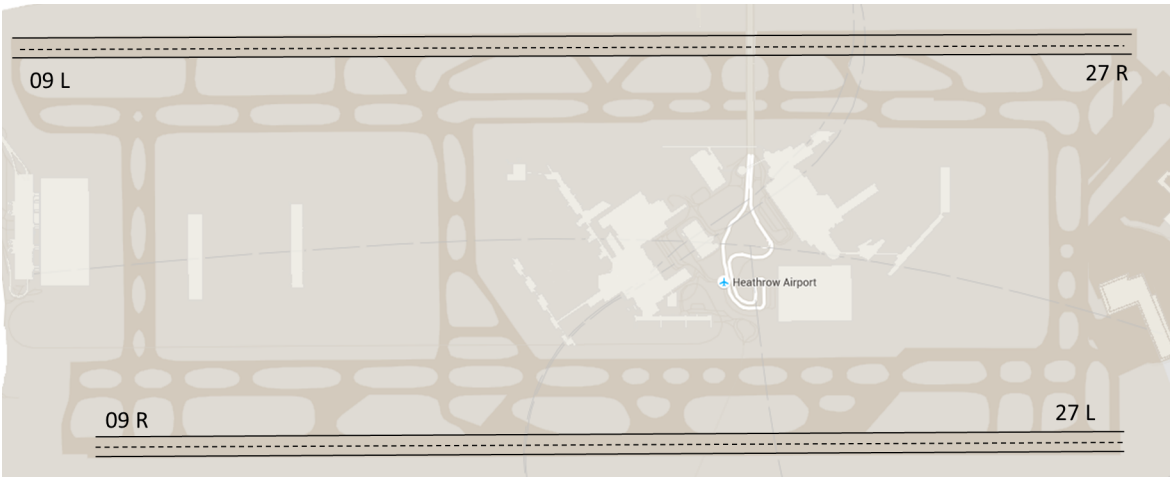


FIGURE 1.2 – Les pistes de l'aéroport de Londres Heathrow (LHR)

Il existe deux types de gestion d'une piste : le mode mixte, où des atterrissages et des décollages peuvent être opérés sur la piste, et le mode séparé, où soit des atterrissages soit

des décollages ont lieu. Deux pistes trop proches l'une de l'autre interfèrent et ne peuvent être opérées en mode mixte indépendamment [5]. Dans ce cas une piste est souvent dédiée aux arrivées et l'autre aux départs, ce qui les rend quasiment indépendantes.

Les pistes de décollage, d'atterrissage et l'espace aérien avoisinant l'aéroport sont sous la responsabilité d'un contrôleur aérien (*Air Traffic Controller, ATC*) situé dans la tour de contrôle du trafic (*Air Traffic Control Tower, ATCT*). Il est appelé contrôleur de piste (*runway controller*) et il arrive qu'il y ait un contrôleur de piste pour les départs et un autre pour les arrivées. Une des préoccupations premières des contrôleurs de piste est d'assurer la sécurité des avions au travers des règles de séparation. Un avion génère des tourbillons de sillage (*wake vortex*) lorsqu'il vole de par la rencontre de masses d'air de pression différentes en bout d'ailes (cf. Figure 1.3).

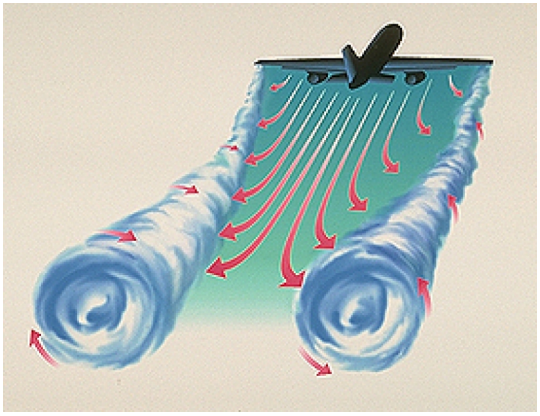


FIGURE 1.3 – Tourbillons de sillage

Une séparation minimale est donc nécessaire entre deux avions pour que ces turbulences se dissipent et que le second avion soit en sécurité. Plus un avion est lourd, plus il génère de tourbillons de sillage. Plus un avion est léger, plus il est sensible aux turbulences. L'Organisation Internationale de l'Avion Civile (OIAC, *ICAO* en anglais) a catégorisé les différents appareils et la séparation minimale entre deux avions dépend de leur catégorie respective. La conception même de l'aile d'un avion a des conséquences sur l'importance des tourbillons de sillage et certaines technologies, telles que les *winglet* (voir Figure 1.4), permettent de les diminuer. En outre, des séparations additionnelles peuvent être nécessaires si deux avions empruntent la même route dans l'espace aérien de l'aéroport.

5. Moins de 2500 pieds ($\approx$ 760m) d'après Balakrishnan and Chandran [2006]

FIGURE 1.4 – *Winglet* de l'Airbus A350

Réseau de routes au sol (*taxiway network*)

Le réseau de routes au sol d'un aéroport relie les pistes aux points de stationnement. Le terme de route se réfère généralement à l'espace aérien et l'anglicisme *taxiway* est utilisé. Une *taxiway* est délimitée par deux lignes latérales et une ligne centrale indiquant la trajectoire que le train de roues avant d'un appareil doit suivre (voir Figure 1.5), ce qui est particulièrement utile pour les virages. Chaque segment de *taxiway* ne se voit pas attribuer un nom précis, le réseau est découpé en secteurs de taille variable dénommés par une lettre et parfois par un chiffre supplémentaire.

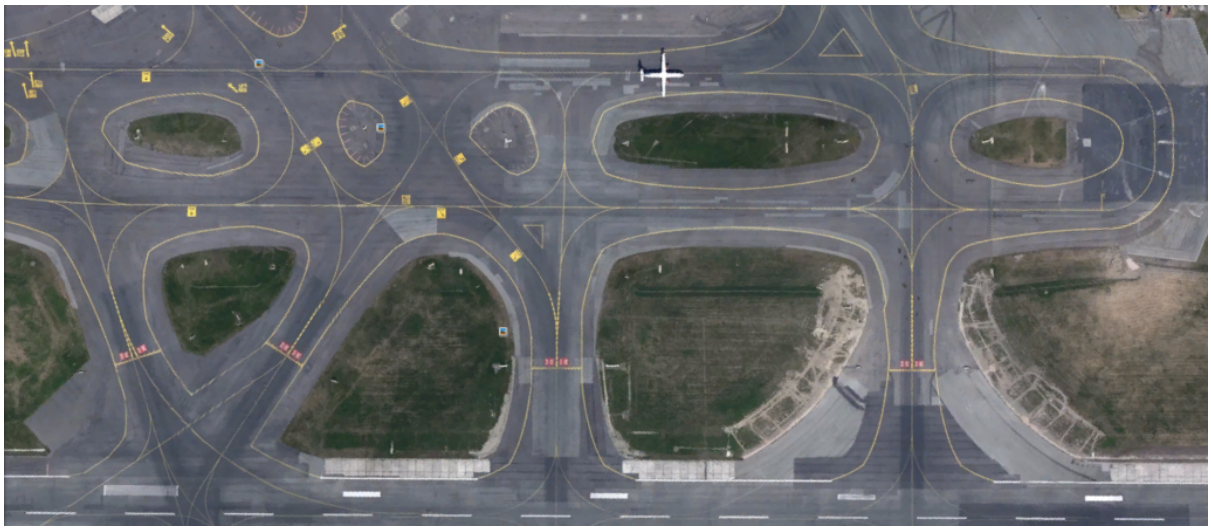


FIGURE 1.5 – Une partie du réseau de *taxiway* de l'aéroport de Copenhague (CPH)

Le réseau de *taxiways* est sous la responsabilité d'un contrôleur aérien lui aussi situé dans la tour de contrôle du trafic (ATCT) et appelé le contrôleur de sol (*ground controller*). Il supervise le roulage des avions afin d'éviter les collisions. Il doit gérer quatre types de mouvement : les arrivées, les départs prêts pour le repoussage, les départs ayant été repoussés et les remorquages des avions.

– Une arrivée doit être routée dès l'atterrissage afin de libérer la piste. L'avion roule ensuite en direction de son point de stationnement.
– Un départ ne peut commencer à rouler sans l'accord du contrôleur de sol, il lui faut l'autorisation d'allumer les moteurs (*start-up approval*) et de quitter le point de stationnement (*push back clearance*). N'étant pas équipé de marche arrière, il est alors repoussé par un tracteur (voir Figure 1.6). En parallèle, le pilote effectue un ensemble de contrôles sur l'appareil (*check list*). L'ensemble de la procédure dure plusieurs minutes. Elle dépend du type d'appareil et de l'agencement du point de stationnement, qui rend parfois la manœuvre très délicate.
– Une fois la procédure de repoussage terminée, l'avion peut commencer à rouler.
– Les avions remorqués n'ont généralement ni pilote ni passager à bord. Le remorquage permet de transférer un appareil vers un autre point de stationnement ou vers un hangar.

FIGURE 1.6 – Tracteurs de repoussage

Quand deux avions veulent utiliser une même ressource (segment de route ou intersection) au même moment, le contrôleur de sol doit gérer un conflit, c'est à dire décider quel avion va utiliser la ressource en premier. L'autre avion est alors soit mis en attente soit routé par un autre chemin.

Points de stationnement

Les points de stationnement sont des zones sur lesquelles les avions sont stationnés lors des opérations au sol. Le premier caractère de leur nom correspond en général au terminal auquel ils sont rattachés (un chiffre ou une lettre), les autres caractères sont arbitraires. Ils sont délimités par une ligne centrale indiquant la position que doit prendre l'avion. Il peut également y avoir deux lignes latérales délimitant leur largeur. Il arrive fréquemment que certains points de stationnement se chevauchent pour offrir la possibilité de garer soit deux petits avions soit un large (voir Figure 1.7(b)).

(a) Un avion stationné en contact (b) Un avion stationné en aire éloignée

FIGURE 1.7 – Points de stationnement en contact et en aire éloignée

En Europe, on distingue deux types de points de stationnement, les stationnements dits en contact (*contact stand*) et les stationnements dits en aire éloignée (*remote stand*). Les stationnements en contact sont physiquement reliés à une ou plusieurs portes d'embarquement (*gate*) par des passerelles ou des rampes. Les passagers rejoignent donc l'avion directement en marchant (voir Figure 1.7(a)). A l'inverse, les stationnements en aire éloignée ne sont pas reliés aux terminaux et un transfert en bus est nécessaire (voir Figure 1.7(b)).

Les points de stationnement sont gérés par les autorités aéroportuaires mais il arrive que la gestion de certains points de stationnement, voire de terminaux entiers, soit déléguée à une compagnie aérienne.

1.3.3 Différences majeures avec les aéroports américains

La forte croissance du trafic aérien et la modernisation des systèmes de gestion des infrastructures ne sont bien sûr pas limitées à l'Europe et ce contexte est similaire aux USA. Cependant, l'organisation des aéroports américains est généralement différente de celle de l'Europe. Le débarquement de passagers en aire éloignée est très peu pratiqué et la plupart des aéroports n'ont que des points de stationnement en contact. De plus, les autorités aéroportuaires ont moins d'emprise sur la gestion du trafic quotidien. Les terminaux ou les points de stationnement sont en grande partie loués aux compagnies aériennes pour plusieurs années, il arrive même parfois qu'elles en soient propriétaires. Elles gèrent directement leurs terminaux ainsi que les *taxiways* avoisinantes (*ramp area*). Elles sont donc responsables du repoussage et du roulage des avions jusqu'à certaines zones appelées *spots*, où l'avion passe sous la direction du contrôleur de sol et

vice versa pour les arrivées. Une plus ample description de ce mode de gestion et des *spots* a été faite par Malik et al. [2010].

1.4 De multiples problèmes d'optimisation

De par leur nature, les pistes, les *taxiways* et les points de stationnement sont souvent considérés comme les ressources les plus critiques des aéroports. Ce sont des infrastructures fixes qui ne peuvent être agrandies à court terme et sans investissement lourd. Ceci explique leur rareté et pourquoi elles sont fortement sollicitées. La bonne gestion de ces ressources est cruciale pour la fluidité du trafic dans l'aéroport et la ponctualité des vols. Cette gestion soulève des problèmes d'optimisation complexes et inter-connectés. Cette thèse s'intéresse particulièrement au problème d'affection aux points de stationnement, au problème de routage au sol et au problème d'ordonnancement à la piste, qui sont décrits ci-dessous.

La gestion des opérations aux points de stationnement soulève également des problèmes d'optimisation. Ils sortent du cadre de cette thèse, mais un aperçu de ce type de problèmes est proposé à la fin de cette partie.

1.4.1 Le problème d'affectation au point de stationnement

Le problème d'affectation aux points de stationnement (*Stand Allocation Problem, SAP*) consiste à affecter des opérations d'avion (arrivée, attente intermédiaire et départ) aux points de stationnement, de manière à optimiser certains indicateurs tout en respectant des contraintes opérationnelles et commerciales. Le plan d'affectation est souvent décidé la veille des opérations, voire plus en amont.

Le séjour d'un avion au point de stationnement peut se diviser en trois opérations macroscopiques : l'arrivée, l'attente intermédiaire et le départ. L'opération d'arrivée regroupe toutes les opérations aux points de stationnement liées à l'arrivée d'un vol, notamment le débarquement des passagers et des bagages. De même, l'opération de départ regroupe toutes les opérations liées au départ d'un vol. Durant l'opération d'attente intermédiaire, l'avion peut être remorqué vers un autre point de stationnement, ce qui permet de libérer certains points de stationnement intéressants comme ceux en contact. L'avion peut également être remorqué vers un hangar pour cause de maintenance. Suivant la durée de la rotation, l'avion peut être remorqué plusieurs fois, par exemple pour débarquer les passagers en contact, attendre en aire éloignée et finalement embarquer les passagers en contact. Cependant le remorquage nécessite un remorqueur qui est

coûteux et disponible en quantité limitée, le nombre de remorquages doit donc rester raisonnable.

Chaque opération ne peut pas être affectée à n'importe quel point de stationnement et des contraintes de compatibilité doivent être respectées, notamment à cause de la taille respective des avions et des points de stationnement ou de l'accès aux services gouvernementaux comme les douanes. Des conflits d'adjacence sont également à prendre en considération : suivant la configuration des points de stationnement, il n'est parfois pas possible de garer deux avions larges côte à côte (voir Figure 1.7(b)).

La qualité d'un plan d'affectation est définie par de nombreux indicateurs, comme par exemple le nombre de passagers ou d'opérations affectés en contact, le respect des préférences des compagnies aériennes, la facilité de connexion pour les passagers et le nombre de remorquages. Il est aussi important que le plan soit robuste aux perturbations arrivant le jour des opérations.

En cas de trop fortes perturbations, le plan d'affectation doit être réparé et un problème de réaffectation doit être résolu.

1.4.2 Le problème de routage au sol

Le problème de routage au sol (*Ground Rounting Problem, GRP*) consiste à planifier les mouvements des avions au sol entre les différentes infrastructures à travers le réseau de *taxiways*, de la manière la plus efficace possible tout en respectant des contraintes opérationnelles. Une arrivée doit être routée entre la piste d'atterrissage où l'avion se pose et le point de stationnement préalablement affecté. Un départ doit être routé entre le point de stationnement où il est actuellement garé et la piste où il va décoller. A cause des forts aléas auxquels est sujet le routage, notamment sur les heures d'entrée des avions dans le réseau de *taxiway*, ce problème est purement opérationnel et le contrôleur de sol route les avions sur un horizon de typiquement 10 à 40 minutes.

Un des objectifs premiers du contrôleur de sol est de garantir un routage sûr des avions, c'est à dire sans collision. Pour cela, une séparation minimale doit être respectée entre chaque avion et à chaque instant. Certaines règles de compatibilité entre avions et *taxiways* doivent également être prises en compte. En particulier, certaines *taxiways* ne peuvent supporter le poids des plus gros appareils. D'autres règles de routage élémentaires doivent également être respectées, telles que des régulations de vitesse et d'accélération ainsi que le degré d'angle d'un virage.

La qualité d'un planning de routage dépend de nombreux indicateurs, comme l'efficacité du routage, le temps de roulage et la ponctualité telle qu'elle est définie dans l'industrie. Le temps de roulage (*taxitime*) mesure le temps qu'un avion passe avec les moteurs allumés, entre l'atter-

rissage et le stationnement pour une arrivée et entre le repoussage et le décollage pour un départ. Il est représentatif de la consommation de carburant et joue donc un rôle prépondérant dans l'impact environnemental de l'aéroport. Il ne comptabilise pas uniquement le temps passé en mouvement mais considère aussi tout temps d'attente à l'arrêt (notamment le temps d'attente à la piste) car les moteurs ne peuvent être éteints après leur allumage. L'efficacité du routage est souvent mesurée par la durée totale du routage dans la littérature, c'est à dire la somme des dates de fin de roulage, ce qui correspond au décollage pour les départs ou au stationnement pour les arrivées (*total completion time* dans la théorie de l'ordonnancement). Dans l'industrie, la ponctualité est mesurée par rapport à l'heure planifiée d'arrivée et de départ du point de stationnement. Les indicateurs majeurs sont le retard total (ou moyen) et l'*On Time Performance* (OTP), c'est à dire le pourcentage de vols ayant moins de 15 minutes de retard.

1.4.3 Le problème d'ordonnancement à la piste

Le problème d'ordonnancement à la piste (*Runway sequencing problem, RSP*) consiste à planifier les opérations de pistes (décollage, atterrissage et traversée) de manière à utiliser au mieux la capacité de piste. Si l'infrastructure de l'aéroport comprend plusieurs pistes, l'affectation de piste peut également faire partie du problème. Comme pour le routage, du fait des forts aléas, ce problème est purement opérationnel et le contrôleur de piste planifie les opérations sur un horizon de typiquement 10 à 40 minutes.

Un des objectifs premiers des contrôleurs de piste est d'assurer la sécurité des avions, c'est à dire le respect de séparations minimales entre les différents appareils afin d'éviter les collisions et la traversée de turbulence (tourbillon de sillage, voir Section 1.3.2). Ce sont ces contraintes qui limitent principalement la capacité des pistes.

La qualité d'une séquence de piste est définie par de nombreux indicateurs. Les principaux indicateurs sont liés à l'efficacité et à la bonne utilisation de la capacité, souvent mesurées dans la littérature par le débit, la somme des dates de fin ou la déviation à une heure de décollage ou atterrissage cible. L'équité entre les différents avions est également un critère important.

1.4.4 Autres problèmes d'optimisation liés à la rotation d'un avion

Comme mentionné en Section 1.3, de nombreuses opérations ont lieu sur un avion lorsqu'il est garé à son point de stationnement : l'avion commence par se stationner, les passagers débarquent, les bagages et la nourriture restante sont déchargés, la cabine est nettoyée, l'avion est ravitaillé en kérosène et rapprovisionné en nourriture et en eau saine, les passagers embarquent et l'avion

est enfin repoussé. Il est également parfois nécessaire que l'avion soit dégivré. De plus amples informations et une description précise de chaque étape sont fournies par van Leeuwen [2007] et Norin [2008].

La gestion de ces opérations impliquent des problèmes d'optimisation. Elles sont souvent très contraintes par le planning d'affectation aux points de stationnement, qui définit des fenêtres de temps fixes et relativement courtes. La plupart des opérations nécessite des opérateurs et des véhicules spécifiques, impliquant des problèmes de planification des forces de travail (*workforce scheduling problem*) et des problèmes de tournées de véhicules avec fenêtre de temps (*vehicle routing problem with time window, VRPTW*). Les véhicules sont parfois obligés de retourner à des dépôts ou aux terminaux entre deux avions ce qui implique plutôt des problèmes d'ordonnancement avec fenêtre de temps.

L'embarquement est l'une des étapes les plus longues des opérations au sol et l'ordre dans lequel les passagers montent à bord influe grandement sur sa durée. En effet, les couloirs entre les sièges étant étroits, les passagers chargeant leur bagage cabine dans les coffres prévus à cet effet bloquent les passagers souhaitant aller plus loin dans la cabine. Une revue de la littérature de ces problèmes liés à l'embarquement est proposée par Jaehn [2015].

Enfin, la répartition des bagages dans l'appareil est contrainte notamment par l'équilibre de la charge. Le chargement des soutes soulève donc des problèmes d'optimisation particulièrement pour les avions cargos (voir par exemple Mongeau and Bes [2003]).

1.5 Résumé des contributions

Nous allons maintenant résumé nos contributions sur les trois problèmes que nous étudions.

Le problème d'affectation aux points de stationnement

Nos contributions principales sur ce problème sont une modélisation par un programme linéaire en nombre entier (PLNE ou *MIP*) ainsi que différentes techniques visant à accélérer le temps de résolution de celui-ci, comme la reformulation de certaines contraintes, le changement de certaines variables ou encore le cassage de certaines symétries du problème. Nous proposons également des heuristiques de décomposition spatiale et temporelle où notre modèle est appliqué sur chaque sous-problème. Nous montrons par une étude numérique approfondie, basée sur des données réelles d'aéroports majeurs européens, que notre modèle peut être résolu exactement en un temps raisonnable, suffisamment court pour pouvoir être utilisé en l'état dans l'industrie. Nous montrons également que nos heuristiques permettent de réduire significativement le temps

de calcul tout en fournissant des solutions très proches de l'optimal. Enfin, une comparaison aux méthodes de la littérature montre que nos méthodes fournissent des plans d'affectation significativement meilleurs. La complexité théorique du problème est également abordée et nous montrons que trouver une solution réalisable au problème est un problème NP-complet. Ce résultat est utilisé pour montrer la NP-difficulté de plusieurs cas particuliers du problème d'optimisation non couverts par la littérature.

Ces travaux ont été publiés dans *European Journal of Operational Research* (Guépet et al. [2015]). Ils ont contribué au développement d'un produit à Amadeus qui a été commercialisé en 2014 [6].

Le problème de routage au sol

Nos principales contributions sur ce problème sont une analyse des relations entre les différents indicateurs de la littérature (temps de roulage et efficacité) et de l'industrie (retard et OTP) à travers une expérimentation basée sur l'aéroport de Copenhague (CPH). Pour cela, nous modélisons les indicateurs de l'industrie dans un programme linéaire en nombre entier issu de la littérature dans lequel les avions sont routés selon un chemin prédéterminé. Ce modèle est généralisé pour considérer des chemins alternatifs. Nos expérimentations révèlent que les indicateurs de l'industrie sont en contradiction avec l'objectif de réduire le temps de roulage des départs. Dans cette optique, nous proposons de nouveaux indicateurs de ponctualité, qui sont à la fois plus écologiques mais aussi plus logiques pour chaque acteur. Elles révèlent également que considérer des chemins alternatifs n'a que peu d'intérêt, en sus d'accroître significativement la difficulté de résolution du modèle. Nous montrons enfin que la piste de décollage et les *taxiways* avoisinant les points de stationnement sont les principaux goulets d'étranglement de l'aéroport de Copenhague, particulièrement pendant les pics de départs.

Ces travaux ont été publiés dans *European Journal of Operational Research* (Guépet et al. [2016]).

Le problème d'ordonnancement à la piste

La revue de la littérature révèle que le problème d'ordonnancement à la piste considéré comme isolé a été très largement étudié et que de nombreuses modélisations et techniques de résolution ont été investiguées. Notre contribution sur le problème isolé est une reformulation des

6. http://www.amadeus.com/web/binaries/blobs/924/195/14AMIT009-CR-BR-Sales-Sheet-11-FixedRMS-V6,0.pdf

contraintes de séparation d'un modèle de la littérature, réduisant significativement le temps de résolution de celui-ci.

Notre principale contribution sur ce problème est l'étude de son intégration avec le problème de routage au sol. Nous adressons le problème intégré par une méthode heuristique séquentielle : dans un premier temps, les avions sont séquencés à la piste, puis routés au sol. L'ordonnancement à la piste est fait par le biais d'une formulation en PLNE innovante et intégrant les principaux conflits du routage que nous avons identifiés : les conflits dans les zones avoisinant les points de stationnement. Nous proposons diverses techniques visant à améliorer le temps résolution de notre modèle. Nous évaluons notre méthode par une étude expérimentale basée sur l'aéroport de Copenhague (CPH). Nous la comparons à deux méthodes proposant une intégration moins élaborée et à une modélisation en PLNE issue de la littérature intégrant directement les deux problèmes. Il en ressort deux résultats principaux. Premièrement, une meilleure intégration de l'ordonnancement à la piste et du routage au sol présente un réel intérêt : le temps de roulage et l'efficacité, mesurée par la somme des dates de fin, sont significativement améliorées. Deuxièmement, notre méthode, avec les améliorations apportées, offre des solutions de haute qualité en des temps de calcul raisonnables, contrairement à la formulation exacte issue de la littérature.

1.6 Plan du manuscrit

Cette thèse est structurée de la façon suivante. Le chapitre 2 présente nos travaux sur le problème d'affectation aux points de stationnement. Le chapitre 3 traite du problème de routage des avions aux sols. Le chapitre 4 introduit le problème d'ordonnancement à la piste et le chapitre 5 présente nos travaux sur son intégration avec le problème de routage au sol. Une conclusion ainsi que les perspectives de la thèse sont proposés dans le chapitre 6.

Chapter 2

Exact and heuristic approaches to the airport stand allocation problem

Abstract

The Stand Allocation Problem (SAP) consists in assigning aircraft activities (arrival, departure and intermediate parking) to aircraft stands (parking positions) with the objective of maximizing the number of passengers/aircraft at contact stands and minimizing the number of towing movements, while respecting a set of operational and commercial requirements. We first prove that the problem of assigning each operation to a compatible stand is NP-complete by a reduction from the circular arc graph coloring problem. As a corollary, this implies that the SAP is NP-hard. We then formulate the SAP as a Mixed Integer Program (MIP) and strengthen the formulation in several ways. Additionally, we introduce two heuristic algorithms based on a spatial and time decomposition leading to smaller MIPs. The methods are tested on realistic scenarios based on actual data from two major European airports. We compare the performance and the quality of the solutions with state-of-the-art algorithms. The results show that our MIP-based methods provide significant improvements to the solutions outlined in previously published approaches. Moreover, their low computation make them very practical.

Keywords: Mixed integer programming, gate assignment problem, heuristic algorithms

J. Guépet, R. Acuna-Agost, O. Briant, and J.P. Gayon. Exact and heuristic approaches to the airport stand allocation problem. *European Journal of Operational Research*, 246(2):597-608, 2015.

2.1 Introduction

Every day, airports deal with different decisions related to aircraft movements. These decisions usually involve the use of fixed and limited resources such as runways, stands (parking positions) and passenger gates. Due to the growing flow of passengers, these resources are falling short of needs while activity planning is increasingly crucial and complex. Consequently, some airports have experienced deterioration in service quality. In one of our partner airports, the number of passengers allocated to remote stands has increased in the last years. This affects passenger connection times, increases bus transfer costs and decreases airport revenue given that airlines usually pay lower fees for flights allocated to remote stands. Since building new terminal gates is expensive and does not provide a short-term solution, value can only be gained from better management of airport resources.

In this chapter, we deal with the Stand Allocation Problem (SAP). This consists in assigning aircraft operations to available stands in line with operational requirements and different objectives. This problem is closely related to the Gate Allocation Problem (GAP). Our work results from close collaboration between the laboratory G-Scop and the company Amadeus. In what follows, we provide a detailed description of the stands, aircraft operations, operational requirements and the different objectives to be taken into account for solving the SAP.

A stand is an aircraft parking position. Figure 2.1 illustrates the two types of possible stands: contact stands (i.e., stands touching an airport terminal gate) and remote stands (i.e., stands where a bus is needed to reach the terminal). Airports and airlines usually prefer contact stands as they are more convenient for passengers and no bus transfer is necessary.

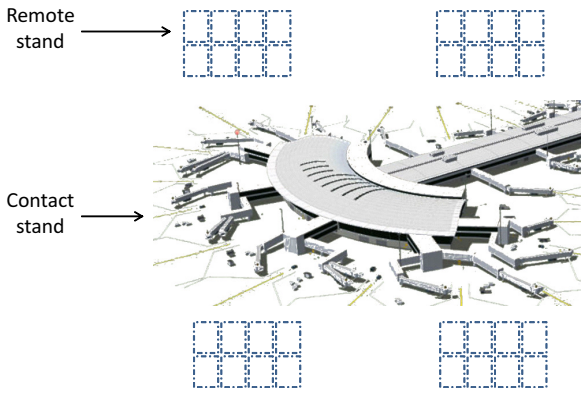


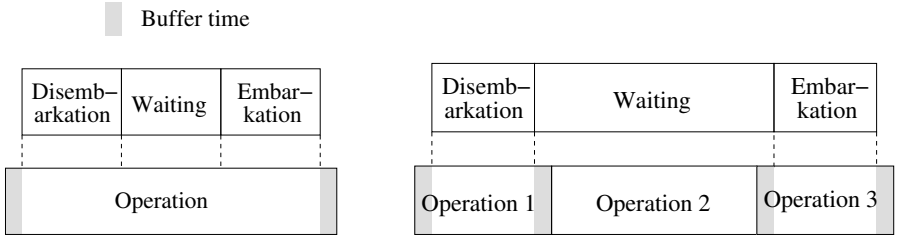
Figure 2.1: Airport stands

The stand operations of an aircraft turnaround can be roughly divided into three parts: disembarkation of the arrival flight, waiting, and embarkation of the departure flight. Disembarkation concerns passengers and luggage and also involves aircraft ground handling operations (refueling, cabin services, catering, etc.) linked to the aircraft's arrival. Similarly, embarkation concerns passengers and luggage and other related ground handling operations. The waiting period can be null if the turnaround is short. During the waiting period, airport operators may decide to tow (move) aircraft to other stands. This can be for several reasons but usually targets a better utilization of valuable stands (e.g. contact stands). However, these operations require an expensive towing tractor (see Figure 2.2) and increase airport congestion. The data provided by our partner airports shows that, at most, two towing operations are performed during a turnaround: one after disembarkation and one before embarkation. Consequently, we assume that turnarounds are split into three operations at most.

Figure 2.2: A towing tractor

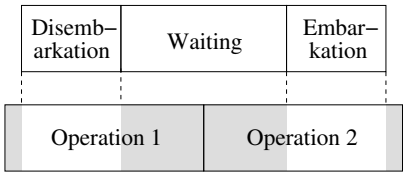
In order to define operations, we need to distinguish between three situations depending on the waiting period length (see Figure 2.3). If the waiting period is too short to move the aircraft (case (a)), then we consider that we only have to schedule a single operation since disembarkation, waiting and embarkation will necessarily take place at the same stand. In order to make the assignment plan robust in the face of small disruptions such as short delays or early arrivals, we add a buffer time at the beginning and end of this single operation. If the waiting period is long enough to move the aircraft twice (case (b)), then we split the turnaround into three operations since an aircraft can potentially disembark at one stand, wait at a second stand and embark at a third stand. We add a buffer time before and after embarkation and disembarkation operations. If the duration of the waiting period is only long enough to move the aircraft once but not twice (case (c)), then the turnaround is split into two operations with the

waiting time equally distributed between both operations and providing of a buffer time. Note that a different distribution of the waiting time is possible, but the one described above seems to be the most natural. We also add a buffer time before the embarkation operation and after the disembarkation operation. When towing is allowed (cases (b) and (c)), the towing time is much shorter than the disembarkation and embarkation times. Hence these can be included in the operations, which simplifies modeling even if it results in a slight overestimation of processing times. Indeed, this approach gives flexibility for actually performing the towing during the operations. In what follows, the set of operations, with fixed start and end time, is considered an input of the problem and is given by the airport.



(a) Short waiting time - Do not split the turnaround

(b) Long waiting time - Split in 3 operations



(c) Medium waiting time - Split in 2 operations

Figure 2.3: Splitting turnarounds in operations and adding buffer times

The assignment of aircraft operations to stands must take into account aircraft-stand compatibility. Indeed, not all aircraft can be assigned to all the stands because of size compatibility but also because of aircraft flight requirements. For example, some stands are forbidden to international flights because they do not offer access to governmental inspection facilities. Furthermore, two overlapping operations must not be assigned to the same stand. Finally, adjacency conflicts, also called shadow restrictions, must be taken into account, e.g. two large aircraft cannot be assigned to adjacent stands simultaneously.

The quality of an assignment plan can be defined using several, often competing criteria, such as the number of unassigned operations, the number of passengers at contact stands, compliance with airline preferences, passenger connection convenience or the number of towing operations.

In practice, an unassigned operation has to be handled manually, either overstepping certain requirements or delaying a flight. One option is to assign an operation to a non compatible stand and to transfer passengers to a compatible terminal area by bus. Another option is to keep the aircraft waiting on the tarmac.

In the literature, several authors consider the objective of minimizing passengers' walking distance or connection time (see Section 2.2). However, this is not always a suitable approach for airports since a large share of their revenue comes from the shops hosted in the terminal. The more passengers walk, the more likely they are to go into a shop and buy something thus boosting the airport's revenue.

For our partner airports, the assignment of aircraft activities is generally decided, at the latest, the day before the operations. In this phase, computation time is not overly problematic. However, on the day the operations are scheduled, disruptions can happen. Many random events may occur, leading to delays and flight cancellations. New flights (e.g. general aviation) and diversions can also impact planning. Hence, the assignment must be robust in the sense that small disruptions must not oblige airport authorities to change the whole assignment plan. Bigger disruption may oblige the airport to reassign aircraft. In this case, computation times need to be very short.

The Stand Allocation Problem (SAP) is closely related to the Gate Allocation Problem (GAP). A gate is the boarding desk where passengers' tickets are checked by the airline and a stand is the position where the aircraft is parked. In many US airports, embarking and disembarking passengers at remote stands is forbidden. Consequently, there is a perfect match between stands and gates, and therefore between the SAP and the GAP. In Europe, this is not often the case since embarking and disembarking can be done at a remote stand that can be associated with different gates (called bus gates). As we work with European airports, we will use SAP terminology.

To explore the SAP, this chapter has been organized in several sections. A review of the literature and a summary of our contributions are presented in Section 2.2. In Section 2.3, we formally introduce the SAP and associated feasibility problem. Section 2.4 proves the NP-hardness of SAP and the NP-completeness of the associated feasibility problem. Section 2.5 presents a mixed integer programming formulation and a number of improvements designed to strengthen. Section 2.6 presents two MIP based heuristic algorithms. Computational experiments are presented in Section 2.7 to show the efficiency of the model and the performance of heuristic algorithms for realistic instances. A conclusion is finally given in Section 2.8.

2.2 Literature review

This literature review focuses on deterministic approaches related to mathematical programming for the GAP. More references to stochastic and expert system approaches can be found in the survey by Dorndorf et al. [2007].

The GAP has been widely studied since 1980. Many models aim at minimizing passenger walking distances or connection times, which naturally leads to a 0-1 quadratic integer program (QIP) close to the Quadratic Assignment Problem. Different methods can be found for solving it. Mangoubi and Mathaisel [1985] propose using the average distance of a gate to other gates and a greedy algorithm to solve the integer program (IP) thus obtained. Yan and Chang [1998] use the same assumption for modeling walking distance and propose a multi-commodity network flow model. They propose a Lagrangian relaxation solved by a sub-gradient algorithm and heuristics. Haghani and Chen [1998] use the classical linearization of the product of binary variables and propose a heuristic algorithm that consists in iterating a greedy algorithm. Xu and Bailey [2001] consider the same model and solve it using a Tabu Search algorithm. Ding et al. [2005] add the objective of minimizing ungated flights and directly solve the quadratic model using a hybrid Tabu Search and Simulated Annealing. Yan and Huo [2001] consider a different IP model minimizing walking distances and connection times. They propose a sensitivity analysis to reduce the number of variables.

Minimizing walking distance tends to concentrate traffic at the best located gates, which can lead to non robust solutions. Indeed, if a flight is delayed, its ground time is increased and it may overlap with the next operation assigned to the same gate. The robustness of an assignment plan is an important objective in the literature. Bolat [2000] proposes a model minimizing the variance of idle times between two consecutive flights assigned to the same gate. He proposes a branch-and-bound algorithm and heuristic algorithms for solving the model. Lim and Wang [2005] propose a stochastic programming model that is transformed into a binary programming model to minimize the expected number of gate conflicts. They propose a hybrid meta-heuristic for solving their model. Yan and Tang [2007] propose a heuristic approach for minimizing flight delays due to gate blockages and reassignments. Their approach consists in iterating between two stages: a planning stage based on a multi-commodity flow network and a real-time stage based on simulations and reassignment rules for updating the planning stage. Diepen et al. [2012] suggest a column generation approach in order to establish robust assignment plan for Amsterdam Airport Schiphol (AAS). They identify gate types, i.e. groups of similar gates, and

proceed in two phases. They first assign flights to the gate type through a column generation process aiming at generating good gate plans. Then a gate plan is assigned to each physical gate.

New objectives have appeared more recently in the literature. Dorndorf et al. [2008] take into account towing operations and shadow restrictions. They model the GAP as a Clique Partitioning Problem. Their model aims at simultaneously maximizing the total flight-gate affinity, minimizing the number of towing operations, minimizing the number of ungated flights and maximizing robustness by minimizing low buffers (idle time shorter than a given limit). A linear combination of these objectives is considered and the problem is solved by an ejection chain algorithm. Dorndorf et al. [2010] extend this model to minimize the deviation from a reference schedule. They suggest a method for building a reference schedule over a multi-period time horizon. Jaehn [2010] proposes a dynamic programming approach to solve a particular case of Dorndorf et al. [2008] where only flight-gate affinities are considered. He also proves the problem NP-hardness with a reduction from the optimal cost chromatic partition problem.

