Sim-Opt in the loop: algorithmic framework for solving airport capacity problems

Paolo Maria Scala, Miguel Mujica, Cheng-Leng Wu, Daniel Delahaye

Proceedings of the 2018 Winter Simulation Conference
M. Rabe, A.A. Juan, N. Mustafee, A. Skoogh, S. Jain, and B. Johansson, eds.

SIM-OPT IN THE LOOP: ALGORITHMIC FRAMEWORK FOR SOLVING AIRPORT CAPACITY PROBLEMS

Paolo Scala
Miguel Mujica

Aviation Academy
Amsterdam University of Applied Sciences
Weesperzijde 190
Amsterdam, 1097DZ, THE NETHERLANDS

Cheng-Leng Wu

School of Aviation
University of New South Wales
Kensington, NSW 2052, AUSTRALIA

Daniel Delahaye

ENAC LAB
Ecole Nationale de l`Aviation Civile
7 Avenue E. Belin
31055 Toulouse Cedex, FRANCE

ABSTRACT

The following paper presents an innovative approach for dealing with complex capacity problems in aviation. We introduce a sliding window framework composed by an optimization method with a simulation component. By applying this framework in diverse problems that are dependent on time it is possible to find feasible and close-to-reality solutions in shorter time than the ones that could be achieved by evaluating the problem in the complete time-horizon. The framework can be applied to solve diverse problems in aviation or similar industries. We exemplify the approach with a model of Paris Charles de Gaulle Airport in France.

1 INTRODUCTION

The air traffic in Europe is facing a constant growth, in 2017 a new peak record was reached with 10.6 million flights, surpassing the 2008 record with 10.2 million of flights, moreover, in 2017 the daily average air traffic movements were 4.3% bigger compared with the previous year (EUROCONTROL 2018a). Air traffic forecasts project an average yearly growth rate of 2.3% for the next seven years (EUROCONTROL 2018b), in spite of the increase of the oil price, that reached an average of 49 euro/barrel in 2017 against 41 euro/barrel in 2016. Due to all these facts, airports are under pressure, finding hard to balance their capacity with the amount of traffic to be handled. The 'majors' airports in Europe in terms of traffic movements and passengers handled, are already struggling with capacity problems and for some of them there are already plans for expansion for the long term horizon (Schiphol Group 2014). In the short term horizon, the main actors that seem the most affected by this situation are the air traffic controllers (ATCOs), since they need to handle all the traffic flowing through the airport. One of the main consequences is the so called 'Air Traffic Flow Management (ATFM) delays' (EUROCONTROL 2016), delays inducted by the air traffic controllers in order to manage the congestion in the airspace and on the ground to avoid disruptions. In this framework, we have developed a tool that could give support to ATCOs in decision-making processes relieving their pressure for handling the traffic. Over the last decades many studies have focused on automatizing ATCOs procedures and

decisions, for instance, tools for optimizing the sequence of aircraft flow in the airspace and on the ground (Beasley et al. 2000; Hu and Cheng 2005; Furini et al.2015; Smeltink et al. 2004; Rolin and Visser 2008; Guepet et al. 2017). Reliability and robustness are two of the most relevant characteristics when developing such tools, the former means allows to be enough precise in making decisions and to be able to predict the real operations accurately. The latter allows to be resilient to perturbation coming from real world operation, and therefore, dealing with the uncertainty present in the system. Most of the previous studies mentioned before, focused on finding an exact optimal solution, assuming the system as a deterministic one, however, this assumption will likely lead to obtain solutions that result unfeasible when applied to the real world instances. In the last years, simulation has been proved to be an excellent tool for dealing with stochastic systems in many different fields like logistic and manufacturing (Longo 2013), its characteristics makes it a good technique for dealing with the uncertainty of the real systems. However, the only use of simulation is not enough for solving combinatorial problems which imply searching in a big state space, and for that reason, the use of simulation has been supported by optimization (Lee 2014; Mujica 2015; Mujica and Flores 2017). Simulation, in this context, can be used as a validation tool for the optimized solution to test its feasibility once the variability is introduced in the simulation model, and also as a means for generating feedback to the optimization model, with the purpose of improving the solution making it more robust. In a previous work of Scala et al. (2017), optimization and simulation were combined together for solving the airport capacity problem regarding the sequencing of aircraft in the airspace and the for the management of the ground capacity. In their work the problem was solved optimizing the schedule by using a metaheuristic, within the framework of a sliding-window approach. They used discrete-event simulation for evaluating if the solution was feasible or not when uncertainty was introduced. The main gap that the authors found in the mentioned work was the lack of automation and also of communication between the optimization and simulation model, which can represent a big step forward in the improvement of the quality and robustness of the methodology. Motivated by this, in the present work, the authors propose the algorithmic architecture consisting of a loop including optimization and simulation. The simulation continuously checks the feasibility of the optimized solution, and based on a logic rule, it triggers the optimization model for generating a new solution. In this way, a loop is generated with the purpose of continuously improving the solution, the loop ends when the logic rule is fulfilled, meaning that an acceptable solution is found.

