Transportation Research Part C 124 (2021) 102937

Contents lists available at ScienceDirect

Transportation Research Part C

journal homepage: www.elsevier.com/locate/trc

An optimization–simulation closed-loop feedback framework for modeling the airport capacity management problem under uncertainty

Paolo Scala[a,*], Miguel Mujica Mota[b], Cheng-Lung Wu[c], Daniel Delahaye[d]

[a] AMSIB, Amsterdam University of Applied Sciences, Netherlands
[b] Aviation Academy, Amsterdam University of Applied Sciences, Netherlands
[c] School of Aviation, UNSW Sydney, Australia
[d] ENAC LAB, Ecole Nationale de l'Aviation Civile, France

ARTICLE INFO

Keywords:
Opt–Sim
Airport capacity
Airspace aircraft sequencing
Runway capacity
Taxiway capacity
Terminal capacity
Stochastic analysis

ABSTRACT

This paper presents an innovative approach that combines optimization and simulation techniques for solving scheduling problems under uncertainty. We introduce an Opt–Sim closed-loop feedback framework (Opt–Sim) based on a sliding-window method, where a simulation model is used for evaluating the optimized solution with inherent uncertainties for scheduling activities. The specific problem tackled in this paper, refers to the airport capacity management under uncertainty, and the Opt–Sim framework is applied to a real case study (Paris Charles de Gaulle Airport, France). Different implementations of the Opt–Sim framework were tested based on: parameters for driving the Opt–Sim algorithmic framework and parameters for driving the optimization search algorithm. Results show that, by applying the Opt–Sim framework, potential aircraft conflicts could be reduced up to 57% over the non-optimized scenario. The proposed optimization framework is general enough so that different optimization resolution methods and simulation paradigms can be implemented for solving scheduling problems in several other fields.

1. Introduction

Most of real-world systems are stochastic in nature. Systems that involve different operators (human factors), different resources and exposure to external factors (such as weather, politics, societal among others) are characterized by uncertainty. This makes many real-world systems difficult to manage, analyze and predict. As such, they are often modeled as stochastic systems. When analyzing stochastic systems with the objective of improving their performance, optimization techniques are mostly used to solve many of these problems. However, especially in the aviation sector, many optimization models are developed using exact methods (Beasley et al., 2000; Smeltink et al., 2005). The main drawback of these models is that they do not consider the variability in system operations. Instead, they tend to simplify real-world operations by considering them as deterministic problems. This simplification makes these analytical models unsuitable for real-world implementation because their results are often less resilient to changes and make them unreliable for decision making and forecasting.

In order to overcome these limitations, simulation techniques are commonly used to consider system variability and interactions between different entities within a system, by allowing them to evolve so as to observe emerging system dynamics. Hence, simulation

* Corresponding author.
E-mail addresses: p.m.scala@hva.nl (P. Scala), m.mujica.mota@hva.nl (M.M. Mota), c.l.wu@unsw.edu.au (C.-L. Wu), delahaye@recherche.enac.fr (D. Delahaye).

https://doi.org/10.1016/j.trc.2020.102937
Received 31 May 2020; Received in revised form 6 November 2020; Accepted 17 December 2020
Available online 28 December 2020

techniques have become a powerful tool for system analysis, ‘what-if’ scenario analysis, bottleneck analysis, and decision-making studies (Banks et al., 2000). Simulation alone, however, is insufficient to optimize system operations due to the complexity and combinatorial nature of system configurations, such as the case of an airline schedule. Therefore, the combination of these two techniques have been explored to take advantage of the synergy that can be achieved through both techniques. They can complement each other: optimization is able to find (sub)optimal solutions, even in a big state space, and simulation is able to reproduce possible solutions and consider a wide range of stochastic elements in a system. This combined approach makes optimized solutions more feasible in a real-world applications.

In the past few years, simulation and optimization have been extensively used, both separately and jointly. The most common application of these two techniques is to use simulation as a validation tool to test the feasibility of optimized solutions or to couple simulation and optimization models for improving feasible solutions. In this paper, simulation and optimization techniques are jointly applied to modeling a stochastic airport system with the objective of improving system reliability and maintaining system performance.

To this purpose, a closed-loop feedback framework, the Opt–Sim framework, was developed. The main contribution of this paper is that we developed an algorithmic architecture to solve the airport capacity management problem simultaneously with any scheduling problem considering the uncertainty of operations in which a simulation component was embedded in an optimization framework. The Opt–Sim framework takes advantage of the simulation model to evaluate proposed solutions from the optimization model and test solutions in a simulated airport system. The developed methodology was applied to the airport capacity management problem by considering a real case study of Paris Charles de Gaulle Airport as a testing environment. The airport capacity management problem, in this context, was focused on the optimization of aircraft landing sequence in the terminal airspace and the mitigation of potential aircraft movement conflicts and congestion on the airport grounds.

The paper is organized as follows. In Section 2, the state of the art concerning implementations of simulation and optimization techniques in aviation problems is presented. Section 3, describes the general architecture of the Opt–Sim framework developed for this study. Section 4, introduces the airport capacity management problem considered in this paper. In Section 5, the main elements that were implemented within the Opt–Sim framework for solving the specific problem tackled in this paper were described. Section 6, presents the case study used for testing the Opt–Sim framework, while in Section 7 numerical experiments were conducted. Finally, in the last section, the paper is concluded and prospects on future research is suggested.

2. State of the art

In this section, a review of simulation and optimization techniques used in combination has been conducted. A subsection is reserved for a review of the main optimization–simulation applications to aviation problem, while, in the last subsection, the contributions of this paper are stated.

2.1. Optimization together with simulation techniques

In the last decades, many researchers have exploited the advantages by using these techniques together, as several reviews about optimization and simulation have been published (Fu, 2002; Fu et al., 2005; Jian and Henderson, 2015; Juan et al., 2015). Jian and Henderson (2015), introduces simulation–optimization and motivates its use by considering two different perspectives: from the simulation perspective, as comparing the effects of different decision variables over complex models; and from the optimization perspective, as including stochasticity in a deterministic model, in order to better capture the real-life system.

In the review of Fu (2002), different approaches for dealing with simulation–optimization were discussed such as: stochastic programming (e.g. using Monte Carlo simulation), and the development of simulation commercial software where optimization packages were integrated. In this review, it was pointed out the need of a standard framework (mentioned in the article as a testbed) where it would be possible to compare various algorithms.

One of the main challenge when implementing simulation–optimization, as stated by Fu et al. (2005) is the trade-off between search and evaluation (or diversification and intensification as it was mentioned in Juan et al. (2015)). Search refers to the ability of an optimization program (exact method, heuristic, etc.) being capable of searching throughout the entire state space;

while evaluation, refers to the capability of the simulation of evaluating the performance of the objective function without being biased by the stochastic nature of their variables. In order to achieve this, the simulation model should run many replications, which, in turn, will increase the computational burden.

Juan et al. (2015), introduced a specific approach within simulation–optimization called sim-heuristics. In this approach, they have integrated simulation into a metaheuristic framework, trying to come up with a good solution by including stochasticity to the problem. In their approach, they have designed a two-stage algorithm where the simulation part has been employed in two different ways: fast simulation process and intensive simulation process.

First, they obtain a set of optimized solutions, by running a metaheuristic resolution method, which are identified as “promising”; these solutions become “elite” after they have been evaluated and ranked according to the simulation performance.

In their approach, their main assumption lies in the fact that they consider two different levels of uncertainty within the problem, moderate and high. Based on this, they identify the so-called “promising” solutions as the ones who are obtained in scenarios with moderate uncertainty. One benefit of this approach is that it manages the trade-off between search and evaluation to avoid too much computational effort; however, the main assumption regarding the level of uncertainty, could lead to a set of “promising” solutions which might not be of good quality.

In the next section, there is a review of the use of optimization and simulation regarding airport capacity problems.

2.2. Optimization together with simulation techniques for aviation problems

In the past few years, many applications combining simulation and optimization methods were developed in different fields. Mujica and Flores (2017) provided various optimization–simulation applications, including those in aeronautical, logistical and industrial case studies. In the aviation field, we can find recent applications of optimization–simulation methods in airport operations as well.

In the work of Odoni (1994), the importance of considering uncertainty when modeling airport and air traffic control operations was highlighted. It classified uncertainties into long, medium, and short-term scenarios. For long-term scenarios, the uncertainty that needs to be considered is related to demand, while for medium and short-term scenarios, the uncertainty is represented by delays, cancellations and weather conditions on the day of operations.

Kleinman et al. (1998) addressed the airport network delay problem by using a simultaneous perturbation stochastic approximation (SPSA) optimization algorithm in order to achieve an optimal ground-holding policy with the objective of minimizing delay costs both in the air and on the ground. The simulation part was implemented by using SIMMOD simulation software (SimmodPro!, 2018) to measure in more detail the performance of the system in terms of ground and air delays.

In Confessore et al. (2005), an airport ground simulation model was developed to simulate taxiway operations, including an optimization module for the minimization of ground delays. In this context, the simulation and optimization modules could communicate with each other to model the system in a more dynamic way. The simulation was based on a discrete event simulation (DES) using a general-purpose simulation software, Arena (2018), while the optimization module was based on an algorithm for solving the shortest path problem.

In the PhD thesis of Lee (2014), the airport surface optimization problem was addressed and uncertainty sources such as aircraft push back times, runway exit times, and taxi speeds were included by simulating the operations using SIMMOD. Mujica (2015) proposed a method for optimizing check-in desk allocation inside an airport terminal with the objective of improving the level of service (LoS) for passenger check-in. The optimization part was based on a evolutionary algorithm considering different operational restrictions, while the simulation part was based on a DES model using the general-purpose simulation software SIMIO (SimioLLC, 2018).

In Mujica et al. (2017), an optimization search algorithm, OptQuest (OptTek, 2018) was embedded in a DES model (by using SIMIO) to improve the turnaround performance of aircraft at an airport. In their work, airport ground and airspace operations were modeled in an integrated way — considering the arriving and departing routes in the airspace, runway, taxiway and turnaround operations. The method included a simulation module, a statistical approach, and an optimization search algorithm. Simulation was used to evaluate the turnaround performance of aircraft, while the Analysis of Variance (ANOVA) was used to identify the main components impacting the turnaround performance. Based on the identification of these components, the state space of the optimization problem was reduced to a smaller instance, such that the optimization search algorithm could efficiently find an optimal solution.