Kim et al. [2009] propose a new 0-1 QIP model for minimizing push back conflicts and taxi blocking. The model is further extended by Kim et al. [2013] to include the minimization of passenger transit times and baggage transport distances. They propose a tabu search and compare it to a linearization of the QIP and to a genetic algorithm for different airport configurations (parallel and horseshoe terminals) with randomly generated operations.

Genç et al. [2012] consider a new GAP, the objective of which is to maximize the total gate occupation time. Time is discretized in time slots of 5 or 10 minutes and the objective is to maximize the number of gate time slots used. They use a Big Bang Big Crunch method for solving instances from Istanbul Atatürk International Airport. Note that maximizing gate occupation time tends to reduce idle time at gates, which can lead to non-robust assignment plans.

In this chapter, we consider the problem introduced by Dorndorf et al. [2008]. The problem is referred to as the SAP since we consider both contact and remote stands. From a theoretical point of view, the SAP and GAP are equivalent. Our contributions to the SAP are summarized in what follows. We first prove that assigning each operation to a compatible stand is NP-complete based on a reduction from the circular arc graph coloring problem. As this corollary, it provides alternative proof for SAP NP-hardness compared with the proof given by Jaehn [2010]. We also prove the NP-hardness of particular cases left open by Jaehn [2010]. While the literature considers heuristic algorithms, we propose a strong mixed integer programming (MIP)

formulation that solves to optimality real-size instances in reasonable computation times. We also introduce two heuristic algorithms based on spatial and time decompositions. In a numerical study, we compare our MIP-based approaches to the ejection chain algorithm described by Dorndorf et al. [2008] and to a simple greedy algorithm that mimics industrial practices (see Section 2.6). All methods are tested on real instances from two major European airports. MIP-based approaches are significantly better while computation times remain short enough for industrial purposes.

2.3 The stand allocation problem

In this section we formally introduce the Stand Allocation Problem (SAP) and the Stand Allocation Feasibility Problem (SAFP).

The ingredients for SAP can be summarized as follows:

- $O = \{1, .., m\}$ the set of operations. Operation $i \in O$ is defined by a start time a_i and an end time d_i, where $a_i < d_i$. a_i and d_i are assumed to be integers. Start and end times will often be referred to as the arrival and departure times.
- $S = \{1, .., n, n+1\}$ the set of stands. Stand $n+1$ is a dummy stand modeling unassignment, i.e. being assigned to stand $n+1$ is equivalent to being unassigned. We will also use the notation $\tilde{S} = S \backslash \{n+1\}$ for the set of real stands.
- $S_i \subset S$ the set of compatible stands for operation $i \in O$. Obviously, $n+1$ belongs to S_i for each operation $i \in O$. We will also use the notation $\tilde{S}_i = S_i \backslash \{n+1\}$.
- $U : O \to O \cup \{0\}$ the successor function. $U(i)$ is the direct successor of operation i for a given aircraft; i.e., if a turnaround is divided in two operations i and i', then $U(i) = i'$. Conventionally, if operation i does not have a successor then $U(i)$ is equal to 0. The end time d_i of an operation $i \in O$ is supposed to be equal to the start time $a_{U(i)}$ of its successor if there is one.
- $Q \subseteq O^2 \times S^2$ the set of shadow restrictions. If $(i, i', j, j') \in Q$ and operation i is assigned to stand j then operation i' cannot be assigned to stand j' and reciprocally.
- $c = (c_{ij})_{O \times S}$ the affinity matrix, i.e. c_{ij} is the affinity realized if operation $i \in O$ is assigned to stand j.

Shadow restrictions represent adjacency conflicts (e.g. two large aircraft cannot be simultaneously assigned to adjacent stands due to space limitations). It should be noted that the dummy stand $n+1$ is not concerned by either overlapping or shadow restrictions.

An assignment can be seen as a mapping $\mathcal{A}$ from the set of operations O to the set of stands S. The evaluation $f(\mathcal{A})$ of an assignment $\mathcal{A}$ is defined as

$$f(\mathcal{A}) = \alpha f_1(\mathcal{A}) - \beta f_2(\mathcal{A}) \tag{2.1}$$

where α and β are non negative and $f_1(\mathcal{A})$ and $f_2(\mathcal{A})$ are the total operation-stand affinity and the number of towing operations, respectively. Without loss of generality, we set $\alpha = 1$ in what follows.

The objective is to find an assignment maximizing $f(\mathcal{A})$ while respecting operation-stand compatibilities, shadow restrictions and overlapping constraints. In order to avoid assignment to the dummy stand, the affinity of an operation i for the dummy stand $n+1$ can be set to a high negative value.

Finally the Stand Allocation Feasibility Problem (SAFP) is the problem of determining whether there is a feasible solution not using the dummy stand.

2.4 Complexity of the stand allocation problem

In this section, we focus on a special case without successor relations ($U(i) = 0, \forall i \in O$ and $\beta = 0$) and without shadow restrictions ($Q = \emptyset$). An instance of the SAP can thus be denoted as $I(O, S, S_i, c)$. An instance of the associated feasibility problem SAFP is denoted as $I(O, \tilde{S}, \tilde{S}_i)$.

We first present the current complexity status of the SAP and highlight a number of open special cases. Then, we show how to formulate the SAFP as a graph coloring problem and prove its NP-completeness by a polynomial reduction from the circular arc graph coloring problem. Finally, we show the NP-hardness of a number of special SAP cases by polynomial reductions from SAFP.

2.4.1 Current complexity status and contributions

Jaehn [2010] proves that the SAP is NP-hard. His proof is based on a special case without compatibility constraints and where operations have the same affinity for each stand ($c_{ij} = c_j \in \mathbb{N}$). This case is proven NP-hard by a polynomial reduction from the Optimal Cost Chromatic Partition Problem (OCCP) in interval graphs. Kroon et al. [1997] prove that the OCCP in interval graphs is polynomial when c_j take at most 2 different values (e.g. $c_j \in \{0, 1\}$). They also show that it is NP-hard when c_j take at least 4 different values (e.g. $c_j \in \mathbb{N}$) while the problem is left open when c_j take exactly 3 different values (e.g. $c_j \in \{0, 1, 2\}$).

In the computational experiments (see Section 2.7), we consider three affinity functions that model different realistic situations : $c_{ij} \in \{0,1\}$, $c_{ij} \in \{0,1,2\}$ and $c_{ij} \in \mathbb{N}$. Jaehn's proof does not provide a conclusion with respect to the complexity status when $c_{ij} \in \{0,1\}$ or $c_{ij} \in \{0,1,2\}$. We will show that these special cases are also NP-hard. We will also prove that the SAP with compatibility constraints is NP-hard, for any of the above affinity functions.

Table 2.1 summarizes the results from the literature and our own contributions.

Affinity	Without compatibility constrains	With compatibility constraints
$c_{ij} \in \{0,1\}$	NP-hard (*)	NP-hard (*)
$c_{ij} \in \{0,1,2\}$	NP-hard (*)	NP-hard (*)
$c_{ij} \in \mathbb{N}$	NP-hard [Jaehn, 2010]	NP-hard [Jaehn, 2010]
$c_{ij} = c_j \in \{0,1\}$	P [Jaehn, 2010, Kroon et al., 1997]	NP-hard (*)
$c_{ij} = c_j \in \{0,1,2\}$	Open [Kroon et al., 1997]	NP-hard (*)
$c_{ij} = c_j \in \mathbb{N}$	NP-hard [Jaehn, 2010]	NP-hard [Jaehn, 2010]

Table 2.1: Complexity status of the stand allocation problem. (*) indicates the new results established in this chapter.

2.4.2 The stand allocation feasibility problem as a graph coloring problem

In this section, we show that the SAFP can be modeled by a graph coloring problem. Let $I(O, \tilde{S}, \tilde{S}_i)$ be an instance of SAFP. Let $G_I = (V \cup W, E)$ be an undirected graph where $V = \{v_1, .., v_n\}$ and $W = \{w_1, .., w_m\}$. Vertex v_j corresponds to stand $j \in \tilde{S}$ and vertex w_i corresponds to operation $i \in O$. To simplify matters, we will speak of stands and operations for vertices of V and W. The edges of the graph are defined as follows:

- $v_j v_{j'} \in E \quad \forall j, j' \in \tilde{S}$ such that $j \neq j'$,
- $w_i w_{i'} \in E \quad \forall i, i' \in O$ such that $i \neq i'$ and $[a_i, d_i[\cap [a_{i'}, d_{i'}[\neq \emptyset$, i.e. if operations i and i' overlap,
- $v_j w_i \in E \quad \forall i \in O, j \in \tilde{S} \backslash \tilde{S}_i$, i.e. if operation i and stand j are incompatible.

Graphs G_I will be denoted as SAFP graphs. Figure 2.4 provides an example of such a graph.

It should be noted that the graph induced by V is the clique K_n, thus G_I cannot be colored with less than n different colors. The graph induced by W is an interval graph. These subgraphs are linked by edges representing incompatibility constraints.

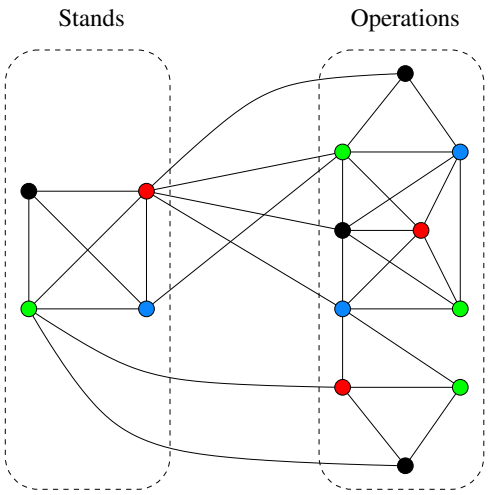


Figure 2.4: A 4-coloration of a SAFP graph

Property 1. *There is a feasible solution to an instance $I(O, \tilde{S}, \tilde{S}_i)$ of SAFP if and only if G_I admits a n-coloring.*

Proof. If G_I can be n-colored, then a feasible solution of I can be built from any n-coloring of G_I. Indeed, we assign each operation of a given color to the stand of the same color. Based on the construction of G_I, operations with the same color do not overlap and operations are compatible with the stand of the same color.

Conversely, if I is a feasible instance, then a n-coloring can be built from any feasible solution of I. A different color is assigned to each stand and each operation is colored with the color of the stand it is assigned to. Based on the construction of G_I, two adjacent nodes do not have the same color. □

This property implies that the SAFP and n-coloring problem of SAFP graphs have the same complexity status.

2.4.3 The circular arc graph coloring problem

The Circular Arc Graph Coloring Problem (CAGCP) was introduced and proven NP-complete by Garey et al. [1980]. A brief overview of this problem is given below.

A circular arc A is a pair of positive integers (e, f) where e and f are different. Let $F = \{A_1, .., A_p\}$ be a set of circular arcs and k the maximum of all e_i and f_i ($k = \max\{e_i, f_i \mid A_i = (e_i, f_i), i \in \{1, .., p\}\}$). Consider a geometric arrangement of circular arcs as follows. A circle can be regarded as divided into k parts defined by k equally spaced points numbered clockwise as $1, 2, .., k$. In such a circle, each circular arc A_i previously defined can be regarded as representing

an arc from point e_i to point f_i again in a clockwise direction. The span $Sp(A_i)$ of an arc $A_i = (e_i, f_i)$ is:

$$Sp(A_i) = \begin{cases} \{e_i + 1, .., f_i\} & \text{if } e_i < f_i \\ \{f_i + 1, .., k, 1, .., e_i\} & \text{if } e_i > f_i \end{cases}$$

Two arcs intersect if the intersection of their spans is not empty, i.e. $Sp(A_i) \cap Sp(A_j) \neq \emptyset$. Note that arcs do not intersect if they only share end points since the first point does not belong to the span.

We can define graph $G = (F, E)$, where $A_iA_j \in E$ if and only if A_i and A_j intersects. G is the circular arc graph induced by the set of circular arcs F. Figure 2.5 presents different representations of a circular arc graph.

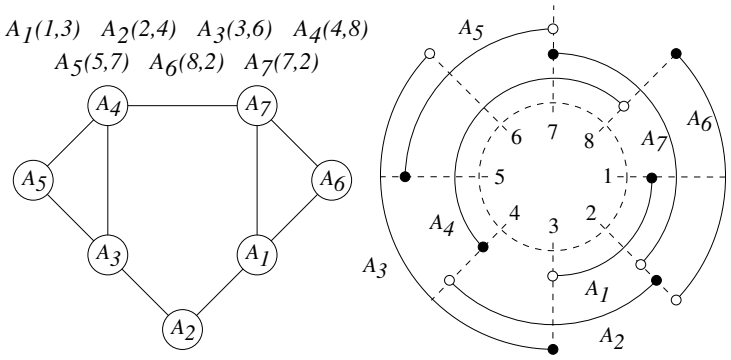


Figure 2.5: Three representations of a circular arc graph

CAGCP is the problem of finding a n-coloring for a circular arc graph. We will now show the relationship between this class of graph and our problem.

2.4.4 Complexity results

The NP-completeness of the SAFP can be shown by a reduction from the CAGCP.

Theorem 1. *The stand allocation feasibility problem (without shadow constraints and successor relations) is NP-complete.*

Proof. The SAFP is in NP as it represents a special case of a n-coloring problem. Let $F = \{A_i, .., A_p\}$ be a set of circular arcs and $G = (F, E)$ the circular arc graph induced by F. It is easy to show that the subgraph induced by $K = \{A_i \in F | e_i > f_i\}$ is a clique and the subgraph induced by $L = \{A_i \in F | e_i < f_i\}$ is an interval graph.

As K is a clique, G cannot be colored with less than $|K|$ colors. Hence, deciding whether G can be n-colored is polynomial if $n < |K|$. Coloring G is trivial if $n \geq p$. Hence, we assume that

$n \in \{|K|, .., p-1\}$ in order to prove the above theorem.

We now build an instance $I(O, \tilde{S}, \tilde{S}_i)$ of the SAFP such that it accepts a solution if and only if G is n-colorable.

- For each circular arc $A_i = (e_i, f_i) \in L$, we define an operation i with the start time $a_i = e_i$ and end time $d_i = f_i$. As $A_i \in L$, $e_i < f_i$ and operation i is well defined.
- For each circular arc A_j of K, we define a stand. Stand j is compatible with operation i if and only if the associated arc A_j does not intersect the associated arc A_i.
- We add $n - |K|$ stands that are compatible with all operations.

Let G_I be the graph associated with I. It should be noted that G and G_I only differ by the vertices associated with the last $n-|K|$ stands. These vertices are only adjacent to other vertices of K. It follows that if G_I is n-colorable, so is G as G is a sub-graph of G_I. The reciprocal is valid because an n-coloring of G_I can be built from an n-coloring of G by assigning the $n - |K|$ colors not used in K to the $n - |K|$ last stands of G_I.

To conclude, G is n-colorable if and only if G_I is n-colorable. Hence coloring G_I is NP-complete. Together with Property 1, this implies the NP-completeness of the SAFP. □

As corollaries of Theorem 1, we now show that some special cases of the SAP, left open by Jaehn [2010], are NP-hard.

Corollary 1. *SAP with compatibility constraints and affinity coefficients verifying* $c_{ij} = c_j \in \{0, 1\}, \forall i \in O, \forall j \in S$ *is NP-hard.*

Proof. Since the SAFP is in NP and a solution can be evaluated in polynomial time, the SAP is in NP. Let us consider an instance $I(O, \tilde{S}, \tilde{S}_i)$ of the SAFP. We define the instance $I(O, S, S_i, c)$ of the SAP as follows:

- $S = \tilde{S} \cup \{|\tilde{S}| + 1\}$, i.e. $|\tilde{S}| + 1$ is the dummy stand,
- $S_i = \tilde{S}_i \cup \{|\tilde{S}| + 1\}$,
- $c_{ij} = \begin{cases} 1 & \forall i \in O, j \in \tilde{S} \\ 0 & \text{otherwise, i.e. for the dummy stand only} \end{cases}$.

$I(O, \tilde{S}, \tilde{S}_i)$ has a feasible solution if and only if $I(O, S, S_i, c)$ has a solution of value $|O|$. Furthermore, we define $I(O, S, S_i, c)$ such that $c_{ij} = c_j \in \{0, 1\}$. This proves the corollary. □

Corollary 2. *SAP without compatibility constraints and affinity coefficients* $c_{ij} \in \{0, 1\}$ *is NP-hard.*

Proof. As in Corollary 1, the SAP is in NP. Let us consider an instance $I(O, \tilde{S}, \tilde{S}_i)$ of the SAFP. We define the instance $I(O, S, S_i, c)$ of the SAP as follows:

- $S = \tilde{S} \cup \{|\tilde{S}| + 1\}$, i.e. $|\tilde{S}| + 1$ is the dummy stand,
- $S_i = S$ (no compatibility constraints),
- $c_{ij} = \begin{cases} 1 & \forall i \in O, j \in \tilde{S}_i \\ 0 & \text{otherwise} \end{cases}$.

$I(O, \tilde{S}, \tilde{S}_i)$ has a feasible solution if and only if $I(O, S, S_i, c)$ has a solution of value $|O|$. This proves the corollary. □

Corollaries 1 and 2 imply the new results presented in Table 2.1. They also provide alternative proof to the results of Jaehn [2010].

The NP-hardness of the special cases considered in this section does not mean that all instances are hard to solve. There may be constraints in industrial problems, making them easier to solve. Nevertheless, we did not identify such sub-structures in the instances considered in Section 2.7.

2.5 A mixed integer programming formulation

In this section, a first mixed integer program (MIP) formulation is presented. This model is then strengthened by reformulating a number of constraints and introducing new variables. Finally, an efficient process to break symmetries is presented.

2.5.1 A natural MIP formulation

Let us introduce the following decision variables:

- $x_{ij} = \begin{cases} 1 & \text{if operation } i \in O \text{ is assigned to stand } j \in S_i \\ 0 & \text{otherwise} \end{cases}$
- $y_i = \begin{cases} 1 & \text{if a towing operation is performed between operation } i \in O \text{ and its successor} \\ & U(i) \text{ if there is one} \\ 0 & \text{otherwise} \end{cases}$

Note that for the sake of simplicity, we define variables $x_{ij} = 0$ for each operation $i \in O$ and each non compatible stand $j \in S \backslash S_i$. Using these variables, the SAP can be formulated as follows:

$$\max \quad \sum_{i \in O} \sum_{j \in S_i} c_{ij} x_{ij} - \beta \sum_{i \in O} y_i \tag{2.2}$$

$$s.t. \quad \sum_{j \in S_i} x_{ij} = 1 \qquad \forall i \in O \tag{2.3}$$

$$x_{ij} + x_{i'j} \leqslant 1 \qquad \forall i, i' \in O, a_i \leqslant a_{i'} < d_i$$
$$\forall j \in S_i \cap S_{i'} \tag{2.4}$$

$$x_{ij} + x_{i'j'} \leqslant 1 \qquad \forall (i, i', j, j') \in Q \tag{2.5}$$

$$x_{ij} - x_{U(i)j} \leqslant y_i \qquad \forall i \in O, U(i) \neq 0,$$
$$\forall j \in S_i \tag{2.6}$$

$$x_{ij} \in \{0, 1\} \qquad \forall i \in O, \forall j \in S_i \tag{2.7}$$

$$y_i \geqslant 0 \qquad \forall i \in O \tag{2.8}$$

MIP 2.1: A natural formulation for SAP

Constraints (2.3) ensure the assignment of each operation to one and only one stand. Constraints (2.4) prevent two overlapping operations from being assigned to the same stand. Constraints (2.5) guarantee that shadow restrictions are respected. Constraints (2.6) ensure that for each operation i towing is needed if the operation is assigned to stand j and not its successor $U(i)$. Note that, according to their definition, $y_i \in \{0, 1\}$ should be imposed. However, since $\beta \geqslant 0$ and since the objective function is maximized, we can simply impose $y_i \geqslant 0$ (2.8). Indeed, in any optimal solution, variable y_i will be set to the smallest value, i.e. 0 or 1 according to constraints (2.6) and (2.8).

2.5.2 A better MIP formulation

We will now strengthen this natural formulation by reformulating a number of constraints, introducing new variables and disrupting the objective function to break symmetries.

Strengthening overlapping and shadow constraints Overlapping constraints (2.4) are weakly formulated and can be reformulated as follows. We introduce overlapping sets O_t as the set of operations overlapping time line t

$$O_t = \{i \in O \mid a_i \leqslant t < d_i\}$$

Overlapping constraints (2.4) can be replaced by

$$\sum_{i' \in O_{a_i}} x_{i'j} \leqslant 1 \quad \forall i \in O, \forall j \in S_i \tag{2.9}$$

This formulation can be proven to be is ideal, i.e. describes the convex hull of integer solutions that satisfy overlapping constraints.

The same principle can be applied to strengthen shadow constraints. Constraint (2.10) is valid for any pair of stands $(j, j') \in S^2$ and any set of operations H and H' such that

1. each pair of operations $(i, k) \in H$ overlap,
2. each pair of operations $(i', k') \in H'$ overlap,
3. there is a shadow restriction (i, i', j, j') between each operation $i \in H$ and each operation $i' \in H'$ on stands j and j'.

$$\sum_{i \in H} x_{ij} + \sum_{i' \in H'} x_{i'j'} \leqslant 1 \tag{2.10}$$

Nevertheless, the number of pairs of sets (H, H') suffers from a combinatorial explosion, even if only maximal sets are considered. We can heuristically aggregate the shadow constraints with the following algorithm.

While there are uncovered shadow restrictions $(i, i', j, j') \in Q$:

1. Let $H = \{i\}$ and $H' = \{i'\}$.
2. Complete set H: for each operation $k \in O$ (by increasing order of start time), add k to H if $(k, i', j, j') \in Q$ and if k overlaps each operation in H.
3. Complete set H': for each operation $k' \in O$ (by increasing order of start time), add k' to H' if for each operation $k \in H$, $(k, k', j, j') \in Q$ and k' overlaps every operation in H'.

$\mathcal{H}_{jj'}$ denotes the set of couples (H, H') generated by our algorithm for stands $j \in S$ and $j' \in S$.

Improving towing formulation The linear relaxation can be strengthened by introducing variables

$$y_{ij} = \begin{cases} 1 & \text{if operation } i \in O \text{ is assigned to stand } j \in S_i \text{ and not its successor (if there is one),} \\ 0 & \text{otherwise.} \end{cases}$$

The objective function becomes

$$\sum_{i \in O} \sum_{j \in S_i} c_{ij} x_{ij} - \beta \sum_{i \in O} \sum_{j \in S_i} y_{ij}$$

and towing constraints (2.6) become

$$x_{ij} - x_{U(i)j} \leqslant y_{ij} \quad \forall i \in O, \forall j \in S_i, U(i) \neq 0 \tag{2.11}$$

Indeed, it can be seen that the linear relaxation of both formulations has the same feasible domain in $x = (x_{ij})_{O \times S_i}$. Furthermore, for a given x, the optimal values of y variables in the linear relaxation are

- $y_i = \max\limits_{j \in S_i}(x_{ij} - x_{U(i)j})$ for the first formulation,
- $y_{ij} = \max\{0, x_{ij} - x_{U(i)j}\}$ for the second formulation.

Consequently, $y_i = \max\limits_{j \in S_i} y_{ij}$ and

$$\sum_{i \in O} y_i = \sum_{i \in O} \max_{j \in S_i} y_{ij} \leqslant \sum_{i \in O} \sum_{j \in S_i} y_{ij}$$

Therefore the formulation using variables y_{ij} is stronger.

Summary To conclude, the problem can be reformulated as MIP 2.2.

$$\begin{aligned}
\max \quad & \sum_{i \in O} \sum_{j \in S_i} c_{ij} x_{ij} - \beta \sum_{i \in O} \sum_{j \in S_i} y_{ij} & \\
s.t. \quad & \sum_{j \in S_i} x_{ij} = 1 & \forall i \in O \\
& \sum_{i' \in O_{a_i}} x_{i'j} \leqslant 1 & \forall i \in O, \forall j \in S_i \\
& \sum_{i \in H} x_{ij} + \sum_{i' \in H'} x_{i'j'} \leqslant 1 & \forall j, j' \in S, \\
& & \forall (H, H') \in \mathcal{H}_{jj'} \\
& x_{ij} - x_{U(i)j} \leqslant y_{ij} & \forall i \in O, U(i) \neq 0, \\
& & \forall j \in S_i \\
& x_{ij} \in \{0, 1\} & \forall i \in O, \forall j \in S_i \\
& y_{ij} \geqslant 0 & \forall i \in O, \forall j \in S_i
\end{aligned}$$

MIP 2.2: An improved formulation for the SAP

Breaking symmetries If coefficients c_{ij} belong to a small set of values, this implies a high multiplicity of optimal solutions limiting the efficiency of branch-and-bound algorithms. For instance, some airports set c_{ij} to 1 for each contact stand and to 0 for each remote stand. A

simple way to break symmetries is to disturb coefficients c_{ij}. We propose the following disruption that does not affect the optimal solution.

Property 2. *Assume that coefficients β and c_{ij} are integer (for all $i \in O$ and $j \in S_i$). Let γ_{ij} be arbitrary real numbers in $[0, 1)$ and $\delta_{ij} = \frac{\gamma_{ij}}{(m+1)}$.*

Thus, any optimal solution of the SAP with coefficients $c'_{ij} = c_{ij} + \delta_{ij}$ is also optimal for the SAP with coefficients c_{ij}.

Proof. Let f and f' be the objective functions of the original and disrupted problem. It should be noted that both problems have the same feasible solutions as they only differ by their objective functions. Let x be a feasible solution, then $f'(x) = f(x) + \varepsilon(x)$ with $\varepsilon(x) = \sum_{i \in O} \sum_{j \in S_i} \delta_{ij} x_{ij}$. We have $0 \leq \varepsilon(x) < 1$. Since coefficients β and c_{ij} are integers, $f(x)$ is also an integer and $\lfloor f'(x) \rfloor = f(x) + \lfloor \varepsilon(x) \rfloor = f(x)$.

Let x^* be an optimal solution for f'. For each feasible solution x we have $f(x) = \lfloor f'(x) \rfloor \leqslant \lfloor f'(x^*) \rfloor = f(x^*)$ and x^* is also optimal for f. □

In the numerical experiments, γ_{ij} is chosen randomly in [0,1) according to a uniform distribution.

2.6 Heuristic approaches

Regardless of how improved an MIP formulation can be, there exists instances that cannot be solved in a reasonable time. In this section, we present four heuristic algorithms that will be numerically compared to the exact MIP method in Section 2.7. The first two algorithms consist in splitting the problem into smaller sub-problems for which the MIP can be solved more quickly. The third algorithm is a greedy algorithm reflecting what was observed in practice in one of our partner airports. The fourth algorithm is the ejection chain algorithm designed by Dorndorf et al. [2008].

2.6.1 Spatial (or stand) decomposition

In the airports we work with, setting the affinity c_{ij} to 0 for remote stands is a reasonable assumption. This is not true for all airports since some remote stands might be preferable to others (e.g. short driving distance, stands that can be reached without a bus transfer, etc).

The stand decomposition method consists in splitting the set of stands into two disjunctive subsets. Subset B_1 contains stands with a positive affinity for at least one operation (typically

contact stands). Subset B_2 contains the other stands with zero affinity for all operations (typically remote stands). Formally, we have $\tilde{S} = B_1 \cup B_2$ with $B_1 = \{j \in \tilde{S} : \exists i \in O, c_{ij} > 0\}$ and $B_2 = \{j \in \tilde{S} : \forall i \in O, c_{ij} = 0\}$.

We relax the assignment constraint (2.3) by

$$\sum_{j \in S_i} x_{ij} \leqslant 1 \quad \forall i \in O$$

The relaxed problem provides an upper bound for the original problem. For the relaxed problem, not every operation may be assigned but any operation cannot be assigned more than once. The contribution of the stands in $B_2 \cup \{n+1\}$ is null or negative. As operations can be unassigned in the relaxed problem, this implies the following property:

Property 3. *The relaxed problem can be solved by considering the stands in B_1 only.*

This property reduces the size of the relaxed problem. We define the stand decomposition method in two phases:

- Phase I: solve the relaxed problem by considering stands in B_1 only,
- Phase II: fix the assignments defined in phase I and solve the SAP for the remaining operations and the stands in $B_2 \cup \{n+1\}$.

The upper bound provided in Phase I can be used to guarantee the solution a posteriori. The following property presents sufficient conditions under which the solution provided by the stand decomposition method is optimal for the original problem.

Property 4. *Conditions of optimality for the stand decomposition method.*
In Phase II, if each operation is assigned to a stand in B_2 without towing, the solution provided by the stand decomposition method is optimal for the original problem.

Proof. Under these conditions, the Phase II solution has the value 0 since the coefficient c_{ij} are all null for stands in B_2 and no towing operation is performed. Therefore, the global solution value is equal to the upper bound provided in Phase I. □

Property 4 can be used for solving Phase II in a more efficient way. Indeed, a Phase II solution with a 0 value is optimal (for Phase II). Consequently, if a heuristic algorithm provides such a solution, it is not necessary to solve a second MIP. In practice, we first apply the greedy algorithm presented in Section 2.6.3 and then solve the MIP only if the greedy algorithm fails to find a 0 value solution.

2.6.2 Time decomposition

The time decomposition consists in splitting the day into smaller intervals and iteratively solving the MIP for each sub-problem from the beginning of the day to the end of the day. Assignments decided in a previous iteration are not questioned in the current one except if the operation is still in progress. To reduce the total computation time, we split the day such that each sub-problem has almost the same size.

2.6.3 Greedy algorithm

The process of one of our partner airports is performed manually and is close to the following greedy algorithm.

1. Sort operations by increasing number of compatible stands.
2. Iteratively assign each operation to the compatible and available stand that maximizes the objective function. In case of multiplicity, choose the stands in lexicographic order.

The complexity of such an algorithm is in $O(m \log m + nm)$.

Once each operation has been assigned, the airport scheduler improves the solution by performing local changes. This process is similar to a descent algorithm using two types of moves : *simple* move (switch the assignment of an operation to another compatible and available stand) and *swap* move (swap the assignment of two operations). Only moves improving the objective are performed.

Such an algorithm ends very quickly in practice but it tends to fall into a local optimum that cannot be overcome as only improving moves are considered.

2.6.4 Ejection chain algorithm

An ejection chain algorithm is a local search meta-heuristic where neighborhoods are defined not only by one move but by a sequence, or chain, of locally optimal moves. Performing more moves with each iteration is supposed to contribute to escaping the local optimum. Dorndorf et al. [2008] applied an ejection chain algorithm to the stand allocation problem. We refer the reader to their paper for further details about their algorithm. In the next section, we compare our approaches to this algorithm, which has been replicated exactly.

2.7 Computational experiments

In this section, we compare the performance of the algorithms on realistic instances generated from the actual data of two major European airports. For the sake of privacy, these airports will be noted I and J.

2.7.1 Instances and tests environment

Computer The results of mixed integer programs presented in this section were obtained using a Cplex 12.4 solver with default parameter tuning on a personal computer (Intel Core i5-2400 3.10Ghz, 4Go RAM) operating with Ubuntu 12.04 LTS operating system. Java Concert API was used to define the models.

Instances Each instance corresponds to an operational day. For the largest airport, we have a single instance I. For the other airport, we have a test set $J = \{J_1, \cdots, J_{83}\}$ of 83 consecutive days. Table 2.2 presents characteristics of the instances with respect to the number of operations, the number of stands and the number of stands compatible with each operation.

Inst.	Ops	Simultaneous ops (peak)	Contact stands	Remote stands	Compatible stands (average)
I	703	92	43	122	131.2
Min J	397	37	60	49	34.8
Avg J	485	43	60	49	35.9
Max J	553	52	60	49	36.7

Table 2.2: Characteristics of the instances (Ops=Operations)

Operation-stand affinity Pricing policies and performances measurements are complex substantially different from one airport to another. However, the operation-stand affinities c_{ij} can capture many practical situations. We will consider three affinity functions that represent different practices.

– *Passenger affinity:* Maximize the number of passengers assigned to contact stands

$$c_{ij} = \begin{cases} \text{number of passengers for operation } i \text{ if stand } j \text{ is a contact stand} \\ 0 \quad \text{otherwise} \end{cases}$$

– *Operation affinity:* Maximize the number of operations assigned to contact stands

$$c_{ij} = \begin{cases} 2 & \text{if operation } i \text{ is a whole turnaround and stand } j \text{ is a contact stand} \\ 1 & \text{if operation } i \text{ is an arrival or a departure operation and stand } j \text{ is a contact stand} \\ 0 & \text{otherwise} \end{cases}$$

– *Bus affinity:* Minimize the number of buses, which is equivalent to maximize the number of avoided buses

$$c_{ij} = \begin{cases} \text{Number of necessary buses for operation } i \text{ if stand } j \text{ is a contact stand} \\ 0 \text{ otherwise} \end{cases}$$

The number of buses required is equal to the ceiling of the number of passengers involved in an operation divided by the capacity of a bus (80 in our numerical study). Note that we set affinity of a waiting operation at a contact stand to 0.

We use subscript $\_op$, $\_bus$ and $\_pax$ to indicate which affinity function is under consideration. For example, I_op corresponds to instance I with the operation affinity function.

Weighting of objectives Coefficient $c_{i,n+1}$ is set to -10^6 to make the assignment of all operations the first priority. Note that all instances allow a feasible solution without using the dummy stand.

Coefficient β is respectively set to 1 for the optimization of operations at contact stands, 2 for optimization of buses and 100 for optimization of passengers. In this case both parts of the objective functions have similar weights.

Buffer time We include buffer times of 10 minutes following the procedure presented in Section 2.1.

2.7.2 MIP 2.1 versus MIP 2.2

In this section, we evaluate the effect of strengthening constraints, towing reformulation and symmetry breaking, with respect to memory consumption, quality of the Linear Programming (LP) relaxation and computation times.

Table 2.3 shows that reformulating overlapping and shadow constraints substantially reduces the number of constraints. Note that the number of binary variables is the same since only continuous variables are added in MIP 2.2.

		Overlapping constraints		Shadow constraints	
Instance	Binary variables	MIP 2.1	MIP 2.2	MIP 2.1	MIP 2.2
I	93 k	4.4 M	36 k	1.3 M	45 k
Avg J	17 k	315 k	10 k	146 k	6 k
Min J	14 k	196 k	8 k	93 k	4 k
Max J	20 k	415 k	11 k	195 k	7 k

Table 2.3: Number of binary variables and constraints

Table 2.4 presents the effect of the MIP formulation on the integrality gap and computation time, for the three affinity functions. A time limit of one hour is set.

		Gap ($z^*_{LP}/z^*_{MIP} - 1$)			CPU time [s]			
Instances		MIP 2.1	MIP 2.2 without y_{ij}	MIP 2.2	MIP 2.1	MIP 2.2 without y_{ij}	MIP 2.2	MIP 2.2 + sym. break.
I_op		OOM	2.4%	0.0%	OOM	313.0	92.8	34.3
I_bus		OOM	3.5%	0.0%	OOM	1717.2	86.3	36.5
I_pax		OOM	2.8%	0.0%	OOM	512.2	74.3	28.2
J_op	Avg	3.2%	1.7%	0.0%	65.5	68.5	3.9	4.2
	Min	1.3%	0.8%	0.0%	2.7	1.6	1.0	1.3
	Max	5.8%	3.2%	0.1%	TL (0.0 %)	TL (0.0 %)	22.4	12.7
J_bus	Avg	4.8%	2.5%	0.0%	10.1	7.8	3.4	3.9
	Min	2.3%	1.2%	0.0%	3.1	1.7	1.0	1.6
	Max	6.7%	4.3%	0.1%	42.3	37.5	9.2	10.0
J_pax	Avg	4.1%	2.0%	0.0%	11.6	9.5	3.2	3.7
	Min	1.6%	0.9%	0.0%	3.0	1.6	1.0	1.2
	Max	5.8%	3.5%	0.1%	53.3	39.4	9.3	9.7

Table 2.4: Gap with the LP solution and computation time (OOM = Out of memory, TL (0.0 %) = Time limit of 1 hour reached, an optimal solution has been found but it cannot be proven because of remaining integrality gap)

We first discuss the results for the large instance (I) that cannot be solved with MIP 2.1 since the model definition phase exceeds the computer's memory. Reformulating overlapping and shadow constraints reduces memory consumption enough to be able to define the model. It also tightens the linear relaxation. Reformulating towing (i.e. replacing y_i by y_{ij}) further strengthens the linear relaxation and yields a zero integrality gap for most instances. MIP 2.2

without towing reformulation provides the optimal solution for all objectives within 5 to 30 minutes. Towing reformulation reduces the computation time to 1 minute and 30 seconds. The symmetry breaking method further reduces the computation time to approximately 30 seconds.

We now discuss the results for the medium-sized airport (J). MIP 2.1 does not exceed the available memory since the 83 instances are much smaller than I. The results with respect to the quality of the LP relaxation are similar to those of I. Furthermore, reformulating the constraints significantly improves the integrality gap, but there is little impact on computation times (probably because Cplex also uses an aggregation method based on cliques). While towing reformulation reduces computation times in a systematic and significant way, symmetry breaking has no effect on them.

These first numerical experiments show that the different reformulations strengthen the model and offer reasonable computational times for all instances and affinity functions under consideration. In what follows, only MIP 2.2 with symmetry breaking will be considered and will be simply referred to as exact MIP.

2.7.3 Comparison of algorithms

In this section, we compare the exact MIP method with the MIP decomposition methods (time and stand), the ejection chain algorithm and the greedy algorithm. For the time decomposition method, we split the day into three intervals for the large airport (I) and into two intervals for the medium-sized airport (J). We have tested other splits and found that these choices offer a good trade-off in terms of solution quality and computation times.

Table 2.5 presents the gap to optimality and the computation time for the three objective functions. The minimum, maximum and average values are presented for instance set J (83 instances).