In this paper, this framework is described in detail, including the main parameters, the logic rule and the results obtained by running some experiments applied to a real-case scenario. The paper is organized as follows: in section 2 the problem under study is described, then, in section 3, the sim-opt framework is described. In section 4 the application of the framework to the problem and the relative results are presented and discussed. In the last section the conclusions are drawn and future steps of this research are discussed.

2 PROBLEM STATEMENT: TMA SEQUENCING AND AIRPORT GROUND CAPACITY MANAGEMENT UNDER UNCERTAINTY

The problem tackled in this work is the terminal manoeuvring area (TMA) sequencing and airport ground capacity management. Due to the growth of the air traffic and the limited capacity at airports, this problem has gained importance in the aviation field. The operations involved in this problem are the following: aircraft arrival sequence from the airspace to the runway, runway operations (arrivals and departures), taxiway and terminal occupancy, Figure 1 provides a schematic representation of the problem. For simplicity, the problem can be divided in two main ones, the arrival aircraft sequencing which involves the airspace surrounding airports (terminal manoeuvring area, TMA), and the ground capacity management which involves all the airside airport components such as: runways, taxiway network and terminals. However, the problem is tackled as one and considers all the aforementioned operations together. The objective of the problem is to come up with a schedule of aircraft that will fly smoothly in the airspace avoiding any potential conflict, and ensuring that the capacity of the main

ground components will not be exceeded. The reader is invited to refer to Scala et al. (2017) for more details about the problem.

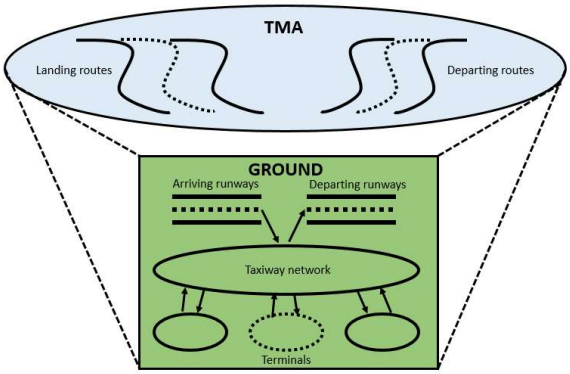


Figure 1: Schematic representation of the operations involved in the problem.

The model proposed in this work, has the purpose of minimizing the objective function, so to reach an ideal situation of zero conflicts. In the following paragraphs the conflicts are defined:

- Airspace conflicts are identified when two consecutive aircraft are separated by a distance which is less than the one established by International Civil Aviation Organization (ICAO) for safety purposes, in Table 1 these distances are showed.

Table 1: ICAO separation minima [NM].

		Leading aircraft			
	Category	A-380	Heavy	Medium	Light
Trailing Aircraft	A-380	-	-	-	-
	Heavy	6	4	3	3
	Medium	7	5	3	3
	Light	8	6	5	3

Ground conflicts, according to the different components, are divided in: runway conflicts, taxiway network conflicts and terminal conflicts.

- Runway conflicts are identified in a similar way as the airspace conflicts, where two consecutive aircraft that are crossing the runway need to keep a certain separation depending on the type of operation conducted on the runway (landing, take off, crossing).
- Taxiway network and terminal conflicts are identified in a similar way, that is, when their capacity is exceeded during the day (capacity overload). The capacity of the taxiway network and of the terminals are defined as the maximum number of aircraft that can simultaneously occupy the respective component, and in this case it is a numeric value chosen by the authors based on the layout of the airport considered in the study. The capacity overload is then calculated in terms of maximum value of capacity overload and the average capacity overload during the day.