In the previous work by Scala et al. (2019), optimization and simulation were combined to solve the airport capacity problem of sequencing aircraft in the airspace and the management of the ground capacity of an airport. In their work, the problem was solved by using a metaheuristic to optimize the schedule through a sliding-window approach. They used discrete-event simulation for evaluating optimized solutions and checked the feasibility of the solutions when uncertainty was introduced. A research gap was identified in Scala et al. (2019) in that the lack of 'direct' communication between the optimization and simulation model was hindering the performance of the simulation–optimization method in terms of solution quality and computational efficiency. Furthermore, there was an inconsistency about the evaluation of the operations between the optimization model and the simulation model, as in the former, a sliding window that broke the whole time-horizon in smaller windows was employed, while in the latter, the whole time-horizon was evaluated in one round. The sliding window approach could lead to an optimized solution more sensitive to uncertainty, since during the construction of the optimized solution for each window uncertainty is not considered, but only in a later stage as it is applied to the whole time-horizon.

2.3. Contributions

This paper proposes an algorithmic architecture consisting of a closed-loop feedback closely linking the optimization component and the simulation component. The main contributions of this paper are threefold.

First, the Opt–Sim framework proposed in this paper is generic so various optimization solution methods can be employed to satisfy the need claimed by Fu (2002) in their review. This general framework could work in other similar scheduling problems, even of other fields (e.g. manufacturing), and it appears to be the first attempt in the literature to develop a closed-loop feedback including optimization and simulation. The proposed Opt–Sim framework, is different from the so called "sim-heuristic" approach (Juan et al., 2015), as uncertainty is kept with the same level throughout the entire Opt–Sim process. In this way, the quality of the solution is not biased by the different uncertainty levels, an aspect that was highlighted as a downside of the sim-heuristics concept (Juan et al., 2015).

Second, the "search" and "evaluation" (Fu et al., 2005; Juan et al., 2015) efficiency of the Opt–Sim framework is improved compared to previous literature. The implementation of the sliding window approach improves the management of the trade-off between "search" and "evaluation" by reducing the computational burden of the problem, while allowing us to incorporate the simulation component (evaluation) in the solution framework. This improves the quality of the solution and fills the research gap highlighted by Scala et al. (2019). Specifically, the "search" ability of the Opt–Sim framework provides feedback that allows the

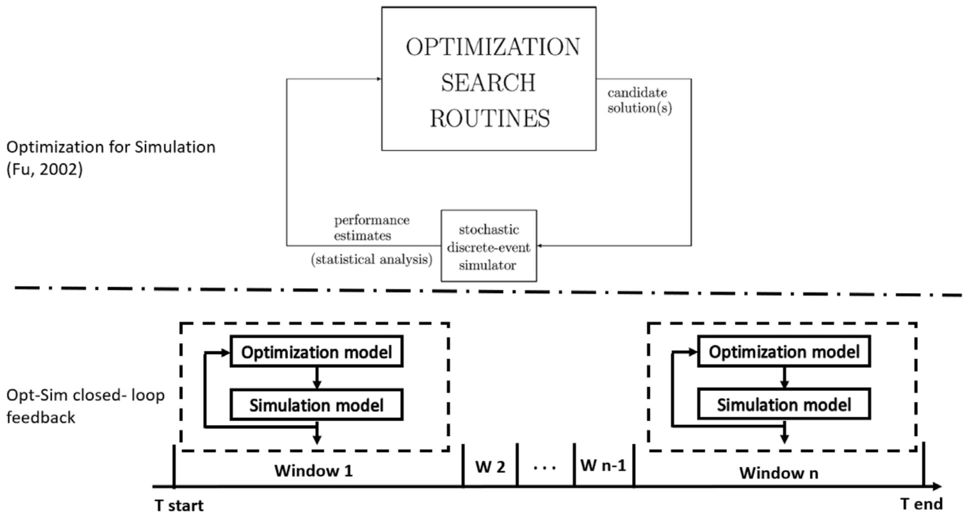


Fig. 1. Comparison between the optimization for simulation framework presented in Fu (2002) and the Opt–Sim closed-loop feedback developed in this work.

optimization model to searching more in depth within the solution space. This was not achieved in Confessore et al. (2005) and Mujica et al. (2017), as they treated the optimization method as a ‘black-box’ within the simulated environment, with the scope of evaluating a limited set of input parameters (limited state space) provided by the simulation model.

Third, the robustness of the solution of the airport capacity management problem is improved by incorporating new measurements such as the number of conflicts and conflict severity. In literature about airport capacity management, the number of conflicts is considered as the only metric for evaluating the performance (Beasley et al., 2000; Hu and Chen, 2007; Ma et al., 2016; Liang et al., 2017; Scala et al., 2019). This paper, raises the attention to potential trade offs between solutions where not only the number of conflicts is considered as a metric, but also the conflict severity. To this end, two different versions of the Opt–Sim framework were implemented: one based on the number of conflicts and the other on conflict severity as the main metric.

3. Opt-Sim framework algorithm

In this study, we are dealing with a stochastic system where many components interact with each other. For instance, perturbations in the airspace can affect the operations executed on the ground. Another aspect to consider is the time frame of the operations. In this study, the airport capacity management problem, is managed at a tactical level (Durand et al., 2016), therefore, the operations considered are within a 24-hours time frame. This fact increases the complexity of the problem, as many more entities (flights) will be part of the operations, and therefore, from an optimization view, increasing the problem state space. The complexity is further increased by the inherent uncertainty which characterizes such a system. A framework like the one developed in this study, must consider all these complexities and uncertainties, and, at the same time, ensure the feasibility and the robustness of a potential solution.

In this context, the Opt–Sim framework developed, fits particularly well as it is able to deal with both the complexity coming from the tactical time frame and from the uncertainty of the system. The former aspect, is tackled by using a sliding window approach that allows to split the entire time frame in smaller instances of the problem, easier to solve. The latter aspect, is tackled by applying a stochastic dynamic simulation model in every instance of the sliding window approach.

The main drawback that the Opt–Sim framework has, is the computational burden. Applying an optimization and a simulation routine for each instance of the sliding window could be computational demanding due to the architecture of the framework, as for improving more the solution at each window, it would require the execution of many feedback loops. If we consider that for each feedback loop we would require to run a certain amount of simulation replications, this would cause an increase of the total computational time for the solution of the total time horizon.

The next section will present the algorithm of the Opt–Sim framework and will describe how all these potential issues have been taken into account and mitigated.

3.1. Algorithm development

To fully integrate the optimization and simulation models, an algorithmic approach is developed where the two models work together to find a better solution. The Opt–Sim framework is further enhanced by introducing an innovative sliding-window technique. For each time window, simulation is combined with optimization in a loop that aims at improving the intermediate solution of the active window. This feature makes it different than the Opt–Sim framework presented in Fu (2002), as it can be seen by the different architectures of these two frameworks shown in Fig. 1. In Fu (2002), the simulation evaluates the candidate solution(s), while in this work, as it can be seen from the lower part of Fig. 1, by applying the sliding window approach, in both optimization and simulation models, the simulation helps in the construction of the overall solution by providing feedback to the optimization model at each window.

The algorithm developed for this framework is summarized in Algorithms 1 and 2. At the beginning of the algorithm, some variables need to be initialized:

- M: objective function tolerance. It is the highest value of objective function that could be accepted, even though it would not give an optimal solution, while trying to minimize the objective function value. Beyond this value, a new iteration (loop) of the Opt–Sim closed-feedback loop is implemented.
- N: maximum number of iterations (loops). This value sets the maximum number of iterations (loops) that can be implemented in the method for each window.
- $nbRep$: number of simulation replications. This value represents the number of simulation replications at each iteration (loop). The simulation model runs several replications to produce statistically significant results due to the stochastic nature of some variables.
- i loop counter. This variable stores the number of iterations (loops) implemented. At each iteration this value is compared with N maximum number of loops. When i is higher than N, the method stops the iterations (loops) for the current window and it continues to the next window.
- $Objective$: This variable stores the objective function value of each iteration (loop).
- $Minimum \quad objective$: This variable stores the minimum value of the objective function between the iterations of each window.

Algorithm 1: Opt–Sim closed-loop feedback

```
    Input: original schedule
    Output: optimized schedule
 1  $M \Leftarrow max \quad objective \quad value \quad tolerance$
 2  $N \Leftarrow max \quad number \quad of \quad loops$
 3  $nbRep \Leftarrow number \quad of \quad replications$
 4  $i \Leftarrow 0$
 5  $objective \Leftarrow 0$
 6  $minObjective \Leftarrow 1000$
 7  $run \quad optimization$
 8  do
 9      run optimization
10      do
11          update simulation database with the optimized schedule
12          run simulation replications
13          if $i$ is equal to $nbRep$ then
14              $objective \Leftarrow objective \quad value \quad calculated \quad by \quad the \quad simulation$
15              $storeBestSchedule(objective)$
16          $i \Leftarrow i + 1$
17      while $i$ is equal to $nbRep$;
18  while closed-loop feedback condition satisfied;
```

Algorithm 2: Store best solution

```
    Input: schedule objective value
    Output: schedule with the minimum objective value
 1  $currentObjective \Leftarrow objective$
 2  if $currentObjective$ is less than $minObjective$ then
 3      $minObjective \Leftarrow currentObjective$
 4      $storetheschedule$
```

The parameters M and N play a crucial role in driving the algorithm efficiency (search) and effectiveness (evaluation). The *storeBestSolution(objective)* function, as it is summarized in Algorithm 2, is used for choosing the solution with the minimum objective value between the different loops of each window, such that eventually the algorithm will come up with an optimized schedule composed of the best schedule from each window.