On the one hand, Table 2.5 reveals that MIP based approaches provide significantly better solutions than the ejection chain and the greedy algorithms. The exact MIP always finds an optimal solution (and proves its optimality) in less than 40 seconds. The stand decomposition heuristic provides an optimal solution most of the time for both airports and the maximum gap is 0.3%. The time decomposition heuristic provides very good solutions with gaps of less than 0.7%. The greedy algorithm offers poor performance for all instances and affinity functions, with a gap of up to 27.3 % for the large airport (I). The ejection chain algorithm outperforms the greedy algorithm with a gap of up to 7.6 % and an average gap of 2.0 % to 4.0 % for the medium-sized airport (J).

Instances		Gap $(1 - z/z^*)$					CPU time [s]				
		MIP	SD	TD	EC	Greedy	MIP	SD	TD	EC	Greedy
I_op		0.0%	0.0%	0.7%	5.0%	18.1%	34.3	20.6	16.3	1.7	0.1
I_bus		0.0%	0.0%	0.4%	6.9%	26.4%	36.5	37.8	20.6	4.1	0.1
I_pax		0.0%	0.0%	0.3%	6.8%	27.3%	28.2	21.1	12.1	3.0	0.1
	Avg	0.0%	0.0%	0.1%	2.0%	6.4%	4.1	2.6	2.7	0.4	<0.1
J_op	Min	0.0%	0.0%	0.0%	0.7%	3.4%	1.2	0.6	1.1	0.2	<0.1
	Max	0.0%	0.3%	0.4%	3.6%	19.9%	13.7	15.1	6.0	0.9	0.2
	Avg	0.0%	0.0%	0.2%	4.0%	7.9%	3.8	2.6	2.6	0.4	<0.1
J_bus	Min	0.0%	0.0%	0.0%	1.3%	3.8%	1.2	0.6	1.2	0.1	<0.1
	Max	0.0%	0.0%	0.7%	7.6%	13.2%	8.8	10.7	5.8	0.8	0.2
	Avg	0.0%	0.0%	0.1%	3.2%	6.5%	3.7	2.6	2.6	0.4	< 0.1
J_pax	Min	0.0%	0.0%	0.0%	1.3%	3.1%	1.2	0.6	1.2	0.1	<0.1
	Max	0.0%	0.0%	0.5%	5.9%	10.9%	9.8	9.4	4.5	0.8	0.2

Table 2.5: Comparison of the different methods (MIP=MIP2+ symmetry breaking, SD=Stand Decomposition, TD = Time Decomposition, EC= Ejection Chain)

On the other hand, Table 2.5 shows that the greedy algorithm and the ejection chain are faster than the MIP based approaches. Nevertheless, the MIP based approaches offer reasonable computation times for industrial applications. They solve all instances of the medium-sized airport (J) in less than 15 seconds and in less than 40 seconds for the large airport (I). Regarding instances I and J, the stand decomposition method generally outperform exact MIP with respect to computation time, but its effect is sometimes more mixed. Time decomposition is the fastest MIP method with computation times approximately halved with respect to the exact MIP method. The differences between the exact MIP and the decomposition methods will be more significant when considering instances with more operations (see Section 2.7.4).

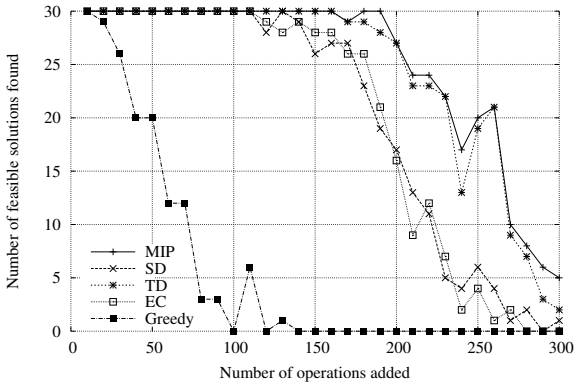
Our experiments lead us to conclude that MIP based approaches are suitable for solving the stand allocation problem for the set of instances considered. Indeed, they offer optimal or near-optimal solutions while ensuring reasonable computation times. The time decomposition method in particular offers the best trade-off between solution quality and computation time.

2.7.4 Feasibility

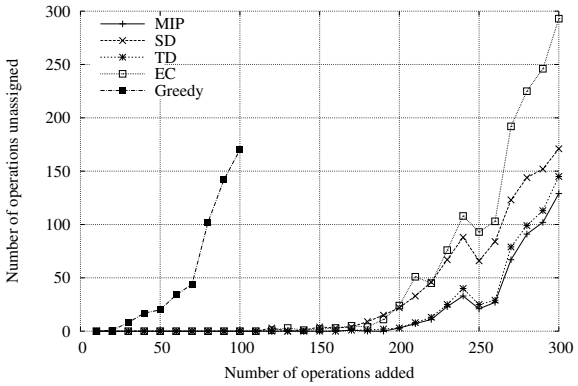
In Section 2.4, we show that deciding whether there is a feasible solution without the dummy stand is NP-complete. In this next section, we illustrate how important this result is from a

practical point of view and compare the ability of each method to find a feasible solution when there is one. Obviously, any algorithm finding a feasible solution leads to the conclusion that an instance is feasible. However only exact methods, such as our MIP formulation, are able to guarantee that there is no feasible solution.

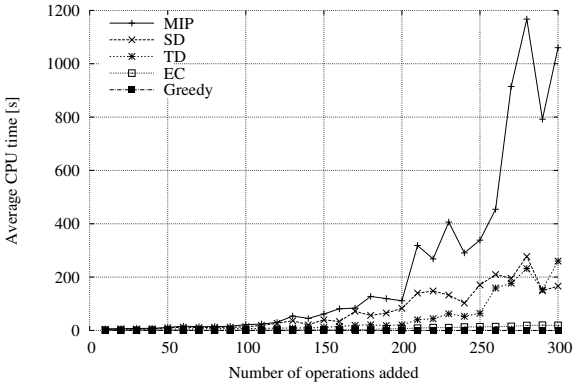
All the instances considered so far admit feasible solutions. In order to test the ability of each algorithm to find a feasible solution, we add a given number s of operations chosen randomly from the 82 other instances in J to the largest instance of J (553 operations). When an operation is added, we also add all the operations involved in the same turnaround while compatibility and objective coefficients are not changed. For each $s = 10, 20, \cdots, 300$, we simulate 30 instances. We then run the 5 algorithms for each of the 900 instances with the passenger affinity function. Note that the optimization is allowed to run to the end, i.e. it is not stopped when a feasible solution is found. Figure 2.6 presents the number of instances for which a feasible solution is found, the total number of unassigned operations and the CPU time for each algorithm.



(a) Number of instances solved



(b) Number of instances solved



(c) Computation time

Figure 2.6: Effect of the number of operations on finding a feasible solution

In Figure 2.6(a), we observe that the greedy algorithm begins to fail to find feasible solutions with only 20 additional operations. The ejection chain algorithm and the stand decomposition method handle all instances with up to 110 operations added while the exact MIP and the time decomposition method can go up to 160 operations. Figure 2.6(b) shows that the number of unassigned operations with the time decomposition method is very close to the exact MIP method. On the contrary, the ejection chains fails for approximately twice as many operations. The number of unassigned operations grows very quickly for the greedy algorithm, which is why it has not been plotted.

In Figure 2.6(c), we observe that the MIP computation times grow exponentially with the number of operations but remain reasonable up to the addition of 250 operations. The time and stand decomposition methods suffer less from this phenomenon since the MIPs solved are smaller.

To conclude, the exact MIP or time decomposition methods are preferable for handling the most congested instances. Once again, the time decomposition algorithm offers the best trade-off in terms of computation time.

2.7.5 Passengers at contact stand versus number of towing operations

Maximizing operation-stand affinity contradicts the idea of minimizing the number of towing operations. A trade-off can be found by tuning coefficient β. However, choosing the values may prove to be a challenging task. This is why we propose to use Pareto curves to support the decision maker. Pareto curves translate the choice of abstract coefficients in terms of business measures. In this section, we consider the passenger affinity function since it provides smoother curves.

Figure 2.7 plots the Pareto curve linking the number of towing operations to the percentage of passengers assigned to contact stands. This curve was obtained by solving Instance I for 100 values of β, from 0.1 to 10 with a step of 0.1. Each point is Pareto optimal, i.e. not dominated by any other solution. A solution is said to dominate another one if it is better, or at least equal, for all objectives simultaneously.

Plotting such a curve might be time consuming as the problem has to be solved several times. However, air traffic is mainly repetitive in the sense that it does not significantly change from one week to the next. Therefore the coefficient values do not need to be discussed everyday and can be previously set with the help of Pareto curves for reference operational days.

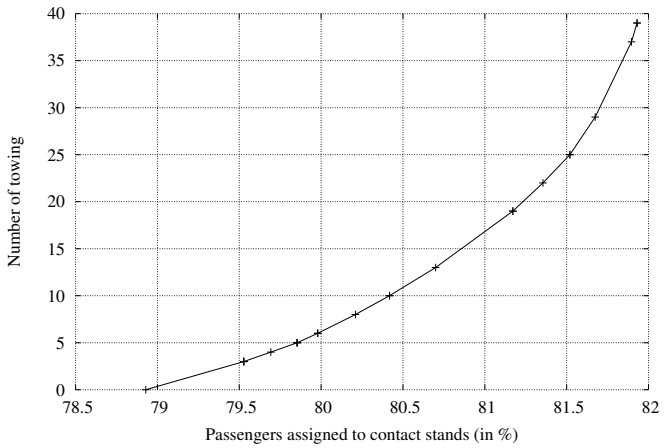


Figure 2.7: Passengers at contact stand versus number of towing operations for I

The extreme left-hand point indicates the minimal number of towing operations. As it is Pareto optimal, it also provides the best possible percentage of passengers assigned to contact stands with this given number of towing operations. On the other hand, the extreme right-hand point indicates the maximal percentage of passengers that can be assigned to contact stands and the associated minimum number of towing operations.

We observe that the first three towing operations enable a 0.6% gain in passengers at the contact stand whereas fourteen additional towing operations are needed to gain the last 0.4% of passengers at the contact stand.

2.8 Conclusion and future prospects

In this chapter, we prove that finding a feasible solution for the stand allocation problem is NP-complete. As a corollary, this proves the NP-hardness of the optimization problem. We then propose a strong MIP formulation and two heuristic algorithms. Our heuristic algorithms are based on the decomposition of the problem (spatial and temporal) where the sub-problems are solved using the MIP formulation. Based on instances from two European airports, we compare our approaches with the ejection chain method proposed by Dorndorf et al. [2008] and a greedy algorithm representing the current practice of a partner airport.

Computational experiments show that our MIP based approaches provide significantly better solutions than the other methods tested but need more computation time. Nevertheless, the computation times are short enough for an industrial application.

The instances considered in this chapter come from large European airports. However, this

does not necessarily mean that the proposed methods can be applied to the biggest airports in the world. Considering the reported computation times, the methods should be able to deal with bigger instances, and when this is not the case, our decomposition can be hybridized to handle the biggest instances: time decomposition can be applied to both phases of the stand decomposition heuristic. The stand decomposition can also be generalized in a terminal decomposition, as many flights are already pre-allocated to terminals in most airports.

Future research might focus on the aggregation of shadow constraints through clique constraints. The cliques used in this chapter were generated heuristically but a theoretical study might lead to better ones being found and consequently stronger constraints.

Future research might also take robustness into more detailed consideration. Buffer times might be managed as an objective and not as a constraint, as Dorndorf et al. [2008] propose. Stochastic optimization and simulation can also be considered as proposed by Lim and Wang [2005] and Yan and Tang [2007].

It would also be interesting to study alternative objectives such as minimizing risky connections or the total walking distance of passengers. Finally, another topic to be researched further is the integration of airports' decision making problems, in particular the integration of the stand allocation problem together with other key airport resources such as runways (i.e. a sequencing problem) and tarmac space (i.e. a routing problem) as Kim et al. [2013] propose.

Appendix

2.A Significant tow times

In this appendix, we explain how to extend our MIP formulation to the case where tow times are significant and cannot be reasonably included in operation processing times. Let $\tau_{jj'}$ be the tow time from stand j to stand j'. For the sake of simplicity, we assume that the tow time does not depend on the aircraft type. This assumption can be easily relaxed.

Consider an operation i that has a predecessor $U^{-1}(i)$, which implies that i is the successor of $U^{-1}(i)$. When the tow time is neglected, the start time of operation i is equal to the end time of its predecessor operation : $a_i = d_{U^{-1}(i)}$. With positive tow times, the starting time of operation i is equal to $a_i^{jj'} = d_i + \tau_{jj'}$ if operation $U^{-1}(i)$ is assigned to stand j and operation i is assigned to stand j'.

Based on this, consider two operations i and i' such that $d_i \leqslant d_{i'}$. We distinguish the case

where i' does not have a predecessor ($U^{-1}(i') = 0$) and the case where i' has a predecessor ($U^{-1}(i') \neq 0$).

Case 1 : $U^{-1}(i') = 0$

In this case, operation i' does not succeed any other operation. Operations i and i' overlap if and only if $a_{i'} < d_i$. Hence the overlapping constraint is the same as (5.3) in MIP 1 :

$$\begin{aligned} x_{ij} + x_{i'j} \leqslant 1 \quad & \forall i, i' \in O, U^{-1}(i') = 0, \\ & \forall j \in S_i \cap S_{i'}, \\ & a_{i'} < d_i \leqslant d_{i'} \end{aligned}$$

Case 2 : $U^{-1}(i') \neq 0$

In this case, operation i' succeeds operation $U^{-1}(i')$. Operations i and i' cannot be assigned to stand j if $U^{-1}(i')$ is assigned to stand j' and $a_{i'}^{j'j} < d_i$, which leads to following overlapping constraints :

$$\begin{aligned} x_{ij} + x_{i'j} + x_{U^{-1}(i')j'} \leqslant 2 \quad & \forall i, i' \in O, U^{-1}(i') \neq 0, \\ & \forall j \in S_i \cap S_{i'}, \forall j' \in S_{U^{-1}(i')}, \\ & a_{i'}^{j'j} < d_i \leqslant d_{i'} \end{aligned}$$

Chapter 3

The aircraft ground routing problem : Analysis of industry punctuality indicators in a sustainable perspective

Abstract

The ground routing problem consists in scheduling the movements of aircraft on the ground between runways and parking positions while respecting operational and safety requirements in the most efficient way. We present a Mixed Integer Programming (MIP) formulation for routing aircraft along a predetermined path. This formulation is generalized to allow several possible paths. Our model takes into account the classical performance indicators of the literature (the average taxi and completion times) but also the main punctuality indicators of the air traffic industry (the average delay and the on time performance). Then we investigate their relationship through experiments based on real data from Copenhagen Airport (CPH). We show that the industry punctuality indicators are in contradiction with the objective of reducing taxi times and therefore pollution emissions. We propose new indicators that are more sustainable, but also more relevant for stakeholders. We also show that alternate paths cannot improve the performance indicators.

Keywords: Ground routing, mixed integer programming, sustainable development

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*, 248(3), 827-839, 2016.

3.1 Introduction

Over the last years, the European air traffic kept growing and Eurocontrol [2012c] predicts an annual increase of the number of flights of 3% between 2014 and 2018. The traffic is expected to double between 2010 and 2030 (Eurocontrol [2010]). Due to this ceaseless increase of the number of flights in Europe, airports are becoming an important bottleneck of air traffic. Hence, using decision support systems and optimization tools is more and more critical.

The aircraft ground movements play an important role in the airport emissions. London Heathrow [2008-09] airport estimates that 54% of the airport NO_x emissions are produced by aircraft on the ground. The ground routing is also key component of the airports carbon footprint. Eurocontrol [2009] estimates that 475,000 tonnes of CO_2 emissions could be earned if only one minute of taxi time per flight could be earned in 50 major European airports. It also represents a non negligible part of airlines fuel cost. Ravizza et al. [2014] demonstrate that a better routing optimization allows to earn $9.6 millions of fuel a year in Zurich airport.

The Ground Routing Problem (GRP) consists in scheduling the movements of aircraft between airport facilities without conflicts and in the most effective way. An arriving aircraft has to be routed from its landing runway to its stand or hangar. A departing aircraft has to be routed from its current parking position to its departure runway. The ground movements occur on a network of roads called taxiways which link airport facilities (see Figure 3.1). In practice this problem is issued by Air Traffic Controllers (ATCs) on an operational window of typically 10 to 40 minutes.

The main constraints of the problem are related to the safety of aircraft: as in airspace, aircraft have to be separated from each other to avoid collisions. Several other routing constraints must also be taken into account such as taxi speeds and acceleration for passengers comfort, turning angle and aircraft / taxiway segment compatibility due to weight or width.

The quality of a routing schedule is defined by several Key Performance Indicators (KPIs). In this chapter, we focus on four of them : the average taxi time, the average completion time, the average delay and the On Time Performance (OTP).

The taxi time measures the time an aircraft spends on the ground with engines on, between push back (i.e. leaving the parking position) and take-off for departures and between landing and park-in for arrivals. It includes any waiting time (e.g. runway queuing time) and not just the time spent moving, as engines cannot be turned off once started up. Pollution emission is

Figure 3.1: Taxiway network in Copenhagen Airport(CPH)

directly related to the fuel consumption. The aircraft fuel consumption is not accurately known for the taxi process nowadays, but various statistical studies conclude that it mainly depends on the taxi time (see e.g. Khadilkar and Balakrishnan [2011], Nikoleris et al. [2011] and Ravizza et al. [2012]). Other influencing factors have been identified, such as the number of stops, turns and accelerations but their effects are less clear and of minor importance in comparison to the taxi time.

We are also interested in minimizing the completion times, i.e. the take-off times for departing flights and the park-in times for arriving flights. In peak hours, the runway is often the main bottleneck of the airport (Idris et al. [1998]). Minimizing take-off times reduces the risk of runway starving and ensures a good use of its capacity. Minimizing park-in times reduces the starting time of passengers debarkation, which increases the quality of service.

In the air traffic industry, the main indicators of punctuality are the average delay per flight and the OTP (see Eurocontrol [2012b] and Eurocontrol [2013a]). The delay is measured with respect to the scheduled times of park-in and push back, which are both printed on passenger tickets. For example, an arriving flight is 5 minutes late if the aircraft arrives at the stand 5 minutes after the scheduled time and a departing flight is 5 minutes late if the aircraft is pushed back from its stand 5 minutes after the scheduled time. The OTP L is the percentage of flights having a delay less than L minutes. The most common value of L in the industry is 15 minutes and OTP 15 is simply called OTP.

Figure 3.2: Runway queue at London Heathrow (LHR) holding point

A common practice in airports is to push back aircraft as soon as possible and to taxi to the runway (see Atkin et al. [2011a] for London Heathrow airport (LHR) and Smeltink et al. [2004] for Amsterdam Schiphol airport (AMS)). It reduces the risk of runway starving and is beneficial for the departure delay and OTP. However, especially during peak hours, the runway capacity is often exceeded and a push back as soon as possible policy results in a take-off queue (see Figure 3.2) in which aircraft engines remain turned on. Pollution emissions can be reduced by transferring the runway queuing time (with engines on) to the stand (with engines off). This process is called *stand holding*. Nevertheless, if an aircraft is held too long, it may not reach the runway in time for take-off and some runway capacity can be wasted. It may also prevent an arriving aircraft from using the stand (stand blockage).

Accurate estimations of aircraft ready times and taxi times are required to schedule push back adequately, i.e. holding stand as much as possible in order to reduce taxi times, but without wasting the runway capacity. Accurate estimations of ready times are not always available in airports: ATCs are often informed of an aircraft ready time only when the pilot calls the control tower for push back and start up approval. That is why Eurocontrol designed the Airport Collaborative Decision Making (A-CDM) project. The main goal of A-CDM project is precisely to improve predictability and information sharing between all stakeholders. In a A-CDM airport, airlines and ground handlers are required to communicate and update an accurate ready time (typically 30 to 40 minutes in advance). The ready time is called the Target Off-Block Time (TOBT) and corresponds to an estimation of the time at which the aircraft will be ready to

push back (all doors closed, boarding bridge removed, etc). In the air traffic industry, the push back scheduling is called often the pre-departure sequencing.

In the literature, it has been shown that stand holding can reduce taxi times significantly without impacting the runway capacity (see Section 3.2). Nevertheless, the impact of stand holding on the KPIs of the industry (OTP and delay) has not been investigated. In this chapter, we propose a model including the OTP and delay indicators. Our model allows several possible paths in the taxiways. We then address the following questions through a numerical study based on realistic instances from Copenhagen Airport (CPH). How do the performance indicators compete ? Are the key indicators of the industry consistent with a sustainable development ? Can we propose better indicators ? Can we reduce taxi times by considering alternate paths ? What is the bottleneck in ground routing operations ? This work is the result of a collaboration with Amadeus company.

The remainder of this chapter is organized as follows. A review of the literature and a summary of our contributions are presented in Section 3.2. In Section 3.3, we propose a model for the GRP and formulate it as a Mixed Integer Program (MIP). We provide details on the data set and instances from CPH in Section 3.4. Then the results of our numerical study are given in Section 3.5. Finally, a conclusion and discussion of our results are presented in Section 3.6.

3.2 Literature review

In this section, we present a literature review of the GRP and related works. More details can be found in a survey by Atkin et al. [2010b]. We also present our contributions.

3.2.1 The ground routing problem

The main differences between the GRP models are the routing options, the link with the take-off schedule and whether the time is discretized or not.

Three different routing options exists: single path, alternate paths and path free. In the single path approach, each aircraft can be routed along one and only one predetermined route. In the alternate paths approach, each aircraft can be routed along a route chosen between a set of predetermined routes. In the path free approach, any route can potentially be assigned to an aircraft. In most of papers, a take-off schedule is given as an input and the objective is

to minimize its deviation. The NMOC slot compliance [1] is also often considered with a high penalty for missed slots and linear cost for deviation inside the slot.

Single path The single path was first studied by Smeltink et al. [2004] who propose a mixed integer program formulation. The (average) taxi time is minimized and the respect of take-off schedule is forced by constraints. Rathinam et al. [2008] simplify this formulation and improve safety by adding separation constraints.

Alternate path Gotteland and Durand [2003] address the alternate path approach with a genetic algorithm whose solutions are evaluated with a Branch & Bound algorithm. Aircraft are allowed to stop once while taxiing and stand holding is not possible. Their objective is to minimize the routing time. Gotteland et al. [2003] adapt this approach to include take-off predictions and NMOC slots. They linearly penalize deviations from take-off predictions and add a large penalty for missed slots. Balakrishnan and Jung [2007] propose a discrete time integer program. Their model minimizes the taxi time and the delay to the take-off schedule. A large penalty is added for each aircraft reaching the runway after its target time. Deau et al. [2008] and Deau et al. [2009] propose a two stages method for optimizing runway scheduling and ground routing in order to minimize the delay. Firstly, the runway scheduling problem is solved, which provides Target Take-Off Times (TTOTs) for departing aircraft. They are used as an input for the ground routing which is solved with the approach of Gotteland et al. [2003] in the second phase.

Path free Marin [2006] models the path free approach with an integer program formulation of the capacitated multi-commodity flow problem in a space-time network (time is discretized). Aircraft are assumed to taxi with a constant speed. The weighted routing time (completion time) is minimized. The model is extended by Marin and Codina [2008] to include other objectives such as the number of controller interventions, the worst taxi time, the delay and the airport throughput. Keith and Richards [2008] propose an integer program for the path free

1. When an air sector of an aircraft flight plan is congested or when the destination airport is facing adverse conditions, the Network Manager Operations Center (NMOC, previously called the Central Flow Management Unit, CFMU) assigns a Calculated Take-Off Time (CTOT). It generally results in delaying the take-off to prevent the situation from NMOC worse in the perturbed sector. The take-off is allowed within the interval $[CTOT - 5\ min; CTOT + 10\ min]$, called a NMOC slot. Otherwise, the aircraft has to wait for another slot from the NMOC.

approach optimizing both the runway scheduling and the ground routing. They minimize a weighted combination of the makespan, the average taxi time and the average distance. The model is slightly adapted by Clare and Richards [2011] to improve computational efficiency. Nevertheless, computation times are still too important: their model needs more than a minute to handle instances with 8 aircraft on a small network with only the runway holding point. Lesire [2010] presents an iterative algorithm for the path free approach. Their method is based on an A^* algorithm and aims to minimize the completion time. Liu et al. [2011] present a hybrid genetic algorithm and simulated annealing algorithm based on the mutli-commodity flow model of Marin [2006]. The idea of their method is to replace usual selection criteria by a simulated annealing temperature principle. Atkin et al. [2011b] and Ravizza et al. [2014] propose an iterative approach. Aircraft are routed by a Quickest Path Problem with Time Window (QPPTW) algorithm. Take-off predictions are taken into account by using a backward version of QPPTW algorithm. The method is tested on Zurich airport and show a potential taxi time reduction of up to 23.6% for Atkin et al. [2011b] and 30.3% Ravizza et al. [2014]. The method offers computation times of less than 50ms by aircraft. Ravizza et al. [2014] propose a swap based method for changing the assignment order, which allows to slightly improve the results while still offering very short computation times.

3.2.2 The push back scheduling problem

The literature on the ground routing problem reveals that most of inefficiencies of the taxi process come from runway congestion and can be reduced by stand holding (see e.g. Balakrishnan and Jung [2007] or Ravizza et al. [2014]), i.e. by scheduling push back time latter. Based on this result, some papers focus on the scheduling of push back times and do not consider a detailed routing.

Deterministic models Malik et al. [2010], Jung et al. [2010, 2011] and Atkin et al. [2012] use a similar decomposition to Deau et al. [2009]: they first optimize the take-off sequence and then allocate push back times to meet this take-off sequence, while trying to absorb as much delay as possible at the stand.

In fact, Malik et al. [2010], Jung et al. [2010, 2011] spot release times are issued and not push back times, which requires a further collaboration with ramp area to transfer the delay

to the gate[2]. Gupta et al. [2012] present the concepts of such a collaboration. The main drawback of this practice is that the complexity of the ramp area structure is hidden and some more appropriate spot release sequences may be missed. On the contrary, Atkin et al. [2012] explicitly consider ramp area contention. A feasible push back time is assigned to aircraft while computing the take-off sequence. A sequence is rejected if such push back times can not be found. Push back times are then re-optimize to improve stand holding.

Note that the problem of considering only push back times is that it requires an accurate taxi time estimation in order not to waste the runway capacity and not to excessively wait at the runway. The ground routing problem can be used for solving the push back scheduling problem, which may lessen the impact of the taxi time prediction.

Queuing models An airport can be seen as a network of queues and the push back scheduling problem has been first modeled as a queuing problem by Pujet et al. [1999]. The whole taxiway network is modeled as a server of infinite capacity with stochastic processing times. They propose a simple threshold policy that regulates the number of aircraft taxiing or waiting at the runway. Carr et al. [2002] adapt this model to take into account departure fix closures. Burgain et al. [2012] refine the modeling of the taxiway network and propose a sequential queues network. It allows to take advantage of recent surface surveillance technology. A comparison to the threshold policy of Pujet et al. [1999] highlights that this technology can be beneficial when the runway is operated at intermediate capacity. Finally, Simaiakis et al. [2014] propose a push back rate control policy, which advises push back frequency for the next 15 minutes. The method was tested in real situation in Boston Logan airport (BOS) over 16 demo period and 8 periods with significant gate-holds were kept for the analysis. They estimate an average earning of one minute and a half in taxi-out times.

3.2.3 Our contributions

We present a MIP formulation of the single path GRP that follows the continuous time models of Smeltink et al. [2004] and Rathinam et al. [2008]. We extend their approach to include alternate paths. This requires to formulate separation constraints temporally as in Clare and

2. In most US airports, particularly in Dallas Fort Worth International airport (DFW) which is considered in these studies, the push back is not managed by the ground controller. Indeed, the ramp area, i.e. the areas around the gates, is managed by flight operators (airlines) or a ramp controller (airport authorities). Their responsibility is to push back aircraft and bring them to predetermined spots (exit point of the ramp area and entry points of the taxiway area), where responsibility is handed over to the ground ATC.

Richards [2011].

Our model differs in several other aspects. First, we include the punctuality indicators of the industry (OTP and delay) in the objective function, in addition to the taxi time and completion time indicators. A second difference is the link with the runway. The input of our model is the take-off sequence while the input of most of the models is the take-off schedule with targeted take-off times. We made this choice because manipulating sequences is more convenient than schedules for ATCs. In that way, our model can be used as a tool for supporting runway sequencing decisions: it provides optimal take-off times from a sequence, while accurately taking into account routing considerations. Finally we add stand blockage constraints.

To the best of our knowledge, the pertinence of the OTP and delay indicators have not been questioned in the context of the GRP. In a numerical study, we analyze the impact of including these KPIs in the optimization. We show that they are in contradiction with the objective to reduce taxi times and pollution emissions in airports. We propose new indicators that are more sustainable, but also more relevant for stakeholders.

Our experiments show that intermediate taxiways, between ramp areas and runways, are not a bottleneck in CPH. It explains why the alternate paths approach does not succeed in improving the KPIs significantly.

3.3 Ground routing problem formulation

In this section, we introduce the main notations and formulate the GRP as a MIP. We first present the model with a single route for each flight. Then the model is generalized to consider a set of possible paths for each flight (alternate paths approach).

For both models, the main inputs are the runway allocation, the take-off sequence, the landing schedule and the stand allocation plan (including the sequence of aircraft for every stand).

3.3.1 Single path model

Taxiway network The taxiway network is modeled as a graph $G = (V, E)$ with V the set nodes and E the set of edges. There is a node for each taxiway intersection and additional nodes for stands. An edge represents an elementary segment of taxiway.

The set of arriving and departing flights is $\mathcal{F} = \mathcal{F}_{arr} \cup \mathcal{F}_{dep}$. For a flight i, the single path from its origin o_i to its destination d_i is $P_i = (o_i, u_2, .., u_{|P_i|-1}, d_i)$. Let $V_i \subset V$ and $E_i \subset E$ the set of nodes and edges that flight i can use. Note that the origin o_i and the destination d_i are

fixed by the runway and stand allocation.

Flight characteristics A flight i is ready to leave its origin o_i at time T_{o_i}. For an arriving flight $i \in \mathcal{F}_{arr}$, it corresponds to the Target LanDing Time ($TLDT$) estimated by ATCs. For a departing flight $i \in \mathcal{F}_{dep}$, it corresponds to the Target Off-Block Time ($TOBT$) estimated by the airline and the ground handlers.

The scheduled time for flight i is SB_i, this time is used to measure the delay and the OTP. For an arriving flight, it is the Scheduled In-Block Time ($SIBT$), i.e. the time the aircraft is supposed to arrive at its stand. For a departing flight, it corresponds to the Scheduled Off-Block Time ($SOBT$), i.e. the time the aircraft is supposed to push back from its stand.

A flight i can spend a minimum (maximum) time T_{iuv}^{min} (T_{iuv}^{max}) on edge $uv \in E_i$. These times can be directly computed from the minimum and maximum speeds allowed on edge uv for aircraft i and from the edge length (see Section 3.4).

The take-off sequence is an input of our model. The position of departure $i \in \mathcal{F}_{dep}$ in the take-off sequence is $\Gamma(i)$.

Interactions between flights Flights i and j must have a minimum separation time at each node $u \in V_i \cap V_j$: if flight i arrives first at node u at time t, then j cannot cross node u before $t + S_{iju}$.

Let $\mathcal{G} \subset \mathcal{F}_{dep} \times \mathcal{F}_{arr}$ the set of possible stand blockages. A pair of flights (i, j) belongs to $\mathcal{G}$ if departure i and arrival j are assigned to the same stand and i is scheduled before j (in the stand allocation plan). In this case, departure i must leave the stand before arrival j parks in.

Decision variables In the single path approach, the main decisions are the time that aircraft reach each node of their path. Our formulation uses the following variables:

- t_{iu} : the time when flight i reaches node $u \in V_i$. The origin time t_{io_i} corresponds to the landing time for an arrival and to the the push back time for a departure. The destination time t_{id_i} corresponds to the take-off time for a departure and to the park-in time for an arrival.
- δ_i : the delay of flight i to its scheduled reference time SB_i. The delay is $\max(0, t_{id_i} - SB_i)$ for an arrival and $\max(0, t_{io_i} - SB_i)$ for a departure.
- $\beta_i = 1$ if flight i is delayed by more than $L \geqslant 0$ with respect to the scheduled reference time SB_i (if $\delta_i > L$), 0 otherwise.
- $z_{iju} = 1$ if flight i arrives before flight j in node $u \in V_i \cap V_j$, 0 otherwise.

Objective function We are interested in minimizing the following performance indicators.

- $\sum_{i \in \mathcal{F}} \beta_i$: Number of flights delayed by more than L
- $\sum_{i \in \mathcal{F}} \delta_i$: Total delay
- $\sum_{i \in \mathcal{F}} (t_{id_i} - t_{io_i})$: Total taxi time
- $\sum_{i \in \mathcal{F}} (t_{id_i} - T_{o_i})$: Total completion time

Note that minimizing the number of flights delayed by more than L is equivalent to maximizing the OTP L (percentage of flights with a delay less than L).

Our objective function is a linear combination of these four indicators using non negative coefficients c^{OTP}, c^{delay}, c^{taxi} and c^{ct}. It can be extended to include the NMOC slot compliance objective (see Appendix 3.A).

MIP formulation The single path problem can be formulated by MIP 3.1.

Constraints (3.2) ensure that departures cannot push back before they are ready to and Constraints (3.3) ensure that arrivals start taxiing as soon as they land, in order to free the runway. Constraints (3.4) ensure that an arrival does not park-in until the previous departure has left (stand blockage constraints). Constraints (3.5) ensure the respect of minimum and maximum time spent on every edge (speed constraints). The maximum time spent on an edge allows to prevent aircraft from stopping in certain taxiway segments (e.g. runway crossing). It also ensures that the capacity of the edge is not exceeded (i.e. no more aircraft that its length allows it). Note that in all the solutions in our experiments, aircraft never taxi at the minimal speed. Constraints (3.6) ensure the definition of sequencing variables z_{iju}, i.e. either flight i arrives before flight j in node $u \in V_i \cap V_j$ or the opposite. Constraints (3.7) ensure that the take-off sequence is respected. Constraints (3.8) and (3.9) prevent the three kinds of conflict illustrated in Figure 3.3. Constraints (3.8) prevent aircraft from bumping into each other at every node (see Figure 3.3(a)), where M is supposed to be a high enough value (e.g. 10 times the time window is largely sufficient, it remains in forcing every aircraft to end taxiing in less than 10 times the time window which is reasonable). Constraints (3.9) prevent two aircraft from using an edge in opposite directions simultaneously (see Figure 3.3(b)). Constraints (3.9) also prevent an aircraft from overtaking another one on an edge, which is physically impossible (see Figure 3.3(c)). Constraints (3.10) to (3.12) ensure the definition of delay variables δ_i and OTP variables β_i. Finally, constraints (3.13) to (3.16) specify the domains of the variables.

$$\min\ c^{OTP}\sum_{i\in\mathcal{F}}\beta_i + c^{delay}\sum_{i\in\mathcal{F}}\delta_i + c^{taxi}\sum_{i\in\mathcal{F}}(t_{id_i}-t_{io_i}) + c^{ct}\sum_{i\in\mathcal{F}}(t_{id_i}-T_{o_i}) \tag{3.1}$$

$$t_{io_i} \geqslant T_{o_i} \qquad \forall i \in \mathcal{F}_{dep} \tag{3.2}$$

$$t_{io_i} = T_{o_i} \qquad \forall i \in \mathcal{F}_{arr} \tag{3.3}$$

$$t_{io_i} \leqslant t_{jd_j} \qquad \forall (i,j) \in \mathcal{G} \tag{3.4}$$

$$T^{min}_{iuv} \leqslant t_{iv} - t_{iu} \leqslant T^{max}_{iuv} \qquad \forall i \in \mathcal{F}, \forall uv \in E_i \tag{3.5}$$

$$z_{iju} + z_{jiu} = 1 \qquad \forall i,j \in \mathcal{F}, \forall u \in V_i \cap V_j \tag{3.6}$$

$$z_{iju} = 1 \qquad \forall i,j \in \mathcal{F}_{dep}, u = d_i = d_j, \Gamma(i) < \Gamma(j) \tag{3.7}$$

$$t_{ju} \geqslant t_{iu} + S_{iju} - M(1 - z_{iju}) \qquad \forall i,j \in \mathcal{F}, \forall u \in V_i \cap V_j \tag{3.8}$$

$$z_{iju} = z_{ijv} \qquad \forall i,j \in \mathcal{F}, \forall uv \in E_i \cap E_j \tag{3.9}$$

$$\delta_i \geqslant t_{io_i} - SB_i \qquad \forall i \in \mathcal{F}_{dep} \tag{3.10}$$

$$\delta_i \geqslant t_{id_i} - SB_i \qquad \forall i \in \mathcal{F}_{arr} \tag{3.11}$$

$$\delta_i \leqslant L + M\beta_i \qquad \forall i \in \mathcal{F}_{dep} \tag{3.12}$$

$$t_{iu} \geqslant 0 \qquad \forall i \in \mathcal{F}, \forall u \in V_i \tag{3.13}$$

$$\delta_i \geqslant 0 \qquad \forall i \in \mathcal{F} \tag{3.14}$$

$$\beta_i \in \{0,1\} \qquad \forall i \in \mathcal{F} \tag{3.15}$$

$$z_{iju} \in \{0,1\} \qquad \forall i \neq j \in \mathcal{F}, \forall u \in V_i \cap V_j \tag{3.16}$$

MIP 3.1: Single path approach

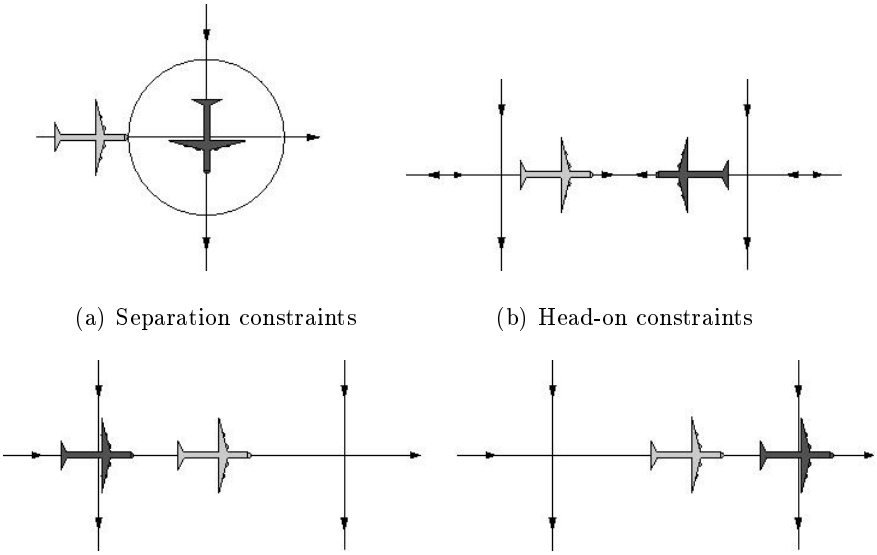
Note that it is possible to differentiate flights for all objectives through a slight change in the objective function (indexing the coefficients):

$$\sum_{i\in\mathcal{F}} c_i^{OTP}\beta_i + \sum_{i\in\mathcal{F}} c_i^{delay}\delta_i + \sum_{i\in\mathcal{F}} c_i^{taxi}(t_{id_i}-t_{io_i}) + \sum_{i\in\mathcal{F}} c_i^{ct}(t_{id_i}-T_{o_i})$$

It particularly makes sense for the taxi time as a big aircraft will consume more fuel (and thus pollute more) than a small one.

