A new multi-commodity flow model to optimize the robustness of the Gate Allocation Problem

Ruixin Wang, Cyril Allignol, Nicolas Barnier, Alexandre Gondran, Jean-Baptiste Gotteland, Catherine Mancel

A New Multi-commodity Flow Model to Optimize the Robustness of the Gate Allocation Problem

Ruixin Wang[a,b,*], Cyril Allignol[b], Nicolas Barnier[b], Alexandre Gondran[b], Jean-Baptiste Gotteland[b], Catherine Mancel[b]

[a]*Laboratory of Complex System Safety and Intelligent Decisions, CAUC-ENAC Joint Research Center of Applied Mathematics for Air Traffic Management, Sino-European Institute of Aviation Engineering, Civil Aviation University of China, Tianjin, China*
[b]*ENAC, Université de Toulouse, France*

Abstract

The Gate Allocation Problem (GAP) is one of the numerous operational problems that all busy airports have to handle and to optimize every day. In this article, we focus on two important objectives for the GAP: the robustness and the taxiing times.We first analyze a basic model which optimizes the robustness of the solution to the GAP and propose a more realistic and compact Multi-commodity Flow Problem model (MFP) to optimize both the robustness and the taxiing times. Based on tests with real data at Paris-Charles-de-Gaulle airport, this new efficient MFP model outperforms the basic one by orders of magnitude.

Keywords: Gate Allocation Problem, Robustness, Multi-commodity Flow Problem, Integer Linear Programming

Introduction

All airports have to handle many challenges regarding traffic increase, delay reduction, traffic predictability and environmental impact. In this context, airports attempt to improve the efficiency of their operations, among which *gate allocation* is one of the most important [6]. Given a set of flights, the schedule of their gate occupancy (i.e. their respective arrival and departure times at the gate) and a list of compatible gates, the Gate Allocation Problem (GAP) consists in assigning a gate to each flight such that at most one aircraft occupies a gate at any time, while optimizing various operating costs.

Classic objectives of the GAP include the optimization of passengers walking distance [20], the allocation of terminal gates over apron stands or the minimization of the number of towing movements [16]. However, the emergence of Aircraft Collaborative

*Principal corresponding author
Email addresses: ruixin.wang@recherche.enac.fr (Ruixin Wang), cyril.allignol@enac.fr (Cyril Allignol), nicolas.barnier@enac.fr (Nicolas Barnier), alexandre.gondran@enac.fr (Alexandre Gondran), alexandre.gondran@enac.fr (Jean-Baptiste Gotteland), catherine.mancel@enac.fr (Catherine Mancel)

Decision Making (A-CDM) [15] in the past decade, and more specifically of Departure Management (DMAN) regulations, tend to increase gate occupancy, which can lead to schedule conflicts and costly disruptions. To overcome this issue, the robustness of the allocation with respect to schedule deviations should be optimized.

Actually, the DMAN has become an essential feature at busy international airports to reduce their environmental impact and optimize their throughput: it computes a pre-departure sequence that aims at holding delayed flights at the gate, engines off and apart from the rest of the traffic, rather than burdening the taxiways. The DMAN process has proven to be very efficient [19], once successfully calibrated to each airport. However, deviations from the plan like late departures at the gate can lead to very complex situations in which arrivals have to wait a long time before a suitable gate becomes available. To alleviate this issue, we propose to design an initial gate allocation method which optimizes the robustness w.r.t. time deviations and assess its benefits.

In order to optimize the robustness of an assignment, [4] suggests to minimize the variance of the durations of gates idle times in order to keep a sufficient amount of buffer between successive flights. The same author then proposes in [5] an Integer Linear Programming (ILP) model to solve the GAP, however, the quadratic number of constraints prevents to solve large scale instances.

In this article, we first improve the above mentioned ILP model by taking into account the heterogeneity of the gates and discarding a number of superfluous variables. Afterwards, we propose a much tighter and smaller model based on the Multi-commodity Flow Problem (MFP) to optimize the robustness of the GAP. Indeed, the GAP is similar to the Unsplittable MFP as explained in Section 4. We introduce the corresponding graph flow model to indicate their similarity, then translate it into an ILP model which reduces the number of constraints by one order of magnitude. Additionally, we propose a more realistic GAP model by taking mutually exclusive gates into account and by integrating taxiing costs in the objective function. Based on tests with real instances of the GAP at Paris-Charles-de-Gaulle international airport, this MFP model outperforms the model introduced in [5] by orders of magnitude.

This paper is organized as follows: Section 1 presents the literature of the GAP, then we provide a formal description of the problem in Section 2. Section 3 discusses the ILP optimization model for the GAP introduced in [5], then Section 4 proposes a new improved ILP optimization model for the GAP. We analyse the performance of our new model with realistic traffic data in Section 5 and we conclude in the last section.

1. Literature Review

The Gate Allocation Problem[1] (GAP) consists in assigning arriving flights with fixed occupancy periods to available compatible gates while maximizing both conveniences to passengers and operational efficiency of airports [7]. [24] was one of the first to introduce the problem in 1974, but the literature is scarce until the 2000s. However, many variants of the GAP have been studied since, as mentioned in [10].

[1] This problem is sometimes called Airport Gate Allocation Problem (AGAP) or Stand Allocation Problem (SAP) in the literature.

If there were no compatibility restriction between gates and aircraft, the corresponding decision problem could be modeled as the *coloring* of an *interval graph*, which is polynomial [17] for the minimization of the number of colors (i.e. gates). Airport gates are generally not equivalent resources though, as they are dimensioned to accommodate specific types of aircraft. Therefore, the set of compatible gates for an aircraft is usually a strict subset of all the available gates and the decision version of the allocation problem is rather a *list-coloring* problem, which is NP-Complete even for interval graphs [3].

Moreover, gates may also be endowed with other secondary features (e.g. compatible airlines, domestic or international, terminal or apron, etc.) which should match the characteristics of the flight and the preferences of airlines as much as possible. These preferences can be modeled as costs associated with each possible assignment, and standard GAP objectives often include the minimization of their sum, which is NP-Hard [11]. From the airport operations perspective, the objective could be to maximize the utilization of the available gates and terminal [24, 28, 21], minimize the flight delay [29] and maximizing the preferences [13] (i.e. certain aircraft should go for particular gates). Other classic objectives can be the passengers walking distance [9, 12, 22] (or other connection means like buses), which is similar to the *Quadratic Assignment Problem*, or the number of towing movements [16].

Even if a flight occupancy is fixed, many real-life factors of uncertainty (e.g. traffic delays, severe weather conditions, equipment failure, etc.), can lead to deviations from the original schedule. If there is not enough buffer time between successive flight occupancies at a given gate, a delay may propagate to other flights, then to other gates, and hinder the operational efficiency of the whole terminal. In order to absorb potential delays and avoid costly disruptions, our study rather focuses on optimizing the *robustness* of the allocation as proposed by [4], which minimizes the variance of idle times of gates to balance and spread them over time and resources. Despite its practical importance, research on the robustness of solutions to the GAP is quite limited.

To solve these very diverse variants of the GAP, many classic combinatorial optimization methods were experimented, depending on the linearity of the model, the size of the instances and the requirements on the execution time of the solver. One of the most used tools is Mixed Integer Programming or Integer Linear Programming (ILP) solvers, like Gurobi or CPLEX, to obtain proven optimal solution like in [5, 16]. Several studies present their ILP model as a Multi-commodity Flow Problem (MFP), like [30] which models re-assignment while minimizing passengers cost. Another complete combinatorial optimization technique, Constraint Programming, was also experimented in [23, 26] to solve the GAP as a *scheduling* problem similar to Fixed Job Scheduling (FJS).

As previously mentioned, all considered variants of the GAP are NP-Complete or NP-Hard, so many publications have suggested various heuristic approaches for solving large instances or non-linear models in reasonable time. [27] developed an algorithm based on the Lagrangian relaxation of the GAP, with subgradient methods, accompanied by a shortest path algorithm and a Lagrangian heuristic to solve an MFP model. Additionally to the ILP model previously mentioned, [5] also proposed a Genetic Algorithm, which is a population-based meta-heuristic, to provide several alternative solutions in case of schedule deviations exceeding the robustness of the implemented solution. [2] proposed a multi-objective heuristic approach base on Breakout Local

Search, with a particular focus on the perturbation strategy. [11] proposed an improved Particle Swarm Optimization algorithm to solve a multi-objective (walking distance of passengers, number of flights at parking apron, etc.) optimization model. [30] used two heuristic algorithms to minimize the weighted sum of the total flight delays. More research work can be found in a recent survey about the GAP [10].

More generally, there are already many studies on the GAP focusing on different criteria. However, few of them except [4, 5] really consider the robustness of the assignment, all the more while minimizing taxiing time simultaneously, which appears to be one of the most important criteria for major international airports like Paris-CDG. Moreover, the resolution process should be efficient enough to be computed in a few minutes, as the GAP does not occur only once (one day before the date of the traffic), but has also to be solved almost in real time when a severe disruption leads to a necessary re-assignment of gates. Using the same decision variables as [5], which does not identify the GAP as a flow problem, we introduce in this article a much more efficient and compact MFP model to optimize its robustness. For an instance with n flights and m gates, we reformulate the original $O(mn^2)$ "continuity" constraints that ensure the feasibility of the assignment, by only $O(mn)$ flow conservation constraints. This tighter model drastically reduces resolution times and allows to optimally solve large-scale instances. Furthermore, we also propose to take taxiing costs, a major concern of large international airports, into account in the objective function, as well as consider subsets of mutually exclusive gates (which occur at Paris-CDG) with additional constraints.

2. The Gate Allocation Problem

After presenting the operational context of the GAP at Paris-CDG and the importance of optimizing the robustness and taxiing times of solutions, we formalize the combinatorial optimization problem independently from its resolution technique and specific model, thoroughly describing its instances components (except mutually dependent gates which are presented in Section 4.4), its feasibility condition and possible objective costs. Eventually, we define a compatibility predicate to help define compact ILP models in Sections 3 and 4.

2.1. Operational Context

The Gate Allocation Problem (GAP) is one of the numerous operational problems that all busy airports have to handle and to optimize every day. It is usually tackled with specific usages and preferences that are difficult to model or list rigorously – some airlines may even manage their own stands area with their own undisclosed strategy. This can lead to a very complex organization, in which the parking stands can have various configurations during each period of the day. Some groups of gates can also be dependent (cf. Section 4.4): only one aircraft can occupy one of them at a time, but these gates cannot be considered as a single one, because they have different entry or exit points, resulting in different taxiing times for the aircraft. Therefore, the GAP is often specific to the considered airport and can be formulated very differently, with various degrees of accuracy.

In the scope of this article, we consider the GAP at Paris-CDG, one of the busiest European international airports. The terminals and their gates (including dependent

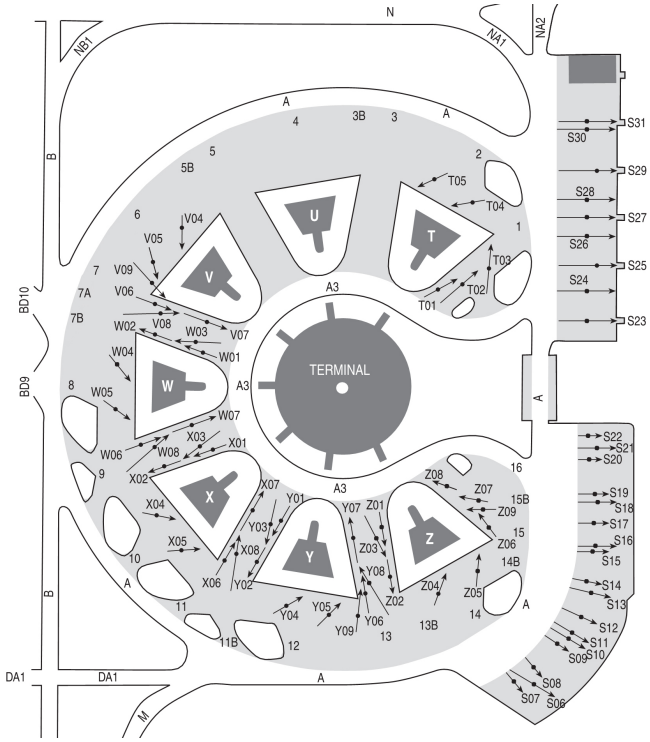


Figure 1: Paris-CDG 1: terminals T, U, V, W, X, Y, Z.