Fig. 2 shows the architecture of the Opt–Sim closed-loop feedback algorithm. Optimization and simulation share a database where they store updated schedules. The interaction between the optimization and the simulation is the following: the optimization looks

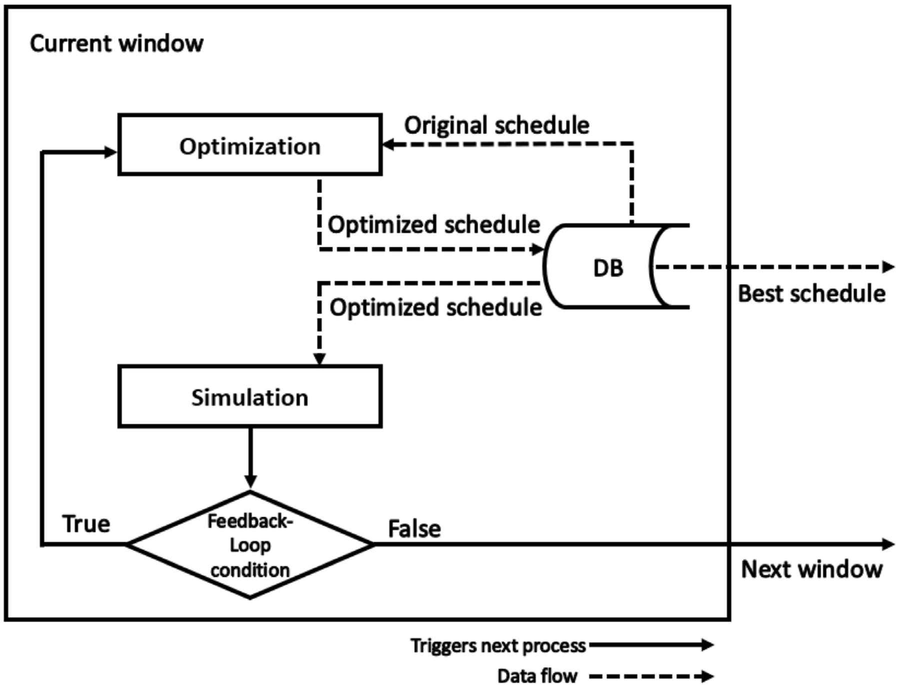


Fig. 2. Architecture of the Opt–Sim closed-loop feedback.

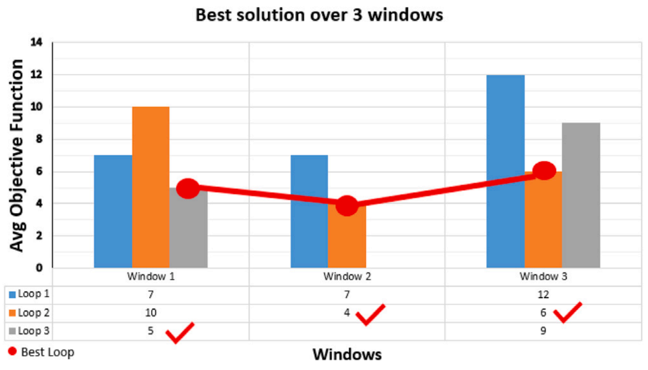
at the database for the original schedule and generates an optimized schedule; this schedule is then used by the simulation model to run the replications. The simulation then identifies the best schedule according to the function, *storeBestSchedule(objective)* and stores it in the database. After the stopping condition of the closed-loop feedback is evaluated (closed-loop feedback condition), it goes back to the optimization model or to the next window. The cycle repeats for each loop of each window in the solution process. The database plays an important role in solving the problem because it allows the two models, optimization and simulation, to communicate with each other and share data between them.

As it was mentioned before, the two parameters M and N are important for the algorithm performance, they set the condition whether the algorithm will run another loop or move on to the next window (see Algorithm 1). For simplicity, we refer to this condition as the *Feedback-loop* condition.

When the closed-loop feedback condition is fulfilled, the optimization model will be triggered and it will generate a new solution for the problem. It will be different than the previous as some of its input parameters will be fine-tuned, by doing this it generates a more robust solution. Later in this paper, we will provide more details on how to fine-tune the parameters in a real case study as they affect the quality of the solution and the computational effort. For example, a large value of M will accept a solution with many conflicts (of diverse severity), meaning that the algorithm will not be effective to find a quality solution. A small M, on the other hand, will force the algorithm to keep searching for another solution, which might increase the computational effort. Similarly, choosing a large value for the parameter N will increase the chance of obtaining better solutions.

3.2. Example of how the overall solution is constructed

To fully understand the potential of the Opt–Sim closed-loop feedback framework, it is convenient to illustrate with an example how the overall solution (entire time-horizon) is constructed. The sliding window approach implemented in this framework, divides the entire time-horizon in smaller windows, for each window, the closed-loop feedback framework uses optimization and simulation to find a solution that is more robust or less sensitive to the uncertainties of real-world system. In this example, a fictional problem of three windows is solved to show how the solution is constructed. In this example, M, which is the objective function tolerance is set to 5, and N, the maximum number of loops equal to 3. For the sake of simplicity, in this example, the optimization and simulation resolution methods are omitted, and we assume that the final results are the ones shown by Fig. 3. Starting from window 1, we can see three bars of different colors (blue, orange and gray) representing three loops. This means that closed-loop feedback condition was fulfilled for the first two loops, until the third loop where it was not fulfilled. This was due to the fact that the value of objective function was, in both first and second loop, higher than M (7 and 10, for loop 1 and loop 2, respectively). The closed-loop feedback condition was not fulfilled at the third loop, because the number of loops reached 3, equalizing the value N. Before continuing to the next window, the closed-loop feedback framework checks, among the loops implemented, which one has provided the best result in terms of objective function value (in this example, we assume that the purpose of the problem is to



	Window 1	Window 2	Window 3
Loop 1	7	7	12
Loop 2	10	4 ✓	6 ✓
Loop 3	5 ✓		9

Fig. 3. Example of the construction of the best solution. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

minimize the value of the objective function). In our example, the third loop has the lowest value, therefore, it is the best solution (schedule) of this window. This schedule will be picked and used as a starting schedule for the next (second) window. In window 2, we notice that only two loops were generated, as in the second loop, the closed-loop feedback condition was not fulfilled. This is because the value of objective function obtained, 4, was lower than M. Obviously, the starting schedule of the third window, will be the schedule provided in the second loop of the second window, as it obtained the best result. Lastly, in window 3, three loops were run, and as none of the loops were able to obtain a value lower than M, then the feedback loop condition was not fulfilled until the third loop. It is worth noticing that, in the third window, the best solution came from the second loop, as it obtained the lowest value of objective function. It is important to highlight that due to the stochasticity of the model variables, the Opt–Sim framework, not necessarily improves the objective function loop after loop. However, the Opt–Sim framework, is able to pick the best solution among the loops and integrate it as part of the best overall solution, as it is shown in Fig. 3 by the connected red dots.

As it has been illustrated, the Opt–Sim framework evaluates the solution at each loop. The new solution, as we will see in the case study provided in this work, is driven either by the randomness of the metaheuristic in finding a new solution or by fine-tuning of some specific input parameters. In this work, the parameters used as drivers for the optimization model are:

- optimization metaheuristic parameter.
- problem-specific parameter.
- objective function parameter.

These parameters will be described in more detail in the numerical experiments section.

4. Airport capacity management problem under operational uncertainty

To test the Opt–Sim framework developed in this work, we have formulated and solved the capacity management (ACM) problem. In the context of this work, this problem refers to the integration of the landing aircraft sequencing in the airspace near an airport and aircraft movement congestion on the ground of an airport. Due to the growth of air traffic and the limited capacity of airports, this problem has gained importance in the industry and attention by researchers. The operations involved in this problem include the following: aircraft arrival sequencing in the airspace (i.e. the Terminal Maneuvering Areas, TMA) before landing on the runway, runway operations (aircraft arrivals and departures), aircraft occupancy of the taxiway and terminal gates, and aircraft departure sequencing. Fig. 4 provides a schematic representation of the study problem.

4.1. Problem description

In the ACM problem operations are modeled at a tactical level, meaning that aircraft sequencing in the airspace can be planned up to 30 min before landing (Durand et al., 2016). Within this time frame, decisions such as trajectory changes, holding procedures, speed changes, runway and gate allocation can be instructed by air traffic controllers. In this work, the airport airspace considered is the TMA. The TMA is a portion of airspace that surrounds airports, its size is usually between 30 and 40 nautical miles radius, and its center is located at the airport location. Within the TMA, the Airspace Traffic Service (ATS) defines and publishes fixed landing and departing routes with the purpose of facilitating the flows of aircraft going to and from the airports. These routes are called Standard Instrument Departure Routes (SIDs), and Standard Arrival Roues (STARs), each of them ensures aircraft to fly at a certain altitude level, under speed restrictions and following some significant points (waypoints). The sequencing of arrival aircraft involves decisions such as TMA entry times, speed changes and runway choice with the objective of avoiding any conflicts in operations. The TMA landing routes were modeled as a network of links and nodes. Each node represents a critical point of a landing route (i.e. waypoints), and the links model the connection between waypoints. Ground operations in an airport were modeled by a macroscopic approach, considering the main components of ground operations as resources characterized by limited

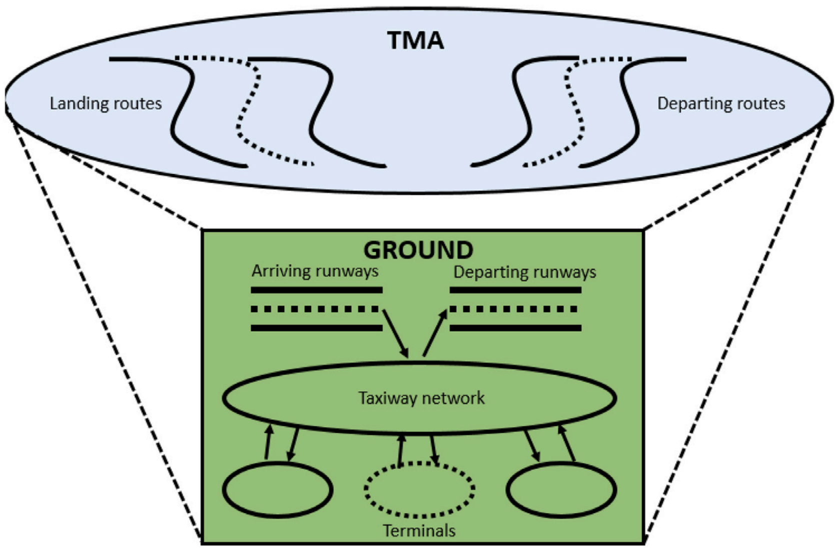


Fig. 4. Representation of the operations considered in the airport capacity management problem.

capacity and service time. In the ACM problem, the ground operation components of an airport include runways, taxiway network and airport terminals. Decisions concerning ground operations such as the taxiway routing of aircraft were not modeled in this approach.