3.3.2 Alternate paths model

In this section, we generalize the previous model in order to allow multiple paths. For each flight i, the path can be chosen in a set of alternate paths $\mathcal{P}_i$. For each flight i and each path $p \in \mathcal{P}_i$, we denote V_i^p and E_i^p the set of nodes and edges used in p. The set of nodes that can



(a) Separation constraints (b) Head-on constraints

(c) Overtaking constraints

Figure 3.3: Safety constraints

be used by flight i is now $V_i = \cup_{p \in \mathcal{P}_i} V_i^p$. Similarly, the set of edges that can be used by flight i is $E_i = \cup_{p \in \mathcal{P}_i} E_i^p$. The aim of this model is to choose a path for every flight and to schedule the moves along this path. Consequently, we need to define the following path selection variables and scheduling variables:

- $x_i^p = 1$ if aircraft i uses path $p \in \mathcal{P}_i$, 0 otherwise.
- $t_{iu}^p \geqslant 0$ the time flight i reaches node $u \in V_i$ through path $p \in \mathcal{P}_i$ if flight i uses path p, 0 otherwise.

Variables β_i, δ_i, z_{iju}, t_{io_i} and t_{id_i} are defined as previously. The alternate paths model can be formulated by MIP 3.2.

Bounding constraints (3.2) and (3.3), stand blockage constraints (3.4), runway sequencing constraints (3.7) and variables definition constraints (3.10) to (3.12) and (3.14) to (3.16) are still valid in this formulation. Other constraints need to be adapted.

Note that $\sum_{\substack{p \in \mathcal{P}_i \\ u \in V_i^p}} x_i^p = 1$ if flight i uses node u, 0 otherwise. Similarly, $\sum_{\substack{p \in \mathcal{P}_i \\ uv \in E_i^p}} x_i^p = 1$ if flight i uses edge uv.

Constraints (3.17) ensure that one and only one path is chosen for every flight. Constraints (3.18) to (3.22) ensure the definition of scheduling variables. Speed constraints (3.5) must be replaced by constraints (3.23). Separation constraints (3.8) must be replace by constraints (3.24). Note that if flight i or j does not use the node $u \in V_i \cap V_j$ then $z_{iju} = 0$ and the constraints is disabled. Sequencing constraints (3.6) must be replaced by constraints (3.25) to (3.28). Note

$$\min \quad c^{OTP}\sum_{i\in\mathcal{F}}\beta_i + c^{delay}\sum_{i\in\mathcal{F}}\delta_i + c^{taxi}\sum_{i\in\mathcal{F}}(t_{id_i} - t_{io_i}) + c^{ct}\sum_{i\in\mathcal{F}}(t_{id_i} - T_{o_i})$$

$$s.t. \quad (3.2-3.4), (3.7), (3.10-3.12), (3.14-3.16)$$

$$\sum_{p\in\mathcal{P}_i} x_i^p = 1 \qquad \forall i \in \mathcal{F} \tag{3.17}$$

$$t_{io_i} = \sum_{p\in\mathcal{P}_i} t_{io_i}^p \qquad \forall i \in \mathcal{F} \tag{3.18}$$

$$t_{id_i} = \sum_{p\in\mathcal{P}_i} t_{id_i}^p \qquad \forall i \in \mathcal{F} \tag{3.19}$$

$$t_{io_i}^p \geqslant T_{o_i} x_i^p \qquad \forall i \in \mathcal{F}_{dep}, \forall i \in \mathcal{P}_i \tag{3.20}$$

$$t_{io_i}^p = T_{o_i} x_i^p \qquad \forall i \in \mathcal{F}_{arr}, \forall i \in \mathcal{P}_i \tag{3.21}$$

$$t_{iu}^p \leqslant M x_i^p \qquad \forall i \in \mathcal{F}, \forall p \in \mathcal{P}_i, \forall u \in V_i^p \tag{3.22}$$

$$T_{iuv}^{min} x_i^p \leqslant t_{iv}^p - t_{iu}^p \leqslant T_{iuv}^{max} x_i^p \qquad \forall i \in \mathcal{F}, \forall p \in \mathcal{P}_i, \forall uv \in E_i^p \tag{3.23}$$

$$\sum_{\substack{p\in\mathcal{P}_j \\ u\in V_i^p}} t_{ju}^p \geqslant \sum_{\substack{p\in\mathcal{P}_i \\ u\in V_i^p}} t_{iu}^p + S_{iju} - M(1 - z_{iju}) \qquad \forall i,j \in \mathcal{F}, \forall u \in V_i \cap V_j \tag{3.24}$$

$$z_{iju} + z_{jiu} \geqslant \sum_{\substack{p\in\mathcal{P}_i \\ u\in V_i^p}} x_i^p + \sum_{\substack{p\in\mathcal{P}_j \\ u\in V_i^p}} x_j^p - 1 \qquad \forall i,j \in \mathcal{F}, \forall u \in V_i \cap V_j \tag{3.25}$$

$$z_{iju} + z_{jiu} \leqslant 1 \qquad \forall i,j \in \mathcal{F}, \forall u \in V_i \cap V_j \tag{3.26}$$

$$z_{iju} \leqslant \sum_{\substack{p\in\mathcal{P}_i \\ u\in V_i^p}} x_i^p \qquad \forall i,j \in \mathcal{F}, u \in V_i \cap V_j \tag{3.27}$$

$$z_{iju} \leqslant \sum_{\substack{p\in\mathcal{P}_j \\ u\in V_i^p}} x_j^p \qquad \forall i,j \in \mathcal{F}, u \in V_i \cap V_j \tag{3.28}$$

$$z_{iju} - z_{ijv} \geqslant \sum_{\substack{p\in\mathcal{P}_i \\ uv\in E_i^p}} x_i^p + \sum_{\substack{p\in\mathcal{P}_j \\ uv\in E_i^p}} x_j^p - 2 \qquad \forall i,j \in \mathcal{F}, \forall uv \in E_i \cap E_j \tag{3.29}$$

$$z_{ijv} - z_{iju} \geqslant \sum_{\substack{p\in\mathcal{P}_i \\ uv\in E_i^p}} x_i^p + \sum_{\substack{p\in\mathcal{P}_j \\ uv\in E_i^p}} x_j^p - 2 \qquad \forall i,j \in \mathcal{F}, \forall uv \in E_i \cap E_j \tag{3.30}$$

$$t_{iu}^p \geqslant 0 \qquad \forall i \in \mathcal{F}, \forall p \in \mathcal{P}_i, \forall u \in V_i^p \tag{3.31}$$

$$t_{io_i}, t_{id_i} \geqslant 0 \qquad \forall i \in \mathcal{F} \tag{3.32}$$

$$x_i^p \in \{0,1\} \qquad \forall i \in \mathcal{F}, \forall p \in \mathcal{P}_i \tag{3.33}$$

MIP 3.2: Alternate paths approach

that constraints (3.25) and (3.26) are equivalent to constraints (3.6) if flight i and j uses node u. Otherwise these constraints have no effect. Also note that constraints (3.27) to (3.28) are not necessary as the optimization will naturally verify them to disable associated separation constraints if necessary. Overtake and head-on constraints (3.9) are equivalent to constraints (3.29) and (3.30) if flight i and j uses edge uv. Otherwise, the right hand side is less or equal to -1 or -2 and the constraints have no effect.

Including the option of choosing a path increases drastically the number of constraints and binary variables, especially if a high number of paths is possible. Clare and Richards [2011] had the same problem in their model. They designed an iterative approach, close to a constraints generation process, to tackle it. It consists is solving the MIP without a subset of constraints (and the binary variables that appear only in these constraints). If the optimal solution of the obtained MIP does not satisfy some constraints not taken into account, they are added to the MIP (with the involved binary variables), which is then solved again. The process is repeated until the current optimal solution satisfies all constraints of the problem, even those that have been relaxed. As shown by Clare and Richards [2011], for this kind of problem, solving smaller and simpler MIP, even several times, is more efficient than solving the whole MIP. It is also the case for our formulation. The constraints that we have chosen to relax are those involving sequencing variables z_{iju}, i.e. separation constraints (3.24), sequencing constraints (3.25) to (3.28) and overtaking / head-on constraints (3.29) and (3.30). This method reduces the computation times significantly, the maximum computation time was divided by approximately 3. We do not present a comparison as it has already been shown by Clare and Richards [2011]. We tried to apply the same approach for the single path model, but no improvement were observed.

3.4 Data set and instances

In this section we present our instances and how they were generated. Each instance represents an operational day in Copenhagen Airport (CPH).

Runway configuration The most frequent runway configuration in CPH is the 22 mode (220° with the north), with take-offs on runway 22R (R=Right) and landings on runway 22L (L=Left). In the month of September 2012, we have selected 8 busy days (with more than 700 flights) in which more than 98% of flights were operated in this runway configuration. The 8

instances have a similar traffic profile. The average number of arrivals and departures by hour is presented in Figure 3.4. Minimum runway separation times used at CPH are presented in Table 3.1.

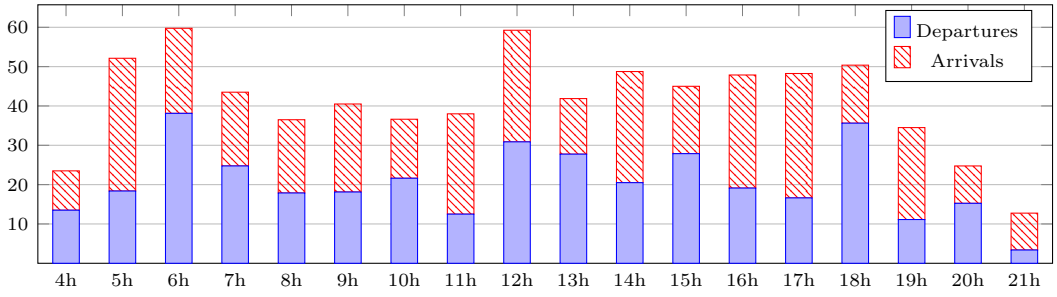


Figure 3.4: Average profile of instances

Time [s]		Trailing aircraft H	M	L
Leading aircraft	H	90	120	120
	M	60	60	90
	L	60	60	60

Table 3.1: Minimum runway separation time at CPH (H = heavy, M = medium and L = light)

Taxiway network Figure 3.5 presents the graph of the taxiway network with the 22 runway configuration. An edge represents an elementary taxiway segment. A node needs to be defined for each taxiway intersection. There is also a node for each stand. The graph is composed of 93 nodes and 235 edges. The standard path between each stand and each runway was provided by the airport, as well as the standard push back scheme and its duration, for each stand. We observed on ground radar data that standard paths were used for more than 83% of flights.

Ground radar data also provides an estimation of the maximal speeds. Based on these data, we assume a maximal speed of 15 m/s for the taxiways around the runway (in blue in Figure 3.5), of 5 m/s for the taxiways around the stands (in red in Figure 3.5) and of 10 m/s for the other taxiways. A minimum speed of 2 m/s is assumed on every edges. The minimum and maximum times spent on an edge (T_{iuv}^{min} and T_{iuv}^{max}) can be directly computed from the minimum and maximum speeds allowed on this edge and from the edge length. The minimum separation time (S_{iju}) between two aircraft is assumed to be 40 seconds for every nodes (except the runways, see Table 3.1), which guarantees a minimum separation of 80 meters between aircraft.

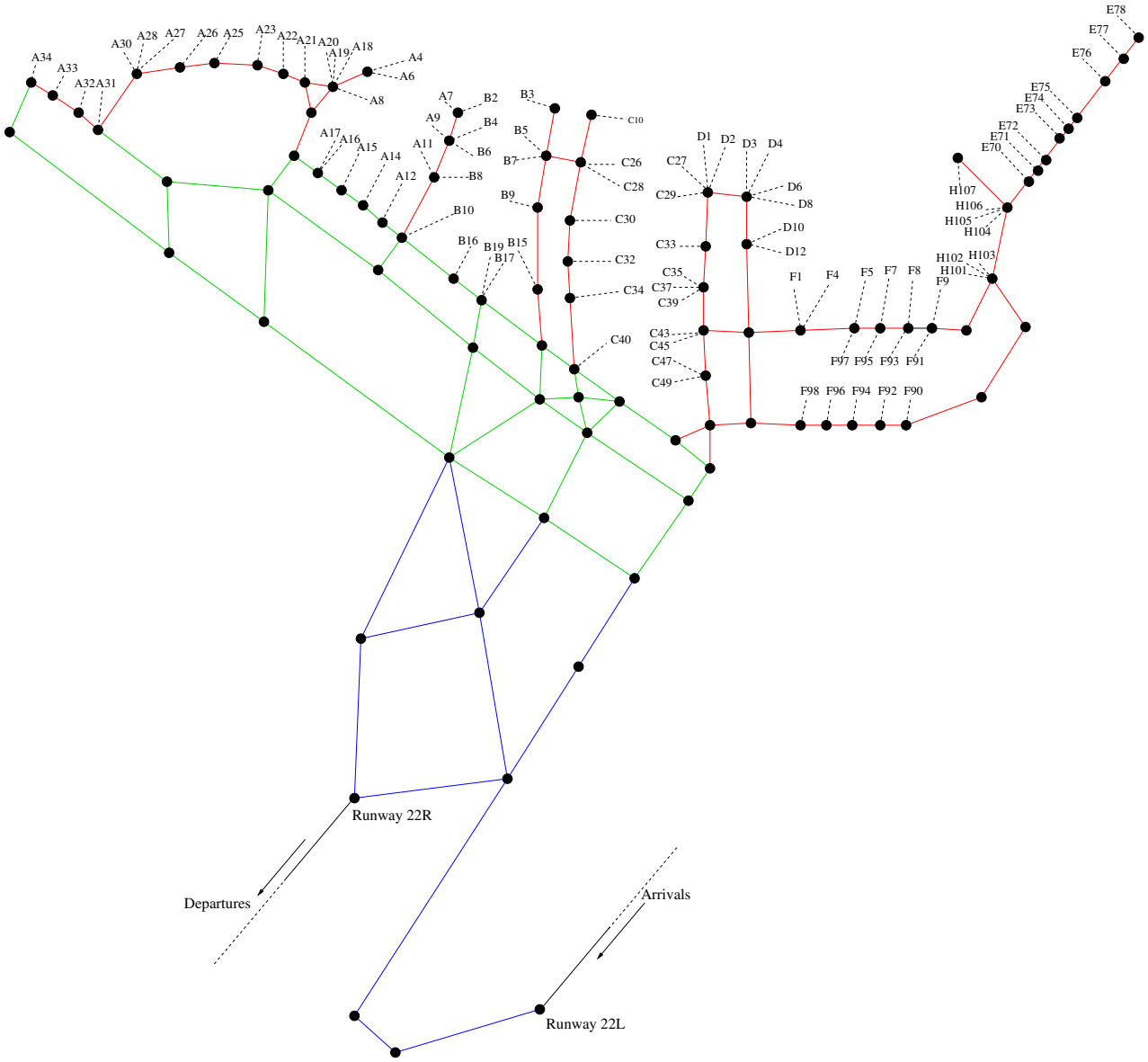


Figure 3.5: The taxiway network when runways are operated in 22 mode

Flights The ground radar data does not contain information on flights but only records composed of aircraft identifier, position in the airport and time stamp. The airport operational database provides other useful data for each flight.

It gives the Scheduled In-Block Time (SIBT) or the Scheduled Off-Block Time (SOBT), denoted by SB_i in our model. For each arriving flight, it also provides the Actual LanDing Time (ALDT) which can be used to define the release date T_{o_i}. However, we did not have access to the release date (or ready time) for departing flights, as CPH was not a A-CDM airport in 2012. As a push back as soon as possible policy is used in CPH most of the time, we have decided to take the Actual Off-Block Time (AOBT) to define the release date T_{o_i} for departing

flights. Finally, the take-off sequence is the actual one which can be derived from the Actual Take-Off Times (ATOTs).

Average performance indicators All results are averaged among the 8 days and the aircraft. For instance, an average taxi time of 8 minutes means that it takes on average 8 minutes for an aircraft to taxi, among the 8 days. We choose the OTP 15 indicator ($L = 15$) which is one of the main punctuality indicators in the air traffic industry.

3.5 Numerical results

The results are presented for the single path problem, except in Section 3.5.3 where we study the effect of the number of paths. We set $c^{taxi} = 1$ without loss of generality, as we vary the other weights c^{ct}, c^{delay} and c^{OTP}. Results of mixed integer programs were obtained with Cplex 12.4 solver using default parameter tuning on a personal computer (Intel Core i5-2400 3.10Ghz, 4Go RAM) under Ubuntu 12.04 LTS operating system. Java Concert API was used to define the models.

3.5.1 Sliding window optimization

As there are many stochastic events, it is not possible to schedule the movements of aircraft for the entire day. Hence the GRP is usually solved dynamically with a sliding window approach, in both literature and practice. The longer the time window is the better the solutions are, but the higher the computation times are.

The optimization does not need to be performed continuously but only when a new aircraft enters the system. Once an aircraft has started taxiing, changing its schedule is not allowed in the next time windows, but it has to be taken into account to ensure a conflict free routing.

In the rest of the numerical study we set a time window of 30 minutes. This assumption seems reasonable in the context of A-CDM project in which airlines and ground handlers are required to communicate accurate ready times 30 to 40 minutes in advance.

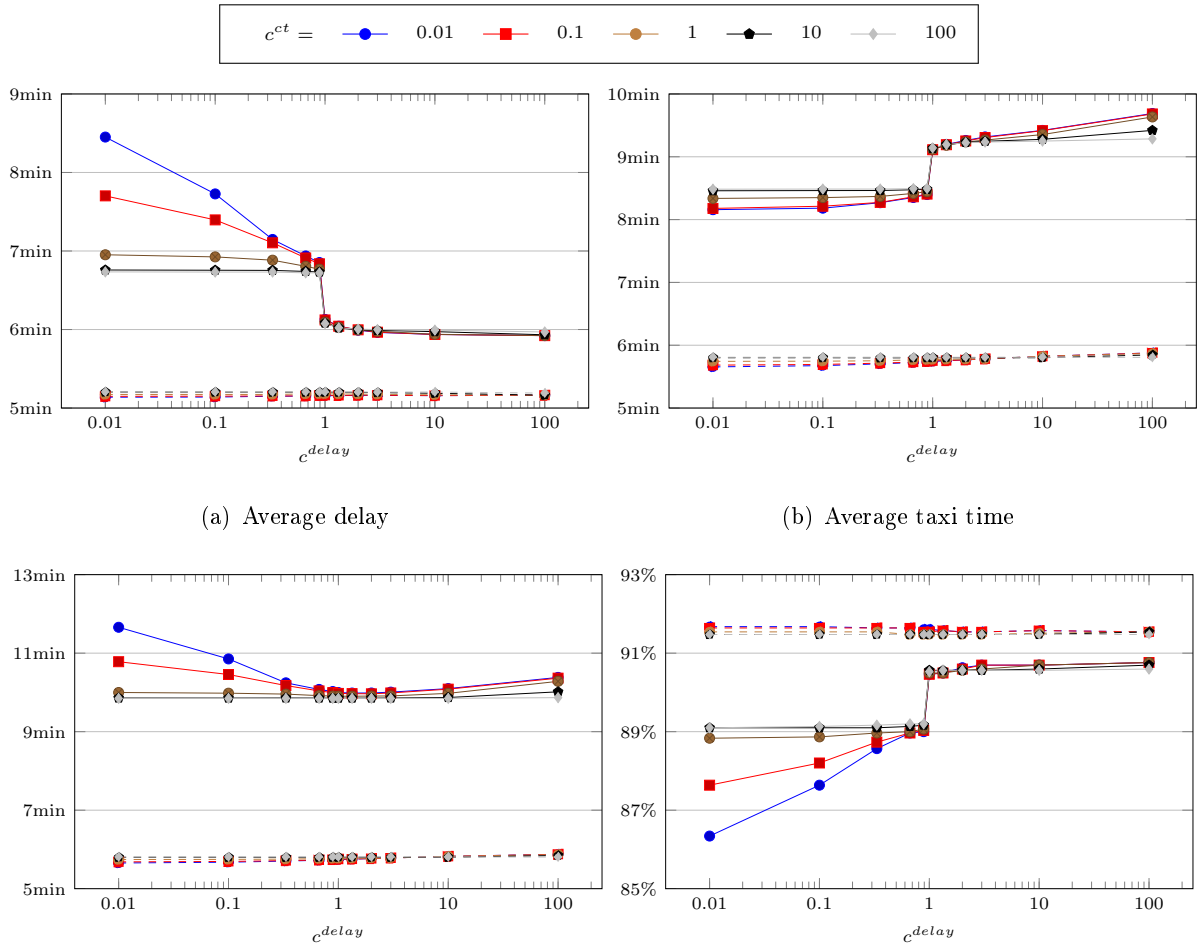
With a 30 minutes time window, computation times were always below two seconds for the single path approach. It appears that a time window of 15 minutes is sufficient in our test case, i.e. longer time windows do not provide better solutions. This value may be airport dependent and cannot be generalized without experiments in other airports.

3.5.2 Including the punctuality key performance indicators

The average delay and the OTP are the main punctuality indicators for airlines and airports. However the literature focuses on taxi time and completion time indicators. In this section, we study the impact of including the average delay and the OTP in the optimization.

3.5.2.1 Average delay

Figure 3.6 presents the effect of the weight c^{delay} on all KPIs for arrivals (dashed lines) and departures (solid lines) and for different values of c^{ct}.



(a) Average delay

(b) Average taxi time

(c) Average completion time

(d) OTP

Figure 3.6: Effect of including the delay indicator in the optimization for arrivals (dashed lines) and departures (solid lines) ($c^{taxi} = 1$, $c^{OTP} = 0$)

For arrivals, we observe that KPIs are not much impacted by c^{delay}, which can be explained as follows. The contribution of a delayed arriving flight $i \in \mathcal{F}_{arr}$ ($t_{id_i} > SB_i$) to objective

function (3.1) is, within a constant and because landing time t_{io_i} is fixed (see constraints (3.3))

$$(c^{delay} + c^{taxi} + c^{ct})t_{id_i} + c^{OTP}\mathbb{1}_{\{t_{id_i} > SB_i + L\}} \tag{3.34}$$

It clearly appears that KPIs are based only on variables t_{id_i}. Therefore including the delay adds redundancy and there is no trade-off to be done between the taxi time and the delay.

On the contrary for departures, increasing c^{delay} reduces delays but increases taxi times. One can even observe a threshold effect between the delay and the taxi time when $c^{delay} = c^{taxi}$. It is explained by the contribution of a delayed departure $i \in \mathcal{F}_{dep}$ ($t_{io_i} > SB_i$) to the objective function, which is, within a constant,

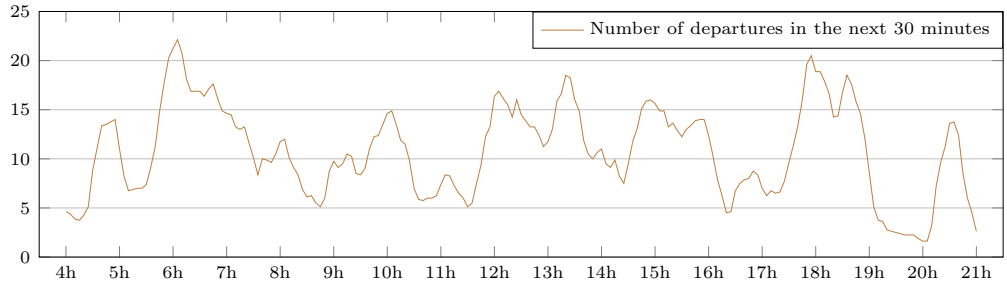
$$(c^{delay} - c^{taxi})t_{io_i} + (c^{taxi} + c^{ct})t_{id_i} + c^{OTP}\mathbb{1}_{\{t_{io_i} > SB_i + L\}} \tag{3.35}$$

It clearly highlights an opposition of the taxi time and the delay for departures. When $c^{delay} - c^{taxi} > 0$, pushing back aircraft earlier (which reduces t_{io_i}) is preferable as it reduces delays. But it leads to longer taxi times when the runway is congested. When $c^{delay} - c^{taxi} < 0$, holding aircraft at stands as much as possible (which increases t_{io_i}) is more profitable and avoids runway queuing. It consequently decreases taxi times, but implies larger delays.

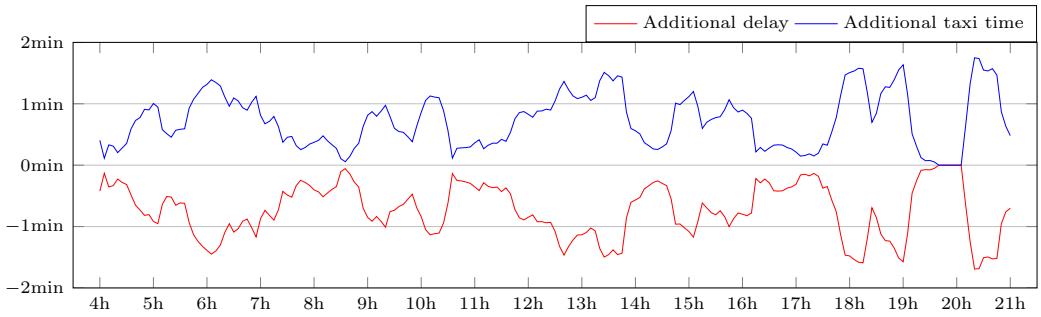
To further illustrate the opposition between the taxi time and the delay for departures, Figure 3.7 details the results along the day with a 30 minutes time window and a 5 minutes step. For instance, at 6h05, 3.7(a) plots the number of departures in the time interval [6h05,6h35[. Figure 3.7(b) plots the additional taxi time and the additional delay when we set $c^{delay} = 2$ instead of $c^{delay} = 0$.

We observe differences mainly during the peaks. In lows, aircraft push back as soon as possible, go to the runway in the shortest time and take-off immediately. Hence, all performance indicators are optimized. In peak hours, the runway is saturated and flights cannot take-off as soon as they reach the runway. They must either wait at stands or at the runway. When $c^{delay} = 0$, stand holding is preferred since it reduces the taxi time. When $c^{delay} = 2$, pushing-back earlier is preferred in order to reduce delays to the scheduled push back time.

In conclusion, delays cannot be significantly reduced without degrading taxi times in peak hours.



(a) Profile



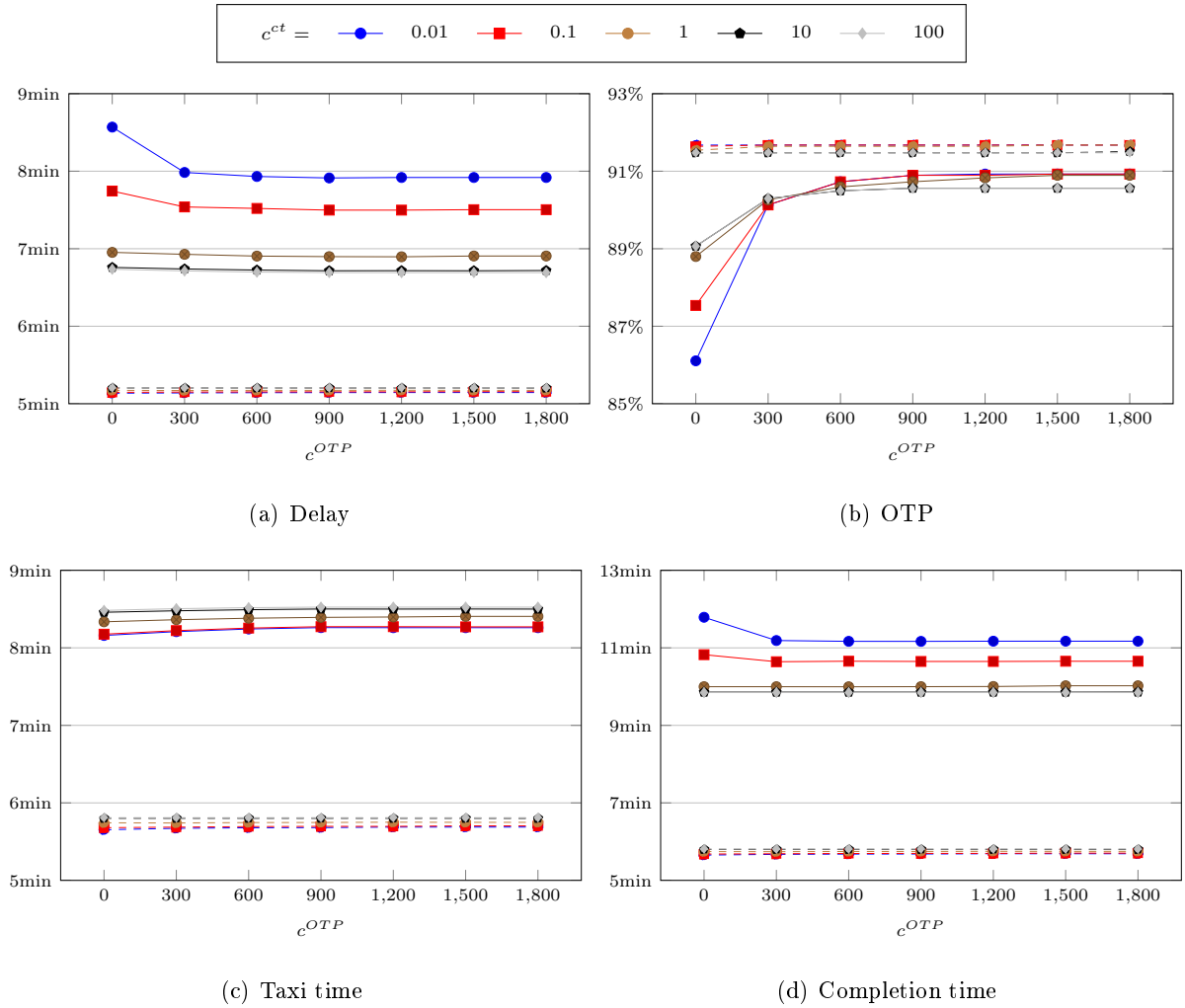
(b) Additional taxi time and delay when $c_{delay} = 2$ instead of $c_{delay} = 0$

Figure 3.7: Effect of including the delay indicator in the optimization along the day (departures only, $c^{ct} = 2$, $c^{taxi} = 1$)

3.5.2.2 OTP 15

Figure 3.8 presents the effect of the weight c^{OTP} on all KPIs for arrivals (dashed lines) and departures (solid lines) and for different values of c^{ct}. The results have some similarities with the previous section, as the OTP 15 is highly correlated to the average delay. However, there are some differences to be highlighted.

We observe that it is possible to significantly improve the OTP without degrading much the taxi time. Moreover, there is no threshold. Here again, the issue is to decide for each departure if the waiting time, due to runway congestion, may be spent at the stand or at the runway. OTP 15 is flexible as it provides a margin of 15 minutes, contrary to the delay for which every minute matters. This 15 minutes margin can be used to hold aircraft at stands and consequently to reduce the waiting time at the runway.



(a) Delay

(b) OTP

(c) Taxi time

(d) Completion time

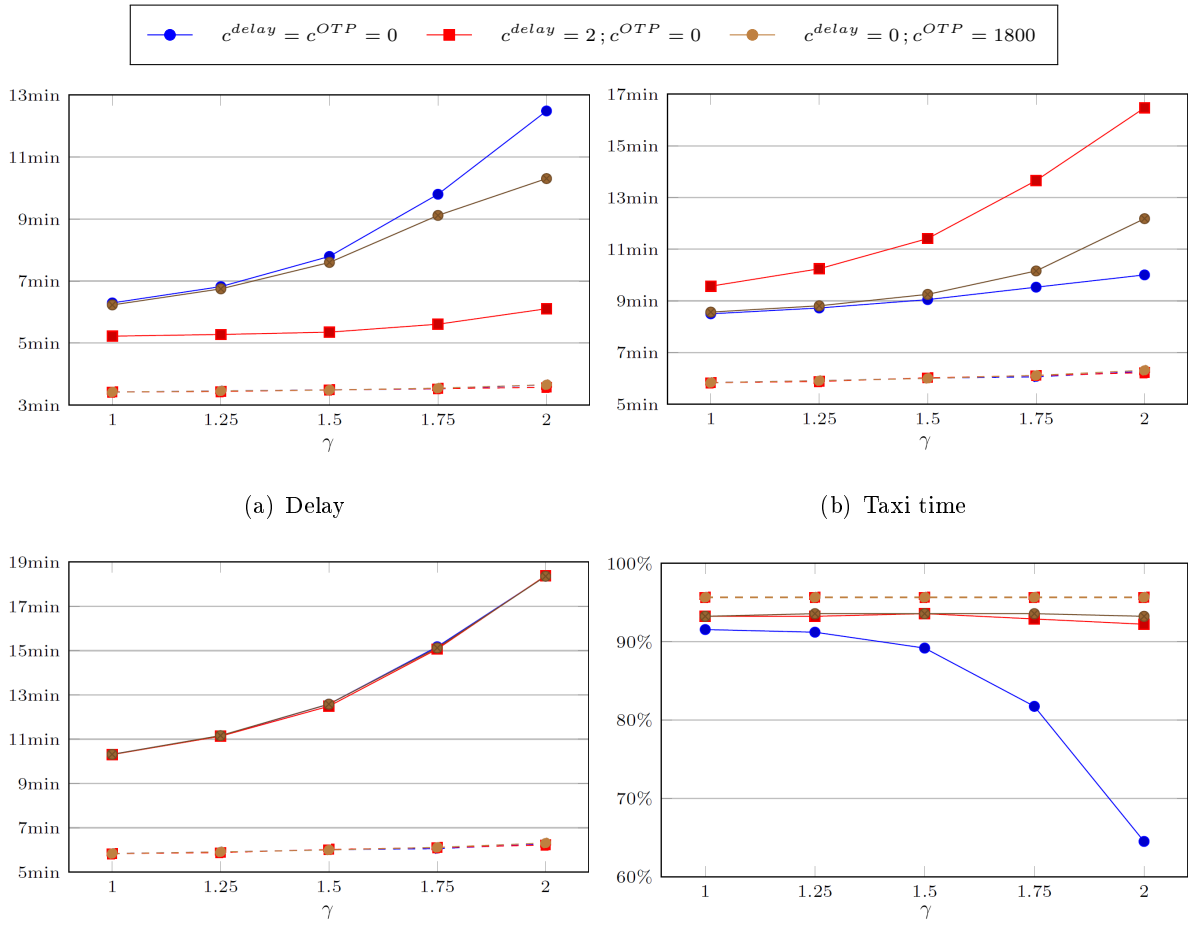
Figure 3.8: Effect of including the OTP indicator in the optimization for arrivals (dashed lines) and departures (solid lines) ($c^{taxi} = 1$, $c^{delay} = 0$)

3.5.2.3 Effect of reducing the network capacity

Figure 3.7(a) reveals that the maximal runway capacity is not exceeded even in departure peaks ($\approx$ 20 aircraft vs a maximal capacity of 30 aircraft), which means that the airport is not over-congested.

To simulate a higher congestion, we focus on the morning peak (from 5 to 7) and reduce network capacity by increasing the minimum separation times at each node, including the runway. All separation times are multiplied by a factor γ. Figure 3.9 presents the effect of γ on the performance indicators, for three different objective functions (without delay and OTP indicators, with delay indicator and with OTP indicator). We observe that the larger γ is, the more there is an opposition between the taxi time and the punctuality indicators. For instance, the

departure taxi time is increased by 7 minutes (resp. 2 minutes) if we include the delay (resp. OTP) in the objective function, when $\gamma = 2$.



