Local Search for Aircraft-Landing Planning

Nicolas Zufferey, Marie-Sklaerder Vié, and Roel Leus

Abstract In collaboration with EUROCONTROL (the *European Organisation for the Safety of Air Navigation*), the considered *Aircraft Landing Planning* (ALP) problem aims at minimizing delays (with respect to the published airline schedules) while satisfying the *separation constraint* (which imposes minimum threshold times between planes, ranging from 90 to 240 s). In this study, the landing sequence of the planes has to be determined first, and subsequently their associated landing times and *Holding-Stack Patterns* (HSPs) needed to meet such landing times. HSPs consist of making a plane wait for its planned landing time by making circular patterns close to the airport. The uncertainty due to winds is taken into account in the simulation procedure (it has an impact on the arrival times). The proposed solution method is a descent local search with restarts. It is quick enough with respect to implementation in real situations as it can be applied within seconds. Furthermore, the obtained results show that the delays can be reduced by approximately 50% on average when compared to a common practice rule.

Keywords Aircraft landing · Scheduling · Descent local search · Uncertainty

1 Introduction

From take-off to landing, planes are subjects to many uncertainties (e.g., wind, traffic). Moreover, the runway capacity of an airport is the bottleneck capacity of the landing process. Therefore, the landing time of each plane arriving at an airport has to be adjusted all along its flight trajectory to use the runway capacity at best despite uncertainty. One way to delay an airplane is to make it perform holding

N. Zufferey (✉) · M.-S. Vié
GSEM, University of Geneva—Uni Mail, Geneva, Switzerland
e-mail: marie-sklaerder.vie@unige.ch; n.zufferey@unige.ch

R. Leus
Faculty of Economics and Business, Leuven, Belgium
e-mail: roel.leus@kuleuven.be

L. Amorosi et al. (eds.), *Optimization in Artificial Intelligence and Data Sciences*,
AIRO Springer Series 8, https://doi.org/10.1007/978-3-030-95380-5_15

stack patterns in the vicinity of the airport (approach area) before its landing, but this creates bigger fuel consumption and noise pollution. Another way is to decrease (sometimes increase) the speed of the plane during its cruise (or make it do a detour using path stretching), but due to larger uncertainties at longer distance from destination, this may create unnecessary additional delays. Also, the fewer number of modifications there are on a landing time, the better it is for the *air traffic controllers*' (ATC) workload, as they have other priorities (mainly ensuring safety and avoiding collisions). The goal of *Aircraft Landing Planning* (ALP) is to optimize the landing times of the planes arriving at an airport runway. For this purpose, the planes to delay have to be selected, as well as how and when to delay them. Safety constraints imposing threshold distances between planes have to be satisfied, obviously.

The paper is organized as follows. A literature review is first conducted in Sect. 2. Second, the considered problem is formally introduced in Sect. 3. Next, the proposed optimization method is proposed in Sect. 4. Results are presented in Sect. 5, followed by conclusions in Sect. 6.

2 Literature Review

A comprehensive review covering the period up to 2011 can be found in [2]. It results that the main objective functions can be divided into four categories, presented below.

- *Safety and efficiency objectives.* Maximize: runway throughput, fairness among the aircrafts. Minimize: approach time before landing, ATCs' workload, aircraft taxi-out time, arrival delays.
- *Airline's objectives.* Maximize: punctuality with respect to the published landing times, adherence to airline priorities within their own flights, connectivity between flights. Minimize: operation costs (mainly fuel costs), total passenger delays.
- *Airport objectives.* Maximize: punctuality according to the operating schedule. Minimize: the need for gate changes due to delays.
- *Government objectives.* Minimize: environmental effects (noise and air pollution).

The review of [2], covering the literature up to 2011, leads to different observations. First, many theoretical studies show a great potential improvement of the runway utilization, but may not be feasible in practice due to ignored operational constraints (e.g., minimum time before landing, precedence constraints) or unreasonable computing times. Indeed, ATCs will always prefer fast (i.e., able to generate a solution within seconds) and satisfying (with respect to the considered objective) solution methods rather than optimal but time-consuming ones. Second, the definition of the objective functions and constraints varies a lot among the articles. Indeed, the involved parties (e.g., airport, companies, customers) have conflicting interests.