Arrival aircraft, in the context of the ACM, start their operations by entering the airspace of the destination airport through one of the arrival routes in the TMA (airspace input). While flying toward the runway, these aircraft will be subject to air traffic control such as speed control, sequencing, and separation minima between any consecutive leading and trailing aircraft along the arrival route. After arrival, aircraft will taxi toward the terminal for turnaround operations for the next departure or head toward the maintenance area for routine maintenance. Due to the macroscopic level of abstraction chosen in modeling the ACM problem, factors such as movements and speed on the taxiway, aircraft turnaround time at a gate are not individually modeled. Instead, the main constraints of taxiway and terminal capacity are represented by the declared values of capacity, which represents a threshold to which the system can still guarantee smooth operations without delays. Therefore, ground operations, starts either when an aircraft has landed (airspace output), or when an aircraft is intended to depart, starting from its terminal area. Departure aircraft starts with a reverse procedure, starting from the terminal area, taxiing to the runway, and then joining the departure sequencing before taking off and leaving the TMA of the airport, as this represents the output of the ground operations.

4.2. Assumptions in modeling the ACM problem

For the model of ACM problem, the following assumptions were adopted in this paper:

- Aircraft fly along standard arrival routes (STARs). These routes are the published routes with altitudes that respect vertical separation minima;
- For departing aircraft, we only consider take off operations on the runway, while the standard instruments departure routes (SIDs) are not considered;
- Aircraft flying the STARs decrease their speed with a constant deceleration rate;
- Aircraft occupying the runway for landing or departing according to an average runway occupancy time;
- Aircraft occupying the taxiway according to an average taxiway occupancy time; and
- Aircraft can park at any gate of the assigned terminal.
- Aircraft turnaround operations are not modeled, however, turnaround times are considered.

4.3. Mathematical formulation of the ACM problem

In this section, the main components of the ACM optimization model are described including decision variables, constraints, an objective function as well as the solution method.

4.3.1. Decision variables

The decision variables used in the optimization model for each flight $f \in F$, are:

Table 1
ICAO separation minima values [NM].

	Wake turbulence category	Leading aircraft		
		Heavy	Medium	Light
Trailing aircraft	Heavy	4	3	3
	Medium	5	3	3
	Light	6	5	3

Entry time in the airspace: it is modeled as a stochastic discrete variable, $t_f \in T_f$ where $T_f = \{T_f^0 + j\Delta T \mid \frac{\Delta T_{min}}{\Delta T} \le j \le \frac{\Delta T_{max}}{\Delta T}, j \in Z\}$. T_f^O is the scheduled entry time of flight f, T_f is a discrete value that lies within a range between a minimum ΔT_{min} and maximum ΔT_{max}. For this specific problem, we adopted ΔT_{min} as −5 min, ΔT_{max} as 30 min, and ΔT as 5 s.
Entry speed in the airspace: it is modeled as a percentage of speed change $s_f \in S_f$ where $S_f = \{S_{fmin} + j\Delta S \mid |j| \le (S_{fmax} - S_{fmin})/\Delta S, j \in Z\}$. S_{fmin} and S_{fmax} are the minimum and maximum speed values that can be assigned to s_f, and ΔS is a discretized time increment for speed adjustment. For this specific problem, we adopted S_{fmin} as 0.9 S_f^0, S_{fmax} as 1.1 S_f^0, and ΔS as 0.01 S_f^0.
Landing runway: choosing a runway $lr_f \in LR_f$ given a set of available landing runways LR_f. For this specific problem, the set of available landing runways is $LR_f 0, 1$, meaning that there are only two landing runways available.
Push back time: the time when an aircraft is cleared for pushing back from the gate for departure. It is modeled as a discrete value of delay $pb_f \in PB_f$ where $PB_f = \{PB_f^0 + j\Delta T \mid 0 \le j \le \frac{\Delta PB_{max}}{\Delta T}, j \in Z\}$ that lies within a range between zero and a maximum value, ΔPB_{max}. For this specific problem, we adopted ΔPB_{max} as 15 min and ΔT as 5 s.
Departing runway: choosing a runway $dr_f \in DR_f$ given a set of available departing runways DR_f. For this specific problem, the set of available departing runways is $DR_f 0, 1$, meaning that there are only two departing runways available.

4.3.2. Constraints and the objective function

The objective of the ACM optimization model is to produce a schedule of arriving and departing aircraft that can be processed by the airport system smoothly by avoiding any potential conflicts in the airspace and ensuring that the capacity of the ground operations at the airport (i.e. runways, taxiways, and terminal gates) is not exceeded. The constraints of this problem are represented by the separation minima between consecutive aircraft in the airspace and on the runway, and by the maximum capacity of the taxiway network and terminals. These constraints are reflected in the objective function as they are considered in the calculation of the conflicts. The optimization model aims at minimizing the total number of aircraft conflicts in operations. In the context of the ACM problem, aircraft conflicts are defined as follows.

Airspace Separation Conflicts are identified when two consecutive aircraft in operation in the airspace are separated by a distance which is less than the standard separation established by the International Civil Aviation Organization (ICAO, 2007) for safety purposes. The separation standard is shown in Table 1. To formalize, let $\mathcal{A}$ be the set of landing aircraft, i and j represent a consecutive pair of aircraft, S_{ij} the separation minima between the leading aircraft type i and the trailing aircraft type j, and D_{ij} the distance between the leading aircraft type i and trailing aircraft type j. Thus, aircraft conflicts are defined as:

$$C_s = \begin{cases} 1, & if \quad D_{ij} < S_{ij}, \quad \forall i, j \in \mathrm{A} \\ 0, & otherwise \end{cases} \tag{1}$$

Sequence Conflicts are detected to avoid any overtaking between consecutive aircraft and ensuring the sequence along the landing route in the TMA of an airport. In the context of the ACM problem, this type of conflict is detected by checking the aircraft sequence at the entry and exit point of each link of a route in the TMA. To formalize this, let $l(u, v)$, $l = 1, \ldots, n$ be a given set of links of a particular route, $order^l_{fu}$ and $order^l_{fv}$ the positions of the aircraft i on the link entry u and on the link exit v, respectively. The order of sequence conflicts is calculated as:

$$C_{seq} = \sum_{l=1}^{n} (\sum_{i \in A} order^l_{iv} - order^l_{iu}) \tag{2}$$

Ground Conflicts in the context of ACM problem include: runway conflicts, and taxiway network and terminal conflicts.

- Runway conflicts are identified in a similar way as airspace conflicts, where two consecutive aircraft that are occupying the runway need to keep a certain separation, depending on the type of operation conducted on the runway, such as landings, take-offs and crossings. Table 2 shows the separation minima standard used in this paper. To formalize this, let A be the set of aircraft, i and j a consecutive pair of aircraft, S_{ij} the separation minima between the leading aircraft type i and trailing aircraft type j, and D_{ij} the detected time difference between aircraft type i and aircraft type j. Thus, runway conflicts can be formalized with the following function:

$$C_r = \begin{cases} 1, & if \quad D_{ij} < S_{ij}, \quad \forall i, j \in A \\ 0, & otherwise \end{cases} \tag{3}$$

Table 2
Runway separation minima [s].
Source: Frankovich (2012).

	Operation-Category	Leading aircraft					
		A-H	A-M	A-L	D-H	D-M	D-L
Trailing aircraft	A-H	96	157	207	60	60	60
	A-M	60	69	123	60	60	60
	A-L	60	69	82	60	60	60
	D-H	60	60	60	96	111	120
	D-M	60	60	60	60	60	60
	D-L	60	60	60	60	60	60

Note: A = Arrivals, D = Departures, H = Heavy, M = Medium, L = Light.

- Taxiway network and terminal conflicts are identified when the capacity of the facility (taxiway and terminal) is exceeded during the day, i.e. capacity overload. The capacity of the taxiway network and the terminals are defined as the maximum number of aircraft that can simultaneously occupy the respective facility. This capacity is dependent on the layout and physical configuration of an airport. The capacity overload is then calculated by two terms: the maximum value of capacity overload ($MaxOL$), and the average capacity overload ($AvgOL$) during the day. To formalize this, let C be the declared capacity, O_t the aircraft occupancy for each discrete time increment $t \in T$, and T the entire time frame considered. Then, the maximum capacity overload during T, $MaxOL$ is given by:

$$MaxOL = \begin{cases} \max_{t\in T}(overload_t), & if \quad O_t > C \\ 0, & otherwise \end{cases} \tag{4}$$

and $AvgOL$ is given by:

$$AvgOL = \frac{\sum_{t\in T} overload_t}{T} \tag{5}$$

where,

$$overload_t = \begin{cases} O_t - C, & if \quad O_t > C \quad \forall t \in T \\ 0, & otherwise \end{cases} \tag{6}$$

Therefore, the total conflicts for ground operations in the aircraft sequencing optimization problem is:

$$C_{(twy)(ter)} = MaxOL + AvgOL \tag{7}$$

The objective function of the ACM optimization model computes the weighted sum of all conflicts as follows. The weights, γ and δ are assigned to the airspace and ground conflicts, respectively. These weights can be adjusted accordingly depending on which component the optimization process focus on,.

$$C = \gamma(C_s + C_{seq}) + \delta(C_r + C_{twy} + C_{ter}) \tag{8}$$

5. Elements of the Opt–Sim closed-loop feedback

In this section, the Opt–Sim framework elements, employed for solving the problem are described. First, the sliding window approach; second, the optimization resolution method (the metaheuristic simulated annealing), and third, the simulation model (discrete event simulation) will be introduced.

5.1. Sliding-window approach

The Opt–Sim framework is based on a sliding-window approach. This approach does not tackle the problem by considering the entire time-horizon in one round, but considers a sequence of time windows of a smaller sizes. The optimization and simulation models are applied to each time window, as the time window is shifted along the entire time-horizon to solve the whole problem. This approach suits well, especially for the type of problems which have a long time-horizon and are of a combinatorial nature. The main advantage of the sliding-window approach is that it reduces the computational effort and the size of the problem to solve making it more tractable. Another advantage is that it allows to treat the whole problem in such a way that, as time flows, new information is updated in each window due to changes in the environment and interactions between entities and the surrounding environment. Fig. 5 shows a schematic representation of the sliding-window approach.

As can be seen from Fig. 5, the entities in our problem can assume four types of status: Completed, On-going, Active and Planned.