(a) Delay

(b) Taxi time

(c) Completion time

(d) OTP

Figure 3.9: Effect of increasing separation times during the morning peak (from 5 to 7 am) for arrivals (dashed lines) and departures (solid lines) ($c^{ct} = 2$ and $c^{taxi} = 1$)

3.5.2.4 New departure punctuality indicators

Stand holding succeeds in reducing the taxi time significantly by transferring runway queuing time with engines on to the stands with engines off. We observed in figures 3.7 and 3.9 that it is particularly efficient during departure peaks. Nevertheless, our analysis also shows that this practice degrades the punctuality indicators. Hence airports and airlines may be reluctant to use stand holding and may prefer a push back ASAP policy to ensure good departure indicators. In this section, we question the relevance of OTP and delay indicators for airlines and airports and propose new punctuality indicators.

British Airways [2008-09] propose to base the measure of departure punctuality for airlines on the ready time and not on the push back time. It seems more appropriate since airlines are not accountable for the delay between the ready time and the push back time.

Measuring the punctuality with respect to push back times is also very artificial for airports as additional delays occur during the taxi process and particularly in the runway queue. Thus, it would be much more natural to base the measure of the punctuality on take-off times. However, a Scheduled Take-Off Time (STOT) does not exist neither in A-CDM, nor (to our knowledge) in the industry. We propose to define STOT as SOBT plus a constant depending on the airport, for instance the average departure completion time ($\overline{ATOT - TOBT}$). Our models can easily be adapted to measure the delay and the OTP with respect to STOT. Constraints (3.10) can be merged with constraints (3.11) as follows :

$$t_{id_i} \leqslant SB'_i + L + M\beta_i \quad \forall i \in \mathcal{F}$$

$$\delta_i \geqslant t_{id_i} - SB'_i \quad \forall i \in \mathcal{F}$$

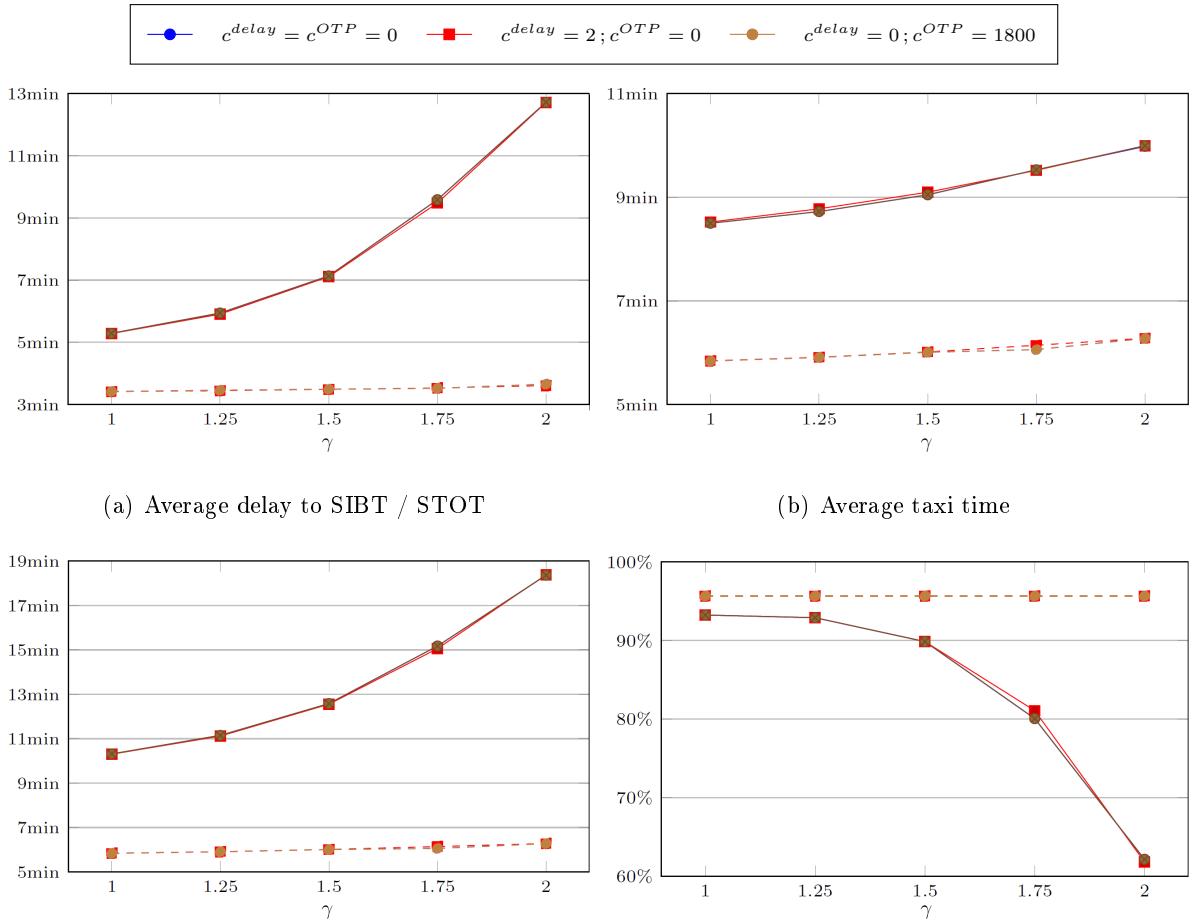
where SB'_i is the Scheduled In-block Time ($SIBT$) for arrivals and the Schedule Take-Off Time ($STOT$) for departures. OTP constraints (3.12) are unchanged.

Figure 3.10 presents the effect of the congestion factor γ with the new definition of the delay and the OTP. We observe that including the delay to SIBT / STOT in the objective function does not impact the KPIs. It can be explained by the contribution of a delayed departure $i \in \mathcal{F}_{dep}$ ($t_{id_i} > SB'_i$) to the objective function (within a constant):

$$c^{taxi} t_{io_i} + (c^{taxi} + c^{ct} + c^{delay}) t_{id_i} + c^{OTP} \mathbb{1}_{\{t_{id_i} > SB'_i + L\}}$$

Both the completion time and the delay are now based on take-off times and are thus redundant. The contradiction between the taxi time and the delay is tackled. From a practical point of view, it means that optimizing the completion times and the taxi times is sufficient to optimize the new delay. We performed the same study for the OTP and the same conclusions were obtained.

The new definition of the departure punctuality indicators will be used in the rest of this chapter, i.e. the delay and the OTP will be measured with respect to STOT.



(a) Average delay to SIBT / STOT

(b) Average taxi time

(c) Average completion time

(d) Average OTP to SIBT / STOT

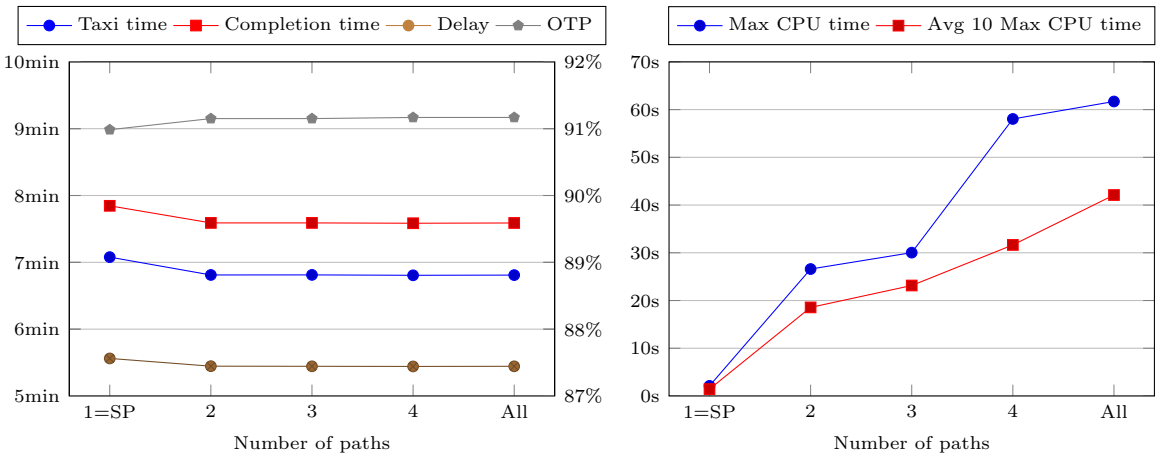
Figure 3.10: Effect of increasing separation times, with the new punctuality indicators, during the morning peak (from 5 to 7 am) for arrivals (dashed lines) and departures (solid lines) ($c^{ct} = 2$ and $c^{taxi} = 1$)

3.5.3 Effect of the number of paths

In this section, we compare the single path approach to the alternate paths one through an analysis of the effect of the number of paths. The alternate paths were generated with a k-shortest paths algorithm (as proposed by Gotteland et al. [2001]) with $k = 10$. Then unrealistic paths were filtered to respect basic routing logic (no cycles, turning-angles, etc...). The final number of paths between a stand and a runway is between 3 and 9 in our data. The first path of the set is always the one used in the single path approach (provided by the airport).

Figure 3.11 shows that the number of paths has a limited effect on KPIs but increases a lot the computation time. One can also notice, that considering more than two paths does not improve the results. Consequently, computation times can be reduced by restricting the number

of paths, which brings the maximum computation time from 61 seconds to 27, but the method is still difficult to implement in peak hours. Nevertheless, experiments were performed with a standard computer and simply increasing computing power may be enough to fit industrial requirements.



(a) Effect on KPIs

(b) Effect on CPU time

Figure 3.11: Effect of the number of paths ($c^{ct} = 2, c^{taxi} = 1, c^{OTP} = c^{delay} = 0$, new OTP and delay)

The lack of gain provided by the alternate path can be explained by the structure of the taxiway network: there are two main parallel taxiways serving the stands and each one is used in a different direction in the single path approach, consequently avoiding most of head-on conflicts between departures and arrivals. On the contrary, the area around the stands is much more intricate and generally offers a single taxiway. This means that the alternate paths do not allow to avoid much more conflicts that the single path approach. This intuition is further explored in the next section through an analysis of the airport bottlenecks.

3.5.4 Bottleneck analysis

The taxiway network can be divided in three distinct parts: the runway, the ramp area (the area around the stands) and the taxiway area. In this section, we evaluate the impact of each area on the routing, by relaxing its constraints in the optimization.

Figure 3.12 presents the results of this analysis. *All constraints* means that all constraints are taken into account. *Taxiway* means that safety constraints of the taxiway area are relaxed. *Ramp* means that safety constraints of the ramp area and stand blockages constraints are relaxed. *Runway* means that separation constraints of the runway are relaxed. Besides no take-off

sequence is forced. *No constraint* is the case where all the above constraints are relaxed and aircraft taxi at maximal speeds without stopping anywhere. The delay and the OTP are measured with respect to SIBT / STOT (as defined in Section 3.5.2.4).

Figure 3.12 shows that the taxiway area has a limited impact on indicators, which join the conclusion of previous section. On the contrary, the ramp area and the runway have a more significant impact on indicators.

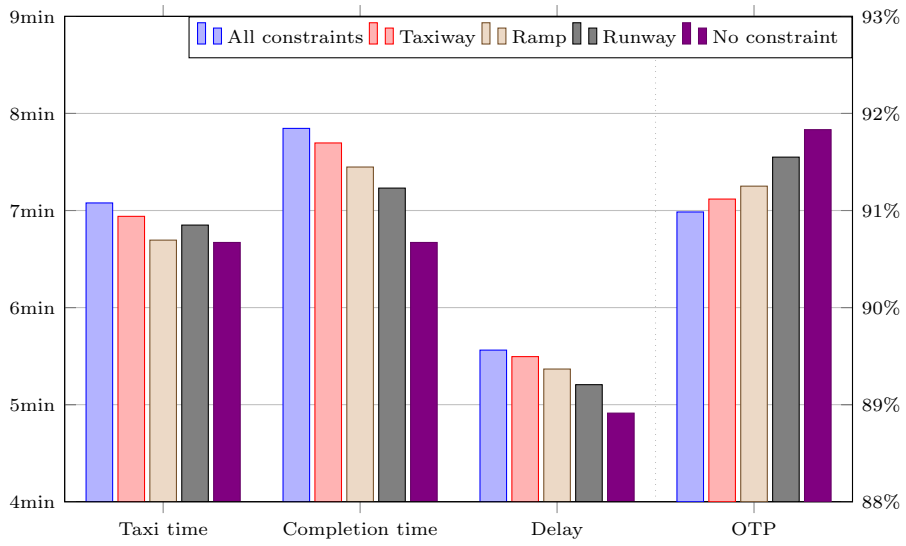


Figure 3.12: Bottleneck analysis ($c^{ct} = 2, c^{taxi} = 1, c^{OTP} = c^{delay} = 0$, new OTP and delay)

3.6 Conclusion

In this chapter, we formulate the ground routing problem as a MIP. We present a formulation with a single path and generalize it to include alternate paths. We consider an objective function that includes the punctuality indicators used in industry (average delay and OTP 15).

In a numerical study based on data from CPH, we first show how the industry punctuality indicators are in contradiction with a sustainable management of airports. The punctuality of departures is currently measured with respect to push back times, which encourages to push back as soon as possible and results in large taxi times in peak hours because of runway congestion. Including the delay in the objective function leads to a taxi time increase of 1 minute in average for departures at CPH. In more congested situations, this increase can reach 6 minutes. Including the OTP 15 in the objective function has less impact in current traffic situations. However, in more congested situations, it also leads to longer taxi times.

We propose to measure the punctuality of airports with respect to take-off times and not with respect to push back times. We show that this new measure of punctuality do not prevent stand holding and limit pollution emissions. Besides they are more appropriate since they capture additional delays between push back and take-off.

We also show that intermediate taxiways are not a bottleneck of CPH Airport and that considering alternate paths do not improve the performance indicators significantly.

Numerical experiments were performed in CPH and we may wonder to what extent our results can be generalized to other airports. In congested airports, the delay and OTP indicators will intuitively not be adequate to measure punctuality, as they encourage to push back as soon as possible and lead to long taxi times. In non congested airports, it will not matter as a push back as soon as possible policy should be nearly optimal.

The main parallel taxiways serving the ramp areas in CPH prevent most of head-on conflicts in the taxiway area. Such a structure is very common and is present in the five most frequented airports in the world[3]. In such configurations, the alternate path approach will probably not bring much with respect to the single path approach. However, the alternate path approach is certainly more beneficial in airports with more complex taxiway layout, typically with runway crossing as shown by Balakrishnan and Jung [2007].

Appendix

3.A Other objectives

– The NMOC slot compliance can easily be included in our model. Maximizing the NMOC slot compliance is equivalent to minimize the number of slots missed. We define new binary variables β'_i indicating if the slot is missed for every regulated departure $i \in \mathcal{F}^{reg}_{dep}$. The NMOC slot compliance objective is

$$\sum_{i \in \mathcal{F}^{reg}_{dep}} c_i^{NMOC} \beta'_i$$

3. In 2014 by passenger traffic according to Wikipedia, Hartsfield-Jackson Atlanta International Airport (ATL), Beijing Capital International International airport (PEK), London Heathrow airport (LHR), Tokyo Haneda airport (HND) and Los Angeles International airport (LAX).

where $c_i^{NMOC} \geqslant 0$. Following constraints are necessary to ensure the definition of variables β'_i.

$$\begin{aligned} t_{id_i} &\leqslant CTOT_i + 10 * 60 + M\beta'_i \quad \forall i \in \mathcal{F}_{dep}^{reg} \\ t_{id_i} &\geqslant CTOT_i - 5 * 60 - M\beta'_i \quad \forall i \in \mathcal{F}_{dep}^{reg} \\ \beta'_i &\in \{0,1\} \quad \forall i \in \mathcal{F}_{dep}^{reg} \end{aligned}$$

The deviation inside the slot can also be considered, it can be modeled in a similar way that the deviation to the take-off schedule.

– We chose not to use the deviation to TTOT as an objective but it can easily be modeled as it is often used in the literature. We need to define new continuous variables δ'_i for every departure $i \in \mathcal{F}_{dep}$ which will measure the deviation. The deviation to TTOTs is

$$\sum_{i \in \mathcal{F}_{dep}} c_i^{TTOT} \delta'_i$$

where $c_i^{TTOT} \geqslant 0$. Following constraints are necessary to ensure the definition of variables δ'_i.

$$\begin{aligned} \delta'_i &\geqslant t_{id_i} - TTOT_i \quad \forall i \in \mathcal{F}_{dep} \\ \delta'_i &\geqslant TTOT_i - t_{id_i} \quad \forall i \in \mathcal{F}_{dep} \\ \delta'_i &\geqslant 0 \quad \forall i \in \mathcal{F}_{dep} \end{aligned}$$

Chapter 4

The runway sequencing problem

Abstract

The Runway Sequencing Problem (RSP) consists in scheduling runway operations (landing, take-off, and crossings) with the objective of maximizing the use of the runway capacity, while respecting operational requirements. This chapter is a preliminary introduction to Chapter 5, which focus on the integration of the RSP and the Ground Routing Problem. We introduce runway operations and review the literature. We then present and compare two models from the literature. Our main contribution is to propose a new formulation of the separation constraints. This reformulation is based on wake vortex categories but it does not imply a loss of generality. It results in significantly reducing computation times, making the model appropriate for an industrial application.

Keywords: Mixed integer programming, runway sequencing

4.1 Introduction

In Chapter 3, the runway has been identified as an important bottleneck of the departure traffic in Copenhagen Airport (CPH), especially during peaks. Such a finding is classical: in both the literature and the industry, runways are recognized as one of the most critical resources due to their scarcity (see e.g. Idris et al. [1998] or Atkin et al. [2007]). As building new runways is not a short term solution and is often not possible, an efficient use of current runway capacity is crucial.

In this chapter, we focus on the Runway Sequencing Problem (RSP). It consists in sequencing runway operations while ensuring safety, i.e. in deciding in which order (and when) each aircraft takes off, lands on or crosses a runway, while respecting minimum separation requirements. Depending on the airport layout, runway assignment can be also part of the problem. This problem is issued by Air Traffic Controllers (ATC) on an operational window of typically 10 to 40 minutes. Longer time windows can hardly be considered because of perturbations occurring the day of operations. Hence a sliding time window scheme is implied and is used in both the literature and the practice. Consequently, the problem has to be solved very often and computation times are critical.

The quality of a runway sequence is defined by different criteria. The main criterion is the efficiency and the good use of the runway capacity. In the literature, it is often modeled by the runway throughput (makespan), the total (weighted) completion time or the total (weighted) deviation to targeted take-off or landing times. Equity is another important criterion, it is often measured by the deviation to the First Come First Serve order (FCFS, the fairest order) or the maximum delay.

Minimum separation requirement is the main limiting factor of runway capacity. Because of wake vortex, an airmass is perturbed when it is crossed by an aircraft and a minimum separation time must be respected between two aircraft to ensure the safety of the second one. The heavier the leading aircraft is, the bigger the wake vortex is and thus the longer the separation is. The lighter the trailing aircraft is, the more it is subject to turbulence and the longer the separation time is. Aircraft are classified in wake vortex categories by the International Civil Aviation Organization (ICAO) and the minimum separation between two aircraft depends on their respective classes (see e.g. Table 4.1).

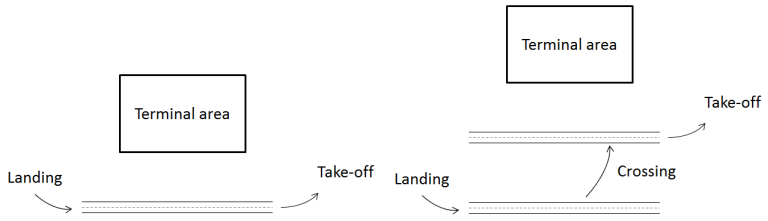
Additional separation may be necessary to prevent conflicts in airspace segment of routes linking entry and exit points of the Terminal Maneuvering Area (TMA) to the runways, also

known as Standard Instrument Departure routes (SIDs) and Standard Terminal Arrival Routes (STARs). Minimum separation requirements are also required for runway crossings.

		Trailing aircraft		
Time [s]		H	M	L
Leading aircraft	H	90	120	120
	M	60	60	90
	L	60	60	60

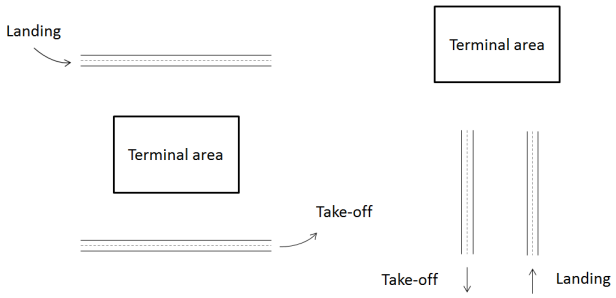
Table 4.1: Minimum take-off separation times (H = heavy, M = medium and L = light)

Two different runway management modes exist: segregated mode and mixed mode. In mixed mode, a runway handles both take-offs and landings whereas in segregated mode, only take-offs or only landings can be processed. Mixed mode is known to be more efficient since separations



(a) Runway operated in mixed mode

(b) Runways operated in segregated mode with runway crossing



(c) Runways operated in segregated mode without runway crossing

Figure 4.1: Frequent runway configurations

between arrivals and departures are shorter (see Ghoniem et al. [2014]), but segregated mode is often used because of airport layouts: two parallel runways interfere when they are too close to each other [1], i.e. minimum separations between two take-offs or two landings must be respected

1. Less than 2500 ft $\approx$ 760m (Balakrishnan and Chandran [2006])

on the adjacent runway too. In that case, one runway is used for departures and the other one for arrivals. Some common configurations are presented in Figure 4.1.

This chapter is organized as follows. A literature review of related works is presented in Section 4.2. Two models with some improvements from the literature are presented in Section 4.3 with our new reformulation. Then a comparison of these approaches is presented in Section 4.4. Section 4.5 draws the key conclusions of this chapter.

4.2 Literature review

As one of the main problems of airport resource management, the Runway Sequencing Problem (RSP) has been broadly studied and many techniques have been proposed for addressing it. There actually are many variants of the RSP, depending on the runway configuration. They mainly differ in traffic type: some variants focus on take-offs with or without runway crossing, some variants focus on landings and some other ones deal with mixed mode. These variants are not fundamentally different but do not involve the same separation standards. The consideration of multiple runways, and thus runway assignment, is another significant distinction.

Bennell et al. [2011] present a detailed description of the problem variants and an extensive literature review. They notice that landing sequencing problems received a greater interest. Since 2011, the interest in mixed mode problems has increased. Runway assignment has also aroused recent works due to its challenging computational difficulty. Based on their extensive classification by solving techniques, we briefly recall the most influencing works prior to 2011 and complete the classification with recent works.

Problem complexity

As shown in Beasley et al. [2000], even the simpler variants of the RSP (single runway and single traffic type) are NP-hard for most of the objectives because of ready times and sequence dependent separation requirements (similarities to a job shop scheduling problem with ready times and sequence dependent processing or set-up times).

Nevertheless, particular cases can be solved in polynomial time using the structure of the separations, which are based on wake vortex categories. Assuming that aircraft of the same category are *similar* allows to sequence them in the First Come First Serve order (FCFS) inside each class without loss of optimality[2]. This property allows to design polynomial algorithms

2. see Briskorn and Stolletz [2014] for a rigorous definition of similar

in the number of aircraft, but non-polynomial in the number of categories (see Briskorn and Stolletz [2014]). These algorithms are not always adapted to take-off sequencing problems since additional separation times can be necessary for safety in SIDs and because of CFMU slots which differentiate aircraft. Moreover, it also forbids any other differentiation, such as the weighted completion times where weights can be the number of passengers on board to minimize the total passengers waiting time.

Other particular cases can be solved in polynomial time without any assumption on the separations but using the so-called Constrained Position Shifting (CPS), where each aircraft can only be shifted by a limited number of positions from the FCFS order (see Balakrishnan and Chandran [2010]). These approaches are polynomial in the number of aircraft but exponential in the maximum number of position shifting. They additionally ensure fairness.

Dynamic programming (DP)

Based on the assumption of classification in wake vortex categories, Psaraftis [1978] proposes a DP for maximizing runway throughput in a landing sequencing problem where ready times are not taken into account. Brentnall [2006] extend the approach to include ready times and model the total completion time. Briskorn and Stolletz [2014] further extend the approach to multiple runway operated in mixed mode for piecewise linear objective functions. Nevertheless the algorithm presented is not tractable, even if polynomial. Lieder et al. [2015] present a dominance criterion to reduce the search space and consequently make the algorithm tractable.

Rathinam et al. [2009] consider holding point constraints in a take-off sequencing problem on a single runway. Aircraft are pre-assigned to runway entry queues and a first come first serve order has to be respected inside each queue. Using this structure, the problem is efficiently solved by a DP that minimizes the total aircraft delay.

Balakrishnan and Chandran [2006, 2007, 2010] propose a polynomial DP under CPS for mixed mode. The approach is based on the so-called CPS graph and they consider several objectives such as the runway throughput or the total delay. They also consider arbitrary aircraft dependent costs through a time discretization (time-space network).

Integer Programming (IP) and Mixed Integer Programming (MIP)

The reference MIP model is presented by Beasley et al. [2000]. The model minimizes the deviation to target landing times on multiple runways not necessarily independent. It uses binary variables for sequencing aircraft and continuous variables for scheduling them. Additional binary

variables are used to deal with runway assignment. They also propose a discrete time model. Solveling et al. [2011] use a similar model for close parallel runways operated in segregated mode with runway crossing. Their objective function gathers several different objectives which are converted into dollars for homogeneity. Briskorn and Stolletz [2014] also use a similar model and consider arrivals and departures on multiple mixed mode runways. Aircraft classes are assumed, which allows to fix sequencing variables inside each class. Ghoniem et al. [2014] propose a similar model for minimizing the makespan for arrivals and departures. They propose several techniques for strengthening the linear relaxation (preprocessing and probing procedures, valid inequalities...). They also derive two heuristic algorithms which consist in fixing some sequencing variables in their model.

Kim et al. [2010] extend the MIP of Beasley et al. [2000] to consider entry and exit points of the Terminal Maneuvering Area (TMA, the airspace around the airport) and gates. Their model aims at minimizing total emissions in the TMA through runway assignment and take-off and landing scheduling. Emissions include taxi phase, take-off/landing phase, climb-out/descent phase and queuing time.

Furini et al. [2012] propose an alternative MIP formulation minimizing the total delay for both arrivals and departures on a single runway. It uses binary variables indicating the position of aircraft in the runway sequence.

Faye [2015] considers the same problem than Beasley et al. [2000]. Time is discretized which allows to model several different objectives. The approach is based on approximating the separation matrix by a rank two matrix. Depending on the choice of approximation, it provides an upper or a lower bound. These bounds are used in a constraint generation algorithm for solving the problem exactly or heuristically.

Greedy algorithms and metaheuristics

Several genetic algorithms have been proposed for solving the landing sequencing problem, more details can be found in Bennell et al. [2011].

Hancerliogullari et al. [2013] minimize the total weighted tardiness on multiple runways operated in mixed mode. They propose three greedy algorithms and two metaheuristics. Their simulated annealing algorithm is shown to provide good solutions in reasonable computation times.

Other approaches

Atkin et al. [2012] consider a single runway take-off sequencing problem taking into account stand contention. Their approach aims at providing simultaneously take-off and push back times. They proceed in two phases: they first compute an optimized runway with feasible push back time. Then, push back times are reoptimized to reduce runway queuing time. The runway optimization algorithm is a complex heuristic which core algorithm is a Branch & Bound (B&B) embedded in a rolling window.

Bosson et al. [2014a] address landing sequencing and airspace movements scheduling under uncertainty. Bosson et al. [2014b] additionally consider ground movement in the approach. They consider stochastic release times and due dates (estimated time of arrival and departure) through sampling (Sample Average Approximation) embedded in a 3-phases decomposition.

D'Ariano et al. [2015] treats a runway sequencing problem taking airspace segment into account and minimizing delay propagation. Their approach is based on Alternate Graphs from job shop scheduling literature. They propose heuristic algorithms and an exact B&B.

Furini et al. [2015] embed the MIP of Beasley et al. [2000] and the MIP of Furini et al. [2012] in a rolling horizon approach for minimizing the total weighted delay on a single runway operated in mixed mode. They also propose a tabu search algorithm. They compare different criteria for splitting the optimization time window.

Ghoniem et al. [2015] present a column generation approach for minimizing the total weighted completion time on multiple runways operated in mixed mode. The master problem is a set partitioning problem and a column represents an assignment plan to a runway (i.e. gathers a group of aircraft assigned to the same runway). Columns are evaluated with a DP assuming wake vortex categories.

Summary of our contributions

The literature review reveals that the RSP has been broadly studied and that many solving techniques have been investigated for the isolated problem, while only few works have studied its integration with other resources. Consequently, our main contribution is the integration of the RSP with the Ground Routing Problem (GRP), treated in Chapter 5. Nevertheless, this chapter contributes in improving a model from the literature through the reformulation of separation constraints. Our reformulation is based on wake vortex categories, but it does not rely on any assumption implying a loss of generality.

4.3 The single runway take-off sequencing problem

In this section, we address the minimization of the total completion time on a single runway problem. The presented models are valid for segregated take-offs, segregated landings or mixed mode, including runway crossing. We focus on the case of segregated take-offs.

Given a set of flights $\mathcal{F}$, the aim is to schedule take-off times such that the total completion time is minimized. Every pair of flights $i, j \in \mathcal{F}$ must be safely separated: a minimum separation time S_{ij} is required between flights i and j, if i precedes j. A flight $i \in \mathcal{F}$ is subject to time window restriction defined by an earliest and a latest take-off time e_i and l_i. Earliest take-off time e_i are due to the readiness of aircraft. Latest take-off times l_i intend to prevent excessive delays. Take-off time windows $[e_i, l_i]$ will be referred to as take-off slots.

As mentioned in Section 4.2, this problem is NP-hard due to similarities to a job shop scheduling problem with ready times and sequence dependent processing times (see Beasley et al. [2000]).

We present a MIP and an IP formulation that have originally been proposed by Beasley et al. [2000] for sequencing landing while minimizing the deviation to some targeted landing times. Many other objectives can be considered and aircraft can be distinguished from each other (by weighting) thanks to the flexibility of linear programming. Improvements from the literature are also presented for both models. We additionally present a new formulation of separation constraints for the IP.

In Chapter 5, we focus on the integration of the RSP with the GRP, which induces differences between aircraft belonging to the same wake vortex category. Consequently, the models we present hereafter do not assume any similarities, even if it could allow to design more efficient algorithms.

4.3.1 Continuous time MIP

4.3.1.1 Original model [Beasley et al., 2000]

The model uses continuous variables for scheduling flights and binary variables for sequencing them:

- $t_i \in [e_i, l_i]$ the take-off time of flight $i \in \mathcal{F}$.
- $z_{ij} = 1$ if flight $i \in \mathcal{F}$ takes off before flight $j \in \mathcal{F}$, 0 otherwise.

The problem can be formulated as follows.

$$
\begin{aligned}
(MIP) \quad & \min && \sum_{i \in \mathcal{F}} t_i && && (4.1)\\
& s.t. && e_i \leqslant t_i \leqslant l_i && \forall i \in \mathcal{F} && (4.2)\\
& && z_{ij} + z_{ji} = 1 && \forall i,j \in \mathcal{F} && (4.3)\\
& && t_j \geqslant t_i + S_{ij} - M_{ij}(1 - z_{ij}) && \forall i,j \in \mathcal{F} && (4.4)\\
& && z_{ij} \in \{0,1\} && \forall i,j \in \mathcal{F} && (4.5)
\end{aligned}
$$

Objective function (4.1) is the total completion. Constraints (4.2) ensure the respect of take-off slots. Constraints (4.3) ensure that either i takes off before j or the opposite. Constraints (4.4) ensure minimum separation requirements, where M_{ij} is a large enough value. As Beasley et al. [2000] remark it, M_{ij} can be defined as $l_i + S_{ij} - e_j$. It must be remarked that this model is not valid if $l_i = +\infty$, since M_{ij} cannot be large enough. Beasley et al. [2000] explain how to strengthen the slots without loss of generality from an upper bound. Such an upper bound can be provided by any feasible solutions, e.g. obtained with a simple FCFS algorithm. This method can be used for bounding l_i consequently allowing the model to deal with infinite take-off slots.

This model is hereafter referred to as *MIP*.

4.3.1.2 Using wake vortex categories

Assuming that aircraft of the same wake vortex categories are similar allows to fix many variables in the model (the length of slots $l_i - e_i$ must be equal for every flight of a given category). Indeed, there exists an optimal solution such that aircraft take off in FCFS order in each class (see Briskorn and Stolletz [2014]). It allows to fix all variables z_{ij} linking two flights of the same wake vortex categories ($z_{iju} = 1$ if $e_i < e_j$), which significantly reduces the number of binary variables. Nonetheless, it also reduces the model generality and it can hardly be generalized to integrate ground routing. However, variables fixing is a significant improvement from the literature and we will evaluate its effect in Section 4.4.

This model is hereafter referred to as *MIP var fix*.

4.3.2 Discrete time IP

In this section, we present a discrete time model. In practice, estimation of ready times at the runway are at a precision of 1 minute and separation standards of Table 4.1 are at a precision of 30 seconds. Therefore, a discretization step of 30 seconds is enough to ensure optimality.

4.3.2.1 Original model [Beasley et al., 2000]

We denote as T_i the discrete set of possible take-off times, i.e. the discretization of the interval $[e_i, l_i]$ ($T_i = \{e_i, e_i + 30, \ldots, l_i\}$). The model uses binary variables $x_{it} = 1$ if flight $i \in \mathcal{F}$ takes off at time $t \in T_i$ and 0 otherwise. The problem can be formulated as follows.

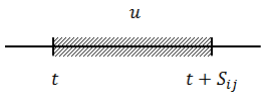
$$(IP) \qquad \min \quad \sum_{i \in \mathcal{F}} \sum_{t \in T_i} c_{it} x_{it} \tag{4.6}$$

$$s.t. \quad \sum_{t \in T_i} x_{it} = 1 \quad \forall i \in \mathcal{F} \tag{4.7}$$

$$x_{it} + x_{ju} \leqslant 1 \quad \forall i, j \in \mathcal{F}, \forall t \in T_i, \forall u \in T_j, t \leqslant u < t + S_{ij} \tag{4.8}$$

$$x_{it} \in \{0, 1\} \quad \forall i \in \mathcal{F}, \forall t \in T_i \tag{4.9}$$

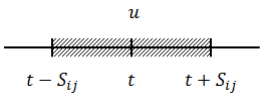
where $c_{it} = t$ for modeling the total completion time in objective function (4.6). Note that coefficients c_{it} can be used to model many other objectives and particularly non linear objectives, unlike the MIP formulation. Constraints (4.7) ensure that flights take-off once and only once in their slot. Constraints (4.8) ensure minimum separation requirements:



This model is hereafter referred to as *IP*. Note that if $l_i = +\infty$, *IP* contains an infinite number of variables and cannot be directly solved with a solver. This problem is tackled in Section 4.3.2.4.

4.3.2.2 Reformulation of separation constraints [Fahle et al., 2003]

This model is not further explored by Beasley et al. [2000], but Fahle et al. [2003] propose to reformulate separation constraints (4.8) using clique constraints as follows



$$x_{it} + \sum_{\substack{u \in T_j \\ t - S_{ji} < u < t + S_{ij}}} x_{ju} \leqslant 1 \quad \forall i, j \in \mathcal{F}, \forall t \in T_i \tag{4.10}$$

IP with such a constraints reformulation will be referred to as *IP Fahle*.

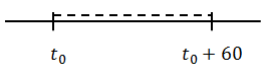
4.3.2.3 Reformulation of separation constraints using wake vortex categories

We propose a stronger reformulation based on wake vortex categories. It must be understood that wake vortex categories are used only to design valid inequalities, which does not lessen the generality of the model: each aircraft is still differentiated from each other. Also, additional separations can still be added with either Constraints (4.8) or (4.10).

Reformulation are presented for the minimum take-off separations of Table 4.1, but a similar approach can be used for other separation standards. We respectively denote $\mathcal{H}$, $\mathcal{M}$, $\mathcal{L}$ the set of flights operated by heavy, medium and light aircraft.

Clique of 60 seconds

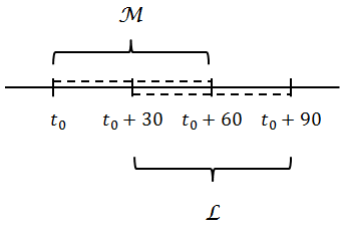
It can be remarked that every aircraft must be separated by at least 60 seconds. Consequently following inequalities are valid and cover the minimum separation of couples M/H, M/M, L/H, L/M, L/L.



$$\sum_{i \in \mathcal{F}} \sum_{\substack{t \in T_i \\ t_0 \leqslant t < t_0 + 60}} x_{it} \leqslant 1 \quad \forall t_0 \in \bigcup_{i \in \mathcal{F}} T_i \tag{4.11}$$

Clique based on medium aircraft category

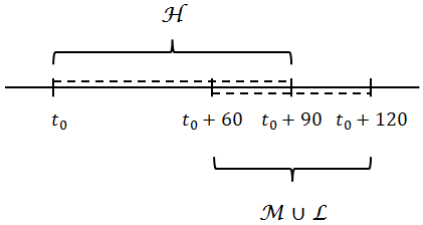
The minimum separation between two medium, two light, and one medium and one light aircraft is respectively 60, 60 and 90 seconds. Consequently, following inequalities are valid and, with inequalities (4.11), they additionally cover couples M/L.



$$\sum_{i \in \mathcal{M}} \sum_{\substack{t \in T_i \\ t_0 \leqslant t < t_0 + 60}} x_{it} + \sum_{i \in \mathcal{L}} \sum_{\substack{t \in T_i \\ t_0 + 30 \leqslant t < t_0 + 90}} x_{it} \leqslant 1 \quad \forall t_0 \in \bigcup_{i \in \mathcal{F}} T_i \tag{4.12}$$

Clique based on heavy aircraft category

The minimum separation between two heavy, one heavy and one medium, one heavy and one light aircraft is respectively 90, 120 and 120 seconds. Consequently, following inequalities are valid and, with inequalities (4.11), they cover minimum separation of couples H/H, H/M, H/L.



$$\sum_{i \in \mathcal{H}} \sum_{\substack{t \in T_i \\ t_0 \leqslant t < t_0 + 90}} x_{it} + \sum_{i \in \mathcal{M} \cup \mathcal{L}} \sum_{\substack{t \in T_i \\ t_0 + 60 \leqslant t < t_0 + 120}} x_{it} \leqslant 1 \quad \forall t_0 \in \bigcup_{i \in \mathcal{F}} T_i \tag{4.13}$$

Several other inequalities can be derived from Table 4.1, but only those presented appeared to be significantly effective in our experiments. All together, inequalities (4.11-4.13) form a reformulation of separation constraints (4.8). *IP* with this reformulation is hereafter referred to as *IP cat*.