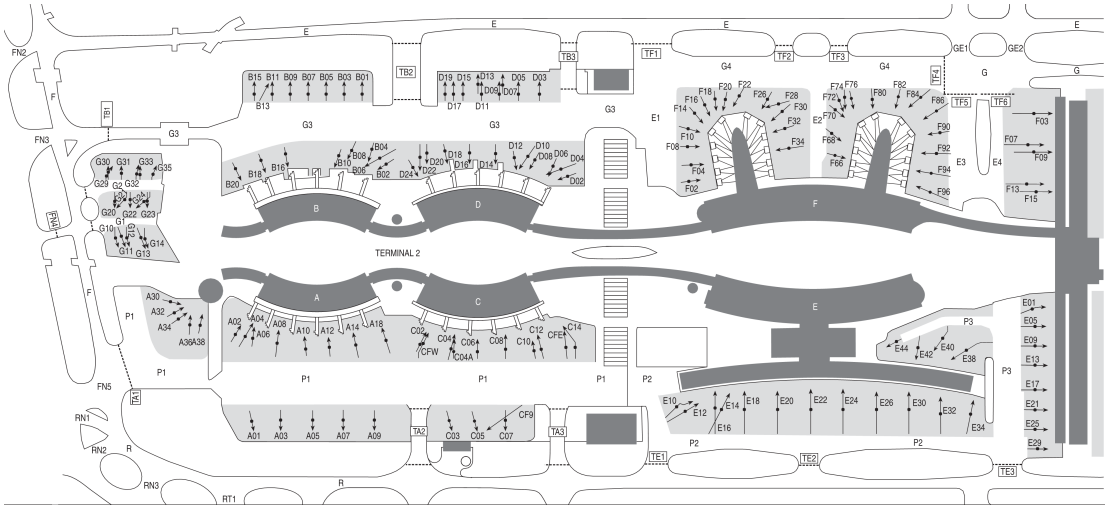


Figure 2: Paris-CDG 2: terminals A, B, C, D, E, F.

ones) are well described, and the types of aircraft that can operationally use a given gate can be easily reconstructed from the available actual traffic records. Figures 1 and 2 are extracted from the official airport charts and show the location of the gates in the main terminals at Paris-CDG. Figure 1 shows how some gates of different terminals can be dependent at Paris-CDG 1, as they cannot be used at the same time: on this figure, arrows represent the position of the aircraft at each stand and, for example, gates V08 and W02 are mutually exclusive. For this reason, instances of the GAP that involve these terminals cannot be solved separately and have to be merged in a single, larger one, even if the precise terminal of each flight is fixed.

Moreover, Paris-CDG has implemented the Airport Collaborative Decision Making (A-CDM) program during the last decade [15]: this program defines which and how accurate information (arrival and departure times, aircraft delay…) can be shared between the different airport stakeholders [14], in order to help them make more efficient decisions in real time. Not only Paris-CDG, many other European and Chinese airports have also obtained A-CDM certification and are currently in the process of its implementation [15, 8]. The A-CDM program also includes a major improvement in the departures management [19]: the delay due to the runways capacities is anticipated and aircraft are preferably delayed at gate, engines off, rather than near the runway after start-up, burning fuel and emitting CO_2. The benefits are twofold: it results in less taxiing traffic, which decreases congestion and nuisance, and also provides a more accurate departure schedule. However, it has a negative effect on gates occupancy, which can significantly increase, especially during peak hours. For these reasons, the robustness of the gate allocation towards additional gate occupancy due to departures delay becomes more and more important for such airports: significant disruptions can be caused in case of gate conflicts between arriving aircraft and delayed departures.

Another important criterion for the GAP is the resulting taxiing times: especially at big airport like Paris-CDG, taxiing times minimization is one of the main objectives for both airlines and airport managers. However, many other constraints like safety and customs services, airlines organization, passengers information, etc. prevent to be able to change the terminal of a flight for this purpose, as it is generally fixed a long time before the day of traffic. Nevertheless, some big terminals offer several distant gates at Paris-CDG, and have different ways to enter or leave, so that taxiing times can significantly vary from a gate to another in the same terminal and must be minimized as well, as shown in Section 4.3.

2.2. Instance Definition

An instance of the basic model of the GAP is defined by:

- $\mathcal{F} = \{f_1, \ldots, f_n\}$: a set of *n flights* (or aircraft) to arrive at the airport during the considered planning horizon and to be assigned to *gates*.
- $\mathcal{G} = \{g_1, \ldots, g_m\}$: a set of *m gates* available during the considered planning horizon.
- $\forall f_i \in \mathcal{F}$:

- f_i^s and f_i^e: the expected *arrival* and *departing times* of flight f_i to its assigned gate, with $f_i^s \leqslant f_i^e$. Note that these arrival and departing times are supposed to be known in advance and not to depend on the gate allocation decisions. Moreover, we suppose that the flights, indexed from 1 to n (noted $[1 \,..\, n]$), are ordered by increasing arrival time:

$$\forall f_i, f_j \in \mathcal{F}, \quad i < j \Rightarrow f_i^s \leqslant f_j^s$$

- $\mathcal{G}_i \subseteq \mathcal{G}$: a subset that specifies the possible gates on which flight f_i can be assigned, depending on the type of the aircraft and the characteristics of the gates. We suppose that $\mathcal{G}_i \neq \varnothing$, otherwise there is no solution.

- $\forall g_k \in \mathcal{G}$:

  - g_k^s and g_k^e: the *opening* and *closing times* of gate g_k. Without loss of generality, we can consider that all gates have the same availability $[g^s \,..\, g^e]$, i.e. $\forall g_k \in \mathcal{G}$, $g_k^s = g^s$ and $g_k^e = g^e$; to represent a period of unavailability of gate g_k between t_1 and t_2 (e.g. for maintenance or while understaffed), we just add an extra flight f_i to $\mathcal{F}$ with $f_i^s = t_1$, $f_i^e = t_2$ and $\mathcal{G}_i = \{g_k\}$. We also consider that $\forall f_i \in \mathcal{F}$, $g^s \leqslant f_i^s$ and $f_i^e \leqslant g^e$, i.e. all flights are scheduled during the period of availability of the gates $[g^s \,..\, g^e]$ (otherwise there is trivially no possible allocation).
  - $\mathcal{F}_k \subseteq \mathcal{F}$: the subset of flights which can be assigned to gate g_k, redundantly defined as $\mathcal{F}_k = \{f_i \in \mathcal{F} \mid g_k \in \mathcal{G}_i\}$ to simplify notations in the following sections.

Subsets $\mathcal{G}_i$ (and $\mathcal{F}_k$) generally have to be built from raw data that specify the *type* of aircraft for each flight (among a given set of types) and the accepted types for each gate. Other characteristics of the flights and the gates can further restrict the possible gates, e.g. the airline, custom or services requirement... In Section 4.4, we refine this model by taking into account mutually exclusive subsets of gates.

2.3. Feasibility

An instance of the GAP is *feasible* if and only if there exists an *assignment* σ of flights to gates such that each flight is assigned a gate from its possible ones (Equation (1)) and flights assigned to the same gate do not overlap (Equation (2)), i.e. $\exists \sigma : \mathcal{F} \mapsto \mathcal{G}$ such that:

$$\forall f_i \in \mathcal{F}, \quad \sigma(f_i) \in \mathcal{G}_i \tag{1}$$

$$\forall f_i, f_j \in \mathcal{F} \text{ s.t. } i < j, \quad \sigma(f_i) = \sigma(f_j) \Rightarrow f_i^e \leqslant f_j^s \tag{2}$$

2.4. Cost

As already mentioned, various kinds of cost can be associated to a feasible allocation. The most usual one is the sum of the individual costs $c_{i,k}^a$ associated to each assignment $\sigma(f_i) = g_k$:

Definition 1 (Assignment Cost). *The* assignment cost *of a solution to an instance of the GAP is defined by:*

$$\text{cost}^a = \sum_{i=1}^{n} c^a_{i,k}$$

However, we focus in this article on the *robustness* of the allocation w.r.t. possible time deviations of the gate occupancies. As proposed by [5], the robustness can be optimized by minimizing the variance of the *idle times* of the gates, such that they are distributed as evenly as possible over the schedule, allowing occupancies to exceed their expected time bounds without incurring gate conflicts. As the overall leeway and the number of idle times are constant (as explained in Section 3), the mean of the idle times is fixed and the minimization of their variance amounts to the minimization of the sum of their squares.

More generally, we can model an idle time as the cost $c^{\text{tr}}_{i,j}$ of the *transition* between two successive flights f_i and f_j assigned to the same gate, considered to be independent of the gate. The transition cost also takes into account costs $c^{\text{tr}}_{0,i}$ of the transition between the opening of each gate, represented by fictive flight f_0, and its first flight f_i, and $c^{\text{tr}}_{i,n+1}$ between the last flight of each gate f_i and its closing, represented by fictive flight f_{n+1}:

Definition 2 (Transition Cost). *The* transition cost *of a solution to an instance of the GAP is defined by:*

$$\text{cost}^{\text{tr}} = \sum_{k=1}^{m} c^{\text{tr}}_{0,\text{fst}(k)} + \sum_{i=1}^{n} c^{\text{tr}}_{i,\text{succ}(i)}$$

where $\text{fst} : [1\,..\,m] \mapsto [1\,..\,n+1]$ is a function such that $\text{fst}(k)$ returns the index (in $[1\,..\,n]$) of the first flight assigned to gate g_k or $n+1$ if the gate is unoccupied, and $\text{succ} : [1\,..\,n] \mapsto [1\,..\,n+1]$ is a function such that $\text{succ}(i)$ returns the index (in $[1\,..\,n]$, or more precisely in $[i+1\,..\,n]$ because flights are ordered by arrival time and therefore $i < \text{succ}(i)$) of the flight that immediately succeeds f_i on its assigned gate, or $n+1$ if it is the last flight.

Definition 3 (Idle Time Duration). *The* idle time duration $I_{i,j}$ *between two successive flights* $f_i \in \{f_0\} \cup \mathcal{F}$ *and* $f_j \in \mathcal{F} \cup \{f_{n+1}\}$ *assigned to the same gate is defined by:*

$$\forall i \in [0\,..\,n],\ \forall j \in [i+1\,..\,n+1],$$

$$I_{i,j} = \begin{cases} f^s_j - f^e_i & \text{if } i \geqslant 1 \wedge j \leqslant n \text{ (successive flights)} \\ f^s_j - g^s & \text{if } i = 0 \wedge j \leqslant n \text{ (first flight)} \\ g^e - f^e_i & \text{if } i \geqslant 1 \wedge j = n+1 \text{ (last flight)} \\ g^e - g^s & \text{otherwise } (i = 0 \wedge j = n+1, \text{ empty gate}) \end{cases}$$

To optimize the robustness of a solution, the transition cost is defined in our model as $c^{\text{tr}}_{i,j} = I^2_{i,j}$ (as explained in Section 3), i.e. the square of the duration of the *idle time* between two successive flights f_i and f_j assigned to the same gate (possibly of null duration if $\text{succ}(i) = j$ and $f^e_i = f^s_j$), or between the opening of a gate and its first flight, or between the last flight of a gate and its closing, or between the opening and closing of an unoccupied gate:

Note that even if this abstract model considers uniform gates, i.e. the transition cost is independent from the gate, the ILP model described in the next section can natively represent gate-dependent costs.