- Completed status: entities are outside the current window and decisions for them have already been made in a previous window.

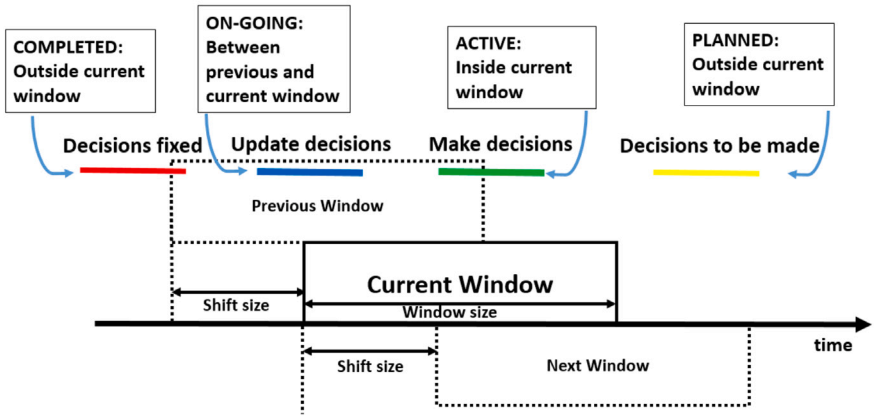


Fig. 5. Representation of the sliding-window approach.

- On-going status: entities are partly outside and partly inside the current window and they have impact on the optimization process of the current window. However, decisions will not be made for these entities.
- Active status: entities are completely inside the current window and, for them, decisions will be made for the current window.
- Planned status: entities are ahead the current window, and they do not have any impact on the current window.

One key consideration in the sliding-window approach is the window length W and the shift length S. It is crucial to choose the right values for them, since a large value in window length might worsen the overall performance of the problem by making decisions for entities too far ahead in time. Moreover, there will also be an increase in computational time. On the other hand, a small value in window length might not have enough information to consider when optimizing aircraft sequences, thereby jeopardizing the solution quality.

In the same way, a large value in shift length will reduce the solution quality of the method, since fewer decisions will be updated from one window to another. Small values in shift length, on the other hand, will increase computational time performance. Hence, choosing the right values for window and shift lengths, depending on the specific problem, will enhance the capability of the sliding-window approach. For our ACM problem, given the time horizon of 24 h and more than 1000 entities involved in the case study, we chose a window size of 2 h and a shift size of 30 min after a number of test runs of the model.

5.2. Resolution method: Simulated annealing metaheuristic

The ACM optimization model is solved by the implementation of the metaheuristic, Simulated Annealing (SA) (Kirkpatrick et al., 1983). The problem under study has been proven to be NP-Hard (Beasley et al., 2000) which is the reason why a metaheuristic has been applied. The main characteristic of this metaheuristic is that it allows jumps in the state space to solutions which are worse in terms of objective values, and at the same time allows exploring the state space more broadly — avoiding being trapped in a local minima. In other similar studies, this metaheuristic has been proven to be very efficient (Ma et al., 2016; Liang et al., 2017).

The name, Simulated Annealing, comes from the analogy of the process of physical annealing with solids, where a crystalline solid is heated and then cooled until it achieves its most possible regular crystalline lattice configuration (Henderson et al., 2003). In the SA metaheuristic, given an initial temperature T_0, a cooling process is conducted in search of an optimal solution. At each value of temperature T, a number of iterations are executed. Once a solution is found, i.e the current one, it can be replaced by a randomly generated neighboring solution. The algorithm accepts all the neighboring solutions which are better than the current solution in terms of objective function value, but also worse neighboring solutions based on an acceptance probability so as to avoid getting trapped in local optima. The acceptance probability is $e^{\frac{\Delta OF}{T}}$, where ΔOF is the difference in objective function between the current and the neighboring solution, and T is the current temperature. During the cooling schedule, if the temperature decreases slowly enough, there will be high chance of converging to a global optima.

In this work, a geometric cooling schedule was implemented, decreasing the temperature T_0 multiplying it by a coefficient α. The termination of the SA algorithm is based on a stopping criteria. In this case, the stopping criteria is either if the maximum number of iterations and minimum temperature are achieved, or if an optimal solution is obtained. For detailed information about the application of this metaheuristic to this problem, refer to Ma et al. (2016).

In this paper, we focus our attention on a specific parameter of this metaheuristic, namely the cooling schedule, α. The cooling schedule parameter, α, is responsible for the convergence toward the optimum of the solution. This parameter is used in the Opt–Sim framework as a parameter to fine-tune in order to obtain a new optimized solution.

Table 3
Airside components characteristics.

Ground component	Capacity
Landing runway	1
Departing runway	1
Taxiway network	20
Terminal 1	11
Terminal 2	89
Terminal 3	55

Table 4
Average taxi time [s].

Terminals	Runways			
	Landing runways (taxi-in time)		Departing runways (taxi-out time)	
	27R	26L	26R	27L
Terminal 1	400	535	720	1400
Terminal 2	730	500	890	760
Terminal 3	680	530	880	710

5.3. Simulation paradigm: Discrete event simulation

A simulation model has been developed to deal with the uncertainties involved in the operation of the airport system and to improve the optimized solution obtained from the optimization model. Within the Opt–Sim framework, the simulation model serves as feedback for the optimization process in order to generate an alternative solution through the optimization model. Simulation in this context can overcome the common drawback from deterministic optimization, where exact solutions are generated without considering the stochastic nature of systems. The simulation approach chosen for this paper is based on discrete event simulation (DES). This DES approach has been consolidated over the last decades since it has been implemented with success by other authors in many fields (Brunner et al., 1998; Banks et al., 2000; Negahban and Smith, 2014). This type of approach fits best for the type of systems that are process-driven, with entities following some predefined paths and undergoing predefined processes and predefined actions.

One of the main features of DES is that the time passes based on specific events, and not in a continuous manner, as it happens for continuous-time simulation approaches. Moreover, this approach differs from agent-based simulation (ABS) due to the fact that the entities involved in the DES system do not have their own logic, but follow pre-defined processes. An airport system, as it was described in Section 4, implies all the characteristics of a DES. Hence, the DES approach was chosen as the simulation method for the ACM problem.

Furthermore, DES differs from Monte Carlo (MC) simulation, as with DES more complex dynamic systems can be modeled, instead, MC simulation is a probabilistic random sampling techniques, which is typically used for non-dynamic models (Chica et al., 2017).

The ACM problem logic and objective function were modeled in the simulation model in the same way as they were modeled by the optimization model. The simulation model was developed by using a commercial general-purpose simulation software Anylogic (2018). Within the simulation model, it was possible to model sources of operational uncertainties in the system, so a solution closer to operational reality could be found. The sources of uncertainties that have been identified include:

- entry time in the TMA, due to speed and human (pilots) factors,
- taxi time, due to layout complexity, congestion, speed and human (pilot) factor, and
- departure (off-block) time, due to ground handling procedures and communications with the aircraft cockpit.

6. Case study: Paris Charles de Gaulle Airport

The framework developed in this work has been applied to a case study to test its validity. The case study chosen was Paris Charles de Gaulle Airport (LFPG) in France. LFPG is one of the major airports in Europe. In 2017, it was the second busiest European airport in terms of air traffic movements (476,000) and passenger numbers (69.5 million) (EUROCONTROL, 2018). The schedule used for the experiments include one day of real operations at LFPG with 562 departures and 554 arrivals, with a total of 1116 movements. The airside layout of LFPG includes four parallel runways used in a segregate mode, two for landings and two for departures respectively, three terminals and a complex taxiway system. Tables 3 and 4 describe the main characteristics of the airside components of the LFPG.

The TMA of LFPG considered in this case study only included aircraft arrival routes. LFPG has two routes coming from the north and two coming from the south with a total of four different entry points for arrivals. Each of the two southern and northern routes converge into two merging points, leading to two descending paths for each of the two landing runways. Therefore, based on the coming direction (route entry point) and on the landing runway, there are 8 paths in total, as depicted in Fig. 6.

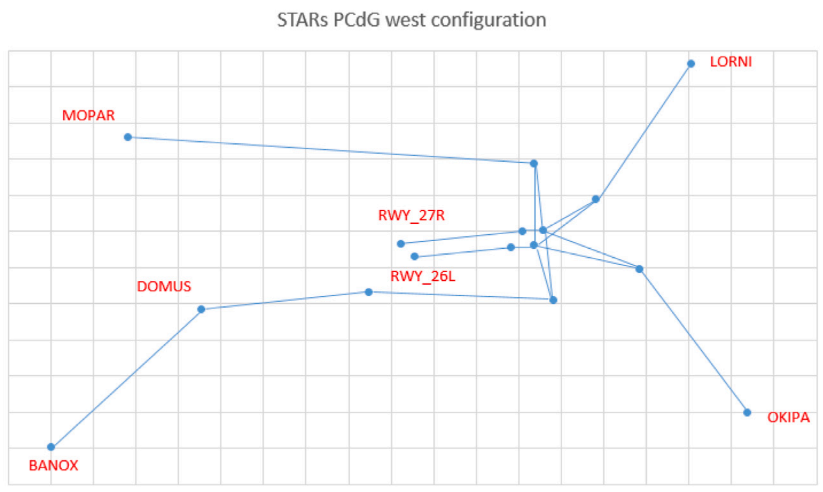


Fig. 6. Airspace landing routes.

7. Numerical experiments

7.1. Experimented scenarios

The proposed Opt–Sim framework was tested on the ACM problem, and four scenarios were created based on the optimization model input parameters to fine-tune. For simplicity, we will mention to them as "input parameters". In the first scenario, no input parameter is fine-tuned so that the new solution generated is guided by the randomness of the metaheuristic. In the second scenario, the input parameter α of the metaheuristic is fine-tuned, and in the third scenario, a constraint of the problem, i.e. the separation minima between aircraft in the airspace and on the runway, is fine-tuned. The fourth scenario is conducted by fine-tuning the objective function weights.