Therefore, selecting an appropriate and realistic objective function is a critical issue. Third, and most importantly, uncertainty occurs at different levels (e.g., weather, precision of the equipment, departure time, accuracy of information). However, in contrast with the supply chain management literature (e.g., [18]), very few studies take this aspect into account and actually use simulation, as [6, 15]. Indeed, most of the literature considers the static case rather than a dynamic (but realistic) environment. Interestingly, it is showed in [7] that: deterministic algorithms are sub-optimal in a dynamic environment; a FCFS method is robust (i.e., not too sensitive to variations in problem characteristics or data quality) and it has many advantages over many existing algorithms (e.g., generated sequence understandable by ATCs, stable and easy-to-estimate delays, already in place in many airports).

From 2012, some work on static ALP has still been published [11, 23]. But as highlighted above, from a practical standpoint, only the approaches that are able to take into account random events are relevant. The three main axes that integrate this dimension are queuing theory [16], robust optimization [12], and on-line optimization [3]. The approaches including uncertainty use Monte-Carlo simulation, and some dynamic solution methods (but without uncertainty, except the appearance of new flights) employ a rolling-horizon approach. Other relevant and recent references can be found in [4, 20, 21].

In the light of the weaknesses of the existing literature, various promising research directions are identified below.

(1) Consideration of uncertainty. Obviously, as most variables in ALP are stochastic, a quick, accurate and dedicated simulation tool is mandatory to evaluate the true quality of a solution. The Monte-Carlo simulation that is usually proposed for ALP is probably not the best tool according to these criteria. Generally, the existing literature does not take uncertainty into account. A scarce literature proposes either robust solution within a static approach, or a dynamic solution method but without any robustness guarantee. An interesting approach could integrate robustness in the on-line process (to avoid a prohibitive number of rescheduling actions) [14].

(2) Appropriate solution methods. The existing metaheuristics (generally genetic and ant colony algorithms) do not seem the most appropriate for ALP. Indeed, they need a long learning phase (e.g., Variable Neighborhood Search [5], Tabu Search [22]), and even though they can provide feasible solutions at any time in the computation, there is little chance that the quality of the solutions is good after only a few seconds of computation. Filtering techniques would be an efficient option to definitely discard unpromising decisions from the solution space (e.g., [13]).

(3) Instance calibration. An instance has to be designed in a realistic way. In most of the literature, instances with hundreds of flights are tackled (with metaheuristics), as well as instances with around 50 flights (with exact methods or metaheuristics). However, everyday and for many important European airports, the demand follows patterns with marked peaks (e.g., hub periods of 3 h) and then lower traffic levels [10]. For these reasons, a relevant instance can

be defined by a time horizon of 3 h (i.e., covering a peak) and a single landing runway. Knowing that the time between two consecutive aircraft belongs to interval [90, 240] s, the most relevant instance size ranges from 90 to 120 flights. Surprisingly, this range is usually not considered in the literature.

(4) *Planning horizon.* In a dynamic case, an important optimization would be the size of the considered rolling horizon H, as it can have a crucial impact on the quality of the obtained solution [19]. A tradeoff has to be found between (1) waiting as much as possible before delaying a flight (which minimizes the uncertainties, but increases the noise pollution and fuel costs), and (2) anticipating and delaying a flight as early as possible (more risky approach if too many uncertainties, but more efficient otherwise). The challenge is that the larger H is, the more flexibility we have in the optimization, but the more uncertainty we have too (mainly because of pop-up flights, as presented below).

3 Considered Problem

Each instance covers a 3-h planning horizon, which allows capturing the peak period of most airports. A rolling planning window $H^t = [t, t + w[$ (with $w = 45$ min) is associated with the current time t, with time steps of $\Delta t = 30$ s. At each time t, we only consider the flights that are in cruise and have their landing planned in H^t. The flights that have their landing in H^t but are not yet in cruise are only considered when they take-off. They are called the *pop-up* flights.

Each flight has different stages: (1) take-off, (2) cruise, (3) approach, (4) landing (the last $L = 15$ min, during which no modification is performed). In this paper, we only consider stages (2) and (3). From a practical standpoint, an initial schedule is first built when each flight enters the planning window (i.e., when it has taken off in the case of a pop-up flight, or when its expected landing time is within the next 45 min). Next, we can reschedule it (within the landing sequence) or make it wait to meet its planned arrival time (through HSPs). The popular *First-Come-First-Served* (FCFS) rule is employed to build the initial schedule. FCFS ranks the flights according to their entry times in H^t (i.e., with respect to increasing published arrival times, like the earliest-due-date rule in job scheduling). FCFS used to be the most employed current-practice approach [9], and it is an optimal rule for the single-machine job-scheduling problems when the maximum tardiness has to be minimized [17] (in our case we have to minimize the average tardiness).