2.5. Flights and Gates Compatibility

For a given instance, not all the transition costs $c^{\text{tr}}_{i,j}$, $\forall i, j \in [0\,..\,n+1], i < j$, need to be defined, but only for the ordered pairs of indices of *compatible* flights, i.e. that have at least a possible gate in common and do not overlap. We define a *compatibility* predicate to specify the corresponding compatible triplets of indices (of two possibly successive flights and a given gate) and help the writing of our ILP models in the following sections, which discard unnecessary decision variables:

Definition 4 (Compatibility Predicate). *The* compatibility *predicate* $\gamma : [0\,..\,n] \times [1\,..\,n+1] \times [1\,..\,m] \mapsto \mathbb{B}$ *on two ordered flight indices i and j and a gate index k is defined by:*

$$\forall i \in [0\,..\,n],\ \forall j \in [i+1\,..\,n+1],\ \forall k \in [1\,..\,m],$$

$$\gamma(i,j,k) = \begin{cases} g_k \in \mathcal{G}_i \cap \mathcal{G}_j \wedge f^e_i \leqslant f^s_j & \text{if } i \geqslant 1 \wedge j \leqslant n \text{ (successive flights)} \\ g_k \in \mathcal{G}_j & \text{if } i = 0 \wedge j \leqslant n \text{ (first flight)} \\ g_k \in \mathcal{G}_i & \text{if } i \geqslant 1 \wedge j = n+1 \text{ (last flight)} \\ \mathit{true} & \text{otherwise } (i = 0 \wedge j = n+1, \text{ empty gate}) \end{cases}$$

f_i, f_j and g_k are said to be compatible *if $\gamma(i,j,k) = true$.*

3. Basic Transition Cost ILP Model

In [5], the author proposes several mathematical models for the GAP, with different kind of optimization criteria, in order to capture the robustness of gate assignment solutions towards flight schedule disruptions. In particular, he proposes to minimize the variance of the *idle times* of the gates to evenly distribute the global leeway over resources and time. This criterion allows for short or long pauses that might be necessary in some instances without hindering the optimization of the rest of the schedule

An idle time is defined as a time period, possibly of null duration, between two successive flights assigned to the same gate and during which the gate is unoccupied. The idle times occurring between the opening of a gate and its first assigned flight, and between its last flight and its closing, are also taken into account as the gates are available to absorb possible delays (or early arrivals), such that there are exactly $n + m$ idle times in any allocation: one after any of the n flights and one between the opening of any of the m gates and its first flight (or its closing for unoccupied ones). The underlying idea of this minimization criterion is that the more the flights are regularly distributed to gates, the less the corresponding gate allocation solution should be sensitive to schedule disruptions.

Indeed, even if the number and the sum of the idle times is considered to be constant in [5], because the arriving and departing times of flights are fixed, their respective durations directly depend on the gate assignment. Hence, the more the idle times are

balanced w.r.t. their durations, the smaller the sum of their squares (or any other convex function). The author proposes an Integer Linear Programming (ILP) model to solve the GAP with the sum of the squares of the idle times as the minimization criterion, with $O(mn^2)$ binary variables $x_{i,j,k}$ stating that flight f_j is the successor of flight f_i on gate g_k, and $O(mn^2)$ constraints.

Before introducing our tighter and more compact flow model in Section 4, which only needs $O(mn)$ constraints, we reformulate this ILP model in the following sections, taking into account the heterogeneity of the gates and discarding variables that can statically be set to 0 because of an incompatibility (see Section 2.5), which was not explicitly done in [5].

3.1. Decision Variables

A binary variable is defined to denote the successive assignment of a pair of compatible flights to the same compatible gate, using compatibility predicate γ of Definition 4:

$\forall i \in [1\,..\,n]$, $\forall j \in [i+1\,..\,n]$, $\forall k \in [1\,..\,m]$, s.t. $\gamma(i,j,k)$,

$$x_{i,j,k} = \begin{cases} 1 & \text{if flights } f_i \text{ and } f_j \text{ are successively assigned to gate } g_k \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

$\forall j \in [1\,..\,n]$, $\forall k \in [1\,..\,m]$, s.t. $\gamma(0,j,k)$,

$$x_{0,j,k} = \begin{cases} 1 & \text{if } f_j \text{ is the first flight assigned to gate } g_k \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

$\forall i \in [1\,..\,n]$, $\forall k \in [1\,..\,m]$, s.t. $\gamma(i,n+1,m)$,

$$x_{i,n+1,k} = \begin{cases} 1 & \text{if } f_i \text{ is the last flight assigned to gate} g_k \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

$\forall k \in [1\,..\,m]$,

$$x_{0,n+1,k} = \begin{cases} 1 & \text{if there is no flight assigned to gate } g_k \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

If the instance is feasible, there is exactly one $x_{i,j,k}$ variable equal to 1 for each flight f_i, so the allocation σ can trivially be deduced from the decision variables:

$$\forall f_i \in \mathcal{F},\ \exists!\, j \in [i+1\,..\,n],\ \exists!\, k \in [1\,..\,m],\ \text{s.t. } x_{i,j,k} = 1, \quad \sigma(f_i) = k$$

or, alternatively, with the linear expression:

$$\sigma(f_i) = \sum_{j \in [i+1..n]} \sum_{\substack{k \in [1..m] \\ \gamma(i,j,k)}} k x_{i,j,k}$$

Note that variables $x_{i,j,k}$ are similar to the ones of typical ILP models of the *Multi-commodity Flow Problem* [25], where they are interpreted as the the amount of a commodity k flowing through an arc from node i to node j in a network. Nevertheless, as shown hereafter, the model proposed in [5] is not formulated as a flow problem model.

3.2. Constraints

Covering constraints: each flight has to be assigned to exactly one compatible gate. In order to guarantee this statement, the following constraints are introduced to ensure that exactly one flight is assigned immediately before a given flight f_j (Equation (7)), and only one flight is assigned immediately after a given flight f_i (Equation (8)):

$$\forall j \in [1 \,..\, n], \sum_{i \in [0..j-1]} \sum_{\substack{k \in [1..m] \\ \gamma(i,j,k)}} x_{i,j,k} = 1 \tag{7}$$

$$\forall i \in [1 \,..\, n], \sum_{j \in [i+1..n+1]} \sum_{\substack{k \in [1..m] \\ \gamma(i,j,k)}} x_{i,j,k} = 1 \tag{8}$$

Moreover, Equation (9) is added to specify that there are exactly $n + m$ binary variables that should be equal to 1 in a feasible solution:

$$\sum_{i \in [0..n]} \sum_{j \in [i+1..n+1]} \sum_{\substack{k \in [1..m] \\ \gamma(i,j,k)}} x_{i,j,k} = n + m \tag{9}$$

Continuity constraint: the successive assignments have to be feasible. Indeed, if a pair of flights (f_i, f_j) with $j \leqslant n$ (i.e. f_i is not the last flight on its gate) are successively assigned to a gate g_k, there must be a flight $f_{j'}$ with $j' \in [j + 1 \,..\, n + 1]$ such that the pair of flights $(f_j, f_{j'})$ are also successively assigned to g_k, but there cannot be any flight $f_{j''}$ such that the pair of flights $(f_j, f_{j''})$ are successively assigned to a gate $g_{k'} \neq g_k$. This continuity constraint is modeled with Equation (10), which states that, for any pair of flights (f_i, f_j) and any gate g_k, either $x_{i,j,k} = 1$, and then no $x_{j,j',k'}$ with $j' > j$ and $k' \neq k$ can be equal to 1 ($\sum_{k' \in [1..m] | k' \neq k, j' \in [j+1..n+1]} x_{j,j',k'} = 0$); either $x_{i,j,k} = 0$, and then at most one $x_{j,j',k'}$ with $j' > j$ and $k' \neq k$ equals 1 (which is always true due to Equations (7) and (8)).

$$\forall i \in [0 \,..\, n-1],\ \forall j \in [i+1 \,..\, n],\ \forall k \in [1 \,..\, m], \gamma(i,j,k), \quad x_{i,j,k} + \sum_{j' \in [j+1..n+1]} \sum_{\substack{k' \in [1..m] \setminus \{k\} \\ \gamma(j,j',k')}} x_{j,j',k'} \leqslant 1 \tag{10}$$

Note that these continuity constraints are not formulated as a flow balance constraint but as the linearization of the logical constraint:

$$\forall i \in [0 \,..\, n-1],\ \forall j \in [i+1 \,..\, n],\ \forall k \in [1 \,..\, m], \gamma(i,j,k), \quad \forall j' \in [j+1 \,..\, n+1],\ \forall k' \in [1 \,..\, m] \setminus \{k\}, \gamma(j,j',k'), \quad x_{i,j,k} = 1 \Rightarrow x_{j,j',k'} = 0$$

These $O(mn^2)$ constraints dominate the number of constraints of this model.

Resource constraints: each gate can be assigned to at most one flight at a time. Due to the previous continuity constraints, it is sufficient to check that each gate should have at most one assigned first flight. Therefore Equation (11) is defined by:

$$\forall k \in [1 \,..\, m], \sum_{\substack{j \in [1..n+1] \\ \gamma(0,j,k)}} x_{0,j,k} \leqslant 1 \tag{11}$$

Domain constraints. The decision variables are binary as mentioned above:

$$\forall i \in [0\,..\,n],\ \forall j \in [i+1\,..\,n+1], \forall k \in [1\,..\,m],\ \gamma(i,j,k),$$
$$x_{i,j,k} \in \{0,1\} \quad (12)$$

3.3. Objective

As previously mentioned, the objective function proposed in [5] consists in minimizing the sum of the squares of the idle times, in order to find robust solutions towards operational arrival and departure delays. Therefore, the following minimization criterion is proposed:

$$\min \sum_{i\in[0..n]} \sum_{j\in[i+1..n+1]} \sum_{\substack{k\in[1..m] \\ \gamma(i,j,k)}} I^2_{i,j} x_{i,j,k} \quad (13)$$

As exactly one binary variable $x_{0,j,k}$ can be equal to 1, $\forall k \in [1\,..\,m]$, i.e. the corresponding flight f_j is the first on gate g_k, and exactly one $x_{i,j,k}$ can be equal to 1 $\forall i \in [1\,..\,n]$, because of the constraint of Equation (8) (i.e. flight f_i has a unique successor f_j), the cost defined by objective of Equation (13) corresponds to the *transition cost* of Definition 1:

$$\text{cost}^{\text{tr}} = \sum_{k\in[1..m]} \sum_{\substack{j\in[1..n+1] \\ \gamma(0,j,k)}} c^{\text{tr}}_{0,j} x_{0,j,k} + \sum_{i\in[1..n]} \sum_{j\in[i+1..n+1]} \sum_{\substack{k\in[1..m] \\ \gamma(i,j,k)}} c^{\text{tr}}_{i,j} x_{i,j,k} \quad (14)$$

Note that with this definition, the gates are not required to be uniform w.r.t. to the transition cost which could depend on k:

$$\text{cost}^{\text{tr}} = \sum_{i\in[0..n]} \sum_{j\in[i+1..n+1]} \sum_{\substack{k\in[1..m] \\ \gamma(i,j,k)}} c^{\text{tr}}_{i,j,k} x_{i,j,k}$$

4. Multi-commodity Flow Model

As mentioned in the previous section, the GAP is similar to a *Minimum-Cost Flow Problem* [25], more precisely an *Unsplittable Multi-commodity Flow Problem* (UMFP) [1] with additional constraints, which translates into a tighter and smaller ILP model. We first introduce the corresponding graph flow model and then describe its translation to ILP as an improvement of the basic model constraints. Eventually, we refine this new GAP flow model to take into account other operational costs and constraints: we propose to combine the robustness with the taxiing cost within the objective and to prevent mutually exclusive gates to be used simultaneously (cf. Section 2.1).