4.3.2.4 The case of infinite take-off slots

As remarked by Fahle et al. [2003], the performance of this model highly depends on the width of the take-off slots: the longer $[e_i, l_i]$ is, the more variables and constraints there are. For an arrival flight, slots are relatively reasonable because of restrictions due to the remaining fuel. On the contrary, a take-off can a priori be delayed endlessly, since waiting can be done at stand with engine off. Beasley et al. [2000] explained how to strengthen the slots without loss of optimality based on a feasible solution (e.g. obtained with a simple FCFS policy). However, the slots remain rather long whereas every aircraft takes off with only a reasonable delay in practice. Consequently, slots can potentially be further reduced without discarding all the optimal solutions.

We propose to arbitrarily truncate slots without sacrificing optimality thanks to dummy binary variables y_i which indicates if further delaying an aircraft can be more interesting. Let assume that every slot is restricted to a length τ, i.e. $T_i(\tau) = \{t \in T_i, t \leqslant e_i + \tau\}$. The formulation, which will be called the τ-restricted formulation and denoted as IP_τ, becomes

$$
\begin{aligned}
& \min && f_\tau(x,y) = \sum_{i\in\mathcal{F}} \sum_{t\in T_i(\tau)} c_{it}x_{it} + \sum_{i\in\mathcal{F}} \tilde{c}_i y_i && && (4.14)\\
(IP_\tau) \quad & s.t. && \sum_{t\in T_i(\tau)} x_{it} + y_i = 1 && \forall i \in \mathcal{F} && (4.15)\\
& && x_{it} + x_{ju} \leqslant 1 && \forall i,j \in \mathcal{F}, \forall t \in T_i(\tau), \forall u \in T_j(\tau), t \leqslant u < t + S_{ij} && (4.16)\\
& && x_{it} \in \{0,1\} && \forall i \in \mathcal{F}, \forall t \in T_i(\tau) && (4.17)\\
& && y_i \in \{0,1\} && \forall i \in \mathcal{F} && (4.18)
\end{aligned}
$$

If $\tau = +\infty$, the τ-restricted formulation simply is the initial formulation (IP), which cannot be solved with a solver if $l_i = +\infty$ since it contains an infinite number of variables.

If objective coefficients $\tilde{c}_i$ are chosen adequately, an optimal solution of the τ-restricted formulation not using any dummy variables is optimal for IP, as stated in Proposition 1.

Proposition 1. *If $\tilde{c}_i \leqslant c_{it}$ for all aircraft $i \in \mathcal{F}$ and for all $t > e_i + \tau$, an optimal solution of IP_τ not using the dummy variables is optimal for IP.*

Proof. Let f be the objective function of IP and f_τ be the objective function of the IP_τ. Let x^* be an optimal solution of IP and $(x^\tau, 0)$ be an optimal solution of IP_τ not using dummy variables. Note that x^τ is feasible for IP and $f_\tau(x^\tau, 0) = f(x^\tau)$.

Assume that $f(x^*) = f(x^\tau)$, then x^τ is optimal for IP since it is feasible. Assume now that $f(x^*) < f(x^\tau)$. For all $i \in \mathcal{F}$, let t_i^* such that $x^*_{it_i^*} = 1$, we build a feasible solution (x, y) of IP_τ from x^* as follows

$$x_{it} = \begin{cases} x^*_{it} & \text{if } t_i^* \in T_i(\tau) \\ 0 & \text{otherwise} \end{cases} \quad \text{and} \quad y_i = \begin{cases} 1 & \text{if } t_i^* \notin T_i(\tau) \\ 0 & \text{otherwise} \end{cases}$$

Let compute the value of (x, y) for IP_τ

$$f_\tau(x, y) = \sum_{\substack{i \in \mathcal{F} \\ t_i^* \in T_i(\tau)}} c_{it_i^*} + \sum_{\substack{i \in \mathcal{F} \\ t_i^* \notin T_i(\tau)}} \tilde{c}_i \leqslant \sum_{\substack{i \in \mathcal{F} \\ t_i^* \in T_i(\tau)}} c_{it_i^*} + \sum_{\substack{i \in \mathcal{F} \\ t_i^* \notin T_i(\tau)}} c_{it_i^*} = f(x^*)$$

since $\tilde{c}_i \leqslant c_{it}$ for all $i \in \mathcal{F}_{dep}$ such that $t \notin T_i(\tau)$. Thus, x is a feasible solution of IP_τ such that $f_\tau(x, y) \leqslant f(x^*) < f_\tau(x^\tau, 0)$ which is a contradiction since $(x^\tau, 0)$ is optimal for IP_τ. Therefore $f(x^*) = f(x^\tau)$ and x^τ is optimal for IP. □

If the optimal solution of the τ-restricted formulation uses dummy variables, one can try the 2τ-restricted formulation and iterate this principle until the solution is proven optimal for IP. Note that this method is not guaranteed to converge: in case of multiple optimal solutions, nothing emphasizes the solutions not using dummy variables. Therefore, it is preferable to set $\tilde{c}_i$ as high as possible (i.e. $\tilde{c}_i = \min_{t > e_i + \tau} c_{it}$ if it exists). In practice, coefficients c_{it} are often increasing with t (or at least increasing from a given t) since taking off early is generally preferred. In this case, $\tilde{c}_i = c_{i(e_i+\tau+1)}$ is advised.

4.4 Experiments

In this section, we present the set of instances and we compare the performances of both formulations (MIP and IP) and their improvements. We finally study the effect of infinite take-off slots.

4.4.1 Instances and test environment

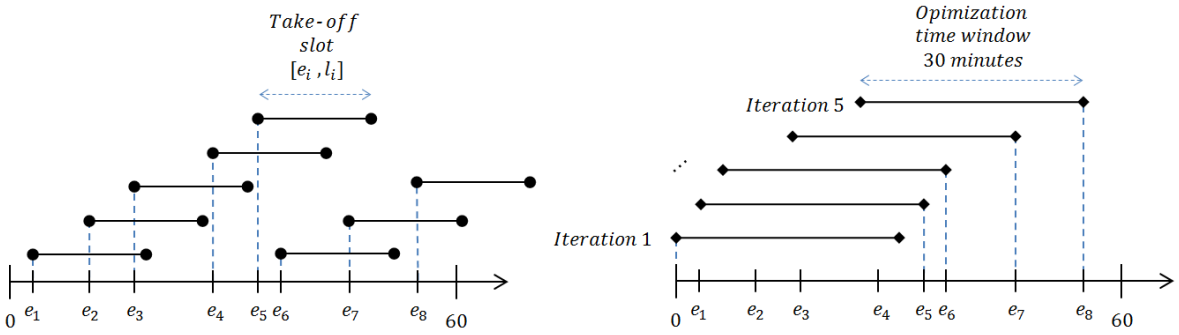
Each instance represents a departure peak of 1 hour on which ready times at the runway e_i are uniformly distributed with a precision of 1 minute. Latest take-off times l_i are set to $e_i + 30$ minutes in Subsection 4.4.2 and to $+\infty$ in Subsection 4.4.3. Instances with 40 and 60 departures are considered, which respectively represents congested and very congested situations. Table 4.1 is used as separation standards and aircraft categories are randomly distributed according to

two different mixes. The first flight mix contains respectively 5%, 90% and 5% of light, medium and heavy aircraft. It will be referred to as 5-90-5. It is close to European airports flight mix, like Copenhagen airport (CPH). The second flight mix contains respectively 10%, 70% and 20% of light, medium and heavy aircraft. It will be referred to as 10-70-20. It is more diverse and closer to bigger US airports flight mix (see e.g. Lee and Balakrishnan [2012]). We generated 10 instances for each flight mix (5-90-5 and 10-70-20) and each size (40 and 60 departures).

As explained in Section 4.1, real operations imply a sliding time window scheme. The length of the optimization time window is set to 30 minutes in our experiments.

The optimization time windows must not be confused with the take-off slots. Take-off restrictions aim to ensure a minimum of equity and prevent excessive completion times, as discussed at the end of the experiments. On the contrary, the optimization time windows define the problems to solve. Both are illustrated in Figure 4.2.

An example of the sliding window optimization is given in Figure 4.2(b). The first time window starts at time 0, instead of time $e_1 - 30$ minutes, in order to make it consistent with the other ones (i.e. it contains as much aircraft as the others). Then, the optimization window is slid over time and a problem has to be solved for every new aircraft. Note that the time axis is limited to 60 minutes since ready times are distributed over 1 hour in our instances. Nevertheless, aircraft can take-off after 60 minutes if necessary. That is why slots of flights 7 and 8 exceed 60 minutes in Figure 4.2(a).



(a) Take-off times windows

(b) Optimization times windows

Figure 4.2: Take-off times windows vs optimization times windows

All problems were solved with Cplex 12.4 through Java Concert API on a personal laptop (Intel Core i5-4300M, 2.60GHz, 4Go RAM) under Windows 7 Professional operating system. Default parameter tuning with a time limit of 5 minutes was used.

4.4.2 Comparison

Table 4.2 presents a comparison of the performances of the models on both flight mixes. Columns *Max CPU* and *Avg CPU* are the maximum and average computation times (in seconds) over every iteration of the sliding window approach. Columns *Nb TL* indicate the number of iterations that reaches the time limit of 5 minutes. When the time limit is reached, optimality is not guaranteed and Column *Gap* presents the gap to the optimal value.

Table 4.2: Comparison of MIP and IP formulations

(a) Flight mix 5-90-5

	40 departures				60 departures			
Model	Max CPU	Avg CPU	Nb TL	Gap	Max CPU	Avg CPU	Nb TL	Gap
MIP	300	13	2 / 106	0%	300	295	143 / 149	1.7%
MIP var fix	5.2	0.1	0 / 106	0%	300	117	52 / 149	0%
IP	1.7	0.4	0 / 106	0%	26	3.3	0 / 149	0%
IP Fahle	2.4	0.3	0 / 106	0%	115	9.2	0 / 149	0%
IP cat	**0.2**	**<0.1**	**0 / 106**	**0%**	**5.5**	**0.3**	**0 / 149**	**0%**

(b) Flight mix 10-70-20

	40 departures				60 departures			
Model	Max CPU	Avg CPU	Nb TL	Gap	Max CPU	Avg CPU	Nb TL	Gap
MIP	300	46	10 / 106	0%	300	300	149 / 152	4.5%
MIP var fix	295	8	0 / 106	0%	300	263	127 / 152	1.5%
IP	7.8	0.8	0 / 106	0%	300	25	5 / 152	0%
IP Fahle	15	0.9	0 / 106	0%	300	62	12 / 152	0%
IP cat	**0.8**	**0.1**	**0 / 106**	**0%**	**96**	**3.6**	**0 / 152**	**0%**

It must be remarked that discrete time formulations (*IP*, *IP Fahle* and *IP cat*) are generally more efficient and more stable than continuous time formulations (*MIP* and *MIP var fix*), which often reaches the time limit.

The experiments also reveal that the reformulation of Fahle et al. [2003] is less efficient than the original formulation of Beasley et al. [2000]. We believe that this is an effect of the solver, which already integrate reformulations based on clique. It seems that their reformulation *hides some structure* to Cplex, whose routines become less efficient.

On the contrary, our reformulation significantly reduces both the average and maximum computation times. All iterations of all instances are solved before the time limit. The maximum

computation time over instances with 60 departures for flight mix 10-70-20 is still long but increasing computational power should be enough to overcome this problem. Using a shorter time limit is an another option, but the approach becomes a heuristic.

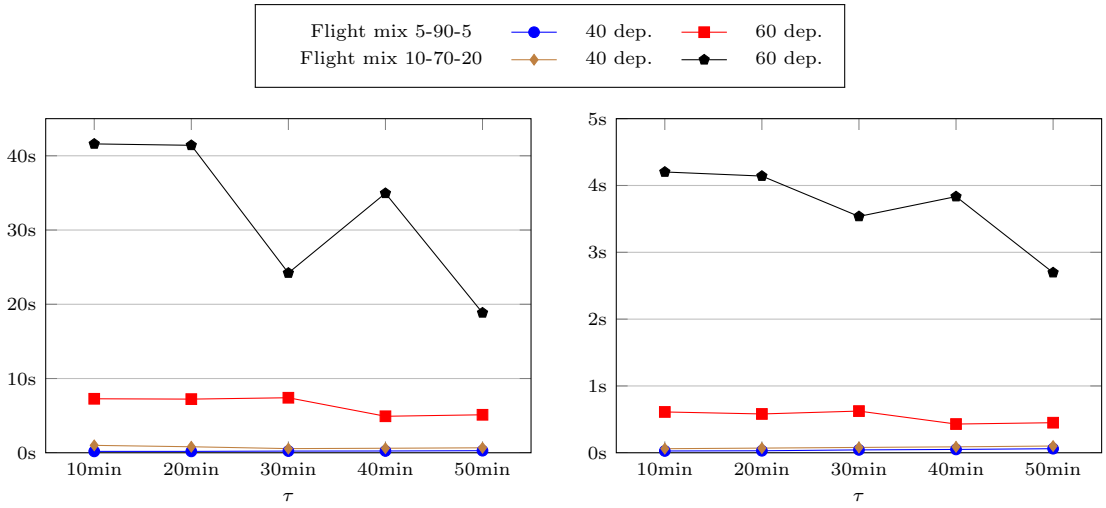
4.4.3 Infinite take-off slots

We are now interested in evaluating the behavior of our model in the case of infinite take-off slots ($l_i = +\infty$). The following study is based on the same instances, the only difference is that latest take-off times l_i are assumed to be $+\infty$. *MIP* and *MIP var fix* models are not expected to have good performances since they already offer poor performances on the bounded case (infinite slots lead to greater M_{ij}, weaker linear relaxation and thus longer computation times). Consequently, we focus on discrete time models and according to the results of the previous section, we consider only *IP cat* and apply the principle of τ-restriction presented in Section 4.3.2.4. Note that τ is reset to its original value for each iteration.

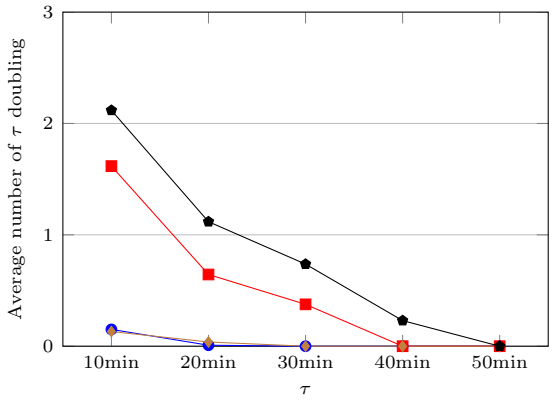
Figure 4.3 presents the performances of this principle, in terms of maximum and average computation times over all instances (Figures 4.3(a) and 4.3(b)). The average number of times that doubling τ was necessary to guarantee optimality is also presented.

The experiments reveal that whatever τ, the time limit of 5 minutes is never reached and the average computation times do not exceed 5 seconds. For instances with 40 departures (brown and blue curves), all values of τ lead to very short computation times and differences are hardly noticeable. Nevertheless, choosing appropriately τ is more important for instances with 60 departures and can lead to significant savings. The most appropriate value of τ appears to be the shortest one not requiring any doubling, i.e. respectively 40 and 50 minutes for flight mixes 5-90-5 and 10-70-20. It lessens the practical advantages of the τ-reformulation principle since it means that a preliminary study is advised to avoid doubling τ.

The black curves of Figures 4.3(a) and 4.3(b) can seem strange because setting τ to 30 minutes is more efficient than 40 minutes. Actually, it is logical for the maximum computation times since some iterations require to double τ for both values. Therefore, some iterations are finally solved with take-off slots of 60 minutes for $\tau = 30$ minutes against 80 minutes for $\tau = 40$ minutes, which implies more variables and longer computation times. In average, it appears that $\tau = 30$ minutes is slightly faster, even if more iterations require to double τ.



(a) Maximum computation times over all iterations (b) Average computation times over all iterations



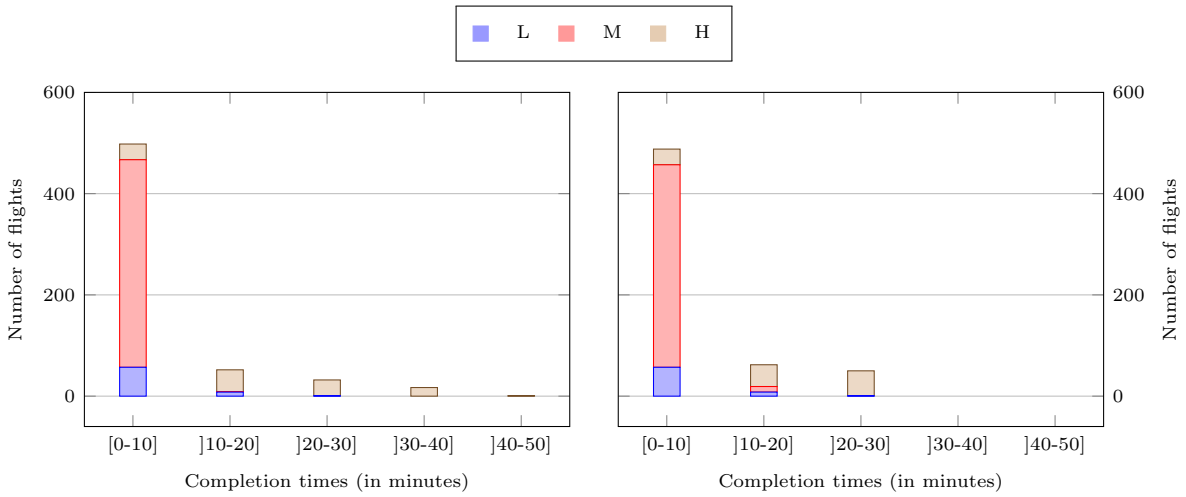
(c) Average number of times that doubling τ was required over all iterations

Figure 4.3: Performances of the τ-restriction principle

Remark on equity

These experiments reveal that limiting take-off slots is necessary to ensure a minimum of fairness. Indeed, completion times of up to 47 minutes have been observed in the case of infinite take-off slots. Figure 4.4 show the distribution of completion times over wake vortex categories on instances with 60 departures with flight mix 10-70-20. It shows that mainly heavy aircraft suffer from excessive completion times. It is actually logical since their separations are the longest (see Table 4.1). Thus, they are shifted until the end of the instance, when their longer separations do not penalize other flight categories. However, it is not acceptable for these flights and highly unfair. Limiting the take-off slots to 30 minutes allows to partially overcome this drawback, since completion times of more than 30 minutes are not possible. But Figure 4.4(b) reveals that heavy aircraft are still disadvantaged, even if the sharing is more equitable. Nevertheless,

limiting take-off slots to 30 minutes impacts efficiency and the total completion time is increased by 2.2 %, which represents 8 seconds by aircraft in average: there is a trade-off between equity and efficiency. Note that heavy aircraft generally transports more passengers, thus the total passenger completion times may be more appropriate.



(a) Infinite take-off slots

(b) 30 minutes take-off slots

Figure 4.4: Distribution of completion times for instances with 60 departures with flight mix 10-70-20

4.5 Conclusion

In this chapter, we have presented the Runway Sequencing Problem. We reviewed the literature, which highlighted that this problem had been broadly studied and that many solving techniques had already been investigated. We focused on the case of a single runway operating only take-offs and compared two models and some improvements from the literature. We propose a new formulation of separation constraints for one of them. We illustrated through computational experiments that this reformulation allows to significantly reduce computation times, making the model adapted to an industrial application while providing optimal solutions. Our experiments also reveal that fairness has to be considered to avoid excessive delays. Fairness was considered through a limitation of take-off slots, but there may be more adapted approaches, such as constrained position shifting (see Balakrishnan and Chandran [2010]), penalization of large shifting (see e.g. Atkin et al. [2010a]) or by considering the total passengers completion time. A study on fairness, including its trade-off with efficiency and a comparison of its different modeling, is one of the main perspectives of this chapter.

Anyway, these models may not be the best way to optimize the runway capacity. The main drawback of the approaches considered is that they rely on estimations of the earliest ready times at the runway (e_i). In practice, it consists in estimating the taxi times, which can lead to inefficiencies when it comes to operation. Indeed, if taxi times are underestimated, some flights will not be able to reach the runway in time for their optimized take-off times and excessive idle time will appear between take-offs, consequently wasting the runway capacity. On the contrary, if taxi times are overestimated, aircraft will reach the runway before their targeted take-off time and some fuel will be excessively burnt. Besides, aircraft will not necessarily reach the runway in the expected order, and the desired sequence may not be achievable (depending on the holding point structure).

These reasons suggests that a larger point of view is more adapted to manage the runway capacity. This idea is developed in the next chapter through an integration of the RSP with the GRP. Such an integration targets a better synchronization of ground movements and runway operations, hopefully resulting in increased runway capacity and reduced taxi times.

Chapter 5

Integration of runway sequencing and ground routing

Abstract

This chapter focuses on the management of the departure process, from push back to take-off, through an integration of the Ground Routing Problem (GRP) and the Runway Sequencing Problem (RSP). The aim is to schedule take-offs and ground movements such that the total completion and taxi times are minimized, while respecting operational requirements and considering interactions with the arrival traffic. We propose a heuristic sequential approach close to A-CDM practice and based on an innovative formulation of the RSP including the conflicts of the ramp area. We explore several directions to improve the solving of this model. We show through numerical experiments that a better integration of both problems allows to significantly reduce the total completion and taxi times. We also show that our heuristic sequential approach provides high quality solutions in reasonable computation time, unlike an exact formulation from the literature directly integrating both problems.

Keywords: Mixed integer programming, runway sequencing, ground routing

5.1 Introduction

In practice, two different managements of the departure process can be observed nowadays. The first one relies mainly on First Come First Serve practices (FCFS). Aircraft are pushed back as soon as possible and taxi to the runway where they can potentially be reordered for increasing the runway efficiency. The second practice proceeds sequentially in three steps. First, an earliest time at the runway is estimated for every aircraft through an estimation of taxi times. Then, the take-off sequence is optimized. Finally, ground movements (or only push back) are scheduled to match the predicted take-off sequence. This approach is recent and promoted by A-CDM project. It targets a better synchronization of ground movements and runway operations. It particularly allows to reduce runway queuing times through a better scheduling of push backs: aircraft can be held at their stand with engines off instead of waiting at the runway with engines on (see Chapter 3).

The aim of a better integration of the GRP and the RSP is to further improve this synchronization. The motivations are twofold: increasing runway efficiency and reducing taxi times. A-CDM approach rely on estimations of taxi times, which are particularly difficult to forecast. Nevertheless, the accuracy of these estimations is crucial. If taxi times are underestimated, aircraft are held too long, consequently creating idle time between take-offs and wasting runway capacity. On the contrary, if taxi times are overestimated, aircraft are not held long enough. Thus, an excessive queue will appear at the departure runway and fuel will be wasted. Furthermore, inaccurate taxi times can make the aircraft reach the runway in a different order than the desired one. Aircraft can be reordered at the runway but Atkin et al. [2009] show that this reordering is constrained by the holding point layout and that not all sequences are feasible. A better synchronization leads to less reordering and potentially enables more efficient sequences.

We address two main research questions. Is a better integration of runway sequencing and ground routing valuable ? How can the integrated problem be solved efficiently ? Indeed, as explained in Chapters 3 and 4, computation times are critical since the GRP and the RSP are considered on an operational horizon of typically 10 to 40 minutes rolling over time. A sliding window scheme is used in both practice and the literature.

To answer these questions, this chapter is organized as follows. A literature review is presented in Section 5.2 with a summary of our contributions. The problem is described in Section 5.3 with a formulation from the literature. A sequential approach is proposed in Section 5.4. Its principle is the same that the sequential approach of A-CDM, but a new runway sequencing

approach is presented. The different methods are tested on Copenhagen Airport (CPH) layout in Section 5.6. We conclude and highlight some directions for future works in Section 5.7.

5.2 Literature review

The works gathered in this section focus on the interactions between runway operations and ground movements. More references on isolated problems can be found in Chapters 3 and 4. The RSP is generally recognized as a more critical problem than the GRP, most of works consequently integrate ground routing perspectives in a RSP. The GRP is either partially or completely integrated. In the second case, a complete routing problem is solved but not in the first one.

Partial integration

Atkin et al. [2004, 2007, 2009] consider routing constraints in the holding point while optimizing take-off sequence on a single runway. They propose a tabu search algorithm in which feasibility in the holding point is heuristically checked when a sequence is evaluated. Infeasible solutions are discarded.

Rathinam et al. [2009] also consider holding point constraints in a take-off sequenceing problem on a single runway. Aircraft are pre-assigned to runway entry queues and a first come first serve order has to be respected inside each queue. Using this structure, the problem is efficiently solved by a Dynamic Program (DP) minimizing the total aircraft delay.

Kim et al. [2010] extend the reference Mixed Integer Program (MIP) of the RSP (see Beasley et al. [2000]). Their model aims at minimizing total emissions in the Terminal Maneuvering Area (TMA, the airspace around the airport) through runway assignment and scheduling of take-off and landing. Constant taxi times (depending on gate / runway) are assumed, benefit in fuel consumption on the ground would thus be lessened if a routing optimization approach was considered (e.g. through stand holding).

Malik et al. [2010] also assume constant taxi times. They compute a take-off sequence with the reference MIP of the RSP (see Beasley et al. [2000]). Then, they deduce spot release times by subtracting the constant taxi times, which reduces runway queuing time. Jung et al. [2010, 2011] use the same approach but replace the runway optimization by an adaptation of the DP of Rathinam et al. [2009].

Atkin et al. [2012] use a similar sequential approach but do not assume constant taxi times and consider stand contention during the design of the take-off sequence: sequences not allowing

feasible push back times are pruned. Push back times are then reoptimized in a second phase. The take-off sequence is optimized with a complex heuristic which core algorithm is a Branch & Bound algorithm (B&B) embedded in a rolling horizon scheme.

Complete integration

Deau et al. [2008, 2009] propose a sequential approach to A-CDM one. A take-off sequencing problem is solved, which provides Target Take-Off Times (TTOTs) to a routing model. The take-off sequencing algorithm is a B&B minimizing departure delay and deviation from NMOC slots [1].

Keith and Richards [2008] propose a MIP directly integrating the RSP and the GRP in a single model. A weighted combination of the makespan, the average taxi time and the average taxi distance is minimized. The model is slightly adapted by Clare and Richards [2011] to improve computational efficiency. Nevertheless, computation times are still too important: their model needs more than a minute to handle instances with 8 aircraft on a small network modeling only a runway holding point.

Lee and Balakrishnan [2012] propose a simplified version of the model of Clare and Richards [2011] by restricting the routing possibility (fix path approach, see Chapter 3). Nevertheless, computation times are still too long and they propose a sequential approach. The take-off sequencing is performed with the algorithm of Balakrishnan and Chandran [2010]. The so-obtained TTOTs are provided to a routing model minimizing a linear combination of deviation to TTOTs and the total taxi time. The take-off sequence can still be changed in their routing model and several minutes are often necessary to solve a time window.

Bosson et al. [2014a] propose a three phase decomposition for sequencing take-offs and landings and scheduling airspace movements under uncertainty. Bosson et al. [2014b] additionally involve ground movements. They consider stochastic release times and due dates (estimated time of arrival and departure) through sampling (Sample Average Approximation) embedded in a 3-phases decomposition. The approach is computationally demanding since an important number of scenarios has to be considered. An instance of less than 15 aircraft is solved in 4

1. When an air sector of an aircraft flight plan is congested or when the destination airport is facing adverse conditions, the Network Manager Operations Center (NMOC, previously called the Central Flow Management Unit, CFMU) assigns a Calculated Take-Off Time (CTOT). It generally results in delaying the take-off to prevent the situation from NMOC worse in the perturbed sector. The take-off is allowed within the interval $[CTOT - 5\ min; CTOT + 10\ min]$, called a NMOC slot. Otherwise, the aircraft has to wait for another slot from the NMOC.

minutes. Besides, routing in the ramp area is not considered.

Summary of our contributions

In this chapter, we address the integration of the RSP and the GRP on a single departure runway, with the objective of minimizing the total completion and taxi times. Optimization of landings is not included. Nevertheless, arrivals are considered in the ground routing and a conflict-free routing schedule is computed for each flight. Landing times are assumed to be fixed inputs but our model can easily be generalized and includes other runway operations.

The literature review reveals that it does not exist an efficient approach for solving this problem. The existing methods are not computationally efficient enough (Clare and Richards [2011], Lee and Balakrishnan [2012] or Bosson et al. [2014b]). Sequential approaches have been proposed but the integration is rudimentary since ground routing is not considered during the design of the take-off sequence (Deau et al. [2008, 2009]). The so-obtained sequences can be inconvenient for ground movements and more efficient global solutions may be missed. On the contrary, Atkin et al. [2012] consider interactions through stand contention but do not use it to minimize fuel consumption during the runway sequencing, potentially missing more fuel efficient solutions.

Our main contribution is to propose an efficient heuristic sequential algorithm based on an innovative formulation of the RSP including conflicts of the ramp area. Several directions are explored for improving its solving. Thought numerical experiments based on Copenhagen Airport (CPH) layout, we highlight that a better integration of both problems is highly valuable and significantly reduces the total completion and taxi times. Our approach is shown to provide high quality solutions in reasonable computation times, unlike an exact formulation from the literature directly integrating both problem in a single MIP.

5.3 The integrated runway sequencing and ground routing problem

The integrated runway sequencing and ground routing problem (I-RSP/GRP) merges the GRP and the RSP presented respectively in Chapters 3 and 4. The aim is to simultaneously schedule ground movements and runway operations such that the total completion and taxi times are minimized, while respecting operational requirements of the GRP and the RSP. A departure has to be routed from its stand to the runway and its take-off time must be optimized. An arrival

has to be routed from the runway it lands on to its stand. Optimization of landing times is not considered and they are assumed to be fixed inputs of the problem, as well as stand allocation and runway assignment. According to the conclusions of Chapter 3, aircraft are routed along a predetermined path.

Notations

The I-RSP/GRP is a generalization of the GRP and uses most of its notations. They are briefly reminded here, more details can be found in Chapter 3.

- The taxiway network is modeled as a graph $G = (V, E)$ with, V the set of nodes and E the set of edges.
- The set of arriving and departing flights is $\mathcal{F} = \mathcal{F}_{dep} \cup \mathcal{F}_{arr}$.
- A flight i must be routed from its origin o_i to its destination d_i while using a pre-determined path $P_i = (o_i, u_2, \ldots, u_{|P_i|-1}, d_i)$.
- $V_i \subset V$ and $E_i \subset E$ are the set of nodes and edges that flight i case use.
- A flight i can spend a minimum (maximum) time T^{min}_{iuv} (T^{max}_{iuv}) on edges $uv \in E_i$.
- Two flights i and j must have a minimum separation time S_{iju} at each node $u \in V_i \cap V_j$ (if i uses node u first). It includes wake vortex separation at the runway.
- $\mathcal{G} \subset \mathcal{F}_{dep} \times \mathcal{F}_{arr}$ is the set of stand blockages.
- A flight i is ready to leave its origin at time T_{o_i} and must reach its destination before L_{d_i}. L_{d_i} is necessary to prevent excessively unfair completion times (see Chapter 4).
- e_{iu} and l_{iu} are the earliest and latest time of flight i at node $u \in V_i$ (see below). Intervals $[e_{iu}, l_{iu}]$ is hereafter referred to as slot $[e_{iu}, l_{iu}]$.

Computing e_{iu} and l_{iu}

A flight i is subject to an earliest time at origin T_{o_i} and a latest time at destination L_{d_i}. Note that flight i cannot reach node $u_k \in P_i$ before

$$T_{o_i} + \sum_{k'=1}^{k-1} T^{min}_{iu_{k'}u_{k'+1}} \tag{5.1}$$

because of the speed restrictions. Also note that flight i must have left node $u_k \in P_i$ before

$$L_{d_i} - \sum_{k'=k}^{|P_i|-1} T^{min}_{iu_{k'}u_{k'+1}} \tag{5.2}$$

to meet latest time at destination L_{d_i}. Also, arrival i has to free the runway as soon as the landing is over, thus it necessarily leaves node $u_k \in P_i$ before

$$T_{o_i} + \sum_{k'=1}^{k-1} T^{max}_{iu_{k'}u_{k'+1}} \tag{5.3}$$

Therefore, flight i is subject to slot restrictions $[e_{iu}, l_{iu}]$ at every node u of its path, where

$$e_{iu} = (5.1) \quad \forall i \in \mathcal{F}, \forall u \in V_i \quad \text{and} \quad l_{iu} = \begin{cases} (5.2) & \forall i \in \mathcal{F}_{dep}, \forall u \in V_i \\ \min\{(5.2), (5.3)\} & \forall i \in \mathcal{F}_{arr}, \forall u \in V_i \end{cases}$$

For a departure i, note that $e_{io_i} = T_{o_i}$ and $l_{id_i} = L_{d_i}$. For an arrival i, note that $e_{io_i} = l_{io_i} = T_{o_i}$ and that $l_{id_i} \leqslant L_{d_i}$.

Direct formulation

A direct way to integrate the GRP and the RSP is to gather both problems in a single MIP, which consists in relaxing runway sequencing constraints (3.7) in the formulation of the GRP presented in Chapter 3. The so-obtained formulation is presented in MIP 5.1. It uses following variables

- $t_{iu} \in [e_{iu}, l_{iu}]$: the time when flight i reaches node $u \in V_i$.
- $z_{iju} = 1$ if flight i arrives before flight j at node $u \in V_i \cap V_j$, 0 otherwise.

$$\min \quad \sum_{i \in \mathcal{F}} c_i^{taxi}(t_{id_i} - t_{io_i}) + \sum_{i \in \mathcal{F}} c_i^{ct}(t_{id_i} - T_{o_i}) \tag{5.4}$$

$$e_{iu} \leqslant t_{iu} \leqslant l_{iu} \qquad \forall i \in \mathcal{F}, \forall u \in V_i \tag{5.5}$$

$$t_{io_i} \leqslant t_{jd_j} \qquad \forall (i,j) \in \mathcal{G} \tag{5.6}$$

$$T^{min}_{iuv} \leqslant t_{iv} - t_{iu} \leqslant T^{max}_{iuv} \qquad \forall i \in \mathcal{F}, \forall uv \in E_i \tag{5.7}$$

$$z_{iju} + z_{jiu} = 1 \qquad \forall i,j \in \mathcal{F}, \forall u \in V_i \cap V_j \tag{5.8}$$

$$z_{iju} = z_{ijv} \qquad \forall i,j \in \mathcal{F}, \forall uv \in E_i \cap E_j \tag{5.9}$$

$$t_{ju} \geqslant t_{iu} + S_{iju} - M_{iju}(1 - z_{iju}) \qquad \forall i,j \in \mathcal{F}, \forall u \in V_i \cap V_j \tag{5.10}$$

$$z_{iju} \in \{0,1\} \qquad \forall i \neq j \in \mathcal{F}, \forall u \in V_i \cap V_j \tag{5.11}$$

MIP 5.1: Complete and direct integration of runway sequencing and ground routing

Such a formulation has been proposed by Lee and Balakrishnan [2012] and is a particular case of the formulation of Clare and Richards [2011].

Most of constraints of MIP 5.1 comes from the GRP and the reader is referred to Chapter 3 for more information. Formula (5.4) is the objective function gathering the total completion and taxi times in a linear combination, where $c_i^{ct}, c_i^{taxi} \geqslant 0$. Constraints (5.5) ensure the respect of the slot at each node. Note that it forces arrival i to start taxiing as soon as the landing is completer since $e_{io_i} = l_{io_i} = T_{o_i}$. Constraints (5.6) prevent stand blockages. Constraints (5.7) ensure the respect of speed limitations. Constraints (5.8) ensure that either i uses node u before j or the opposite. It ensures the definition of variables z_{iju}. Constraints (5.9) prevent overtake and head-on conflicts on a single edge. Constraints (5.10) ensure the separation of aircraft at every node of the taxiway network, including the runway, where M_{iju} can be defined as $l_{iu} + S_{iju} - e_{ju}$.

Note that for two flights i and j, if $l_{iu} \leqslant e_{ju}$ at node $u \in V_i \cap V_j$, then z_{iju} have to be equal to 1. These constraints can be added to strengthen the model and one of the two separation constraints (5.10) can be removed. Furthermore, if $l_{iu} + S_{iju} \leqslant e_{ju}$, then the separation is naturally forced. The associated variables z_{iju}, z_{jiu} and constraints involving them can be removed.

MIP 5.1 is hereafter referred to as *Full* in Section 5.6. Note that it is an exact formulation of the I-RSP/GRP. One may remark that it is not only a generalization of the GRP (by relaxing runway sequencing constraints (3.7)) but also of the RSP since it extends the continuous time MIP presented in Chapter 4 (see Section 4.3.1). We saw in Section 4.4 that this model already offers poor performances. The main source of inefficiencies is a weak linear relaxation because of the so-called *big M constraints* modeling the separation requirements (Constraints (5.10)). M_{iju} cannot be further reduced without loss of generality and we did not succeed in strengthening this model.