We propose the following mathematical model (P^t) for each time t. Among the flights that have already taken off, we only consider the flights with planned landing times up to time $t + w$. Let J^t be the set of (say n) flights considered in H^t. For each flight $j \in J^t$, the following data is given:

- r_j: release date (i.e., take-off time).
- d_j: due date (i.e., published landing time).
- p_j^t: processing time (i.e., remaining time—in seconds—during the cruise phase).

- $s_{j,j'}$: set up time between flights j and j'. More precisely, for each pair (j, j') of flights such that j has to land before j', their landing times must be separated by $s_{j,j'} \in \{90, 120, 150, 180, 210, 240\}$ s, depending on the involved plane types.

We have two types of decision variables:

- determine the vector Π^t of the positions of the flights involved at time t (i.e., improve the current landing sequence by performing an optimization method);
- for each flight j, determine a feasible landing time C_j^t (with respect to the separation constraint) and assign a HSP of duration W_j^t in order to meet C_j^t.

The objective function f to minimize is the sum of all positive delays (i.e., the total tardiness), and it is given in Eq. (1). In contrast with the production-planning literature, negative delays are not penalized (e.g., [24]).

$$f = \sum_{j \in J^t} \max\{C_j^t - d_j, 0\} \tag{1}$$

Below, Constraints (2) impose that two flights are not scheduled in the same position. Constraints (3) capture the separation constraints. Constraints (4) determine the expected landing times. Constraints (5) are the domain constraints.

$$\Pi_j^t \neq \Pi_{j'}^t \qquad \forall j, j' \in J^t \tag{2}$$

$$C_{j'}^t \geq C_j^t + s_{j,j'} \qquad \forall j, j' \in J^t \text{ such that } \Pi_j^t + 1 = \Pi_{j'}^t \tag{3}$$

$$C_j^t = t + p_j^t + W_j^t + L \qquad \forall j \in J^t \tag{4}$$

$$\Pi_j^t \in \{1, \ldots, n\}, C_j^t \geq 0, W_j^t \geq 0 \qquad \forall j \in J^t \tag{5}$$

This problem can be seen as a variant of a single-machine total-tardiness problem with setup times, which is NP-hard even without setup times [8].

4 Optimization Method

Algorithm 1 presents how to roll the planning window H^t over the full 3-h planning horizon.

In Step 2, the landing positions Π^t of the new flights are computed with the following insertion rules used in practice:

- each pop-up flight j that just entered H^t (i.e., $t \geq r_j$ but $t - \Delta t < r_j$, and $t \geq d_j - w$) is added to the landing sequence at a position Π^t such that its due date is respected (i.e., j is placed before all flights j' such that $C_{j'}^t \geq d_j$ but after all the other flights);

Algorithm 1: Optimization for each time step t

Initialization: set $t = 0$, $J^t = J^{t-\Delta t} = \emptyset$ and $\Pi^t = \Pi^{t-\Delta t} = ()$.
While (not all flights have landed), **do:**

1. Update J^t: remove the flights that have started landing (i.e., each flight j for which $t \geq C^t_j - l$), and add the flights that have just entered the updated planning window H^t (i.e., each flight j for which $t \geq r_j$ and $t \geq d_j - w$).
2. Compute the positions of the new flights (i.e., the flights that are in J^t but not in $J^{t-\Delta t}$) to obtain the vector Π^t, based on $\Pi^{t-\Delta t}$ and the insertion rules.
3. Update the remaining cruise time for each flight j: set $p^t_j = p^{t-\Delta t}_j - \Delta t \cdot (1 + u^t_j)$.
4. Update C^t and W^t according to the new flight positions Π^t and the processing times p^t.
5. Improve solution (Π^t, C^t, W^t) with a solution method.
6. Move to the next time step: set $t = t + \Delta t$ and $H^t = [t, t + w[$.

- each flight j that took off a while ago but just entered H^t (i.e., $t \geq r_j$ and $t \geq d_j - w$, but $t - \Delta t < d_j - w$) is put at the end of the landing sequence (FCFS rule).