4.1. Graph Model

The GAP can be seen as a kind of Unsplittable Multi-commodity Flow Problem (UMFP) [1] with unit capacity for each arc and unit demand for each commodity. But our problem is defined on a directed *multigraph* $G = (V, E)$ with *parallel arcs* between nodes v_i and v_j (corresponding to flights f_i and f_j) whenever $f^e_i \leqslant f^s_j \ \wedge\ \left|\mathcal{G}_i \cap \mathcal{G}_j\right| > 1$ that restrict the compatible commodities rather than their amount. Arcs are *labeled* with the corresponding compatible gate index k (represented by a distinct color in the example of Figure 3) s.t. $\gamma(i, j, k)$ are specified by triplets of $V \times V \times [1\,..\,m]$. More classically, the opening of gates are the *sources* of G and their closing are the *sinks*.

Definition 5 (Compatibility (Multi)Graph). *The* compatibility (multi)graph *of a GAP instance* $G = (V, E, w, d)$ *with* weight $w : E \mapsto \mathbb{R}_{\geqslant 0}$ *and* supply $d : V \mapsto \mathbb{R}$ *is defined by:*

- $V = V_{\mathcal{F}} \cup V^s_{\mathcal{G}} \cup V^e_{\mathcal{G}}$*:*

$$V_{\mathcal{F}} = \{v_i,\ \forall i \in [1\ ..\ n]\} \tag{15a}$$

$$V^s_{\mathcal{G}} = \{v^k_0,\ \forall k \in [1\ ..\ m]\} \tag{15b}$$

$$V^e_{\mathcal{G}} = \{v^k_{n+1},\ \forall k \in [1\ ..\ m]\} \tag{15c}$$

  $V_{\mathcal{F}}$ corresponds to regular flights, $V^s_{\mathcal{G}}$ to fictive openings of the gates and $V^e_{\mathcal{G}}$ to their endings, and $|V| = 2m + n$.

- $E = E_{\mathcal{F}} \cup E^s_{\mathcal{G}} \cup E^e_{\mathcal{G}} \cup E_{\mathcal{G}}$*:*

$$E_{\mathcal{F}} = \{(v_i, v_j, k),\ \forall i \in [1\ ..\ n-1],\ \forall j \in [i+1\ ..\ n],\ \forall k \in [1\ ..\ m] \quad s.t.\ \gamma(i, j, k)\} \tag{16a}$$

$$E^s_{\mathcal{G}} = \{(v^k_0, v_j, k),\ \forall j \in [1\ ..\ n],\ \forall k \in [1\ ..\ m],\ s.t.\ g_k \in \mathcal{G}_j\} \tag{16b}$$

$$E^e_{\mathcal{G}} = \{(v_i, v^k_{n+1}, k),\ \forall i \in [1\ ..\ n],\ \forall k \in [1\ ..\ m],\ s.t.\ g_k \in \mathcal{G}_i\} \tag{16c}$$

$$E_{\mathcal{G}} = \{(v^k_0, v^k_{n+1}, k),\ \forall k \in [1\ ..\ m]\} \tag{16d}$$

  $E_{\mathcal{F}}$ corresponds to idle times between two regular successive flights, $E^s_{\mathcal{G}}$ to idle times between the opening of the gates and their first flights, $E^e_{\mathcal{G}}$ to the ones between their last flights and the closing of the gates, and $E_{\mathcal{G}}$ to empty gates. $|E|$ is equal to the number of compatible pairs of (real) flights plus $2\sum_{k\in[1..m]} |\mathcal{F}_k|$, the arcs from the opening and closing of the gates to compatible flights, i.e. in $O(2mn + n^2)$ in the worst case.

- *Each arc has a* unit capacity *1 and a* weight *(or* cost*) w defined by:*

$$w((v_i, v_j, k)) = c^{\text{tr}}_{i,j} = I^2_{i,j}$$

  (also valid for start and end vertices v^k_0 and v^k_{n+1}). Note that our model is generic enough to take into account non uniform resources (gates) for which the cost may depend on k as shown in Section 4.3 where taxiing times are added to the idle times.

- *Each node has a* supply *value d defined by:*

$$d(v) = \begin{cases} 1 & \text{if } v \in V^s_{\mathcal{G}} \text{ (openings)} \\ -1 & \text{if } v \in V^e_{\mathcal{G}} \text{ (closings)} \\ 0 & \text{otherwise (i.e. } v \in V_{\mathcal{F}}\text{, regular flights)} \end{cases}$$

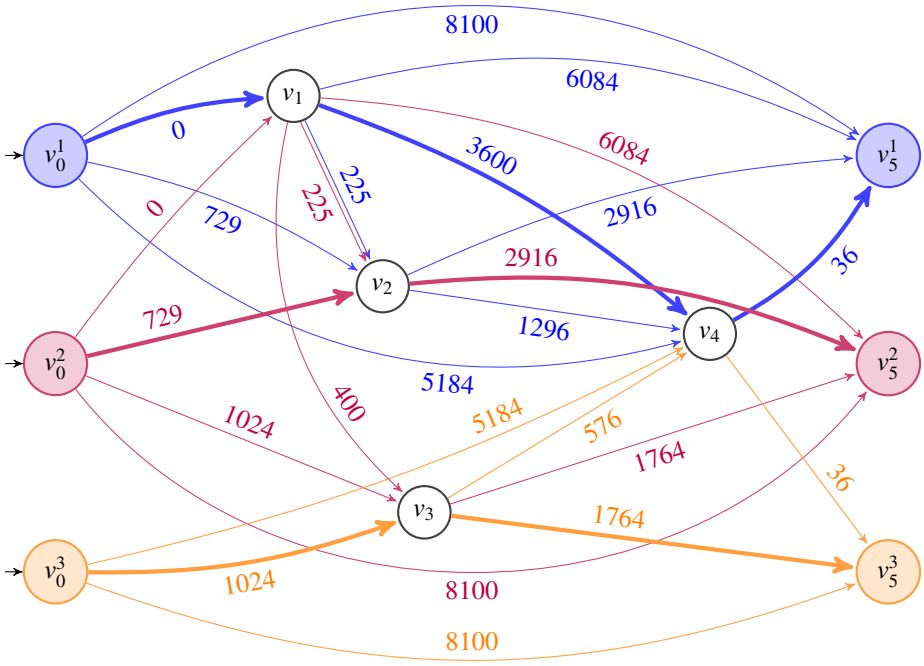


Figure 3: Coloured representation of the *compatibility graph* of Example 1 where "commodities", i.e. the three gates, are represented by different colors. Fictive nodes v_0^k and v_{n+1}^k are associated respectively to the opening and ending of gate k, and nodes v_i, $\forall i \in [1\,..\,4]$, to actual flights. A coloured arc joins two nodes if they can succeed each other on the corresponding gate, therefore multiple arcs may join the same pair of nodes (as between nodes v_1 and v_2). They are labelled with their weight (i.e. the cost of the associated idle time) indicated in hundreds of min^2. The flow corresponding to the optimal solution is shown with bold arcs.

As an example, Figure 3 represents the graph model of the following GAP instance:

Example 1 (GAP Instance with 4 Flights and 3 Gates). $\mathcal{F} = \{f_1, f_2, f_3, f_4\}$, $\mathcal{G} = \{g_1, g_2, g_3\}$ *with* g^s *= 06:00,* g^e *= 21:00 (in hh:mm format) and:*

i	f_i^s	f_i^e	$\mathcal{G}_i$
1	*06:00*	*08:00*	$\{g_1, g_2\}$
2	*10:30*	*12:00*	$\{g_1, g_2\}$
3	*11:20*	*14:00*	$\{g_2, g_3\}$
4	*18:00*	*20:00*	$\{g_1, g_3\}$

Feasibility. Like a standard flow problem, an instance of the GAP is *feasible* iff there exists a *binary flow* $\phi : E \mapsto \{0, 1\}$ such that the imbalance between outgoing and incoming flows at each node is equal to the supply:

$$\forall v \in V, \quad \sum_{e \in E_v^+} \phi(e) - \sum_{e \in E_v^-} \phi(e) = d(v) \tag{17}$$

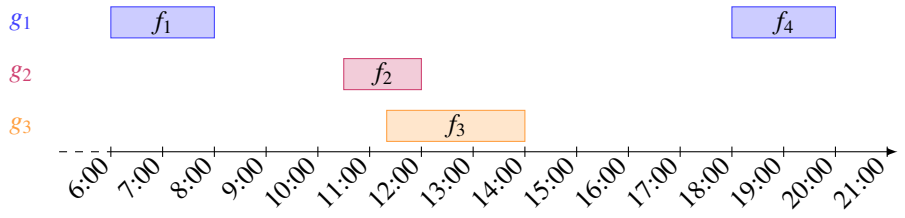


Figure 4: Gantt diagram of the optimal solution to Example 1.

with $E_v^+ = \{e, \forall e = (u, v, k) \in E\}$ the *incoming* arcs of node v and $E_v^- = \{e, \forall e = (v, w, k) \in E\}$ its *outgoing* ones. However, for the GAP, the flow must also satisfy the following additional constraints.

Additional constraints. In general, a solution to a UMFP is not a solution to the GAP, because all internal nodes (corresponding to real flights) must be covered by the flow (Equation (18)) and paths must be "monochromatic" (Equation (19)):

- A solution to the GAP must cover all flights of $\mathcal{F}$, therefore all nodes of $V_{\mathcal{F}}$ must have a unit inflow and outflow:

$$\forall v \in V_{\mathcal{F}}, \sum_{e \in E_v^+} \phi(e) = \sum_{e \in E_v^-} \phi(e) = 1 \tag{18}$$

- The same gate must be used to enter and leave a node, therefore the flow imbalance Equation (17) must be expressed individually for each gate (or "color"):

$$\forall v_i \in V_{\mathcal{F}}, \forall g_k \in \mathcal{G}_k, \sum_{(v_{i'}, v_i, k) \in E_v^+} \phi((v_{i'}, v_i, k)) = \sum_{(v_i, v_{i''}, k) \in E_v^-} \phi((v_i, v_{i''}, k)) \tag{19}$$

The flow of each edge (v_i, v_j, k) (or (v_0^k, v_j, k), (v_i, v_{n+1}^k, k), (v_0^k, v_{n+1}^k, k)) corresponds to the binary decision variable $x_{i,j,k}$ (or $x_{0,j,k}$, $x_{i,n+1,k}$, $x_{0,n+1,k}$) of the basic ILP model described in Section 3.1, thus the assignment of flights to gate can be deduced accordingly.

Objective. The cost of a flow is defined by:

$$\sum_{e \in E} w(e)\phi(e) \tag{20}$$

As mentioned in the previous paragraph, this cost corresponds to the one of the basic ILP model described in Section 3.3, and is therefore also equal to the transition cost of Definition 1. Figure 4 depicts the Gantt diagram of an optimal solution to Example 1 with cost: $\text{cost}^{\text{tr}} = 10069 \times 100\text{min}^2$.