Consequently, we propose a heuristic approach in the next section.

5.4 Sequential approach

The principle of the sequential approach is explained by Lee and Balakrishnan [2012], it proceeds in three steps.

1. Estimate the arrival time at the runway for departures.
2. Sequence take-offs using these ready times.
3. Route aircraft in the taxiway network based on the take-off sequence of Step 2.

This sequential scheme is convenient for airports since it decomposes the whole problem in three modules, which can be changed independently from each other. Consequently, existing systems can be reused. Most of big airports already has a runway optimization advisory system. Besides, it respects current airport organization since an Air Traffic Controller (ATC) manages the taxiway network (the ground controller) and another ATC manages the runway (the runway controller). Finally, this sequential scheme is close to A-CDM approach, which means that it is already accepted in the industry.

Three different approaches are considered for Step 2. They will be compared in Section 5.6.

(a) *FCFS*: aircraft are sequenced according to the First Come First Serve order on the estimation of arrival time at the runway provided by Step 1.

(b) *IP*: aircraft are sequenced with the discrete time IP formulation of the RSP presented in Chapter 4 (see Section 4.3.2).

(c) *IP Ramp*: aircraft are sequenced with a new formulation of the RSP presented in the next section. This model is innovative and is one of the main contributions of this chapter.

Step 3 will be solved with the single path formulation of the GRP presented in Chapter 3 (MIP 3.1). It provides a detailed schedule of ground movements, i.e. a time at each node for every aircraft. The real interest of Step 3 is that it provides an accurate estimation of push back and take-off times, which can be different from the take-off times computed in Step 2 when building the sequence.

5.5 Integer program considering the conflicts of the ramp area

In this section, we present a new IP formulation of the RSP that considers conflicts of the ramp area. Chapter 3 reveals that the runway and the ramp area are the two main bottlenecks of the ground traffic in CPH, especially during departure peaks. Based on this result, our approach consists in scheduling take-off and push back times simultaneously in Step 2. Park-in times are also scheduled for arrivals. This formulation is a partial integration of the GRP and the RSP since the main conflicts of the routing are taken into account during the design of the take-off sequence. It aims at capturing the main fluctuations of the taxi times, which are thus simply estimated by the unimpeded taxi times.

5.5.1 Formulation

The new formulation generalizes the IP formulation of the RSP presented in Chapter 4. Variables are added to schedule push back and park-in times. The conflicts of the ramp area are modeled by separation constraints on the push back and parking times.

This idea was originally proposed by Atkin et al. [2012], they called it stand contention. However, they did not propose an IP formulation and use a complex heuristic based on a B&B algorithm (see Section 5.2). Besides, they do not explain how to compute the stand contention. Section 5.5.4 presents two simple algorithms for identifying conflicts of the ramp area, they compute the minimum separation between push back and park-in times. A flight is said to be in conflict with another one (in the ramp area) if its push back / park-in prevents the other one from pushing back / parking in.

Main notations

Our model needs additional notations.

- $EXOT_i$ is the estimated taxi-out time of departure i is used in Step 1 for estimating the earliest time at the runway. Similarly, $EXIT_i$ is the estimated taxi-in time of arrival i.
- T_i^r is the set of possible runway times for flight i, i.e. take-off times if i is a departure and landing times if i is an arrival. It is a discretization of slots $[e_{id_i}, l_{id_i}]$ for departures and $[e_{io_i}, l_{io_i}]$ for arrivals. Note that interval $[e_{io_i}, l_{io_i}]$ is actually a singleton for arrivals.
- T_i^s is the set of possible stand times, i.e. push back times if i is a departure and park-in times if i is an arrival. It is a discretization of slots $[e_{io_i}, l_{io_i}]$ for departures and $[e_{id_i}, l_{id_i}]$ for arrivals.
- S_{ij} is the minimum runway separation time between flight i and j if i uses the runway before j.
- To prevent conflicts in the ramp area between two flights i and j, we define $a_{ij} \leqslant b_{ij} \in \mathbb{R}$ such that if i pushes back / parks in at time t, then j cannot push back / park in during the interval $[t + a_{ij}, t + b_{ij}]$ (see Section 5.5.4 for more details).

Variables and objective coefficients

Our model uses following variables.

- $x_{it}^r = 1$ if flight $i \in \mathcal{F}$ takes off / lands on at time $t \in T_i^r$.
- $x_{it}^s = 1$ if flight $i \in \mathcal{F}$ pushes back / parks in at time $t \in T_i^s$.

As a reminder, objective function (5.4) of MIP 5.1 is

$$\sum_{i \in \mathcal{F}} c_i^{taxi}(t_{id_i} - t_{io_i}) + \sum_{i \in \mathcal{F}} c_i^{ct}(t_{id_i} - T_{o_i})$$

Note that for a departure i, time at destination t_{id_i} is equivalent to $\sum_{t \in T_i^r} tx_{it}^r$ and time at origin t_{io_i} is equivalent to $\sum_{t \in T_i^s} tx_{it}^s$. Objective coefficients are thus defined as follows

$$\begin{aligned} c_{it}^r &= tc_i^{ct} + tc_i^{taxi} \quad && \forall i \in \mathcal{F}_{dep}, \forall t \in T_i^r \\ c_{it}^s &= -T_{o_i} c_i^{ct} - tc_i^{taxi} \quad && \forall i \in \mathcal{F}_{dep}, \forall t \in T_i^s \end{aligned}$$

Similarly, for arrivals, objective coefficients are defined as follows

$$\begin{aligned} c_{it}^r &= -T_{o_i} c_i^{ct} - tc_i^{taxi} \quad && \forall i \in \mathcal{F}_{arr}, \forall t \in T_i^r \\ c_{it}^s &= tc_i^{ct} + tc_i^{taxi} \quad && \forall i \in \mathcal{F}_{arr}, \forall t \in T_i^s \end{aligned}$$

The model

The problem can be formulated as MIP 5.2.

$$\min \quad \sum_{i \in \mathcal{F}} \sum_{t \in T_i^r} c_{it}^r x_{it}^r + \sum_{i \in \mathcal{F}} \sum_{t \in T_i^s} c_{it}^s x_{it}^s \tag{5.12}$$

$$s.t. \quad \sum_{t \in T_i^r} x_{it}^r = 1 \qquad \forall i \in \mathcal{F} \tag{5.13}$$

$$\sum_{t \in T_i^s} x_{it}^s = 1 \qquad \forall i \in \mathcal{F} \tag{5.14}$$

$$\sum_{t \in T_i^r} tx_{it}^r - \sum_{t \in T_i^s} tx_{it}^s \geqslant EXOT_i \quad \forall i \in \mathcal{F}_{dep} \tag{5.15}$$

$$\sum_{t \in T_i^s} tx_{it}^s - \sum_{t \in T_i^r} tx_{it}^r \geqslant EXIT_i \quad \forall i \in \mathcal{F}_{arr} \tag{5.16}$$

$$\sum_{t \in T_i^s} tx_{it}^s \leqslant \sum_{t \in T_j^s} tx_{jt}^s \qquad \forall (i,j) \in \mathcal{G} \tag{5.17}$$

$$x_{it}^r + x_{ju}^r \leqslant 1 \qquad \forall i,j \in \mathcal{F}_{dep}, \forall t \in T_i^r, \forall u \in T_j^r, t \leqslant u < t + S_{ij} \tag{5.18}$$

$$x_{it}^s + x_{ju}^s \leqslant 1 \qquad \forall i,j \in \mathcal{F}, \forall t \in T_i^s, \forall u \in T_j^s \cap [t + a_{ij}, t + b_{ij}] \tag{5.19}$$

$$x_{it}^r \in \{0,1\} \qquad \forall i \in \mathcal{F}, \forall t \in T_i^r \tag{5.20}$$

$$x_{it}^s \in \{0,1\} \qquad \forall i \in \mathcal{F}, \forall t \in T_i^s \tag{5.21}$$

MIP 5.2: Runway sequencing model integrating ramp congestion

Constraints (5.13) and (5.14) ensure that one and only one runway time and stand time is assigned to every flight. Constraints (5.15) and (5.16) ensure that minimum taxi-out and taxi-in times are respected. Constraints (5.17) prevent stand blockages. Constraints (5.18) and (5.19) ensure that minimum runway and ramp separation requirements are respected. Other constraints ensure the definition of the variables.

Note that according to our assumptions, T_i^r is a singleton if i is an arrival, variable x_{it}^r and constraints involving it can thus be removed from the model (for arrivals only). Similarly, if departure i pushes back before the current time window of the sliding window scheme, T_i^s becomes a singleton. Departure i can be removed from the instance once its take-off is completed before the current time window, i.e. when separations with other flights are necessarily respected.

Optimization of landings can be easily added to this formulation by considering a larger set and by adding runway separation constraints similar to (5.18).

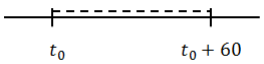
5.5.2 Constraints reformulation and filtering

Constraints have been expressed in a natural way, which is not necessarily strong. We propose reformulations that will be shown to improve computation times in Section 5.6. We first recall the reformulation of runway separation constraints (5.18) presented in Chapter 4. The second type of reformulations focuses on constraints involving time in the coefficients (Constraints (5.15)-(5.17)). Finally, we propose a heuristic filtering of ramp separation constraints (5.19).

Reformulation of runway separation constraints

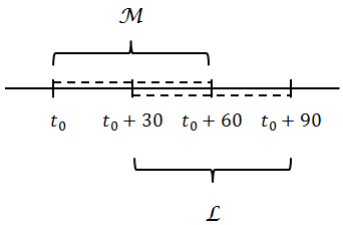
Runway separation constraints (5.18) can be reformulated using clique constraints based on wake vortex categories presented in Section 4.3.2. They are recalled here and adapted to the new notations.

– Clique of 60 seconds



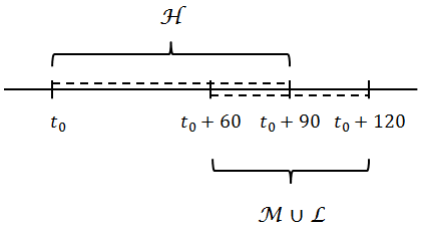
$$\sum_{i \in \mathcal{F}} \sum_{\substack{t \in T_i^r \\ t_0 \leqslant t < t_0+60}} x_{it}^r \leqslant 1 \quad \forall t_0 \in \bigcup_{i \in \mathcal{F}} T_i^r \tag{5.22}$$

– Clique based on medium aircraft category



$$\sum_{i \in \mathcal{M}} \sum_{\substack{t \in T_i^r \\ t_0 \leqslant t < t_0+60}} x_{it}^r + \sum_{i \in \mathcal{L}} \sum_{\substack{t \in T_i^r \\ t_0+30 \leqslant t < t_0+90}} x_{it}^r \leqslant 1 \quad \forall t_0 \in \bigcup_{i \in \mathcal{F}} T_i^r \tag{5.23}$$

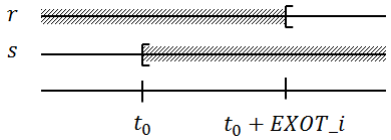
– Clique based on heavy aircraft category



$$\sum_{i\in\mathcal{H}} \sum_{\substack{t\in T_i^r \\ t_0\leqslant t<t_0+90}} x_{it}^r + \sum_{i\in\mathcal{M}\cup\mathcal{L}} \sum_{\substack{t\in T_i^r \\ t_0+60\leqslant t<t_0+120}} x_{it}^r \leqslant 1 \quad \forall t_0 \in \bigcup_{i\in\mathcal{F}} T_i^r \tag{5.24}$$

Reformulation of taxi time and stand blockages constraints

Taxi-out time constraints (5.15) can be reformulated as follows



$$\sum_{\substack{t\in T_i^s \\ t\geqslant t_0}} x_{it}^s + \sum_{\substack{t\in T_i^r \\ t<t_0+EXOT_i}} x_{it}^r \leqslant 1 \quad \forall i \in \mathcal{F}_{dep}, \forall t_0 \in T_i^s \tag{5.25}$$

i.e. if departure i pushes back after t_0 then it cannot take off before t_0+EXOT_i and reciprocally. The interest of such a reformulation can be illustrated with a simple example with only one departure: assume a flight i such that $EXOT_i = 600$, $T_i^s = \{0, 30, 60\}$ and $T_i^r = \{600, 630, 660\}$. The original formulation of its taxi-out time constraint is

$$600x_{i600}^r + 630x_{i630}^r + 660x_{i660}^r - 0x_{i0}^s - 30x_{i30}^s - 60x_{i60}^s \geqslant 600$$

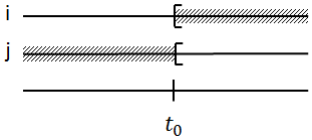
The reformulation of its taxi-out time constraint is

$$(\star) \begin{cases} x_{i0}^s + x_{i30}^s + x_{i60}^s \leqslant 1 & \text{for } t_0 = 0 \\ x_{i30}^s + x_{i60}^s + x_{i600}^r \leqslant 1 & \text{for } t_0 = 30 \\ x_{i60}^s + x_{i600}^r + x_{i630}^r \leqslant 1 & \text{for } t_0 = 60 \end{cases}$$

It can be remarked that the fractional solution $x_{i0}^s = 0.5$, $x_{i60}^s = 0.5$ and $x_{i630}^r = 1$ ($x_{i30}^s = x_{i600}^r = x_{i660}^s = 0$) respects the original formulation but violates constraints ($\star$) for $t_0 = 60$.
The same reformulation can be done for taxi-in time constraints (5.16).

$$\sum_{\substack{t\in T_i^r \\ t\geqslant t_0}} x_{it}^r + \sum_{\substack{t\in T_i^s \\ t<t_0+EXIT_i}} x_{it}^s \leqslant 1 \quad \forall i \in \mathcal{F}_{arr}, \forall t_0 \in T_i^r \tag{5.26}$$

The same idea can be used for reformulating stand blockages constraints (5.17)



$$\sum_{\substack{t\in T_i^s \\ t\geqslant t_0}} x_{it}^s + \sum_{\substack{t\in T_j^s \\ t<t_0}} x_{jt}^s \leqslant 1 \quad \forall (i,j) \in \mathcal{G}, \forall t_0 \in T_i^s \cap T_j^s \tag{5.27}$$

i.e. if departure i pushes back after t_0, then arrival j cannot park-in before t_0 and reciprocally.

Filtering ramp separation constraints

Not all the ramp separation constraints (5.19) may be necessary and some may be naturally respected. Filtering them allows to reduce the size of the model and computation times can potentially be improved.

In Chapter 3, we saw that aircraft taxi at the maximum speed most of the time. We rely on this finding to heuristically filter the ramp separation constraints that are shorter than the runway separation constraints. For example, let consider two medium aircraft. Their runway separation is 60 seconds (see Table 5.1). If they both taxi at maximal speed, they will be separated by 60 seconds on taxiways (assuming the same maximal speed for both aircraft). Consequently, ramp separation constraints shorter than 60 seconds are naturally respected and can be filtered.

This principle is heuristic since slower speeds may be necessary because of conflicts with other aircraft. However, the sequential approach is already heuristic, so its generality is not lessened. This filtering appears to be efficient in many cases without degrading the quality of the final solution, i.e. after the GRP of Step 3 which, anyway ensure the respect of these separations.

5.5.3 Using the dynamic nature of the problem

We can exploit the fact that the problem is addressed continuously through a sliding time window approach. It seems reasonable to think that push backs scheduled in the near future during the previous time window will still be scheduled in the near future in the current one. This idea can be used to heuristically reduce slots $[e_{iu}, l_{iu}]$ as follows.

Let $i \in \mathcal{F}$, a flight that was present in the previous time window. Let $t^*_{io_i}$ and $t^*_{id_i}$ the time at origin and destination that was computed. We reduce slots $[e_{io_i}, l_{io_i}]$ and $[e_{id_i}, l_{id_i}]$ as follows

$$e_{io_i} \leftarrow \max\{e_{io_i}, t^*_{io_i} - \Delta_i\} \qquad e_{id_i} \leftarrow \max\{e_{id_i}, t^*_{id_i} - \Delta_i\}$$

$$l_{io_i} \leftarrow \min\{l_{io_i}, t^*_{io_i} + \Delta_i\} \qquad l_{id_i} \leftarrow \min\{e_{id_i}, t^*_{id_i} + \Delta_i\}$$

where, assuming that current time window starts at time 0 (within a translation),

$$\Delta_i = \begin{cases} 1 \text{ minutes} & \text{if } 0 \leqslant t^{s^*}_{i\,k-1} < 5 \text{ minutes} \\ 3 \text{ minutes} & \text{if } 5 \leqslant t^{s^*}_{i\,k-1} < 10 \text{ minutes} \\ 5 \text{ minutes} & \text{if } 10 \leqslant t^{s^*}_{i\,k-1} < 15 \text{ minutes} \\ 10 \text{ minutes} & \text{if } 15 \leqslant t^{s^*}_{i\,k-1} < 20 \text{ minutes} \\ +\infty & \text{otherwise} \end{cases}$$

This principle, which is illustrated in Figure 5.1, remains in almost fixing the flights with the earliest scheduled push back / park-in while giving more flexibility to further flights. Note that take-off time windows are also reduced according to the proximity of the scheduled push back time. It allows to reduce the number of variables and constraints but the quality of the solution can be deteriorated.

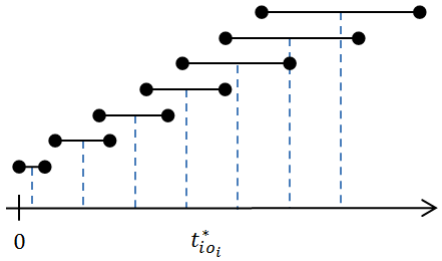
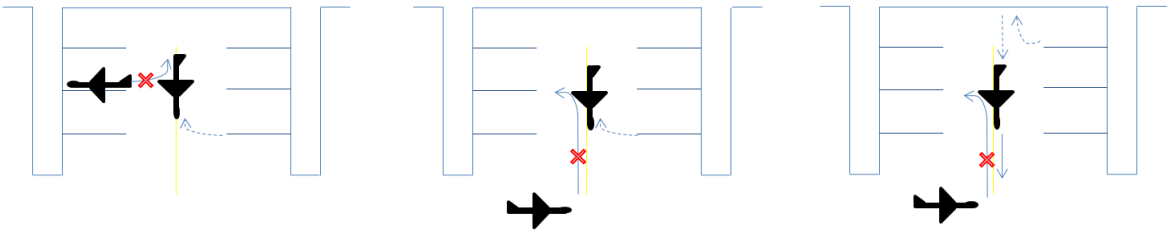


Figure 5.1: Using previous solution to reduce intervals $[e_{iu}, l_{iu}]$

5.5.4 Conflicts of the ramp area

There are many different conflicts in the ramp area. Among them, the push back conflicts and the head-on conflicts between departures and arrivals are the most common ones (see Figure 5.2). Ramp area configurations are very diverse and finding a literal formula for every case is not convenient. Consequently, we propose two efficient algorithms for computing conflicting intervals.



(a) Push back conflicts (b) Head-on conflict

Figure 5.2: Common conflicts of the ramp area

Consider two flights i and j, we will determine if they are in conflict in the ramp area, i.e. compete for the same taxiway segment(s) or node(s). In that case, we will determine a conflicting interval, i.e. two bounds a_{ij} and b_{ij} such that if i pushes back / parks in at time t then j cannot push back / park in during the interval $[t + a_{ij}, t + b_{ij}]$. The interval must be as large as possible in order to capture the whole conflict (the largest in the sense of the inclusion).

It must be remarked that a_{ij} and b_{ij} do not depend on time t since separations S_{iju} neither. Hence, we assume that i pushes back / parks in at time 0 in what follows.

Algorithms 1 and 2 computes respectively a_{ij} and b_{ij}. They use four methods:

- $buildSchedule(i,t)$ computes the shortest schedule in the ramp area such that flight i pushes back / parks in at time t. More precisely, for a departure i, let $(o_i, u_2, .., u_{k_i})$ the path of i in the ramp area (i.e. u_{k_i} is the last node of P_i of the ramp area), it returns the schedule $(t_{io_i}, t_{iu_2}, .., t_{iu_{k_i}})$ such that

$$t_{io_i} = t$$
$$t_{iu_k} = t_{iu_{k-1}} + T^{min}_{iu_{k-1}u_k} \quad \forall k = 2, .., k_i$$

  The principle is the same for arrivals but backwards.

  Note that the duration of the shortest schedule does not depend t since minimum travel times T^{min}_{iuv} neither. This duration is hereafter referred to as $duration(i)$.
- $isFeasible(schedule_i, schedule_j)$ indicates if $schedule_i$ and $schedule_j$ are compatible in the ramp area, i.e. respect minimum separation at each node and overtake and head-on constraints on each edge of the ramp area.
- $lb(i,j)$ returns a lower bound of a_{ij}. For example, $-(duration(i)+duration(j)+\sum_u Sjiu)$ fits: it is clear that if j pushes back / parks at this time, it uses the ramp area first and does not interfere with i (pushing back / parking in at time 0). Note that any other lower bound can be used.
- Similarly, $ub(i,j)$ is an upper bound of b_{ij}, e.g. $duration(i) + duration(j) + \sum_u Siju$.

The principle of Algorithm 1 is to look for the earliest time of conflict a_{ij} iteratively from $lb(i,j)$ to $ub(i,j)$. If upper bound $ub(i,j)$ is reached, flights i and j are not in conflict in the ramp area. In the other case, Algorithm 2 is called for computing the latest time of conflict b_{ij}. Its principle is similar to Algorithm 1.

Note that depending on the structure of the ramp area, there could be compatible times in the interval $[t+a_{ij}, t+b_{ij}]$, but they would require an accurate and tight synchronization, which is not robust and explain why we do not consider them. Algorithm 1 and 2 can be adapted to find these times and compute multiple conflicting intervals.

Algorithm 1: Computing a_{ij}

Data: flights i and j

Result: a_{ij}

$schedule_i = buildSchedule(i, 0)$

$a_{ij} = lb(i, j)$

$schedule_j = buildSchedule(j, a_{ij})$

while $isFeasible(schedule_i, schedule_j)$ and $a_{ij} \leqslant ub(i, j)$ *do*

  $a_{ij} = a_{ij} + 1$

  $schedule_j = buildSchedule(j, a_{ij})$

end

Algorithm 2: Computing b_{ij}

Data: flights i and j, a_{ij}

Result: b_{ij}

$schedule_i = buildSchedule(i, 0)$

$b_{ij} = ub(i, j)$

$schedule_j = buildSchedule(j, b_{ij})$

while $isFeasible(schedule_i, schedule_j)$ and $b_{ij} \geqslant a_{ij}$ *do*

  $b_{ij} = b_{ij} - 1$

  $schedule_j = buildSchedule(j, b_{ij})$

end

5.6 Experiments

5.6.1 Instances and test environment

To test the behavior of the models, we generate random traffic in the layout of Copenhagen airport (CPH) presented in Chapter 3 (see Section 3.4). Instances of 1 hour with 40 or 60 departures and 0 or 20 arrivals are generated to represent intense and very intense departure peaks with or without arrivals. Ready times are uniformly distributed with a precision of 1 minute. Aircraft type are randomly generated according to the flight mixes of Chapter 4: 5-90-5 and 10-70-20 (5%, 90%, 5% and 10%, 70%, 20% of light, medium and heavy aircraft). The minimum take-off separation times are also the same, they are reminded in Table 5.1.

Stands are randomly allocated for departures first and such that two departures do not use the same stand (which is rarely the case on one hour). Then stands are randomly allocated to arrivals such that, on each stand, there are at least 35 minutes between an arrival and a

consecutive departure to respect minimum turnaround times.

It results in 8 test sets, 10 instances are generated for each one.

Time [s]		Trailing aircraft H	M	L
Leading aircraft	H	90	120	120
	M	60	60	90
	L	60	60	60

Table 5.1: Minimum take-off separations time (H = heavy, M = medium and L = light)

As in Chapters 3 and 4, real conditions are simulated with a sliding time window approach. Its length is set to 20 minutes. Coefficients c_i^{ct} and c_i^{taxi} are respectively set to 2 and 1 to emphasize efficiency against taxi times, while not making taxi times negligible. The same coefficients are used in the GRP of Step 3 of the sequential approaches for the sake of consistency (with $c_i^{OTP} = c_i^{delay} = 0$ according to the conclusions of Chapter 3).

For the sequential approaches, it must be remembered that the results presented are the final results of the sequential scheme, i.e. the results of the GRP solved in Step 3.

All MIPs were solved with Cplex 12.4 through Java Concert API on a personal computer (Intel Core i5-2400 3.10 Ghz, 4Go RAM) under Ubuntu 12.04 LTS. Default parameter tuning was used with a time limit of 5 minutes.

5.6.2 Algorithm comparison

In this section, we compare the performances of the different approaches presented previously. As a reminder, *Full* refers to MIP 5.1 (Equations (5.4)-(5.11)) which directly and completely integrates both problems. *FCFS*, *IP*, *IP Ramp* refer to the sequential approach presented in Section 5.4 where the take-off sequence (Step 2) is computed with respectively a simple FCFS algorithm, the IP considering only the runway presented in Chapter 4 (Equations (4.6),(4.7),(4.9) and (4.11)-(4.13)) and our new IP integrating the conflicts of the ramp area (Equations (5.12)-(5.17), (5.19)-(5.24)). Note that the reformulation of runway separation constraints is included in *IP* and *IP Ramp* since its efficiency has been proven in Chapter 4. Other reformulations are the subject of Section 5.6.3.1 and are not used here.

Tables 5.2 and 5.3 compares the performances of the algorithms. Table 5.2 presents the average and maximum gaps to the best known solutions over all instances. Note that best

		Flight mix 5-90-5				Flight mix 10-70-20			
		FCFS	IP	IP Ramp	Full	FCFS	IP	IP Ramp	Full
40 dep.	Avg	6.46 %	4.97 %	0.51 %	**0 %**	7.02 %	3.63 %	0.16 %	**0 %**
0 arr.	Max	12.9 %	9.31 %	0.90 %	**0 %**	13.9 %	9.37 %	0.63 %	**0 %**
40 dep.	Avg	6.38 %	5.19 %	0.60 %	**0.06 %**	6.71 %	3.20 %	0.92 %	**0 %**
20 arr.	Max	12.2 %	13.9 %	1.45 %	**0.55 %**	17.2 %	6.09 %	3.15 %	**0 %**
60 dep.	Avg	11.9 %	6.45 %	0.46 %	**0.10 %**	20.5 %	9.84 %	**0.18 %**	0.89 %
0 arr.	Max	18.3 %	11.8 %	1.17 %	**0.91 %**	29.2 %	21.1 %	**1.11 %**	2.17 %
60 dep.	Avg	14.7 %	8.63 %	0.68 %	**0.23 %**	19.2 %	9.54 %	1.11 %	**0.97 %**
20 arr.	Max	26.0 %	14.5 %	3.33 %	**0.96 %**	27.0 %	16.6 %	2.35 %	**2.15 %**

Max= maximum gap to the best known solutions over all instances

Avg= average gap to the best known solutions over all instances

Table 5.2: Comparison of the gap to the best known solutions of the different approaches

		Flight mix 5-90-5				Flight mix 10-70-20			
		FCFS	IP	IP Ramp	Full	FCFS	IP	IP Ramp	Full
40 dep. 0 arr.	Max	<0.1s	0.2s	24s	300s	<0.1s	1.0s	4.0s	300s
	Avg	<0.1s	<0.1s	0.8s	6.8s	<0.1s	<0.1s	0.4s	20s
	% solved	-	-	-	99.3 %	-	-	-	95.3 %
40 dep. 20 arr.	Max	<0.1s	0.3s	46s	300s	<0.1s	0.4s	8.5s	300s
	Avg	<0.1s	<0.1s	1.7s	44s	<0.1s	<0.1s	0.9s	23s
	% solved	-	-	-	87.5 %	-	-	-	96.0 %
60 dep. 0 arr.	Max	<0.1s	1.8s	300s	300s	<0.1s	5.5s	300s	300s
	Avg	<0.1s	0.1s	12s	262s	<0.1s	0.8s	41s	267s
	% solved	-	-	99.0 %	15.9 %	-	-	94.1 %	12.2 %
60 dep. 20 arr.	Max	<0.1s	3.2s	107s	300s	<0.1s	5.3s	300s	300s
	Avg	<0.1s	0.2s	8.6s	270s	<0.1s	0.9s	80s	292s
	% solved	-	-	-	11.3 %	-	-	91.8 %	3.6 %

Max= maximum computation time over all iterations

Avg= average computation time over all iterations

% solved= percentage of iterations solved before the time limit

Table 5.3: Comparison of the computation times of the different approaches

known solutions are not necessarily provided by *Full* because of the time limit. Indeed, when the time limit is reached, the solution provided is not necessarily optimal and *IP Ramp* sometimes provides better solutions. Also, because of the sliding window, even an exact formulation is not

guaranteed to find the best final solution (over all the instance). Table 5.3 presents the average and maximum computation times over all iterations. It also presents the percentage of instances that were solved before the time limit. Two main results come out from these tables.

Firstly, Table 5.2 shows that *Full* and *IP Ramp* perform significantly better than *IP* and *FCFS*. It leads to conclude that a better integration of ground routing and runway sequencing is valuable. Nevertheless, Table 5.3 reveals that the computation times of *Full* and *IP Ramp* are long and do not match the requirements of an industrial application for all instances. *Full* particularly suffers from long computation times: the time limit is reached very often on instances with 60 departures.

Secondly, *IP Ramp* offers high quality solutions. It even finds better solutions for some instances with 60 departures. It highlights that most of the conflicts of the routing are successfully captured in our runway sequencing approach. Furthermore, it performs highly better in terms of computation times. Average computation times are short enough on instances with 40 departures, but they are too long on instances with 60 departures, particularly for flight mix 10-70-20. Moreover, maximum computation times are not acceptable for instances with 60 departures and the time limit is sometimes reached.

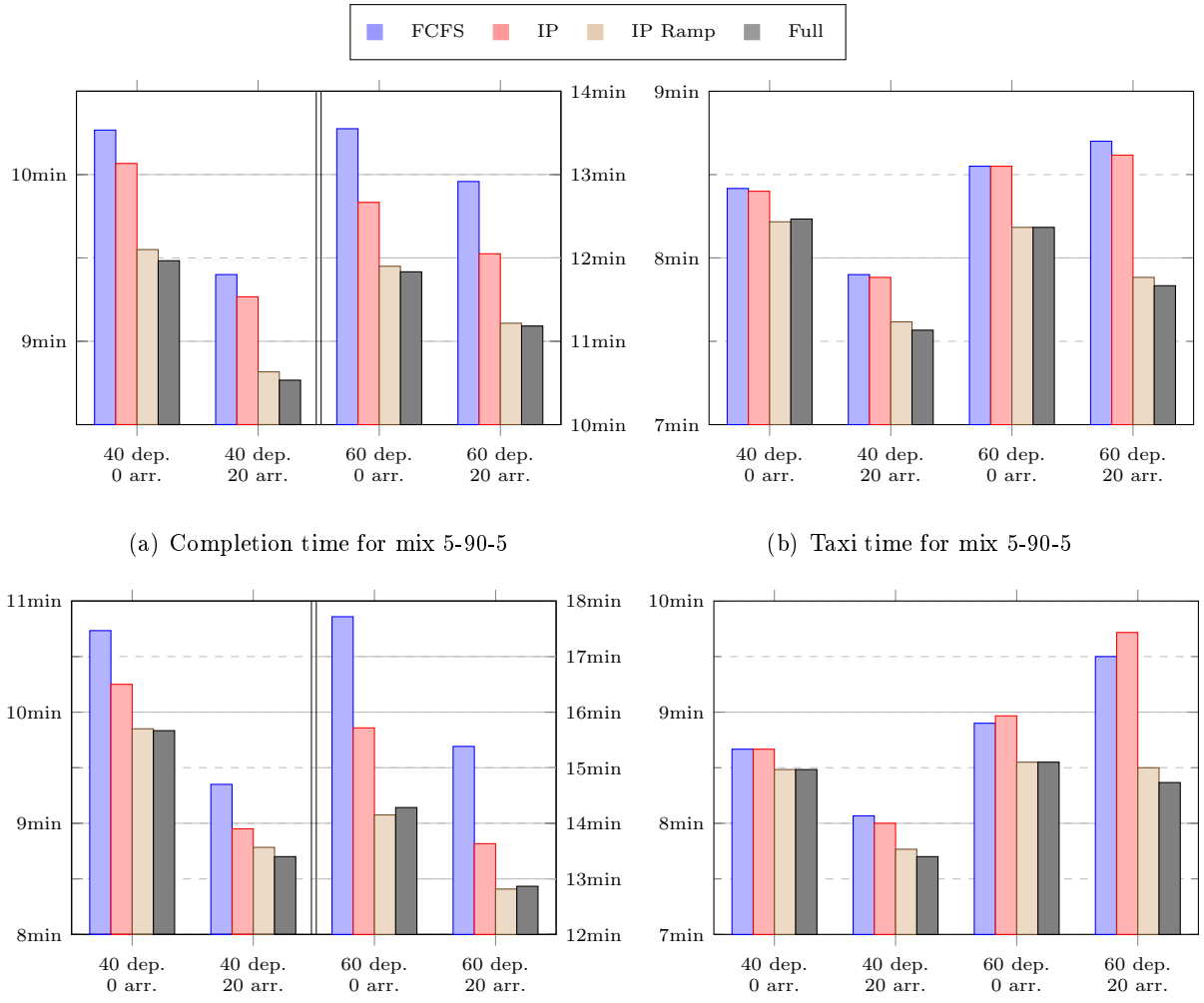
Figure 5.3 additionally presents the results in terms of average completion and taxi times. It reveals that average completion times are significantly improved by *Full* and *IP Ramp*: approximately 30 seconds are earned on instances with 40 departures and 1 minute on instances with 60 departures. Improvements in average taxi time are less important but still significant.

In conclusion, significant benefit rises from a better integration of the GRP and the RSP. Nevertheless, the proposed methods are not suitable for a direct applications because of excessive computation times. In the next section, we evaluate the different improvements proposed for *IP Ramp* aiming at tackling this problem.

5.6.3 Improving computation times of *IP Ramp*

5.6.3.1 Constraints reformulations and filtering

In Section 5.5.2, we proposed some techniques aiming at improving the solving of *IP Ramp*. We proposed a reformulation for taxi time and stand blockages constraints (5.15)-(5.17) and a heuristic filtering of ramp separation constraints (5.19) that have strong chances to be naturally respected.



(a) Completion time for mix 5-90-5

(b) Taxi time for mix 5-90-5

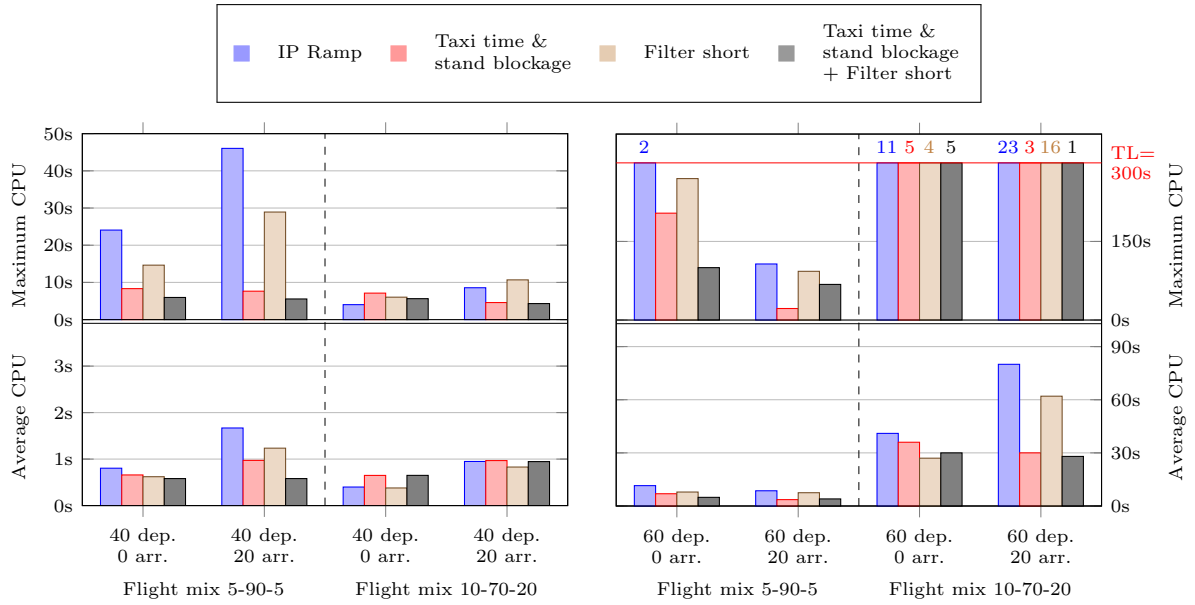
(c) Completion time for mix 10-70-20

(d) Taxi time for mix 10-70-20

Figure 5.3: Comparison of algorithms performances

In Figure 5.4, *IP Ramp* is the same model than in previous section (Equations (5.12)-(5.17), (5.19)-(5.24)). *Taxi time & stand blockage* refers to *IP Ramp* where taxi time and stand blockages constraints (5.15)-(5.17) have been reformulated by constraints (5.25)-(5.27). It aims at evaluating the isolated effect of these reformulations. *Filter short* refers to *IP Ramp* where «short» ramp separation constraints has been filtered (see Section 5.5.2 for a detailed explanation of «short»). It aims at evaluating the isolated effect of the filtering. *Taxi time & stand blockage + Filter short* uses both techniques.