The sum of all these components will compute the objective function, in Table 2 detailed structure of the objective function is presented. Once the objective function has been defined the optimization model will generate the optimized schedule based on decisions such as: entry time in the airspace; entry speed in

the airspace; choice of the landing runway; and the time when the aircraft can leave the gate for departing (push back time). The reader is invited to refer to Ma et al. (2016), for a more detailed explanation and definition of the problem decision variables.

Table 2: Objective function broken down in all its components.

Objective function	Airspace	Aircraft conflicts	Separation minima violation
	Ground	Runway conflicts	Separation minima violation
		Taxiway network conflicts	Maximum capacity overload
			Average capacity overload
		Terminal conflicts	Maximum capacity overload
			Average capacity overload

In this problem have been implemented some sources of uncertainty in order to make it closer to reality. The sources of uncertainty have been identified as:

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

3 METHODOLOGY: SIM-OPT IN THE LOOP

In the following section the formalization of the methodology is presented.

3.1 Sliding-Window Framework

The methodology developed in this work is based on a sliding window approach. This approach assumes that the whole time horizon is divided into windows of smaller size that are then solved by using optimization techniques. This approach suits well especially for the type of problems which have a long time horizon and for big combinatorial problems. The main advantage coming from the sliding-window approach is that it reduces the computational time and the size of the problem to solve. Another main advantage is that it allows to treat the problem in a dynamic way, considering the decisions taken in one window that affect the next one. Figure 2 shows a schematic representation of the sliding window approach.

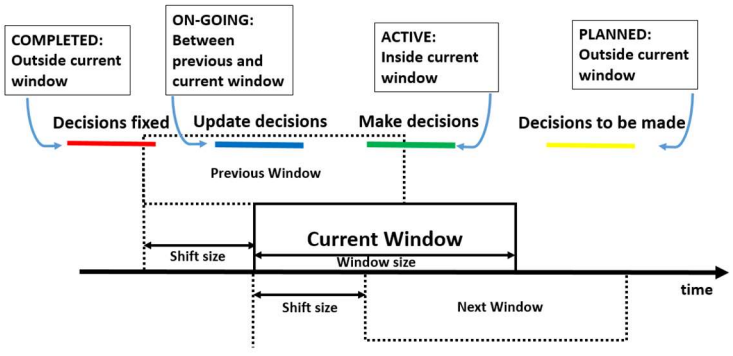


Figure 2: Sliding window approach scheme

As can be seen from Figure 2, the entities subject of the problem can assume four types of status: *Completed, On-going, Active and Planned.* In the *Completed* status entities are outside the current window and decision for them have already taken in a previous window; *On-going* status is for the entities that are partly outside and partly inside the current window and they have impact in the optimization process of the current window; *Active* status is for the entities that are completely inside the current window and for them decision will be taken for the current window; *Planned* status is for entities that are ahead the current window, they do not have any impact in the current window.

The main parameters of the sliding-window approach are the window size and the shift size, it is important to choose these values accurately since they affect the efficiency and effectiveness of the approach. Window and shift size should be chosen based on the size and the nature of the problem tackled. For the problem tackled in this work, the time horizon was 24 hours, and the entities involved were more than 1000, it was chosen a window size of 2 hours and a shift size of 30 minutes. Please refer to Scala et al. (2017) for further details about these two parameters.

3.2 Algorithmic Framework

In this work, for each window, simulation is combined together with optimization in a loop that aims at improving the intermediate solution of that window. The algorithm developed for this framework is summarized in Algorithm 1, where the parameters M and N play a crucial role as they drive the algorithm efficiency and effectiveness. In this algorithm, 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, so eventually the algorithm will come up with an optimized schedule composed by the best schedule of each window.