4.2. ILP Model

The previously described GAP model as a UMFP with additional constraints can be translated in an ILP model equivalent to the basic model of Section 3, based on the same

$x_{i,j,k}$ binary decision variables and cost expression but with tighter and fewer constraints. The UFMP flow imbalance constraints of Equation (17) ensures that all sources (or gate openings) have a unit outflow as their supply value is 1:

$$\forall k \in [1\,..\,m], \quad \sum_{\substack{j\in[1..n+1]\\ \gamma(0,j,k)}} x_{0,j,k} = 1 \tag{21}$$

and that all sinks (gate closings) have a unit inflow (supply value of -1):

$$\forall k \in [1\,..\,m], \quad \sum_{\substack{i\in[0..n]\\ \gamma(i,n+1,k)}} x_{i,n+1,k} = 1 \tag{22}$$

which amounts to $O(m)$ constraints.

The covering constraints of Equation (18) for nodes of $V_{\mathcal{F}}$ (i.e. regular flights) lead to Equation (8) of the basic model, i.e. all internal nodes have a unit outflow:

$$\forall i \in [1\,..\,n], \quad \sum_{j\in[i+1..n+1]} \sum_{\substack{k\in[1..m]\\ \gamma(i,j,k)}} x_{i,j,k} = 1$$

which amounts to $O(n)$ constraints.

More specifically, the constraints of Equation (19) ensuring that the path of a commodity is monochromatic, can be directly translated to the flow balance constraint:

$$\forall i \in [1\,..\,n],\ \forall k \in [1\,..\,m], \quad \sum_{\substack{i'\in[0..i-1]\\ \gamma(i',i,k)}} x_{i',i,k} = \sum_{\substack{i''\in[i+1..n+1]\\ \gamma(i,i'',k)}} x_{i,i'',k} \tag{23}$$

which amounts to $O(mn)$ constraints.

Other constraints of the basic model that ensure that nodes of $V_{\mathcal{F}}$ have a unit inflow (cf. Equation (7)), the continuity of path (cf. Equation (10)) or the number of variables equal to 1 (cf. Equation (9)) would be redundant in our flow model and are therefore omitted. Overall, this model has only $O(nm)$ constraints in the worst case instead of $O(mn^2)$ for the basic one.

4.3. Cost with Combined Taxiing

Even if the robustness of the schedule is our main operational concern in this study, the fuel consumption during taxiing has become of paramount importance in the last decade. Generally, the taxiing time cost of a flight does not only depend on the assigned gate, but also on the flight itself. As mentioned in Section 5.1, big international airport (like Paris-CDG) usually offer several runways for departure and arrival, thus the taxiing times of two flights assigned on the same gate could differ from each other, as their distance to a given gate is different if they do not use the same runway.

To attempt to minimize the fuel consumption induced by the amount of taxiing times, we introduce a new parameter $t_{i,k}$ that represents the taxiing time cost of aircraft f_i if assigned to gate g_k. Note that in the results presented in Section 5.5, $t_{i,k}$ is a taxiing time that only depends on the runway used by f_i and the gate, but it could also depend

on the type of aircraft to represent fuel consumption or CO_2 emission. If flight f_i is assigned to g_k, there is exactly one $x_{i,j,k}$ variable equal to 1, $\forall j \in [i+1 \,..\, n+1]$, and all other $x_{i,j,k'}$ with $k' \neq k$ are equal to 0, so the taxiing cost c_i^t of f_i can be expressed as:

$$\forall i \in [1 \,..\, n], \quad c_i^t = \sum_{j \in [i+1..n+1]} \sum_{\substack{k \in [1..m] \\ \gamma(i,j,k)}} t_{i,k} x_{i,j,k} \tag{24}$$

and the global taxiing cost for all flights as:

$$\text{cost}^t = \sum_{i \in [1..n]} c_i^t = \sum_{i \in [1..n]} \sum_{j \in [i+1..n+1]} \sum_{\substack{k \in [1..m] \\ \gamma(i,j,k)}} t_{i,k} x_{i,j,k} \tag{25}$$

which can be integrated into the transition (or robustness) cost summation as is apparent from Equation (14). Obviously, no taxiing cost corresponds to the openings of the m gates taken into account in the first term of the sum of Equation (14), but we can obtain a uniform expression similar to Equation (13) by setting $t_{0,k} = 0$, $\forall k \in [1 \,..\, m]$:

$$\text{cost}^t = \sum_{i \in [0..n]} \sum_{j \in [i+1..n+1]} \sum_{\substack{k \in [1..m] \\ \gamma(i,j,k)}} t_{i,k} x_{i,j,k} \tag{26}$$

To combine both costs in the objective, we choose a linear scalarization with a single weight parameter $\alpha \in [0, 1]$, and normalize them because the range of robustness (min^2 of gates idle times) is a priori much larger than the one of taxiing times (expressed in min in our study, but which can be chosen as any convenient operational cost, e.g. fuel consumption). To normalize the costs, we first compute their respective lower and upper bounds as follows.

Idle Time Upper Bound. The GAP can be relaxed as a Bin Packing (BP) problem where gates serve as bins of capacity $g^e - g^s$ and flights f_i as items of volume $f_i^e - f_i^s$. The classical items configuration (stacked and divided as needed to completely fill the first bins) that corresponds to the BP lower bound on the number of bins, can be used to compute a simple upper bound for the sum of any idle time convex cost, in particular their square:

$$\text{ub}^{\text{tr}} = m'\delta(g)^2 + (\delta(g) - r)^2 \tag{27}$$

where function $\delta : \mathcal{F} \cup \mathcal{G} \mapsto \mathbb{R}_{\geqslant 0}$ returns the duration of its parameter, i.e. $g^e - g^s$ for any gate and $f_i^e - f_i^s$ for flight f_i, and is extended to take a set parameter and returns the sum of the duration of its elements, e.g. $\delta(\mathcal{F}) = \sum_{f_i \in \mathcal{F}} \delta(f_i)$, and $m' = m - \left\lceil \frac{\delta(\mathcal{G}) - \delta(\mathcal{F})}{\delta(g)} \right\rceil$ the number of empty gates and $r = (\delta(\mathcal{G}) - \delta(\mathcal{F})) \mod \delta(g)$ the time taken by the flights scheduled on the last non-empty gate.

Idle Time Lower Bound. Still relaxing the GAP as a kind of BP problem, the lower bound for the overall sum of the squares of idle times is reached when all idle times have the same size and are evenly distributed along a continuous virtual gate made of all the gates stacked one after another. As mentioned in Section 3, there are exactly $n + m$

idle time periods (possibly of null duration), therefore we can compute the following global lower bound on the idle time cost:

$$\mathrm{lb}^{\mathrm{tr}} = (n+m)\left\lfloor \frac{\delta(\mathcal{G}) - \delta(\mathcal{F})}{n+m} \right\rfloor^2 \tag{28}$$

Taxiing Time Bounds. As mentioned in Section 5.1, each departure and arrival at Paris-CDG airport is usually assigned to a fixed terminal and a fixed runway. Hence, the taxiing time bounds for a flight f_i are the minimal and maximal times taken between any of its compatible gates $\mathcal{G}_i$ and both its fixed arrival and departure runways. Therefore its upper bound can be obtained as:

$$\mathrm{ub}^{\mathrm{taxi}} = \sum_{i\in[1..n]} \max_{g_k\in\mathcal{G}_i} (tt(\mathrm{ar}_i, g_k) + tt(g_k, \mathrm{dr}_i)) \tag{29}$$

where ar_i and dr_i are the arrival and departure runways of flight f_i, and tt is a function that returns the taxiing time between the location of its two parameters. The lower bound $\mathrm{lb}^{\mathrm{taxi}}$ is trivially computed by substituting the min function to the max one in Equation (29).

The combined cost that takes into account both robustness and taxiing time, cost^α, can then be defined with the normalized terms (e.g. $\frac{c}{\mathrm{ub}-\mathrm{lb}}$ for cost c and its bounds) or equivalently (i.e. with the same optimal solution) by:

$$\forall\alpha\in[0,1],\quad \mathrm{cost}^\alpha = \sum_{i\in[0..n]} \sum_{j\in[i+1..n+1]} \sum_{\substack{k\in[1..m]\\ \gamma(i,j,k)}} \left(\alpha\rho^{\mathrm{tr}} t_{i,k} + (1-\alpha)\rho^{\mathrm{taxi}} I^2_{i,j}\right) x_{i,j,k} \tag{30}$$

with $\rho^{\mathrm{tr}} = \mathrm{ub}^{\mathrm{tr}} - \mathrm{lb}^{\mathrm{tr}}$ and $\rho^{\mathrm{taxi}} = \mathrm{ub}^{\mathrm{taxi}} - \mathrm{lb}^{\mathrm{taxi}}$ the respective ranges of both costs. The coefficient of the linear scalarization α can then be adjusted to influence the respective importance of the two criteria. Note that this new definition of the objective does not change the rest of our model.

4.4. Mutually Exclusive Gates

Depending on the layout of the terminals, some subsets of the gates cannot simultaneously be assigned to more than one flight, as they would otherwise block each other. Therefore, a set $\mathcal{H}$ of r subsets of $\mathcal{G}$ representing groups of mutually exclusive gates is additionally specified in the data: $\mathcal{H} = \{\mathcal{H}_1, \ldots, \mathcal{H}_r\}$, with $\mathcal{H}_h \subseteq \mathcal{G}$, $\forall h \in [1\,..\,r]$. We could consider such a subset as a single gate, but gates are not equivalent w.r.t. the taxiing time and their contribution to the cost must be individually accounted for.

To add this constraint in our flow model, we define $O(mn)$ auxiliary variables $y_{i,k}$ that represent the allocation of flight f_i to gate $g_k \in \mathcal{G}_i$:

$$\forall i \in [1\,..\,n],\ \forall g_k \in \mathcal{G}_i,\quad y_{i,k} = \sum_{\substack{j\in[i+1..n+1]\\ \gamma(i,j,k)}} x_{i,j,k} \tag{31}$$

Then for each mutually exclusive gates subset $\mathcal{H}_h$, we constrain that all overlapping flights pairs cannot be assigned to two of its gates simultaneously:

$$\forall h \in [1 \,..\, r],\ \forall g_k, g_l \in \mathcal{H}_h,\ k < l,$$
$$\forall f_i \in \mathcal{F}_k,\ \forall f_j \in \mathcal{F}_l,\ [f_i^s, f_i^e[\cap [f_j^s, f_j^e[\neq \varnothing,$$
$$y_{i,k} + y_{j,l} \leqslant 1 \quad (32)$$

Groups of mutually exclusive gates $\mathcal{H}_h$ usually comprise only a small number of gates (w.r.t. m), therefore the worst-case $O(n^2 \sum_{h\in[1..r]} |\mathcal{H}_h|^2)$ additional constraints does not burden much the model.

5. Results

The models described in the previous sections have been implemented and run over real traffic data at Paris-CDG. All experiments were carried out on a standard workstation consisting of a 2.0 GHz Intel® Xeon® 32-core processor with 48 GB of memory running Debian GNU/Linux 9.6. We used the Gurobi Commercial Optimizer 8.1 [18] for the Integer Linear Programming models. Computation times to solve a 1-day instance using flow models for densest terminals are less than 100 s, and about 1 s is needed to optimally solve most other instances. A detailed report is provided in Section 5.3.