Figure 5.4 presents the effect of the different improvements on the maximum and the average computation times. The figures above the bars reaching the time limit are the number of iterations ended prematurely.



(a) Instances with 40 departures

(b) Instances with 60 departures

Figure 5.4: Impact of the constraints reformulation on computation times

For instances with 40 departures, using both techniques is the fastest approach for flight mix 5-90-5, but their effect is more mitigated for flight mix 10-70-20. Nevertheless, it brings the maximum computation times below 10 seconds and average computation times below 1 second. These instances can now be solved in reasonable computation times.

However, this is not the case of instances with 60 departures and particularly for flight mix 10-70-20. The time limit is still reached for some iterations and the average computation times are still excessive. Nonetheless, our improvements significantly reduce computation times. For flight mix 5-90-5, average computation times fall below 5 seconds, which is reasonable. But maximum computation times are too long and the effect of the different techniques is mitigated: using only the constraints reformulations is more efficient for instances with 20 arrivals than using both techniques (22 vs 68 seconds), but not on instances without arrivals (204 vs 100 seconds).

Remark on gaps

Every iteration can allow several optimal solutions. Therefore, using exact reformulations, such as (5.25)-(5.27), does not guarantee to find the same solution. Also, our filtering of ramp separation constraints is heuristic and finding the same solution is not guaranteed too. Because of the dynamic nature of the problem implying a sliding time window approach, it can potentially lead to significant differences in the final solutions. We observed that solutions are actually

different, but the average gaps of every approach are 0.2 % close. Besides, no general trend comes out, i.e. no approach appears to be generally better than the others.

In conclusion, these experiment highlight that using both our constraints reformulations and our filtering is generally preferable and significantly reduces computation times. Unfortunately, not enough for a direct application.

5.6.3.2 Using the previous time window

As explain in Section 5.5.3, the solution of the previous time window can be used to heuristically reduce the number of variables and constraints. It is supposed to speed up the solving, but it can deteriorate the quality of the solution.

Table 5.4 presents the performances of such a principle (see Section 5.5.3 for more details). The gap are computed with respect to the best known solutions so far. We also recall the results of IP Ramp with our both improvements, referred to as *IP Ramp*$^+$, for the sake of comparison. The computation times presented do not account for the first iteration since no previous solution is available, thus they are not representative of the method. They are not counted for *IP Ramp*$^+$ too.

		Using previous solution				IP Ramp$^+$			
		Gap		CPU		Gap		CPU	
		Max	Avg	Max	Avg	Max	Avg	Max	Avg
Flight mix 5-90-5	40 dep. 0 arr.	0.86 %	0.44 %	1.1s	0.2s	1.05 %	0.52 %	5.1s	0.5s
	40 dep. 20 arr.	1.66 %	0.74 %	1.7s	0.3s	1.52 %	0.62%	5.6s	0.9s
	60 dep. 0 arr.	1.71 %	0.22 %	17s	0.8s	1.45 %	0.5%	100s	5.1s
	60 dep. 20 arr.	4.12 %	1.38 %	11s	0.9s	2.81 %	0.71 %	68s	4.1s
Flight mix 10-70-20	40 dep. 0 arr.	0.46 %	0.11 %	2.1s	0.2s	0.46 %	0.19 %	4.1s	0.6s
	40 dep. 20 arr.	1.94 %	0.87 %	1.1s	0.3s	2.18 %	0.8 %	4.3s	0.9s
	60 dep. 0 arr.	0.56 %	0.14 %	20s	2.5s	1.11%	0.28 %	300s	32s
	60 dep. 20 arr.	3.14 %	0.88 %	45s	3.0s	2.29 %	0.91 %	300s	29s

Table 5.4: Using the solution of previous iteration

Table 5.4 highlights that this process is very efficient. Firstly, almost all test sets are solved with an average gap below 1 %. Note that the gaps are sometimes less than the gaps of *IP Ramp*$^+$. It is again due to the sliding time window scheme. Secondly, the average computation times are appropriate to an industrial application and the time limit is never reached. The

maximum computation time over the instances with 60 departures and 20 arrivals are still a bit long for flight mix 10-70-20, but it remains reasonable. Note that a fine tuning of Δ_i can allow to better control computation times, but it may further deteriorate the quality of the solutions.

5.7 Conclusion

This chapter focuses on the management of the departure process, from push back to take-off, through an integration of the GRP and the RSP: the I-RSP/GRP. The main conclusion of this chapter is that a better integration of both problems actually results in a better synchronization of ground movements and take-offs, leading to an increased efficiency and decreased taxi times. Our main contribution is an efficient heuristic sequential algorithm based on an innovative IP formulation of the RSP that takes conflicts of the ramp area into account. We also propose different techniques aiming at improving its solving. In a numerical study based on Copenhagen Airport (CPH) layout, we show that our approach provides high quality solutions, while offering significantly shorter computation times than an exact approach from the literature directly integrating both problems in a single MIP. Our approach first suffered from excessive computation times, preventing an industrial application, but further experiments revealed that the improvements we proposed tackle this problem. Our approach finally fits the industrial requirements.

A direct perspective of our works is to strengthen ramp separation constraints. Unfortunately, the layouts of the ramp area can be very diverse and we did not succeed in identifying a general structure allowing to design efficient cuts or reformulations. Nevertheless, we believe that some frequent situations can be identified and used for such a purpose.

Another challenging direction for future works is to consider the optimization of landings in the approach. We explained how adapting our new IP formulation of the RSP to deal with arrivals, but computational complexity will be increased. We cannot assert whether or not our approach will still be fast enough without experiments.

Numerical experiments were performed on Copenhagen Airport (CPH) layout and one may wonder to what extents our results can be generalized. Our approach is actually based on the result that the ramp area and the runway are the two main bottlenecks in CPH during departure peaks. Even if it is the case of other airports (e.g. London Heathrow Airport (LHR), see Atkin et al. [2012]), it may not be valid anywhere. Runways are generally recognized as an important

bottleneck of airports (see e.g. Idris et al. [1998]) but the criticality of the ramp area probably may depend on its layout.

However, the methodology we followed to design our method can be applied to any airports. We first analyze the bottlenecks of the traffic through an analysis of ground movements. Then, we proposed a simplified model of integration focusing on these bottlenecks. It is consequently more likely to be tractable than a global and direct integration. Furthermore, our algorithms identifying the conflicts in the ramp area (Algorithms 1 and 2) can be applied to other airport areas, since they are only based on building shortest schedules. The ramp separations introduced in the model can also concern other airport areas.

Chapitre 6

Conclusion

Dans cette thèse, nous nous intéressons à trois problèmes d'optimisation des opérations aéroportuaires : l'affection aux points de stationnement (chapitre 2), le routage au sol (chapitre 3), l'ordonnancement à la piste (chapitre 4) et son intégration avec le problème de routage au sol (chapitre 5).

Nos principaux résultats sont rappelés dans ce chapitre. Nous présentons également les perspectives globales de nos travaux ainsi que les directions de recherches futures qu'ils indiquent.

Résultats principaux

Le problème d'affectation aux points de stationnement

Nos résultats sur le problème d'affection aux points de stationnement sont à la fois théoriques et pratiques. Sur le plan théorique, nous montrons que trouver une solution réalisable, c'est à dire affecter chaque avion à un point de stationnement compatible, est un problème NP-complet. Nous en déduisons la NP-difficulté de plusieurs cas particuliers intéressants du problème d'optimisation laissés ouverts par la littérature.

D'un point de vue pratique, nous proposons une modélisation en Programme Linéaire en Nombres Entiers (PLNE) ainsi que plusieurs techniques visant à accélérer son temps de résolution (reformulations de contraintes, changement de variables et cassage de symétries). Au travers d'une expérimentation basée sur des instances réalistes provenant de deux aéroports européens majeurs, nous montrons que les améliorations apportées à notre modèle le rendent opérationnel pour une application industrielle. Ce modèle est également embarqué dans deux heuristiques de décomposition (une spatiale et une temporelle) permettant de réduire davantage les temps de calculs tout en conservant un très haut niveau de qualité de solutions. Les performances

de celles-ci nous laissent penser qu'elles peuvent être appliquées efficacement aux plus grands aéroports du monde. Une comparaison aux méthodes de la littérature, qui sont heuristiques, révèle que la résolution exacte du problème permet d'atteindre des solutions significativement meilleures. Il en résulte une meilleure utilisation des points de stationnement, un revenu plus important pour l'aéroport et une meilleure qualité de service pour les passagers.

Le problème de routage au sol

Nos résultats sur le problème de routage au sol sont principalement d'ordre expérimental. Nous effectuons une analyse des relations liant les différents indicateurs caractérisant la qualité d'un planning de roulage en se basant sur l'aéroport de Copenhague (CPH). Pour cela, nous nous appuyons sur une modélisation en PLNE issue de la littérature et routant les avions selon un chemin prédeterminé. Nous lui ajoutons une modélisation des principaux indicateurs de ponctualité de l'industrie : le retard mesuré aux points de stationnement et l'*On Time Performance*. Nos expérimentations montrent que ces indicateurs sont en contradiction avec l'objectif de réduire le temps de roulage des départs, garant de la consommation de carburant et donc de l'impact environnemental. Par conséquent, nous proposons de nouveaux indicateurs de ponctualité qui sont à la fois plus écologiques et plus logiques pour chaque acteur.

Ce modèle est généralisé pour considérer des chemins alternatifs. Nos expérimentations révèlent que cela ne permet pas d'améliorer significativement les différents indicateurs tandis que la difficulté de résolution du modèle est grandement accrue. Enfin, nous proposons une méthodologie expérimentale permettant d'identifier les goulets d'étranglement du trafic au sol. Elle révèle que la piste de décollage et les *taxiways* avoisinant les points de stationnement impactent fortement la fluidité du trafic, contrairement au reste du réseau.

Le problème d'ordonnancement à la piste et son intégration avec le routage au sol

Nos principaux résultats sur le problème d'ordonnancement à la piste concernent son intégration avec le routage au sol. Nous proposons une méthode heuristique séquentielle basée sur une formulation en PLNE innovante du problème d'ordonnancement à la piste. Le principe de cette méthode est d'optimiser la séquence de décollage dans un premier temps, puis le routage au sol dans second temps. Notre formulation tient compte des conflits du routage proches des points de stationnement et inclut donc les principaux goulets d'étranglement identifiés précédemment. Nous proposons également diverses techniques visant à accélérer la résolution de ce modèle. Dans une étude numérique basée sur l'aéroport de Copenhague (CPH), nous montrons qu'une

meilleure intégration des opérations de pistes et des mouvements au sol permet d'améliorer la gestion de la piste tout en réduisant le temps de roulage. Nous montrons également que notre heuristique fournit des solutions de bonne qualité en un temps raisonnable, contrairement à une formulation exacte issue de la littérature.

Perspectives de recherches

Nous ne rappelons pas ici les perspectives liées à chaque problème qui ont été présentées à la fin de chaque chapitre (cf. sections 2.8, 3.6 et 5.7), mais nous présentons des perspectives plus globales ainsi que les directions de recherches futures indiquées par cette thèse.

Intégration du problème d'affectation aux points de stationnement

Le plan d'affectation aux points de stationnement est une entrée importante du problème de routage au sol. Il paraît donc naturel de vouloir intégrer ces deux problèmes, ou plus généralement d'intégrer les trois problèmes abordés dans cette thèse. Cependant, ils ne considèrent pas le même horizon de temps. En effet, le problème d'affectation aux points de stationnement est généralement résolu au plus tard la veille des opérations pour toute la journée. Au contraire, les problèmes de routage au sol et d'ordonnancement à la piste sont opérationnels et concernent au maximum l'heure à venir. Une intégration directe est donc difficilement envisageable. Nous voyons ici deux directions de recherches principales.

La première est une intégration préventive dans le problème d'affectation aux points de stationnement par de la robustesse. Nous avons vu que les conflits dans les zones des points de stationnement contraignent fortement le routage. Ce sont principalement des conflits de repoussage et des conflits face-à-face entre arrivées et départs (*head-on conflicts*). Ces conflits ont été résolus lors du routage, mais une affection aux points de stationnement différente ne les aurait pas engendrés. Il en est de même pour les conflits de blocage de points de stationnement.

La difficulté de l'intégration réside donc dans le fait que les conflits ne sont pas connus de façon précise lors de l'affectation aux points de stationnements, du fait des nombreuses perturbations se produisant le jour des opérations. Ainsi, une intégration préventive nous paraît la plus adaptée. Il est probablement préférable de répartir au maximum les opérations censées se dérouler en même temps dans l'aéroport. La mise au point d'indicateurs ou de contraintes garantissant une bonne répartition est une direction de recherches futures. Un dimensionnement adéquat des temps tampons et des contraintes d'adjacence additionnelles sont notamment une

première idée qui mériterait d'être explorée.

Deuxièmement, lorsque de trop fortes perturbations apparaissent le jour des opérations, le plan d'affectation n'est plus réalisable et un problème de réaffectation doit être résolu. Ce problème prend place sur un horizon de temps beaucoup plus proche des opérations et il est possible que des informations plus précises soient exploitables. Une intégration directe avec le problème de réaffectation paraît donc plus envisageable et soulève ainsi une autre direction de recherches prometteuse.

Intégration des opérations aux points de stationnement

Les opérations aux points de stationnement sont aujourd'hui souvent effectuées par des entreprises spécialisées et indépendantes des autorités aéroportuaires et des contrôleurs aériens (cf. chapitre 1). A notre connaissance, leur intégration dans les processus de décision que nous avons étudiés est plus que restreinte.

Pourtant, les opérations aux points de stationnement jouent un rôle déterminant dans la ponctualité des vols et particulièrement sur l'heure à laquelle un départ va être prêt pour le repoussage. Ces heures étant une des entrées principales des problèmes de routage au sol et de l'ordonnancement à la piste, une intégration avec ces problèmes a donc du sens et offre de nouvelles opportunités. Une meilleure synchronisation des opérations pourrait permettre une préparation plus rapide des vols critiques du routage au sol et du décollage. Il en résulterait une efficacité accrue du processus de départs. En contrepartie, cela donnerait plus de flexibilité aux opérateurs de sol sur les vols moins critiques. Ils pourraient ainsi mieux optimiser leurs coûts.

Ces opérations sont également fortement contraintes par le plan d'affectation aux points de stationnement, qui définit des fenêtres de temps relativement courtes. Il paraît donc aussi naturel de vouloir les prendre en compte lors de l'affectation aux points de stationnement, ce qui à notre connaissance, n'a encore jamais été considéré dans la littérature. Une meilleure intégration pourrait également permettre une réduction des coûts opérationnels et une efficacité accrue, conduisant ainsi à une réduction du temps de préparation d'un vol.

Stochasticité et simulation

Pour finir, l'ensemble des problèmes considérés dans cette thèse ont été abordés de manière déterministe et principalement par la PLNE. Cependant, ces problèmes ont bien des aspects stochastiques car de nombreuses perturbations se produisent durant les opérations. Un vol peut être retardé en l'air et atterrir plus tard que prévu, par exemple à cause de mauvaises conditions

météorologiques. Il peut également arriver en avance s'il rencontre des vents favorables. De même, des perturbations peuvent venir des opérations aux points de stationnement et retarder le repoussage. Le temps de roulage est également sujet à des aléas, ce qui crée de l'incertitude sur les heures de décollage et d'arrivée aux points de stationnement.

Le plan d'affectation aux points de stationnement peut ainsi devenir irréalisable, malgré les temps tampon considérés. Une étude approfondie de ces aléas permettrait de les prendre en compte plus finement, notamment par un dimensionnement adéquat des temps tampon, et peut ainsi conduire à des plans d'affectation plus robustes.

Ces perturbations ont également des conséquences sur le routage au sol et la séquence de décollage. L'approche par horizon glissant, utilisée dans cette thèse, dans la littérature et dans la pratique, permet de réagir à ces aléas, puisque tout est réoptimisé à chaque nouvel évènement (arrivée d'avion dans le système ou perturbation). Cependant, il est possible qu'une approche stochastique pro-active soit plus efficace que cette approche réactive. Les solutions obtenues peuvent notamment être plus stables d'une itération à l'autre et ainsi plus appropriées aux besoins de l'industrie. Ces idées ont déjà suscité des travaux, notamment pour l'ordonnancement à la piste (cf. par exemple Solveling and Clarke [2014] et Bosson et al. [2014a,b]).

Quoiqu'il en soit, il est difficile d'évaluer a priori la performance d'une méthode déterministe dans un cadre stochastique. De même, il est ardu de comparer une méthode déterministe à une méthode stochastique. La simulation est un moyen de répondre à ces problématiques et Amadeus travaille actuellement sur le développement d'un simulateur de trafic au sol. Le développement d'un tel simulateur nécessite de bien comprendre la dynamique aléatoire qui régit le routage au sol et l'ordonnancement à la piste. Nous avons commencé à travailler sur des modélisations simples du processus de départ par des files d'attente dans ce but. Des travaux similaires ont déjà été proposés, par exemple par Pujet et al. [1999], Carr et al. [2002] et Simaiakis et al. [2014].

Une des directions de recherches principales de cette thèse est l'embarquement de nos méthodes d'optimisation dans un tel simulateur, afin d'évaluer leurs comportements dans un cadre stochastique. Nous espérons ainsi une validation, ou dans le cas contraire une mise en valeur des manquements de chaque méthode et donc les axes de recherches futures pour palier les éventuels défauts identifiés.

References

Bibliographie

J.A.D. Atkin, E.K. Burke, J.S. Greenwood, and D. Reeson. A metaheurisitc approach to aircraft departure scheduling at London Heathrow Airport. In *Electronic proceeding of the 9th international conference on computer-aided scheduling of public transport*, San Diego, California, USA, 2004.

J.A.D. Atkin, E.K. Burke, J.S. Greenwood, and D. Reeson. Hybrid metaheuristics to aid runway scheduling at London Heatrhow Airport. *Transportation Science*, 41(1) :90–106, 2007.

J.A.D. Atkin, E.K. Burke, J.S. Greenwood, and D. Reeson. An examination of take-off scheduling constraints at London Heatrhow Airport. *Public Tranport*, 1 :169–187, 2009.

J.A.D. Atkin, E.K. Burke, and J.S. Greenwood. TSAT allocation at London Heathrow : The relationship between slot compliance, throughput and equity. *Public Transport*, 2(3) :173–198, 2010a.

J.A.D 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, Budapest, Hungary*, 2010b.

J.A.D. Atkin, E.K. Burke, and J.S. Greenwood. A comparison of two methods for reducing take-off delay at London Heathrow Airport. *Journal of Scheduling*, 14 :409–421, 2011a.

J.A.D. Atkin, E.K. Burke, and S. Ravizza. A more realistic approach for airport ground movement optimisation with stand holding. In *proceedings of the 5th multidisciplinary international scheduling conference, Phoenix, Arizona, USA*, 2011b.

J.A.D. Atkin, G. De Maere, E.K. Burke, and J.S. Greenwood. Addressing the pushback time allocation problem at Heathrow airport. *Transportation Science*, 47(4) :584–602, 2012.

H. Balakrishnan and B. Chandran. Scheduling aircraft landings under constrained position shifting. In *AIAA Guidance, Navigation and Control Conference and Exhibit*, Keystone, Colorado, USA, 2006.

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 R&D Seminar*, 2007.

H. Balakrishnan and B. Chandran. Algorithms for scheduling runway operations under constrained position shifting. *Operations Research*, 58(6) :1650–1665, 2010.

H. Balakrishnan and Y. Jung. A framework for coordinated surface operations planning at Dallas-Fort Worth International Airport. In *proceedings of AIAA Guidance, Navigation and Control Conference, Hilton Head, USA*, 2007.

J. E. Beasley, M. Krishnamoorthy, Y. M. Sharaiha, and D. Abramson. Scheduling aircraft landings - the static case. *Transportation Science*, 34(2) :180–198, 2000.

J. A. Bennell, M. Mesgarpour, and C. N. Potts. Airport runway scheduling. *4OR - A Quaterly Journal of Operations Research*, 9 :115–138, 2011.

A. Bolat. Procedures for providing robust gate assignments for arriving aircrafts. *European Journal of Operational Research*, 120(1) :63–80, 2000.

C. Bosson, M. Xue, and S. Zelinski. Optimizing integrated terminal airspace operations under uncertainty. In *Digital Avionics Systems Conference (DASC), 2014 IEEE/AIAA 33rd*, pages 1A3–1, 2014a.

C. Bosson, M. Xue, and S. Zelinski. Optimizing integrated arrival, departure and surface operations under uncertainty. 2014b.

A.R. Brentnall. *Aircraft arrival management*. PhD thesis, University of Southampton, UK, 2006.

D. Briskorn and R. Stolletz. Aircraft landing problems with aircraft classes. *Journal of Scheduling*, 17 : 31–45, 2014.

British Airways. Annual report and accounts, 2008-09. URL http://www.britishairways.com/cms/global/microsites/ba_reports0809/our_business/kpi3.html.

P. Burgain, O.J. Pinon, E. Feron, J.P. Clarke, and D.N. Mavris. Optimizing pushback decisions to valuate airport surface surveillance information. *IEEE Transactions on Intelligent Transportation Systems*, 13(1) :180–192, 2012.

F. Carr, A. Evans, J.P. Clarke, and E. Feron. Modeling and control of airport queueing dynamics under severe flow restrictions. In *American Control Conference, 2002. Proceedings of the 2002*, volume 2, pages 1314–1319, 2002.

G.L. Clare and A.G. Richards. Optimization of taxiway routing and runway scheduling. *Intelligent Transportation Systems, IEEE Transactions on*, 12(4) :1000–1013, 2011.

A. Cook and G. Tanner. European airline delay cost reference values, 2011. URL http://www.eurocontrol.int/sites/default/files/content/documents/sesar/business-case/european_airline_delay_cost_reference_values_2011.pdf.

A. D'Ariano, D. Pacciarellio, M. Pistelli, and M. Pranzo. Real-time scheduling of aircraft arrivals and departures in a terminal maneuvering area. *Networks*, 65(3) :212–227, 2015.

R. Deau, J.-B. Gotteland, and N. Durand. Runways sequences and ground traffic optimization. In *proceedings of the 3rd International conference on Research in Air Transportation, Fairfax, USA*, 2008.

R. Deau, J.-B. Gotteland, and N. Durand. Airport surface management and runways scheduling. In *proceedings of the 8th USA/Europe Air Traffic Management R&D Seminar, Napa, USA*, 2009.

G. Diepen, J.M. Van Den Akker, J.A. Hoogeven, and J.W. Smeltink. Finding a robust assignment of flights to gates at Amsterdam Airport Schipol. *Journal of Scheduling*, 15 :703–715, 2012.

H. Ding, A. Lim, B. Rodrigues, and Y. Zhu. The over-constrained airport gate assignment problem. *Computers and Operations Research*, 32(7) :1867–1880, 2005.

U. Dorndorf, A. Drexl, Y. Nikulin, and E. Pesch. Flight gate scheduling : State-of-the-art and recent developments. *Omega*, 35(3) :326–334, 2007.

U. Dorndorf, F. Jaehn, and E. Pesch. Modelling robust flight-gate scheduling as a clique partitioning problem. *Transportation Science*, 42(3) :292–301, 2008.

U. Dorndorf, F. Jaehn, and E. Pesch. Flight gate scheduling with respect to a reference schedule. *Annals of Operations Research*, 194 :177–187, 2010.

Eurocontrol. Airport CDM leaftlet, January 2009. URL http://www.euro-cdm.org/library/cdm_leaflet.pdf.

Eurocontrol. Long-term forcecast, 2010. URL http://www.eurocontrol.int/sites/default/files/content/documents/official-documents/forecasts/long-term-forecast-2010-2030.pdf.

Eurocontrol. Airport CDM implementation, 2012a. URL http://www.euro-cdm.org/library/cdm_implementation_manual.pdf.

Eurocontrol. Performance review report, 2012b. URL http://www.eurocontrol.int/sites/default/files/prr-2012.pdf.

Eurocontrol. Medium-term forcecast, 2012c. URL http://www.eurocontrol.int/sites/default/files/content/documents/official-documents/forecasts/medium-term-forecast-flights-2012-2018.pdf.

Eurocontrol. Coda digest : Delays to air transport in europe, 2013a. URL http://www.eurocontrol.int/sites/default/files/content/documents/official-documents/facts-and-figures/coda-reports/coda-digest-annual-2013.pdf.

Eurocontrol. Challenges of growth 2013, 2013b. URL https://www.eurocontrol.int/sites/default/files/article//content/documents/official-documents/reports/201306-challenges-of-growth-2013-task-4.pdf.

Eurocontrol. Seven-year forecast, 2015. URL https://www.eurocontrol.int/sites/default/files/content/documents/official-documents/forecasts/seven-year-flights-service-units-forecast-2015-2021-Feb2015.pdf.

T. Fahle, R. Feldmann, S. Gotz, and B. Monien. The aircraft sequencing problem. *Computer Science in perspective*, pages 152–166, 2003.

A. Faye. Solving the aircraft landing problem with time discretization approach. *European Journal of Operational Research*, 242(3) :1028–1038, 2015.

F. Furini, A. C. Persiani Alfredo, and P. Toth. Aircraft sequencing problems via a rolling horizon algorithm. In *Combinatorial Optimization*, pages 273–284. 2012.

F. Furini, M.P. Kidd, C.A. Persiani, and P. Toth. Improved rolling horizon approaches to the aircraft sequencing problem. *Journal of Scheduling*, 15(5) :1–13, 2015.

M. Garey, D. Johnson, G. Mille, and C. Papadimitriou. The complexity of coloring circular arcs and chords. *SIAM Journal on Algebraic Discrete Methods*, 1(2) :216–227, 1980.

H.M. Genç, O.K. Erol, I. Eksin, and M.F. Berber. A stochastic neighborhood search approach for airport gate assignment problem. *Expert Systems with Applications*, 39 :316–327, 2012.

A. Ghoniem, H.D. Sherali, and H. Baik. Enhanced models for a mixed arrival-departure aircraft sequencing problem. *Journal on Computing*, 26(3) :514–530, 2014.

A. Ghoniem, F. Farhadi, and M. Reihaneh. An accelerated branch-and-price algorithm for multiple-runway aircraft sequencing problems. *European Journal of Operational Research*, 2015.

J.-B. Gotteland and N. Durand. Genetic algorithms applied to airport ground traffic optimization. In *proceedings of the congress on Evolutionary Computation, Canberra, Australia, vol. 1*, 2003.

J.-B. Gotteland, N. Durand, J.M. Alliot, and E. Page. Air ground traffic optimization. In *proceedings of the 4th International Air Traffic Management R&D Seminar ATM, Santa Fe*, 2001.

J.-B. Gotteland, N. Durand, and J.M. Alliot. Handling CFMU slots in busy airports. In *proceedings of the 5th USA/Europe Air Traffic Management R&D Seminar, Duapest, Hungary*, 2003.

J. Guépet, R. Acuna-Agost, O. Briant, and J.P. Gayon. Exact and heuristic approaches to the airport stand allocation problem. *European Journal of Operational Research*, 246(2) :597–608, 2015.

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*, 2015.

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*, 248(3) :827–839, 2016.

G. Gupta, W. Mali, and Y. Jung. An integrated collaborative decision making and tactical advisory concept for airport surface operations management. In *12th AIAA Aviation Technology, Integration, and Operations (ATIO) Conference, Indianapolis, IN*, 2012.

A. Haghani and M. Chen. Optimizing gate assignments at airport terminals. *Transportation Research Part A : Policy and Practice*, 32(6) :437–454, 1998.

G. Hancerliogullari, G. Rabadi, A.H. Al-Salem, and M. Kharbeche. Greedy algorithms and metaheuristics for a multiple runway combined arrival-departure aircraft sequencing problem. *Journal of Air Transport Management*, 32 :39–48, 2013.

Heathrow. Heathrow air quality, 2008-09. URL http://www.heathrow.com/file_source/Company/Static/PDF/Communityandenvironment/air-quality-strategy_LHR.pdf.

H.R. Idris, B. Delcaire, I. Anagnostakis, W.D. Hall, J.P. Clarke, R.J. Hansman, E. Feron, and A.R. Odoni. Observations of departure processes at Logan airport to support the development of departure planning tools. In *2nd USA/Europe air traffic management R&D seminar*, pages 1–4, 1998.

F. Jaehn. Solving the flight gate assignment problem using dynamic programming. *Zeitschrift für Betriebswirtschaft*, 80 :1027–1039, 2010.

F. Jaehn. Airplane boarding. *European Journal of Operational Research*, 244 :339–359, 2015.

Y. Jung, T. Hoang, J. Montoy, G. Gupta, W. Malik, and L. Tobias. A concept and implementation of optimized operations of airport surface traffic. In *10th AIAA Aviation Technology, Integration, and Operations (ATIO) Conference, Fort Worth, TX*, 2010.

Y. Jung, T. Hoang, J. Montoya, G. Gupta, W. Malik, L. Tobias, and H. Wang. Performance evaluation of a surface traffic management tool for Dallas/Fort Worth International Airport. In *Ninth USA/Europe Air Traffic Management Research and Development Seminar*, pages 1–10, 2011.

G. Keith and A. Richards. Optimiztion of taxiway routing and runway scheduling. In *proceedings of AIAA Guidance, Navigation and Control Conference, Honolulu, Hawaii, USA*, 2008.

H. Khadilkar and H. Balakrishnan. Estimation of aircraft taxi-out fuel burn using flight data recorder archives. In *proceedings of the AIAA Guidance, Navigation, and Control Conference*, 2011.

B. Kim, L. Li, and J.P. Clarke. Runway assignment by minimizing emissions in terminal airspace. In *Proceedings of AIAA Guidance, Navigation and Control Conference, Toronto, Canada*, 2010.

S.H. Kim, E. Feron, and J.P. Clarke. Assigning gates by resolving physical confllicts. In *AIAA Guidance, Navigation and Control Conference*, Chicago, 2009.

S.H. Kim, E. Feron, and J.P. Clarke. Gate assignment to minimize passenger transit time and aircraft taxi time. *Journal of Guidance, Control, and Dynamics*, 36 :467–475, 2013.

L.G. Kroon, A. Sen, H. Deng, and A. Roy. The optimal cost chromatic partition problem for trees and interval graphs. In *Graph-Theoretic Concepts in Computer Science*, volume 1197 of *Lecture Notes in Computer Science*, pages 279–292. Springer Berlin Heidelberg, 1997.

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*, pages 4E1–1, 2012.

C. Lesire. Iterative planning of airport ground movements. In *proceedings of the 4th International Conference on Research in Air Transportation (ICRAT), Budapest, Hungary*, 2010.

A. Lieder, D. Briskorn, and R. Stolletz. A dynamic programming formulation for the aircraft landing problem with aircraft classes. *European Journal of Operational Research*, 243 :61–69, 2015.

A. Lim and F. Wang. Robust airport gate assignment. In *ICTAI '05 : Proceedings of the 17th IEEE international conference on tools with artificial intelligence*, 2005.

Q. Liu, T. Wu, and X. Luo. A space-time network model based on improved genetic algorithm for airport taxiing scheduling problems. *Procedia Engineering 15*, 15 :1082–1087, 2011.

W. Malik, G. Gupta, and Y. Jung. Managing departure aircraft release for efficient airport surface operations. In *AIAA Guidance, Navigation, and Control Conference*, 2010.

R. S. Mangoubi and D. F. X. Mathaisel. Optimizing gate assignments at airport terminals. *Transportation Science*, 19(2) :173–188, 1985.

A. Marin. Airport management : taxi planning. *Annals of Operations Research*, 143(1) :191–202, 2006.

A. Marin and E. Codina. Network design : taxi planning. *Annals of Operatins Research*, 157(1) :135–151, 2008.

M. Mongeau and C. Bes. Optimization of aircraft container loading. *Aerospace and Electronic Systems, IEEE Transactions on*, 39(1) :140–150, 2003.

T. Nikoleris, G. Gupta, and M. Kistler. Detailed estimation of fuel consumption and emissions during aircraft taxi operations at Dallas/Fort Worth International Airport. *Transportation Research*, 16(4) : 304–308, 2011.

A. Norin. Airport logistics - modeling and optimizating the turn-around process, 2008. URL http://www.diva-portal.org/smash/get/diva2:133720/FULLTEXT01.pdf.

H. N. Psaraftis. A dynamic approach to the aircraft sequencing problem. Flight Transportation Laboratory, MIT, USA, 1978.

N. Pujet, B. Declaire, and E. Feron. Input-output modeling and control of the departure process of congested airports. In *Proceedings of AIAA Guidance Navigation and Control Conference, and Exhibit*, Portland, OR, USA, 1999.

S. Rathinam, J. Montoya, and Y. Jung. An optimization model for reducing aircraft taxi times at the Dallas Fort Worth International Airport. In *proceedings of the 26th International Congress of the Aeronautical Sciences*, 2008.

S. Rathinam, Z. Wood, B. Sridhar, and Y.C. Jung. A generalized dynamic programming approach for a departure scheduling problem. In *AIAA Guidance, Navigation, and Control Conference*, pages 10–13, 2009.

S. Ravizza, J. Chen, J.A.D. Atkin, E. K. Burke, and P. Stewart. The trade-off between taxi time and fuel consumption in airport ground movement. In *conference on Advanced Systems for Public Transport, Stantiago, Chile*, 2012.

S. Ravizza, J.A.D. Atkin, and E.K. Burke. A more realistic approach for airport ground movement optimisation with stand holding. *Journal of Scheduling*, 17(5) :507–520, 2014.

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*, 66 : 251–267, 2014.

J.W. Smeltink, M.J. Soomer, P.R. de Waal, and R.D. Van Der Mei. An optimization model for airport taxi scheduling. In *proceedings of the INFORMS Annual Meeting Denver, USA*, 2004.

G. Solveling and J.P. Clarke. Scheduling of airport runway operations using stochastic branch and bound methods. *Transportation Research Part C : Emerging Technologies*, 45 :119–137, 2014.

G. Solveling, S. Solak, J.P. Clarke, and E. Johnson. Scheduling of runway operations for reduced environmental impact. *Transportation Research Part D*, 16 :110–120, 2011.

P. van Leeuwen. Caed d2 : Modelling the turnaround process, April 2007. URL https://www.eurocontrol.int/eec/gallery/content/public/documents/projects/CARE/CARE_INO_III/CAED_D2_v2.0.pdf.

J. Xu and G. Bailey. The airport gate assignment problem : Mathematical model and a tabu search algorithm. In *Proceedings of the 34th Annual Hawaii International Conference on System Science*, 2001.

S. Yan and C.M. Chang. A network model for gate assignment. *Journal of Advanced Transportation*, 32 (2) :176–189, 1998.

S. Yan and C. Huo. Optimization of multiple objective gate assignments. *Transportation Research Part A : Policy and Practice*, 35(5) :413–432, 2001.

S. Yan and C.H. Tang. A heuristic approach for airport gate assignments for stochastic flight delays. *European Journal of Operational Research*, 180 :547–567, 2007.

RÉSUMÉ

Le cadre de cette thèse est l'optimisation des opérations aéroportuaires. Nous nous intéressons à trois problèmes de gestion des avions dans un aéroport : l'affectation aux points de stationnement, le routage au sol entre les pistes et les points de stationnement, et l'ordonnancement des décollages et des atterrissages.

Ce travail a été réalisé en collaboration étroite avec la société Amadeus. Nos approches ont été testées et validées avec des données réelles provenant d'aéroports européens.

Nous proposons une formulation en Programme Linéaire en Nombres Entiers (PLNE) du problème d'affectation aux points de stationnement. Nous montrons que trouver une affectation réalisable est un problème NP-Complet et nous proposons diverses améliorations visant à réduire le temps de résolution de notre modèle. Nous obtenons ainsi des solutions de meilleure qualité que celles de la littérature, tout en conservant un temps de calcul raisonnable.

Le problème de routage au sol est modélisé en adaptant un PLNE de la littérature. Nous montrons que les indicateurs de l'industrie sont en contradiction avec l'objectif de réduction du temps de roulage, et donc des émissions de pollutions. Nous proposons de nouveaux indicateurs basés sur l'heure de décollage, et non sur l'heure de départ du point de stationnement.

Enfin, nous nous intéressons à l'intégration de l'ordonnancement à la piste avec le routage au sol. Nous montrons qu'une meilleure intégration permet de réduire le temps de roulage et d'améliorer la gestion de la piste. Nous proposons une heuristique séquentielle basée sur une modélisation en PLNE innovante du problème d'ordonnancement à la piste. Nous montrons que cette heuristique fournit des solutions de bonne qualité en temps raisonnable, contrairement à l'approche exacte de la littérature.

MOTS-CLÉS *Aéroport, gestion des opérations, programmation linéaire en nombres entiers, affectation aux points de stationnement, routage au sol, ordonnancement à la piste.*

ABSTRACT

In this thesis, we address the optimization of aircraft ground operations at airports, focusing on three main optimization problems: the stand allocation, the ground routing between stands and runways, and the sequencing of take-offs and landings.

These works result from a close collaboration with Amadeus. Our approaches have been tested and validated with real data from European airports.

The stand allocation problem is formulated as a Mixed Integer Program (MIP). We show that finding an allocation plan respecting operational requirements is NP-Complete and we strengthen our model in several directions. We obtain better solutions than the literature withing reasonable computation times for an industrial application.

The ground routing problem is modeled by a MIP formulation adapted from the literature. We show that the main indicators of the industry are in contradiction with the objective of reducing taxi times and therefore air pollution. We propose new indicators based on take-off times instead of push back times.

Lastly, we focus on the integration of the runway sequencing with the ground routing. We highlight that a better integration allows to reduce taxi times while improving the management of the runway. We propose a sequential heuristic based on an innovative MIP formulation of the runway sequencing problem. This heuristic is shown to provide high quality solutions in reasonable computation times, unlike the exact approach from the literature.

KEY WORDS *Airport, operations management, mixed integer programming, stand allocation, ground routing, runway sequencing.*