Algorithm 1 Sim-Opt Loop

```
1: procedure simOptLoop
2:   M ← max objective value tolerance;
3:   N ← max number of loops;
4:   nbRep ← number of replications;
5:   i ← 0;
6:   objective ← 0;
7:   minObjective ← 10000;
8:   do
9:     run optimization;
10:    do
11:      update simulation database with the optimized schedule;
12:      run simulation replications;
13:      if number of replications run is equal to nbRep then
14:        objective ← objective value calculated by the simulation;
15:        storeBestSolution(objective);
16:      end if
17:    while replication number is equal to nbRep;
18:    i ← i + 1;
19:  while objective is greater than M and i less than N;
20: end procedure
```

As it was mentioned before, the two parameters M and N are important for the algorithm, these two parameters (see Algorithm 1), are part of the condition for which the algorithm will continue with another loop or moving to the next window. For simplicity we refer to this condition as *loop condition.*

Algorithm 2 Store Best Solution

```
1: procedure storeBestSolution(objective)
2:   currentObjective ← objective;
3:   if currentObjective is less than minObjective then
4:     minObjective ← currentObjective;
5:     store the schedule;
6:   end if
7: end procedure
```

The *loop condition* checks if the objective value of the current window is greater that 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. It is important to choose the right values for these two parameters as they will affect the quality of the solution and also the computational feasibility. For example a big value of M will accept solution with many conflicts, meaning that that the algorithm will not be really effective in finding a quality solution, while a small number of M will force the algorithm to keep searching for another solution which might be costly in terms of computing performance. At the same way, choosing a big value for the parameter N will increase the chances of obtaining better solutions, so at least a minimum number of loops should be kept. On the other hand, too many loops might be hampering the performance time of the algorithm.

Figure 3 shows the architecture of the sim-opt loop algorithm, as it can be seen both optimization and simulation share a database where they store their schedules. The chronological order of the actions between the optimization and the simulation is the following: the optimization looks at the database for the original schedule and generates an optimized schedule; this optimized 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; the cycle repeats for each loop of each window. The database plays an important role since it allows the two model, the optimization and the simulation, to communicate between them and to share data.

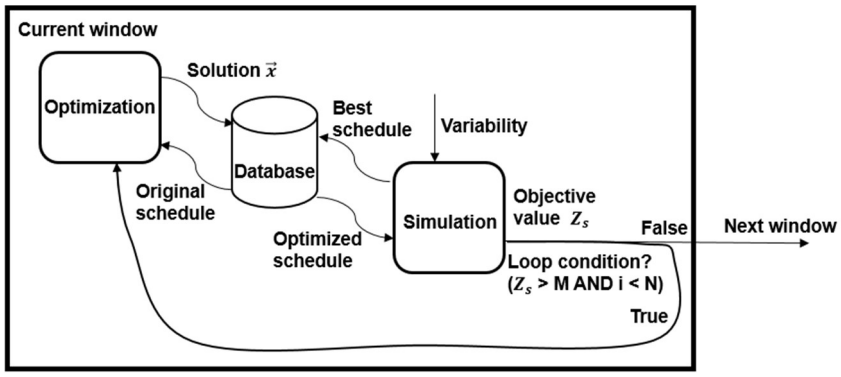


Figure 3: Architecture of the Sim-Opt Loop

In this work is presented also an extension of the algorithm, that is, a functionality able to modify a parameter of the optimization metaheuristic every time a new loop is triggered. In this way, for every loop, the optimization will generate a new solution not only based on the metaheuristic randomness of the search in the state space, but it will be based on different metaheuristic parameters that can drive the search differently. This parameter will be explained in next section, when the metaheuristic implemented for solving this problem is introduced.

3.2.1 Optimization by Simulated Annealing

The optimization process within this framework is based on Simulated Annealing (Kirkpatrick et al. 1983). The problem under study has been proved 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 value, and at the same time allows to explore the state space more broadly, avoiding to be trapped in a local minima. In other similar studies this metaheuristic has been proved to be very efficient (Ma et al. 2016; Man et al. 2017), for detailed information about the application of this metaheuristic to this problem please refer to (Ma et al. 2016). In this work we put our attention to a one of the main parameters of this metaheuristic, that is, the cooling schedule *alpha.* The cooling schedule parameter *alpha* is responsible for the convergence toward the optimum of the solution, since a big value will lead to have likely a better solution at the cost of having a longer convergence to the optimum, since it will search broadly in the state space. A smaller value of the cooling schedule parameter *alpha*, will lead to have a faster algorithm that will likely end with a local minima.