In this section, we first describe how we have extracted the traffic demand and the available gates from actual traffic records at Paris-CDG airport in Section 5.1 to generate a set of Gate Allocation Problem instances. Then, in Section 5.3 we compare the computation times of the different methods used to solve these instances. The implementation of DMAN can be very sensitive to the robustness of the allocation, as the slightest delay in a fragile allocation might quickly propagate and disrupt the departure sequence. Moreover, whereas ATC is concerned mainly with the feasibility of the assignment, our approach also focuses on minimizing the taxiing cost, which is directly related to fuel consumption, while preserving enough robustness in the schedule. Hence, Sections 5.4 and 5.5 discuss the benefits of a robust gate allocation in terms of idle time distribution and taxiing time differences between initial and robust traffic.

The results presented in this section refer to the actual allocation implemented by ATC on the day of traffic as "Initial", to the models presented in Section 3 as "JORS", and to the models presented in Section 4 as "Flow". Each model has two variants: R when only robustness is being maximized, and R+T when robustness is being maximized and taxiing time is being minimized.

5.1. Data

The data sample used for this study is available at:

`http://recherche.enac.fr/~wangrx/gap`

It consists in:

- actual traffic demands during a whole heavy month (July, 2017) at Paris-CDG airport;

Table 1: Number of flights and gates by terminal (per day in average). The density is the ratio of the average number of flights by the number of gates.

Terminal	Flights	Gates	Density
A	47	17	2.8
B	15	10	1.5
C	14	21	0.7
D	84	18	4.6
E	111	23	4.8
F	185	27	6.8
I	46	51	0.9
J	72	20	3.6
K	53	19	2.8
L	18	7	2.6
M	13	10	1.3
Q	50	17	3.0
R	12	12	1.0
TVWXYZ	96	43	2.2
Total	816	295	2.8

- a set of allowed aircraft types for each gate;
- an estimation of the average taxiing times needed between each runway and each gate (from each runway to each gate for arrivals and from each gate to each runway for departures).

The traffic demand is extracted from the data sample which provides for each flight: the arrival time at the gate, the departure time from the gate, the aircraft type, the runway and the gate used. The corresponding allocation serves as a reference solution named "Initial" in the following sections. Table 1 gives the number of flights (in average per day) and the number of gates by terminal in this data sample.

In some cases (probably due to some gate-to-gate movements that were not recorded), one of the two times was missing. The data were completed as follows: when the arrival (resp. departure) time is missing, the flight is considered at the gate from (resp. until) the beginning (resp. end) of the day if there is no other flight occupying the same gate before (resp. after) it; otherwise, the flight is considered at the gate 30 min before (resp. after) its departure (resp. arrival) time, which did not cause any gate conflict in our data set.

The sets of allowed aircraft types at each gate is also deduced from the actual traffic sample: we consider that the only aircraft types allowed at a given gate are the ones that actually used this gate during the month.

The estimation of taxiing times between gates and runways are computed using the *standard* routing rules described in the operations manual of the airport, after considering the average taxiing speed on each taxiway portion, as measured on actual radar records.

5.2. Bi-objective Parameterization

For the R+T strategies, we have tested different values for α ranging from 0 to 1, in objective function Equation (30), with 0 corresponding to optimizing taxiing time only and 1 to optimizing robustness only. Figure 5 shows how each individual cost is impacted by the value of the α parameter, for the 7th July 2017 instance (other instances exhibit a very similar pattern).

In Figure 5a, the value of both robustness cost and total taxiing time is plotted with respect to α. We notice that robustness cost is rapidly increasing for values of α below 0.5. On the other hand, with α over 0.9 (or even 0.8 for many instances), there is not much gain in the taxiing time. Thus, to achieve an interesting compromise between those two criteria, we recommend values of α in the interval $[0.5, 0.8]$. In the following sections, results are presented for $\alpha = 0.7$.

Figure 5b provides a Pareto visualization of the same data. In an operational context, this parameter could be adjusted by airport managers to meet their own performance objectives.

5.3. Computation Times and Quality of the Solutions

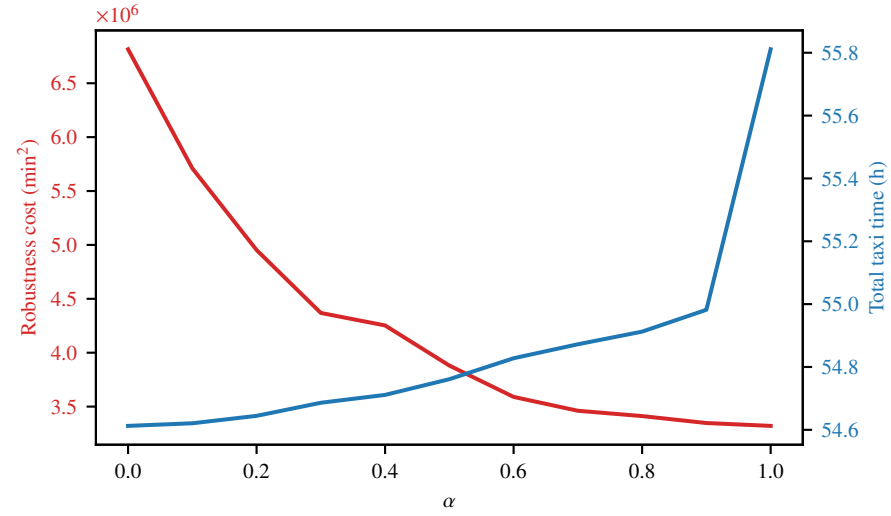
We report in this section the performances of the various models with the criteria described in Sections 3 and 4 to solve the instances at Paris-CDG airport described in Section 5.1. We first show that an optimal solution to small instances of the problem (i.e. at the least busy terminals) can be easily reached by all models. We then discuss the weaknesses of JORS-R and JORS-R+T models on larger instances compared to their Flow-R and Flow-R+T counterparts.

Figure 6 displays the time to optimally solve a 1-day instance (averaged over the entire month of July, 2017) for each terminal and each implemented model. The four models are distinguished by their color in the figure. For the sake of readability, the figure has been separated in two parts, with Figure 6a focusing on small instances (with a density lower than 3 flights per gate) and Figure 6b showing results for larger instances.

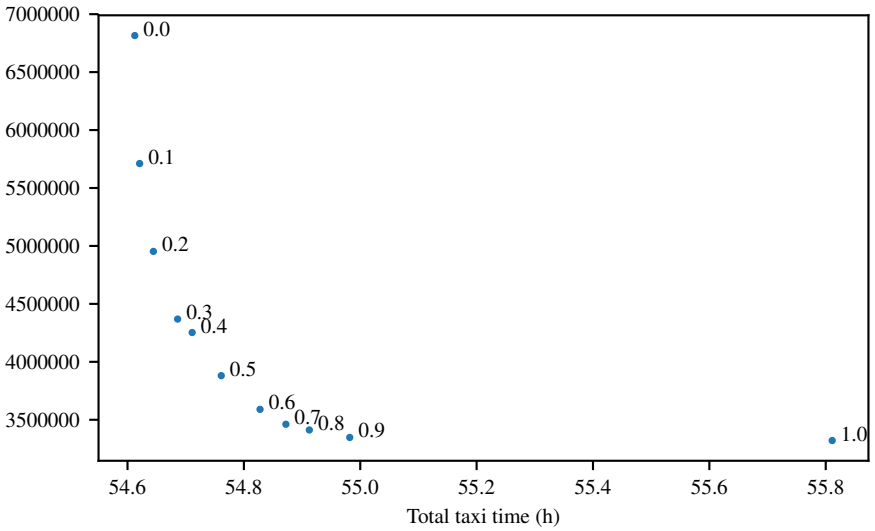
For small instances (i.e. terminals A, B, C, I, L, M, R, and TVWXYZ[2]), Figure 6a shows that all models can obtain an optimal solution, and the optimality is always proved in very limited time (less than 1 s with Flow-R and Flow-R+T models). Flow models systematically outperform their JORS counterparts in terms of computation times, sometimes by two orders of magnitude (e.g. on terminals A and I). Interestingly, taking taxiing-time into account with the Flow-R+T model does not add to the computation time compared to the robustness-only Flow-R model.

Large instances (i.e. terminal D, E, F, J, K and Q) are significantly more challenging, as shown in Figure 6b. Considering only robustness as an optimization criterion, both Flow-R and JORS-R models still find solutions, again with an advantage of one order of magnitude in favor of the Flow-R model. For the largest and densest terminal F, the Flow-R model reaches optimality in less than 1 min, where JORS-R needs more than 20 min to obtain the same result.

[2]Terminals T, V, W, X, Y, and Z must be solved simultaneously as there are mutually exclusive gates belonging to different terminals in this set. Dependent gates information is accessible at `http://recherche.enac.fr/~wangrx/gap`.

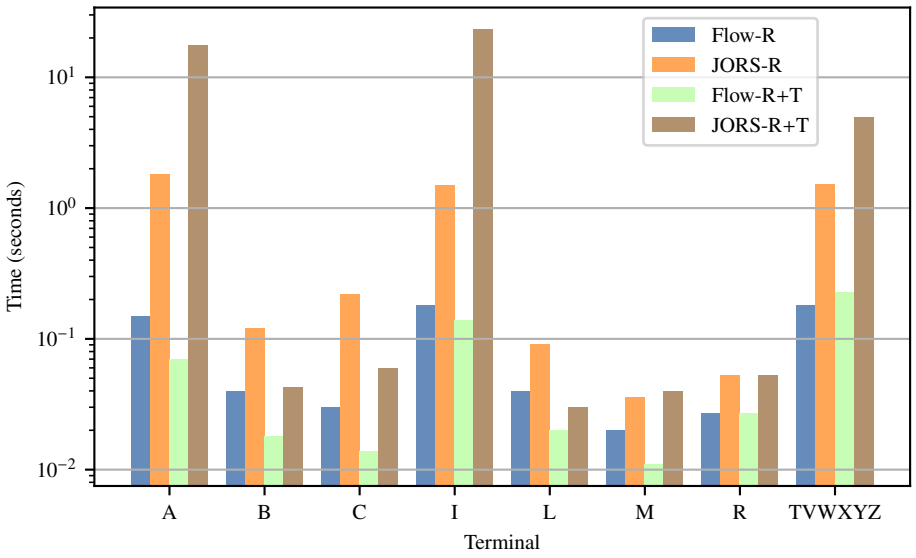


(a) Influence of α on individual objectives.

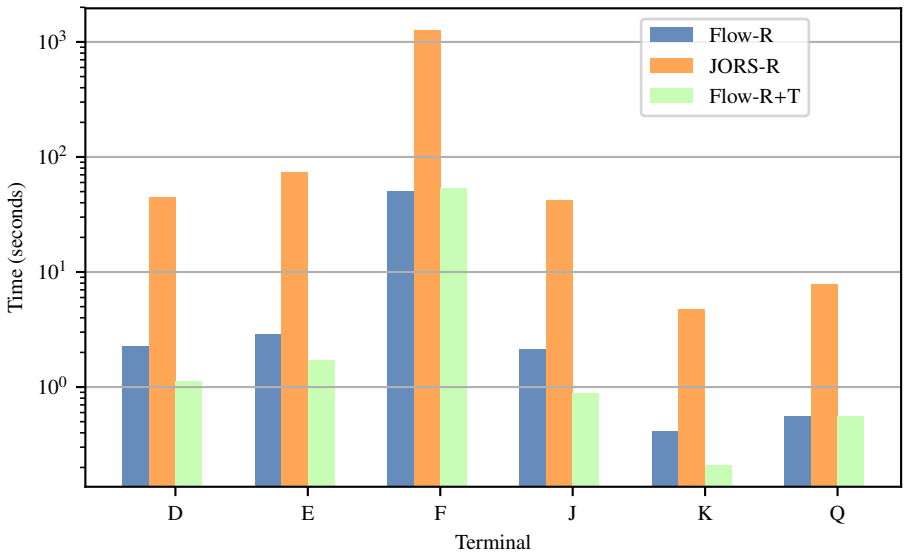


(b) Visualization of the Pareto front. The value of α is indicated next to each dot.