- Scenario 1 (S1): at each loop, the optimization model generates a new solution based on the randomness of the metaheuristic when searching in the state space. We will refer to this scenario as the "Default"
- Scenario 2 (S2): at each loop, we increase the simulated annealing cooling schedule parameter, α, such that the optimization model will generate a solution by exploring the state space more in depth. We will refer to this scenario as "Alpha"
- Scenario 3 (S3): at each loop, we increase the separation minima constraint (i.e. constraint relaxation). In theory, we shall obtain an optimized solution which will be more conservative. However, when we simulate the solution, it can potentially absorb the effect of the uncertainty. We will refer to this scenario as "Sep. min."
- Scenario 4 (S4): at each loop, we modify the objective function weights γ and δ. At each loop, depending on the value of the airspace or ground side conflicts in the objective function, we will increase its weight such that the next solution will focus more on those conflicts. The rationale behind this is that if most of the conflicts are generated by a specific area (airspace or ground side), then it is better to focus the optimization process on that specific area (by increasing the weight of that specific area). We will refer to this scenario as "O.F. weight."

Table 5 shows the values of adopted input parameters in various scenarios for each simulation loop in the Opt–Sim framework. In scenario S1, none of these input parameters is fine-tuned, thus the cooling schedule, α, parameter is set to 0.96, the separation minima are the ones shown in Tables 1 and 2, and the objective function weights, γ and δ, are both set to 1. It is important to point out that the values of the objective function weights of Table 5 refer to the component of the objective function that impact the most the overall objective function value (it can be airspace or ground). Moreover, an extra experiment was conducted, based on a variant of the closed-loop feedback condition, as it was introduced in Section 3. In this variant, conflicts (airspace and runway) severity was considered. Before presenting this variant of the closed-loop feedback condition, let us introduce two parameters:

- $Conflict\ severity$: This variable represents, in percentage, the extent to which the separation minima between two consecutive aircraft, in the airspace or on the runway, has been violated.
- ϵ: conflict severity tolerance. This variable represents the maximum value of conflict severity accepted for a solution.

In the following, the two closed-loop feedback conditions are formulated:

$$closed-loop\ feedback\ condition\ 1:\ objective\ function \geq M\ and\ i \leq N \tag{9}$$

Table 5
Values of the input parameters of the methodology.

Scenario (Input parameter)	Loop 1	Loop 2	Loop 3
S1 (Default)			
S2 (Alpha)	0.96	0.97	0.98
S3 (Sep. min.)	+10%	+20%	+30%
S4 (O.F. weight)	+10%	+20%	+30%

Table 6
Sources of uncertainty included in the simulation model with their values.

Entry time in the airspace deviation	[−30 s , +30 s]
Taxiway time deviation	[−10% , +10%]
Push back time deviation	[−30 s , +30 s]

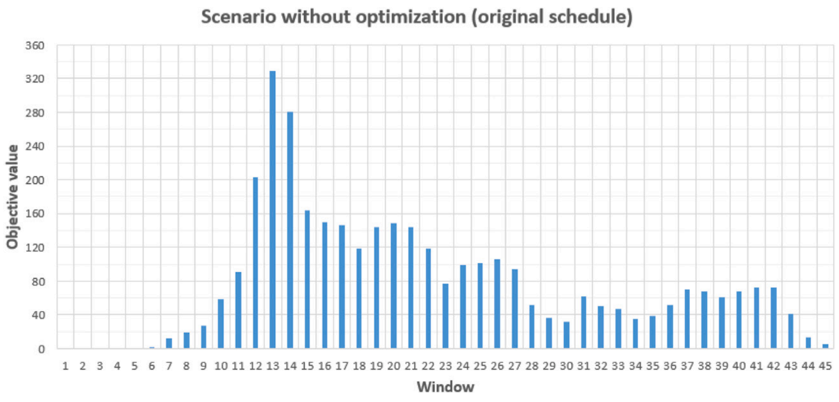


Fig. 7. Aircraft conflicts for each time window for the scenario without schedule optimization.

$$closed-loop\ feedback\ condition\ 2:\ objective\ function \geq M\ and\ i \leq N\ or\ conflict\ severity \geq \epsilon\ and\ i \leq N \tag{10}$$

The closed-loop feedback condition 1 (9), checks if the objective value of the current window is greater than M and if the number of loops is less than N. The parameter M defines the objective value tolerance that can be accepted by the algorithm, meaning that a certain number of conflicts can be accepted by a solution. Parameter N defines the maximum number of loops that can be implemented for a window.

The closed-loop feedback condition 2 (10), is similar to the previous one, but it additionally checks if the conflict severity is greater than a certain conflict tolerance ϵ. The rationale behind this condition is that a conflict with a low severity is preferred to a conflict with a high severity.

For each loop implemented, the simulation performed 10 replications at each loop. Due to this fact, the results will be presented in terms of averages.

7.2. Parameter setting for the Opt–Sim framework

Regarding the parameter M, objective value tolerance, it was calculated as 10 percent of the average number of conflicts for all windows for the scenario run without schedule optimization. This means that in every window we want to keep the conflicts below this threshold, and in general we accept an amount of conflicts up to 10 percent of the average number of conflicts for all windows. Fig. 7 presents the results in terms of average objective function value for the schedule run without optimization for all time windows. The average number of conflicts for all windows, rounded to the nearest integer, was 78. Hence, the parameter M was chosen as 8 (7.8 rounded to the nearest integer). The conflict tolerance ϵ value was chosen as 20% of the total separation minima violation. Both M and ϵ criteria were arbitrarily chosen and could be further improved in future studies.

Regarding the parameter N, maximum number of loops, the authors chose three as a value. The value of N impacts the computational aspect of the framework and can be adjusted based on the availability of computing hardware. However, a minimum of loops must be kept in order to evaluate the performance of the framework. In the simulation model, some sources of uncertainties have been included, as summarized in Table 6. Values have been chosen based on domain knowledge.

Table 7
Comparison between the different scenarios.

	Scenario 1	Scenario 2	Scenario 3	Scenario 4
Average objective function value	37.95	33.08	37.20	37.38
Conflicts severity airspace (%)	37.85%	34.23%	39.40%	37.82%
Conflicts severity runway (%)	55.93%	55.08%	55.81%	56.07%
Computational time (sec.)	16 083.67	14612.9	13 863.37	14 813.11

7.3. Results Opt–Sim implementation: closed-loop feedback condition 1

Table 7 shows the main results after running the different scenarios. By looking at the first row, it can be noticed that the best scenarios in terms of objective function value is Scenario 2 (S2). This is the scenario where "Alpha" cooling schedule parameters has been fine-tuned. These values shows the average objective function for the different windows. By ordering from best until worst, we have that: S2 obtained 33.08, followed by Scenario 3 (S3) "Sep. min." with 37.20, Scenario 4 (S4) "O. F. weight" with 37.38 and Scenario 1 (S1) "Default" with 37.95. If we look at the severity performance, for both airspace and runway, we found that S2 has obtained the best values, 37.85% and 55.08% for airspace and runway, respectively. These values are represented in terms of averages calculated for different windows of the time horizon. Computational times are presented in the last row of Table 7, and they fall between 3,85 Hrs (S3) and 4.46 Hrs (S1), revealing that computational performance represent a burden for the methodology.

According to Table 7, the best scenario results are S2, "Alpha", as for both objective function value and conflict severity it obtains the lowest value when compared with the other scenarios. The graph in Fig. 8, shows the value of objective function for each window for the S2 scenario. In this graph, the different components of the objective function are highlighted with different colors. These results reveal that the main contributor to the objective function value is airspace conflicts (yellow bar), at least from window 9 until window 20. From window 21 until the last window, the main contributor to the value of objective function is runways conflicts. Taxiway and terminals overload contribute marginally to the overall objective function value. Fig. 8, shows a clear trend regarding the airspace conflicts (yellow bar), as we see a high concentration of conflicts between windows 12 and 18. These windows represents the time of the day between 5.30 AM and 10.30 AM, where the peaks of arrival occur according to the original schedule. The blue bar, representing the runways (arrival/departure) conflicts, seems to be equally distributed among all windows. This suggests that in the arrivals peak, until window 18, most of the runway conflicts belong to the arrival runway, while from window 19 until the last window, most of the runway conflicts belong to the departure runway. This trend is also highlighted by the terminal overload, red bar, as they appear mostly between windows 13 and 18, during the arrivals peak. Taxiway network overload peaks appear mostly between windows 20 and 28, and between windows 37 and 43, suggesting that, the departing flow is the main reason for the congestion on the runway.

Fig. 9 illustrates the amount of aircraft that were involved in the Opt–Sim process. We could notice that from window 10 to window 14, there was a rapid increase in the number of aircraft during the morning traffic peak. In the following windows, the number of aircraft increases further, albeit with a less steep trend, until reaching the maximum peak in window 20. Compared with results shown in Fig. 8, even though the number of aircraft reaches its peak after window 14, the number of conflicts in the optimized solution, especially in the airspace, decreases. This suggests that in these windows most of the aircraft are on the ground side (taxiway network and terminals) rather than in the airspace. With this, we also observed terminal overload in Fig. 8 getting severe after window 13/14, while runway conflicts and taxiway overload sustained for the rest of the day.

The graph of Fig. 10, shows the trends of the conflict severity for both airspace (yellow bar) and runways (blue bar). Regarding airspace conflict severity, within the arrival peak (windows 12–18), severity values are between 60 and 80%, with some peaks over 80% (windows 10, 18, and 32). Especially for windows 10 and 32, the high value of conflict severity is not so relevant, as in these windows we find a small amount of conflicts. Regarding the runway conflict severity, we found a uniform trend with severity values between 50 and 80%.

7.4. Results Opt–Sim implementation: closed-loop feedback condition 2

Table 8 shows the main results after running the different scenarios when severity is also included in the closed-loop feedback condition. By looking at the first row, it can be noticed that the best scenario in terms of objective function value is Scenario 4 (S4). This is the scenario where the objective function weights are fine-tuned. The values of the average objective function for the different windows are 36.61 for S4, followed by Scenario 2 (S2) "Alpha" with 37.01, Scenario 3 (S3) "Sep. min." with 37.34 and Scenario 1 (S1) "Default" with 37.61. These results show that there is not a big difference between the different scenarios, as the value gap between best and worst scenario is around 1. If we consider the severity performance, the best scenario is S2 "Alpha", having 33.38% and 53.97% conflict severity for airspace and runway, respectively. On the other hand, S4 has the worst value of conflict severity for airspace (39.82%) and the second best value for conflict severity for runway (55.56%). Regarding the computational times, they are slightly longer than the ones found in the previous experiments. This results show a trade-off between a solution (S2) with the lowest conflict severity, and the solution (S4) with the lowest objective function value. While the difference in objective function between S4 and S2 is not too big (0.4), the difference in airspace and runway conflict severity between S2 and S4 is bigger, as they have obtained 6.43% and 1.50%, respectively.