In this work we have extended the algorithm with a feature that increases the cooling schedule parameter *alpha* at each loop, so to have in the next loop a solution that explores the state space more in depth.

3.2.2 Simulating with DES

Simulation has been included in this algorithmic framework in order to deal with the uncertainty related to the system and to improve the optimized solution, serving as a feedback for the optimization process in order to generate an alternative solution. Simulation in this context, is able to overcome the flaw deriving from deterministic optimization, where exact solution are generated without considering the stochastic nature of systems. The simulation approach chosen for this work is discrete event simulation (DES). This 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, where entities follows some predefined path, undergo predefined processes and perform predefined actions. One of the main feature of DES is that the time passes based on specific events, and not in a continuous manner as it happens for continuous simulation approaches. Moreover, this approach is different than an agent-based simulation (ABS) due to the fact the entities involved in the system do not have their own logic, has it happens for ABS, but they follow the logic of the processes in which they are involved. An airport system, as it was described in section 2, implies all the characteristics described for DES, and that is why, for this problem, DES approach was chosen rather than other simulation approaches.

4 APPLICATION TO A REAL CASE STUDY

The framework developed in this work has been applied to a real case study in order to test its validity. The case study chosen was Paris Charles de Gaulle Airport (LFPG), in France. LFPG is one of the major airport in Europe in terms of air traffic movements and number of passengers transported per year. In the 2017 it was the second European airport regarding air traffic movements (476K) and number of passengers transported, 69.5 million (EUROCONTROL 2018a). The schedule used for the experiments includes one day of real operations, having 562 departures, 554 arrivals for a total of 1116 movements. These traffic characteristics makes it a good example for testing the effectiveness of the proposed framework. The airside layout of LFPG incudes four parallel runways used in a segregate mode, 2 for landings and 2 for departures, respectively; 3 terminals and a complex taxiway system. Tables 3 and 4 describe the main characteristics of the airside components.

The TMA of LFPG was considered only with its arrival routes, which are 2 routes coming from north and 2 coming from the south, having in total 4 different entry points. Each of the two southern and

northern routes converge into two merging points, which in turn split into other 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 routes in total.

Table 3: Airside components capacity.

Ground component	Capacity
Arrival runways	1
Departing runways	1
Taxiway network	20
Terminal 1	11
Terminal 2	89
Terminal 3	55

Table 4: Average taxi time [s].

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

4.1 Parameters of the Algorithmic Framework

As it was mentioned in section 3.2, the two parameters M and N of the algorithmic framework are critical for the efficiency and effectiveness in solving the problem, that is why they need particular attention. Regarding the parameter M, objective value tolerance, it was calculated as the 10 percent of the average number of conflicts for all the windows for the scenario run without optimizing the schedule. This means that in every window we want to keep the conflicts below this threshold identified by the parameter M, and in general we accept an amount of conflicts up to the 10 percent of the average number of conflicts for the windows. This criterion was arbitrarily chosen, but as it will be mentioned in the section about future steps of the research, it represents an area for future development of the framework. In the graph of Figure 4 presents the results in terms conflicts for the schedule run without optimization for all the windows. The average number of conflicts for all the windows, rounded to the nearest integer, was 168, therefore the parameter M was chosen as 17 (16.8 rounded to nearest the integer).

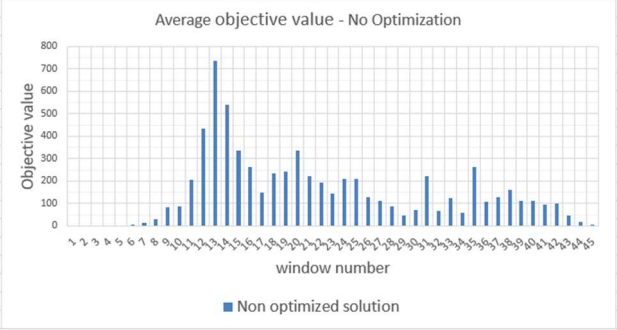


Figure 4: Average objective value without optimization.