Figure 5: Influence of α on bi-objective optimization (7th July, 2017 at Paris-CDG airport).



(a) Easy instances.



(b) Hard instances.

Figure 6: Comparison of computation times to find an optimal solution with JORS and Flow model.

Table 2: Average gap between the cost of optimal solution and the best solution found with JORS-R+T model within a 5 min time limit.

Terminal	K	Q	J	D	E	F
Best solution	+1.08 %	+1.11 %	+4.07 %	+3.11 %	+4.63 %	–

Table 3: Average idle time between two consecutive aircraft at the same gate.

Allocation	Initial	Flow-R	Flow-R+T
Average idle time	120 min	157 min	157 min

Additionally taking taxiing time into account with the Flow-R+T model does not incur additional costs compared to Flow-R: even the largest instances are optimally solved in less than 1 min. The JORS-R+T model, however, could not optimally solve any single instance within a reasonable time (the computations were stopped after 3 h). Even for the medium-sized terminal K, only 25 % of instances could be optimally solved. For instances at the largest and busiest terminal F, finding a first solution with JORS-R+T model took at least 30 min. If we impose a 5 min time limit for computation, all instances are optimally solved with the Flow-R+T model. As explained before, the JORS-R+T model could not reach (and even less prove) optimality; however, in many instances, it could still find good solutions within the time limit. Table 2 shows the discrepancy to the optimal cost by terminal for the JORS-R+T model, obtained after 5 min of computation. There is no figure for Terminal F as no solution could be found within the time limit. For other terminals, the solutions found are close to optimal, with approximately 1 % to 5 % additional cost.

As the Flow-R and Flow-R+T models are strictly and significantly faster than JORS-R and JORS-R+T respectively, the following sections only present results for the first two, as well as comparisons with the initial gate allocation extracted from the data (see Section 5.1). Also, these computation times are suitable for an allocation process performed the day before operations, or even on the day of operations if some event breaks the solution and a new allocation needs to be devised. In such a case, an additional criterion could be introduced that would help find a new solution as close as possible to the previous one, so that the gate allocation would remain mostly unchanged. This is, however, not covered in the current study.

5.4. Robustness of the Gate Allocation

To assess the robustness of a gate allocation, we measure the idle times occurring in the corresponding schedule, as too small idle times are likely to cause issues on the day of operations. Table 3 shows the average idle time: the average for the Initial allocation is 2 h, when the average for both Flow-R and Flow-R+T allocations are 2 h 40 min.

More interestingly, Figure 7 shows the distribution of idle times for the entire airport. Those distributions are plotted for the Initial allocation (in blue) and for both Flow-R (in orange) and Flow-R+T (in green) models. As we can see, the Initial allocation leads to a high number of short idle times: more than 1000 idle times between 0 and 10 min and

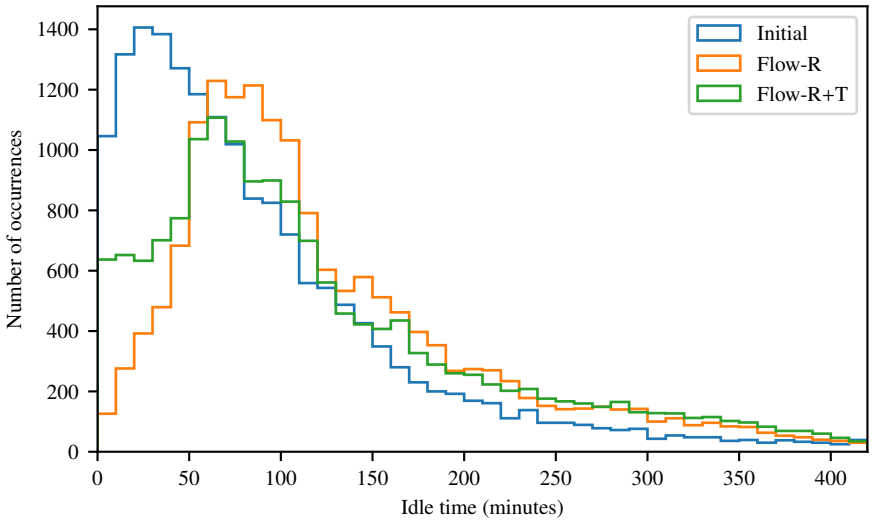


Figure 7: Distribution of idle times for all terminals over all tested instances.

Table 4: Average idle time between two consecutive aircraft at the same gate for terminal F only.

Allocation	Initial	Flow-R	Flow-R+T
Average idle time	65 min	73 min	72 min

more than 1300 idle times between 10 and 20 min. The Flow-R allocation drastically reduces the number of idle times of less than an hour, thus increasing the robustness of the allocation as expected. With this model, only 150 idle times are less than 10 min. With the Flow-R+T model, the solutions are still significantly more robust than the Initial allocation, with again a high reduction of short idle times. Overall, such idle times can absorb a large part of the uncertainty that can be observed on flight arrival and departure times, with no re-allocation needed in most cases.

The presence of long idle times in all distributions (more than 2 h) comes from the fact that all terminals are aggregated in this figure, even the smallest ones with a very low density.

Table 4 and Figure 8 present similar results, focusing on terminal F only. Even for this particularly busy terminal, both Flow-R and Flow-R+T models manage to provide a solution that significantly reduces occurrences of smaller idle times. Also, there are very few idle times of more than 2 h, as the density (flights-to-doors ratio) is very high at this terminal.

5.5. Impact on Taxiing Time

Another main objective of the GAP is the minimization of taxiing times. As mentioned in Section 2, Paris-CDG large terminals offer several gates with various ways

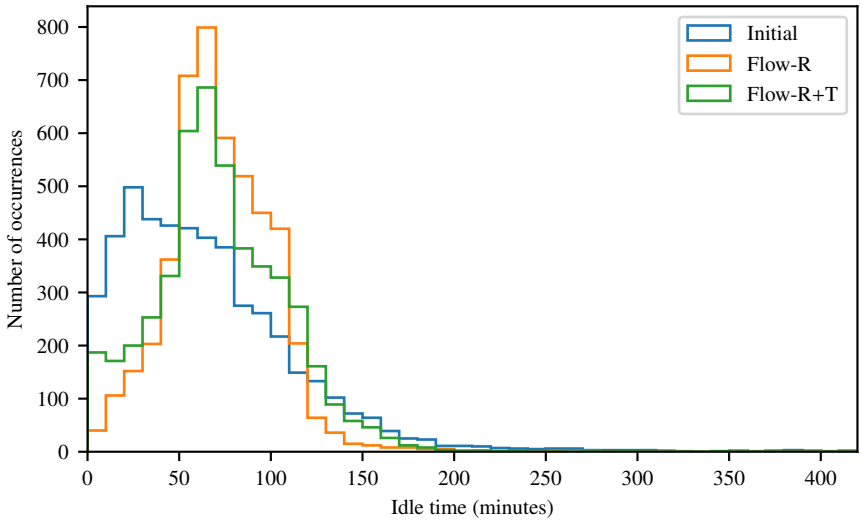


Figure 8: Distribution of idle times at terminal F over all tested instances.

to enter or leave the terminal, so that taxiing times can be very different from one gate to another in the same terminal.

Table 5 gives the total amount of saved taxiing time during the month of July 2017, comparing the Initial (actual) gate allocation with the two gate allocation strategies (Flow-R and Flow-R+T). With the Flow-R strategy, the total taxiing time is almost unchanged, which shows that the gain in robustness commented in Section 5.4 does not affect taxiing times. With the Flow-R+T strategy, a slight reduction by 2.67 % of the taxiing time is observed, which represents 180.97 h of saved taxiing time during the month, i.e. 6 h per day in average. As discussed in Section 5.4, this gain in taxiing time does not significantly penalize the robustness of the gate allocation.

Results are also detailed by terminal to point out the ones with the greatest reduction in taxiing time. Terminal R appears to be the most influenced one with a decrease of 14.54 % in taxiing time, but this result is not significant as the number of flights in this terminal is very low. More important terminals, from A to F, reach a decrease between 0.63 % to 4.08 %, which represents 77 h of saved taxiing time during July 2017, contributing to 42 % of the total saved taxiing time.

5.6. Merging Terminals

As described in Section 5.3, Flow-R and Flow-R+T models can solve instances of all the terminals in less than 50 s. In this part, we test the performances of our best models on bigger instances by merging the terminals at Paris-CDG 1 and 2 (see Figure 2), where passengers can walk from one terminal to another. These instances also provide an estimation of the gain that can be expected in terms of robustness and taxiing time, were the terminal of each flight not fixed — which would be outside of the scope of current

Table 5: Taxiing time comparison between Initial, Flow-R and Flow-R+T. The total gain in taxiing time is for the entire month of July, 2017.

Terminal	Flights	Taxiing time (and difference w.r.t. Initial)		
		Initial	Flow-R	Flow-R+T
A	2,372	8 min 35 s	8 min 34 s (−0.19 %)	8 min 14 s (−4.08 %)
B	723	10 min 39 s	10 min 39 s (0.00 %)	10 min 35 s (−0.63 %)
C	1,176	8 min 10 s	8 min 11 s (+0.20 %)	8 min 01 s (−1.84 %)
D	4,344	10 min 59 s	10 min 58 s (−0.15 %)	10 min 46 s (−1.97 %)
E	5,657	8 min 33 s	8 min 33 s (0.00 %)	8 min 22 s (−2.14 %)
F	9,530	10 min 36 s	10 min 35 s (−0.15 %)	10 min 27 s (−1.42 %)
G	136	8 min 37 s	8 min 39 s (+0.39 %)	8 min 28 s (−1.74 %)
H	343	11 min 15 s	11 min 09 s (−0.89 %)	10 min 58 s (−2.52 %)
I	1,284	13 min 01 s	12 min 57 s (−0.51 %)	12 min 32 s (−3.71 %)
J	3,968	9 min 49 s	9 min 44 s (−0.85 %)	9 min 29 s (−3.40 %)
K	2,541	9 min 45 s	9 min 52 s (+1.20 %)	8 min 58 s (−8.03 %)
L	860	9 min 40 s	9 min 42 s (+0.34 %)	9 min 26 s (−2.41 %)
M	533	10 min 44 s	10 min 39 s (−0.78 %)	10 min 32 s (−1.86 %)
N	93	12 min 02 s	11 min 59 s (−0.42 %)	11 min 56 s (−0.83 %)
P	303	11 min 26 s	11 min 17 s (−1.31 %)	10 min 51 s (−5.10 %)
Q	2,653	9 min 39 s	9 min 32 s (−1.21 %)	9 min 22 s (−2.94 %)
R	399	7 min 27 s	7 min 26 s (−0.22 %)	6 min 22 s (−14.54 %)
S	130	8 min 27 s	8 min 25 s (−0.39 %)	8 min 03 s (−4.73 %)
T	712	7 min 43 s	7 min 48 s (+1.08 %)	7 min 41 s (−0.43 %)
V	678	8 min 22 s	8 min 12 s (−1.99 %)	8 min 05 s (−3.39 %)
W	1,505	6 min 32 s	6 min 39 s (+1.79 %)	6 min 30 s (−0.51 %)
X	1,363	6 min 42 s	7 min 11 s (+7.21 %)	6 min 40 s (−0.50 %)
Y	548	7 min 59 s	7 min 54 s (−1.04 %)	7 min 52 s (−1.46 %)
Z	431	7 min 17 s	7 min 12 s (−1.14 %)	7 min 08 s (−2.06 %)
All	42,282	9 min 37 s	9 min 36 s (−0.02 %)	9 min 21 s (−2.67 %)
Total			−0.33 h	−180.97 h

operational constraints at Paris-CDG airport. Three different sets of merged terminals were considered:

- TVWXYZ at Paris-CDG 1;
- ABCD or ABCDEF at Paris-CDG 2.