According to Table 8, the best scenario is S4, "O.F. weight", as it obtains the lowest value of objective function when compared with the other scenarios. Although, the best scenario according to conflict severity is S2, "Alpha". The graphs in Figs. 11 and

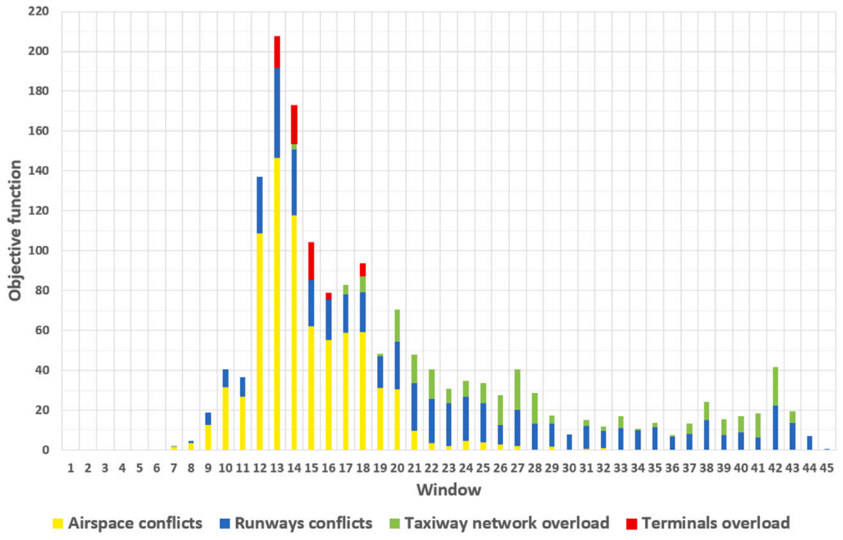


Fig. 8. Objective function values of the “Alpha” scenario (S2). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

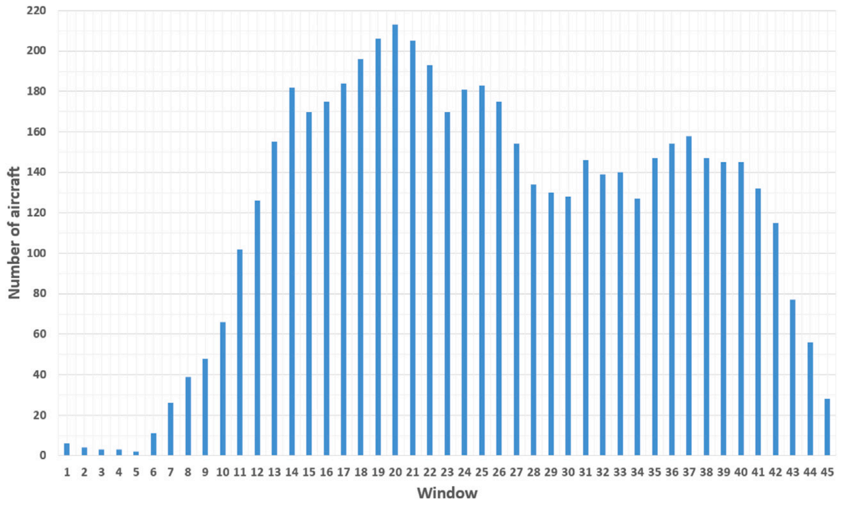


Fig. 9. The number of aircraft within each solution window.

Table 8
Comparison between the different scenarios. Airspace and runway severity included in the feedback loop condition.

	Scenario 1	Scenario 2	Scenario 3	Scenario 4
Average objective function value	37.61	37.01	37.34	36.61
Conflicts severity airspace (%)	39.12%	33.38%	39.64%	39.81%
Conflicts severity runway (%)	56.21%	53.97%	55.94%	55.56%
Computational time (sec.)	14 340.31	14 997.92	15 055.74	14 671.48

12, show the trends of the objective function and conflict severity, which are similar to the previous experiment. Regarding the correlation between the number of aircraft within each window and the results obtained, similar conclusions as previous scenarios can be drawn. For this scenario, the results are affected by the number of aircraft involved in each window (see Fig. 9), where the distribution of arrival and departure peaks throughout the time horizon can be deducted. The graph of Fig. 12, shows the trends of the conflict severity for both airspace (yellow bar) and runways (blue bar). Regarding airspace conflict severity, within the arrival peak (windows 12–18), severity values are between 60 and 80%. In this experiment we found high airspace conflict severity in many more windows (7,8,10, 18, 28, 34, 43, 44), although the number of airspace conflicts for these windows results low. Regarding the runway conflict severity, as for the previous experiment, we have a uniform trend with severity values between 50 and 80%.

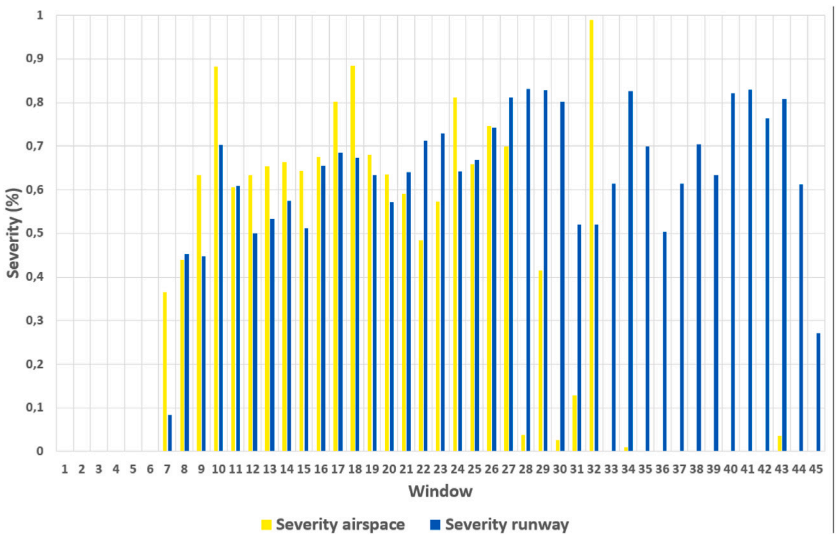


Fig. 10. Airspace and runway conflicts severity for the "Alpha" scenario (S2) throughout all the windows. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

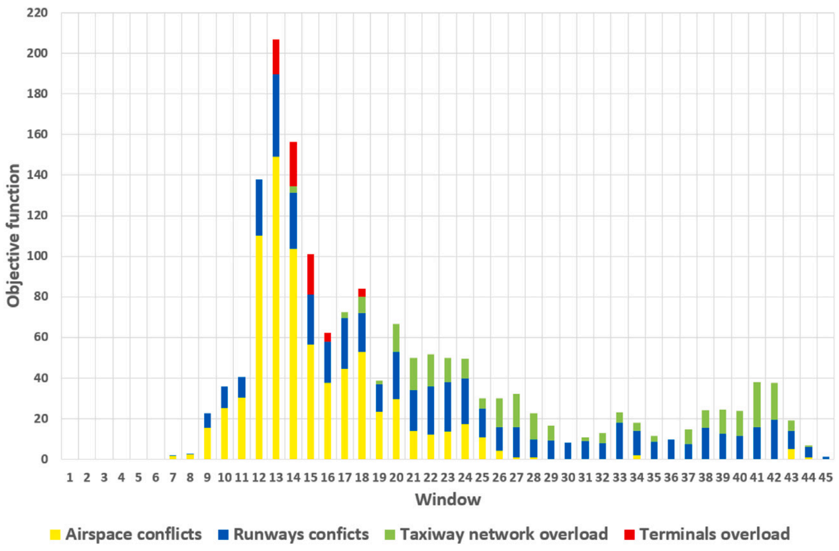


Fig. 11. Objective function for the "O.F. weight" scenario (S4) throughout all the windows. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

7.5. Comparison between the best scenarios among the two different Opt–Sim implementations

In Table 9, the results of the two best scenarios from the two different Opt–Sim implementations have been compared with the non-optimized case (original schedule), which was used as base case scenario. From these results we conclude that the Opt–Sim implementation 1, scenario "Alpha", achieved the overall best results as it improved the objective function value of 57.64%, and reduces conflict severity by 30.76% and 6.34% for airspace and runways, respectively. The best scenario from the Opt–Sim implementation 2 improved the performance when compared to the base case scenario in 56.96%, (reduction in objective function value) which is similar to the best scenario of the Opt–Sim implementation 1. Regarding conflict severity, the Opt–Sim implementation 2, obtained the following improvements:19.47% for airspace severity and 5.53% for runway severity. This comparison suggests that, under a specific setting of the main parameters M, N and ϵ, the evaluation of the conflict severity within the closed-loop feedback condition does not improve the performance for both objective value and conflict severity. In general, the conflict severity, as well as the objective function, were almost always higher than the parameters M and ϵ for each window, therefore, in most of the windows, the feedback loop condition was stopped mainly due to the maximum number of loops reached,

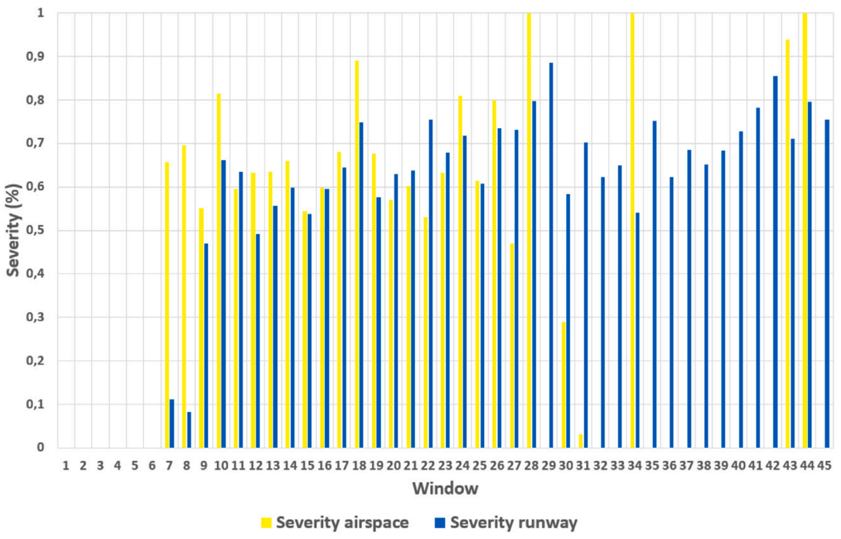


Fig. 12. Airspace and runway conflicts severity for the "O.F. weight" scenario (S4) throughout all the windows. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Table 9
Comparison between the best scenarios among the two Opt–Sim implementations and the non-optimized scenario.