It is important to mention that in the system some sources of uncertainty have been included as they are summarized in Table 5. These values were chosen at discretion of the authors as they leave as a further research a more accurate analysis for their choice.

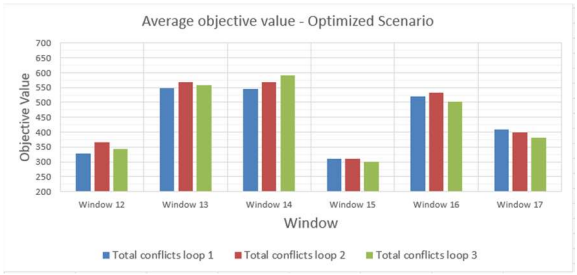
Table 5: Sources of uncertainty related to the problem.

Entry time in the airspace deviation	[-30 sec. , +30 sec.]
Taxiway time deviation	[-10% , +10%]
Off block time deviation	[-30 sec. , +30 sec.]

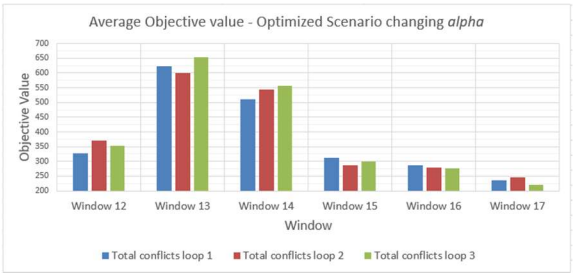
Regarding the other parameter of the framework, N maximum number of loops, the authors chose 3 as a value. The value of N impact especially in the computational aspect of the framework, therefore it can be adjusted based on the hardware resources of the machine that is being used. However, a minimum of loops must be kept in order to evaluate the performance of the framework.

4.2 Experimenting with the Framework

The framework was tested with two versions, one where the loop triggers the generation of a new solution without tuning any of the metaheuristic parameters, and another one where the cooling schedule parameter *alpha* of the metaheuristic is progressively increased according to the number of loops made. From now on we will refer to the former as "OPT-version" and the latter as the "modified-*alpha* OPT-version". The parameter *alpha* is a parameter with values that ranges between 0 and 1, in this work, it has a default value equal to 0.96. For every loop, *alpha* will be increased according to the following equation: $alpha = alpha + (i * 0.01)$, where i is the number of loop triggered. In the graphs of Figures 5(a) and 5(b), the two versions are compared for the windows with the highest number of conflicts, keeping in mind that the objective is to minimize the objective function value.



(a) OPT-version



(b) modified-*alpha* OPT-version

Figure 5: Average objective value for the windows with the most conflicts for the two different version of the framework.

Looking at both Figure 5(a) and 5(b), it can be noticed that for each window there are three bars of different colors, blue, red and green, representing loop 1, 2 and 3 , respectively. It can be furthermore noticed that the objective value changes between the loops, and so for example we can see in Figure 5(a) that the blue bar (loop 1) outperforms in the first three windows from the left to the right, while the last two windows are outperformed by the green bar (loop 3). In this case the algorithm will pick those schedules that have outperformed as they will be part of the overall best schedule. In figure 6 there is graph that shows an overview of the best solutions for the two different versions of the framework, so they can be compared with each other and also with the non-optimized schedule.

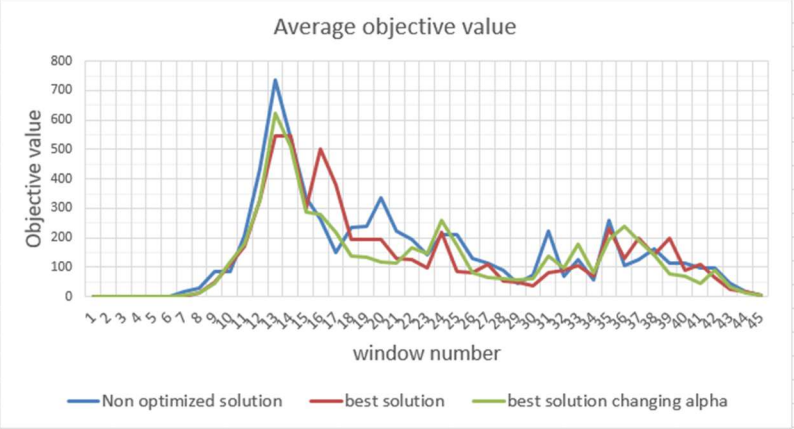


Figure 6: Comparison of the average objective value between the non-optimized scenario and the two version of the optimized framework.