Table 6: Computation time for merged terminals instances with the Flow model.

Terminals	Flights	Gates	Time R	Time R+T
ABCD	167	59	22.7 s	6.8 s
ABCDEF	461	109	409.6 s	2631.9 s
TVWXYZ	95	42	2.6 s	2.94 s

Table 6 gives the mean computation time needed to find and prove an optimal solution with models Flow-R and Flow-R+T. The resolution time ranges from a few seconds to 45 min, which remains reasonable for a gate allocation process that is usually performed the day before operations. If the allocation process had to be run again to accommodate delays on the day of operations, the instances would be much smaller (by not considering the entire day of traffic), so that computing times would probably be small enough. The most time-consuming instance comes from the merging of all Paris-CDG 2 terminals (i.e. A to F).

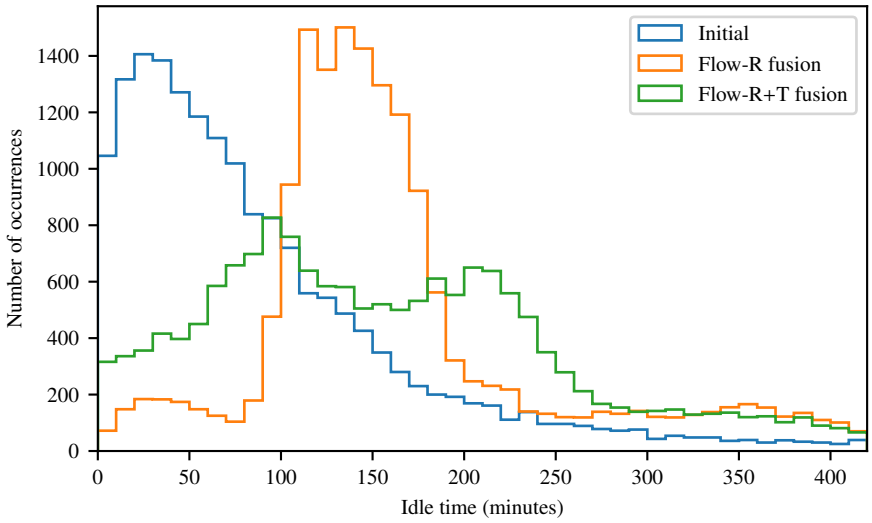


Figure 9: Distribution of idle times for all terminals with ABCDEF and TVWXYZ combinations.

Figure 9 shows the distribution of idle times resulting from the merging of terminals TVWXYZ at Paris-CDG 1 and ABCDEF at Paris-CDG 2. Compared to the distributions

Table 7: Taxiing time comparison between the different gate allocation strategies.

Terminal	Flights	Taxiing time (and difference w.r.t. Initial)		
		Initial	Robust	Robust + Taxiing
ABCD	8,615	9 min 54 s	9 min 50 s (−0.67 %)	8 min 36 s (−13.13 %)
ABCDEF	23,802	9 min 52 s	9 min 55 s (+0.68 %)	8 min 24 s (−14.86 %)
TVWXYZ	5,237	7 min 12 s	7 min 20 s (+0.18 %)	7 min 04 s (−1.85 %)
others	13,243	10 min 06 s	10 min 03 s (−0.49 %)	9 min 40 s (−4.3 %)
All	42,282	9 min 37 s	9 min 39 s (+0.37 %)	8 min 38 s (−10.1 %)
Total			+0.33 h	−684.04 h

obtained in Section 5.4 (see Figure 7), we can observe a new significant improvement in the robustness of the gate allocation with the two tested criteria, as the number of short idle times (i.e. less than one hour) is further decreased.

Table 7 gives the variation of the average taxiing time for the three sets of merged terminals. We can notice the significant taxiing time reduction that can be expected using the Robust+Taxiing criterion at Paris-CDG 2: 13.13 % by merging terminals A, B, C and D, and 14.86 % by adding terminals E and F. Globally, with merged terminals, the R+T criterion applied on the whole airport allows to save 684 h of taxiing time during the month of July 2017, which corresponds to almost 23 h per day. These results demonstrate how beneficial the relaxing of the terminal allocation of some flights could be at Paris-CDG airport, in terms of gate allocation robustness and saved taxiing time.

Conclusion

In this study, we present and test a new efficient multi-commodity flow model for the gate allocation problem at big airports. To the best of our knowledge, this is the first multi-commodity flow model that can handle complex constraints like mutually exclusive gates, while optimizing both the robustness and the taxiing times, which are obviously two main criteria for large airports. The efficiency of the model is tested on actual instances, extracted from traffic records at Paris-CDG airport, during one of its heaviest month (July, 2017). All the instances was successfully solved with the ILP solver.

We compare two versions of the min-cost flow model (Flow-R and Flow-R+T) to the two corresponding versions of a more classical model (JORS-R and JORS-R+T). Instances of small density are optimally solved in less than one minute by all of the four models. For larger instances, we defined a 300 s limit for the time computation, in order to make the approach compatible with a real-time context (e.g. gate re-allocation in case of operational disruption). With this limit, the Flow-R and Flow-R+T models still provide optimal solutions for all the terminals in less than 100 s, while JORS-R and JORS-R+T models not always reach optimality, and sometimes do not even find a solution.

Results show the overall improvement in robustness and taxiing time after gate allocation: the idle times distributions are better balanced, with a significant reduction

of the number of small idle times, and the taxiing times are reduced by a total of 180 h during the whole month, which represents 2.67 % of the total taxiing time.

Flow-R and Flow-R+T models can also be applied on larger instances after merging all the terminals of Paris-CDG 2 (i.e. A, B, C, D, E, F) and of Paris-CDG 1 (i.e. T, V, W, X, Y, Z). The results show that the robustness of the gate allocation is still improved while the taxiing time is decreased by 684 h (i.e. 10.1 % of the total taxiing time), moreover our min-cost flow model still finds an optimal solution in a reasonable time for these larger instances.

In further works, we project to test the gate allocation into some more realistic simulations of the airport traffic, considering the conflicts that can appear between aircraft entering or exiting the gates areas. This will provide a more accurate estimation of the gain that can be expected in terms of gates conflicts and taxiing times diminution with our gate allocation model.

Funding

This article is supported by the funds of basic scientific research in central universities, Civil Aviation University of China. [No. 3122021084].

References

[1] Alvelos, F. and Valério de Carvalho, J. M. [2003], Comparing branch-and-price algorithms for the unsplittable multicommodity flow problem, *in* 'International Network Optimization Conference (INOC'2003)', Évry/Paris, pp. 7–12.

[2] Benlic, U., Burke, E. K. and Woodward, J. R. [2017], 'Breakout local search for the multi-objective gate allocation problem', *Computers & Operations Research* **78**, 80–93.

[3] Biró, M., Hujter, M. and Tuza, Z. [1992], 'Precoloring extension. I. Interval graphs', *Discrete Mathematics* **100**(1), 267 – 279.

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

[5] Bolat, A. [2001], 'Models and a genetic algorithm for static aircraft-gate assignment problem', *Journal of the Operational Research Society* **52**(10), 1107–1120.

[6] Bouras, A., A. Ghaleb, M., S. Suryahatmaja, U. and M. Salem, A. [2014], 'The airport gate assignment problem: A survey', *The Scientific World Journal* .

[7] Bouras, A., Ghaleb, M. A., Suryahatmaja, U. S. and Salem, A. M. [2014], 'The airport gate assignment problem: a survey', *The scientific world journal* **2014**.

[8] China ACDM [2018], 'China airport ACDM implementation route'.
URL: *www. chinaacdm. com/ home/ page/ acdm_ in_ china*

[9] Daş, G. S. [2017], 'New multi objective models for the gate assignment problem', *Computers & Industrial Engineering* **109**, 347–356.

[10] Daş, G. S., Gzara, F. and Stützle, T. [2020], 'A review on airport gate assignment problems: Single versus multi objective approaches', *Omega* **92**, 102146.

[11] Deng, W., Zhao, H., Yang, X., Xiong, J., Sun, M. and Li, B. [2017], 'Study on an improved adaptive PSO algorithm for solving multi-objective gate assignment', *Applied Soft Computing* **59**, 288–302.

[12] Dijk, B., Santos, B. and Pita, J. [2019], 'The recoverable robust stand allocation problem: a GRU airport case study', *OR Spectrum* **41**, 615–639.

[13] Dorndorf, U., Drexl, A., Nikulin, Y. and Pesch, E. [2007], 'Flight gate scheduling: State-of-the-art and recent developments', *Omega* **35**(3), 326–334.

[14] EUROCONTROL [2015], *A Guide to the Network Manager Operations Centre*. **URL:** *www. skybrary. aero/ bookshelf/ books/ 3525. pdf*

[15] EUROCONTROL [2017], *Airport CDM Implementation Manual*, EUROCONTROL Airport CDM Team.

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

[17] Gupta, U. I., Lee, D. T. and Leung, J. Y. [1982], 'Efficient algorithms for interval graphs and circular-arc graphs', *Networks* **12**(4), 459–467.

[18] Gurobi Optimization, L. [2018], 'Gurobi optimizer reference manual'.

[19] Huet, D. and Pickup, S. [2017], Local and network impact assessment of airport collaborative decision making (a-cdm), *in* '12th USA/Europe Air Traffic Management Research and Developpment Seminar'.

[20] Kim, S. H. [2013], Airport Control Through Intelligent Gate Assignment, PhD thesis, Georgia Institute of Technology.

[21] Li, W. and Xu, X. [2012], Optimized assignment of airport gate configuration based on immune genetic algorithm, *in* 'Measuring Technology and Mechatronics Automation in Electrical Engineering', Springer, pp. 347–355.

[22] Mokhtarimousavi, S., Talebi, D. and Asgari, H. [2018], 'A non-dominated sorting genetic algorithm approach for optimization of multi-objective airport gate assignment problem', *Transportation Research Record: Journal of the Transportation Research Board* **2672**(23), 59–70.

[23] Simonis, H. [2007], 'Models for global constraint applications', *Constraints* **12**(1), 63–92.

[24] Steuart, G. N. [1974], 'Gate position requirements at metropolitan airports', *Transportation Science* **8**(2), 169–189.

[25] Wang, I.-L. [2018], 'Multicommodity network flows: A survey, part I: Applications and formulations', *International Journal of Operations Research* **15**, 145–153.

[26] Wang, R. and Barnier, N. [2018], Propagation of idle times costs for fixed job scheduling, *in* '2018 IEEE 30th International Conference on Tools with Artificial Intelligence (ICTAI)', pp. 718–725.

[27] Yan, S. and Chang, C.-M. [1998], 'A network model for gate assignment', *Journal of Advanced Transportation* **32**(2), 176–189.

[28] Yan, S. and Huo, C.-M. [2001], 'Optimization of multiple objective gate assignments', *Transportation Research Part A: Policy and Practice* **35**(5), 413–432.

[29] Yan, S. and Tang, C.-H. [2007], 'A heuristic approach for airport gate assignments for stochastic flight delays', *European Journal of Operational Research* **180**(2), 547–567.

[30] Zhang, D. and Klabjan, D. [2017], 'Optimization for gate re-assignment', *Transportation Research Part B: Methodological* **95**, 260–284.