	Non-optimized scenario	Opt–Sim 1: "Alpha" scenario (difference %)	Opt–Sim 2: "O.F. weight" scenario (difference %)
Average objective function value	78.1	33.08 (−57,64%)	36.61 (−56.96%)
Conflicts severity airspace (%)	49.44%	34.23% (−30.76%)	39.81% (−19.47%)
Conflicts severity runway (%)	58.81%	55.08% (−6.34%)	55.56% (−5.53%)

N. Further research will investigate a better setting of the values of the feedback loop condition parameters (M, N, ϵ), to evaluate if different settings would be able to achieve significant results.

8. Conclusion

The results presented in this paper show that the proposed algorithmic framework is able to drastically reduce aircraft conflicts during a long time horizon by fine tuning some key parameters. The experiments were based on two different Opt–Sim implementations, one based on the number of loops N and the objective function value tolerance M (Opt–Sim implementation 1), and another one based on additionally considering the conflict severity tolerance ϵ (Opt–Sim implementation 2). For each Opt–Sim implementations, 4 scenarios were tested: "Default", "Alpha", "Sep. Min." and "O.F. Weight". Although the methodology was not able to keep the conflict-free solution generated by the optimization element in the simulation environment, the best two scenarios from two different feedback loop condition implementations were able to improve the original non-optimized solution by about 57.64% (Opt–Sim implementation 1, Scenario "Alpha") and 56.96%(Opt–Sim implementation 2, Scenario "O.F. weight") which is a significant value compared to the close-to-reality scenario. Conflict severity for airspace and runway, a new metric introduced by this work, saw a decrease compared to the non-optimized solution by about 30.76% and 6.34% for the Opt–Sim implementation 1 - Scenario "Alpha"; 19.47% and 5.53% for the Opt–Sim implementation 2 - Scenario "O.F. weight". This confirmed the potential of the methodology for optimizing aircraft sequencing in airport operations under more realistic conditions than the ones considered with exact solutions. On the other hand, during some of the busiest windows of the operation, the algorithm showed some limitations as it showed the presence of conflicts accompanied by high values of conflict severity, this might be because of the level of congestion of the case study; for that reason, the methodology should be evaluated with another airport under different levels of congestion.

An aspect that is important to stress is that this framework is not problem-driven or metaheuristic-driven but a more Opt–Sim paradigmatic architecture, meaning that it can be applied to similar problems in other fields. Moreover, the fact that the Opt–Sim framework includes the inherent stochasticity of dynamic operations, makes it a suitable approach for many real-world systems that are affected by operational uncertainties, such as sea ports, manufacturing scheduling, and resource scheduling among others.

Further investigation can focus on the impact of the main parameters M (objective value tolerance), N (maximum number of loops), and ϵ (conflict severity tolerance) with the objective of reaching a balance between solution quality and computational effort. The results in this paper have shown that the main parameter that determined the stop of the feedback loop condition was mainly N, as in most of the windows the methodology needed several loops to obtain an improvement. Therefore, future work could focus on the implementation of a dynamic rule for setting the parameter N at each window, and as well as for the other two parameters, M and ϵ. In this manner, the efficiency of the Opt–Sim framework could be improved, as a solution could be obtained by running less feedback loops, which would lead to a reduction in the total computational time. In order to improve the quality of the solution, attention should also be given to the parameters that can be fine-tuned at each loop, such as problem-specific parameters or metaheuristic parameters, as it was indicated by the results when the best outcomes were provided by scenarios where the metaheuristic and objective function parameters were modified.

CRediT authorship contribution statement

Paolo Scala: Conceptualization, Methodology, Software, Validation, Formal analysis, Data curation, Investigation, Writing - original draft, Writing - review & editing, Visualization. **Miguel Mujica Mota:** Conceptualization, Methodology, Resources, Writing - review & editing, Supervision, Project administration, Funding acquisition. **Cheng-Lung Wu:** Conceptualization, Methodology, Resources, Writing - review & editing, Supervision, Project administration. **Daniel Delahaye:** Conceptualization, Methodology, Resources, Data curation, Supervision, Project administration.

Acknowledgments

The authors would like to thank the Aviation Academy and AMSIB faculty of the Amsterdam University of Applied Sciences for supporting this study and the Dutch Benelux Simulation Society (www.dutchbss.org) and EUROSIM for disseminating the findings of this study.

References

Anylogic, 2018. http://www.anylogic.com accessed online the 23rd of July 2018.

Arena, 2018. https://www.arenasimulation.com/, accessed online the 23rd of July 2018.

Banks, J., Carson, J.S., Nelosn, B.L., Nicol, D.M., 2000. Discrete-Event System Simulation, third ed. Prentice-Hall.

Beasley, J.E., Krishnamoorthy, M., Sharaiha, Y.M., Abramson, D., 2000. Scheduling aircraft landings: the static case. Transp. Sci. 34 (2), 190–197.

Brunner, D.T., Cross, G., McGhee, C., Levis, J., Whitney, D., 1998. Toward increase use of simulation in transportation. In: Proceedings of the 1998 Winter Simulation Conference, pp. 1169–1176.

Chica, M., Juan, A.J., Cordon, O., Kelton, W.D., 2017. Why simheuristics? Benefits, limitations, and best practices when combining metaheuristics with simulation. SSRN.

Confessore, G., Liotta, G., Grieco, R., 2005. Tactical decisions in the apron of Rome-Fiumicino airport. In Proceedings of the 2005 Winter Simulation Conference, pp. 1596–1605.

Durand, N., Gianazza, D., Gotteland, J.B., Alliot, J.M., 2016. Metaheuristics for Air Traffic Management, first ed. Wiley.

EUROCONTROL, 2018. Industry monitor. Technical Report, 2000, EUROCONTROL.

Frankovich, M.J., 2012. Air Traffic flow Management at Airports: A Unified Optimization Approach (Ph.D. thesis). Sloan School of Management at Massachusetts Institute of Technology.

Fu, M.C., 2002. Optimization for simulation: Theory vs practice. INFORMS J. Comput. 14 (3), 192–215.

Fu, M.C., Glover, F.W., April, J., 2005. Simulation optimization: A review, new develpments, and applications. In: Kuhl, M.E., Streiger, N.M., Amstrong, F.B., Joines, J.A. (Eds), Proceedings of the 2005 Winter Simulation Conference, Orlando, FL (USA), pp. 83–95.

Henderson, D., Jacobson, S.H., Johnson, A.W., 2003. The theory and practice of simulated annealing. In: Handbook of Metaheuristics. Springer, pp. 287–320.

Hu, X., Chen, W., 2007. Receding horizon control for aircraft arrival sequencing and scheduling. IEEE Trans. Intell. Transp. Syst. 8 (2), 254–263.

2007. Air Traffic Management. Doc. 4444. Technical report, ICAO.

Jian, N., Henderson, S.G., 2015. An introduction to simulation optimization. In: Yilmaz, I., Chan, W.K.V., Moon, I., Roeder, T.M.L., Macal, C., Rossetti, M.D. (Eds.), Proceedings of the 2015 Winter Simulation Conference, Huntington Beach, CA (USA) pp. 1780–1794.

Juan, A.J., Faulin, J., Grasman, S.E., Rabe, M., Figueira, G., 2015. A review of simheuristics: Extending metaheuristics to deal with stochastic combinatorial optimization problems. Oper. Res. Perspect. 2, 62–72.

Kirkpatrick, S., Gelatt, C.D., Vecchi, M.P., 1983. Optimization by simulated annealing. Science 220 (4598), 671–680.

Kleinman, N., Hill, S.D., Llends, V.A., 1998. Simulation optimization of air traffic delay cost, In: Proceedings of the 1998 Winter Simulation Conference, pp. 1177–1181.

Lee, H., 2014. Airport Surface Traffic Optimization and Simulation in the Presence of Uncertainties (Ph.D. thesis). Department of Aeronautics and Astronautics at Massachusetts Institute of Technology.

Liang, M., Delahaye, D., Marechal, P., 2017. Integrated sequencing and merging aircraft to parallel runways with automated conflict resolution and advanced avionics. Transp. Res. C 85, 268–291.

Ma, J., Delahaye, D., Sbihi, M., 2016. Integrated optimization of terminal manoeuvring area and airport. In: Proceedings of the 6th SESAR Innovation Days, pp. 265–270.

Mujica, M., 2015. Check-in allocation improvements through the use of a simulation-optimization approach. Transp. Res. Part A 77, 320–335.

Mujica, M., Flores, I., 2017. Applied Simulation and Optimization, Vol. 2, first ed. Springer.

Mujica, M., Scala, P., Delahaye, D., 2017. Improving Airport Performance through a Model-based and Optimization Approach. Springer, pp. 109–129.

Negahban, A., Smith, J.S., 2014. Simulation for manufacturing system design and operation: Literature review and analysis. J. Manuf. Syst. 33, 241–261.

Odoni, A., 1994. The importance of probability theory in the airport and air traffic control sectors. Int. J. Contin. Eng. Educ. 4, 105–113.

OptTek, 2018. http://www.opttek.com/products/optquest/, accessed online the 23rd of July 2018.

Scala, P., Mujica, M., Delahaye, D., Ji, M., 2019. Tackling uncertainty for the development of efficient decision support system in air traffic management. IEEE Trans. Intell. Transp. Syst. 1–14. http://dx.doi.org/10.1109/TITS.2019.2924981, (early access).

SimioLLC, 2018. https://www.simio.com/index.php, accessed online the 23rd of July 2018.

SimmodPro!, 2018. http://www.atac.com/itl/itl/simmod-pro.html accessed online the 23rd of July 2018.

Smeltink, J.W., Soomer, M.J., de Wall, P., van der Mei, R., 2005. An optimizaton model for airport taxi scheduling. In: Thirtieth Conference on the Mathematics of Operations Research.