In Figure 6 it can be noticed that almost for all the windows the blue line (non-optimized scenario) is outperformed by the green line (OPT-version) and the red one (modified-*alpha* OPT-version). In some windows, the blue line has less conflicts than the other, this behavior is explained by the following: when the optimization takes decisions for the aircraft (entry time in the airspace, entry speed, push back time), they will be moved along the time line, so they will be shifted to the next windows. For this reason in some windows the non-optimized scenario will have less aircraft inside the window and so the conflicts detected will be less. In general if we sum-up the conflicts detected in all the windows the "modified-*alpha* OPT-version" (green line) outperforms the other two. In the windows 16 to 20 (ranging from 60 to 224 less conflicts), 27 (around 45 less conflicts), 39-41 (ranging from 21 to 123 less conflicts) the "modified-*alpha* OPT-version" (green line) outperforms the other two, while the "OPT-version" (red line) outperforms the other two in the windows 13 (around 76 less conflicts), 22 to 25 (ranging from around 39 to 87 less conflicts), 30 (around 22 less conflicts), 31 (around 55 less conflicts), 33 (around less 75 less conflicts) and 36 (around 105 less conflicts). In general, the graph of Figure 6 shows that the green line reduces the conflicts to a bigger extent than the red line, meaning that the framework "modified-*alpha* OPT-version", is the best one in this case.

5 CONCLUSIONS AND FUTURE DEVELOPMENTS

The results presented in section 4.2 proved that the algorithmic framework developed in this work, was able to improve the solution for each window and therefore for the overall time horizon. However, the main findings showed the presence of a big amount of conflicts, especially in the busiest windows, due to the variability. In general, from a theoretical perspective the algorithmic framework proved its potential, while from a practical perspective it was not able to reach a feasible solution of zero conflicts. An aspect that is important to stress is the fact that this framework is not problem-driven or metaheuristic-driven, meaning that it can be applied to similar problems in other fields and that the metaheuristic implemented can be different as well. Moreover, the fact that it is able to include uncertainty, thanks to the simulation element of the framework, makes it a suitable approach for the types of systems that are affected by variability, a characteristic that most of the real-world systems have in common.

The future directions for the improvement of the framework are toward the choice of the main parameters of the framework M (objective value tolerance), and N (maximum number of loops), with the objective of reaching a balance between quality of the solution and computational effort. In order to improve the quality of the solution, particular attention should be given also to the parameters that can be fine-tuned at each loop, such as problem-specific parameters or metaheuristic parameters, as it was done in the experiments by fine-tuning the cooling schedule parameter *alpha*. For instance the weights of the

objective function can be fine-tuned so to drive the optimization to a certain direction but also other metaheuristic parameters for making the search in the state space more effective. Moreover, other techniques such as constraint relaxation can be implemented in order to get new alternative solutions at each loop.

ACKNOWLEDGMENTS

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

REFERENCES

Banks, J., J. S. Carson, B. L. Nelson, and D. M. Nicol. 2000. *Discrete-Event System Simulation*. 3rd ed. Upper Saddle River, New Jersey: Prentice-Hall, Inc.

Beasley, J.E., M. Krishnamoorthy, Y. M. Sharaiha, and D. Ambramson. 2000. "Sceduling Aircraft Landings – The Static Case." *Transportation Science* 2:180-197.

Brunner, D. T., G. Cross, C. McGhee, J. Levis, and D. Whitney. 1998. "Toward Increase Use of Simulation in Transportation". In *Proceedings of 30th Conference on Winter Simulation, edited by D. Medeiros, E. F. Watson, J. Carson, and M. Manivannan, 1169-1176. Washington, DC.*

EUROCONTROL. 2018a. Industry monitor. Issue 200.

EUROCONTROL. 2018b. EUROCONTROL Seven-Years Forecast February 2018. Flight Movements and Service Units 2018-2024.

Schiphol Group. 2014. Ondernemingsplan Lelystad Airport. March 2014.

EUROCONTROL. 2016. ATM Lexicon. ATFM Delay. https://ext.eurocontrol.int/lexicon/index.php/ATFM_delay, accessed April 30th, 2018.

Furini, F., M. P. Kidd, C. A. Persiani, and P. Toth. 2015. "Improved Rolling Horizon Approaches to the Aircraft Sequencing Problem". *J Sched* 18:437-447.

Guepet, J., O. Briant, J.-P. Gayon, and R. Acuna-Agost. 2017. "Integration of Aircraft Ground Movements and Runway Operations". *Transportation Research Part E* 104:131-149.

Hu, X. B., and W.-H. Chen. 2005. "Receding Horizon Control for Aircraft Arrival Sequencing and Scheduling". *IEEE Transactions on Intelligent Transportation Systems* 6: 189-197.

Kirkpatrick, S., C. D. Gelatt, and M. P. Vecchi. 1983. "Optimization by Simulated Annealing". *SCIENCE* 220:671-680.

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, Cambridge, Massachusetts. Available via https://dspace.mit.edu/handle/1721.1/87480.

Longo, F. 2013. "On the Short Period Production Planning in Industrial Plants: A Real Case of Study". *International Journal of Simulation & Process Modelling* 8:17-28.

Ma, J., D. Delahaye, and M. Sbihi. 2016. "Integrated Optimization of Terminal Maneuvring Area and Airport". In *Proceedings of the 6th SESAR Innovation Days,* 265-270, Delft, The Netherlands.

Liang, M., D. Delahaye, and P. Marechal. 2017. "Integrated Sequencing and Merging Aircraft to Parallel Runways with Automated Conflict Resolution and Advanced Avionics". *Transportation research Part C* 85:268-291.

Mujica, M. 2015. "Check-in Allocation Improvements Through the Use of a Simulation-Optimization Approach". *Transportation Research Part A* 77:320-335.

Mujica, M, and I., Flores. 2017. 2017. *Applied Simulation and Optimization 2,* 1st ed. Springer.

Negahban, A., and J. S. Smith. 2014. "Simulation for Manufacturing System Design and Operation: Literature Review and Analysis". *Journal of Manufacturing Systems*, 33:241-261.

Roling, P., and H. G., Visser. 2008. "Optimal Airport Surface Traffic Planning Using Mixed-Integer Linear Programming". *Int. J of Aerospace Engineering* 2008:1-11.

Scala, P., M. Mujica, and D. Delahaye. 2017. "A Down to Earth Solution: Applying a Robust Simulation-Optimization Approach to resolve Aviation Problems". In *Proceedings of the 2017 Winter Simulation Conference,* edited by W. K. V. Chan et al., 2626-2637. Las Vegas, Nevada: IEEE.

Smeltink, J. W., M. J. Soomer, P. de Waal, and R. van der Mei. 2004. "An Optimization Model for Airport Taxi Scheduling". *Informs Journal on Computing – INFORMS.*

AUTHOR BIOGRAPHIES

PAOLO SCALA is a PhD student at ENAC (Ecole Nationale de l'Aviation Civile) (France) sponsored by the Aviation Academy of the Amsterdam University of Applied Sciences (the Netherlands). His research interests lie in modeling and simulation and optimization applied to the aviation field. His email address is p.m.scala@hva.nl.

MIGUEL MUJICA is an Associate Professor at the Aviation Academy of the Amsterdam University of Applied Sciences in the Netherlands. His research interests lie in simulation techniques and O.R. in industrial, logistics and aviation problems. His email address is m.mujica.mota@hva.nl and web page www.mmujicamota.com.

CHENG-LENG WU is an Associate Professor at the School of Aviation of the University of New South Wales (Australia). His research interests lie in airport operations, airline operations, schedule optimization and passenger's behavior modelling. His email address is c.l.wu@unsw.edu.au.

DANIEL DELAHAYE is a Professor at ENAC (Ecole Nationale de l'Aviation Civile) (France). His research interests lie in stochastic optimization for airspace design and large-scale traffic assignment. His email address is delahaye@recherhe.enac.fr.