In Step 3, each remaining processing times p^t_j is updated while considering an uncertainty parameter u_t randomly generated following the EUROCONTROL specifications. u_t generates a deviation (e.g., due to wind) of the cruise speed of around 7% (with an average of 0%, as positive deviations are compensated by negative ones). In Step 4, and after each modification of Π^t, the values of C^t and W^t are updated with the following current-practice rules. First, we re-number all flights of J^t as $j_1, j_2, \ldots, j_n$ such that $\Pi^t_{j_1} < \Pi^t_{j_2} < \ldots < \Pi^t_{j_n}$. Next, for $k = 1$ to n, we perform steps (S1) and (S2).

- *Step (S1).* $C^t_{j_k} = \max\{C^t_{j_{k-1}} + s_{j_{k-1}, j_k}, t + p^t_{j_k} + L\}$ (i.e., the arrival time of j_k is as close as possible to the arrival time of the previous flight j_{k-1}, or as soon as j_k can land).
- *Step (S2).* $W^t_{j_k} = C^t_{j_k} - (t + p^t_{j_k} + L)$ (i.e., the flight turns over the airport if it is too early with respect to the planned landing time).

As (1) the considered problem is NP-hard, (2) up to 24 flights are involved in H^t, and (3) the allowed computing-time limit T is very short ($T = \Delta t = 30$ s), quite a number of potential solution methods are not suitable for Step 5. Indeed, exact methods, cumbersome population-based metaheuristics (e.g., genetic algorithms, ant algorithms) or metaheuristics using a somewhat long learning process (e.g., simulated annealing) are too slow. In contrast, a descent local search (DLS) appears as a promising candidate.

DLS takes as input the solution from Step 4. At each iteration, a neighbor solution S' is generated from the current solution $S = (\Pi^t, C^t, W^t)$ by performing the best *Reinsert* move on S. A move *Reinsert* consists of changing the position Π^t_j of a flight $j \in J^t$ within the landing sequence. After each modification of Π^t, the associated variables (C^t, W^t) must be updated to have a feasible solution S' (separation constraint) and to know $f(S')$. The search process stops when no

improvement of S is achieved during an iteration. In order to use the full time budget T, DLS is restarted when it encounters a local minimum (which occurs almost every second). The best visited solution is returned at the end.

At each iteration, two mechanisms are used for reducing the computational effort. First, the new position for the investigated flight j must be in $[\Pi_j^t - 5; \Pi_j^t + 5]$. This kind of *Constrained Position Shifting* is standard [1]. Indeed, from a practical standpoint, it seems straightforward to reschedule a flight not too far away from its initial position. Second, only a random proportion ρ (tuned to 50%) of the possible neighbor solutions is generated. These mechanisms allows to perform more iterations during T seconds, which increases the exploration capability of DLS.

5 Results

The algorithms were coded in C++ (under Linux, 3.4 GHz Intel Quad-core i7 processor, 8 GB of DDR3 RAM).

Table 1 compares the proposed DLS approach with FCFS (i.e., a common practice rule, see Algorithm 1 without Step 5). Average results (over all the instances) are indicated in bold face in the last line. For each instance (provided by EUROCONTROL), the following information is provided: the number N of flights, the largest number n_{max} of flights encountered in a planning window, the average delay and the maximum delay (for both DLS and FCFS). The two latter quantities

Table 1 Comparison of FCFS with DLS for 15 instances provided by EUROCONTROL

			FCFS		DLS		% Gain	
Instance	N	n_{max}	Avg. delay	Max delay	Avg. delay	Max delay	Avg. delay	Max delay
1	59	16	91.36	305.00	59.96	450.20	34%	−48%
2	35	10	156.08	528.20	96.98	490.40	38%	7%
3	64	20	154.41	447.60	93.05	456.00	40%	−2%
4	79	24	388.30	782.60	228.31	1672.60	41%	−114%
5	53	14	189.53	545.00	110.36	538.40	42%	1%
6	79	21	328.86	709.20	181.40	1629.20	45%	−130%
7	75	18	208.88	558.80	111.38	475.40	47%	15%
8	75	24	288.57	651.40	143.88	1480.60	50%	−127%
9	62	16	240.26	539.20	118.33	505.20	51%	6%
10	70	22	207.41	503.20	99.57	563.40	52%	−12%
11	72	22	280.79	631.20	131.57	800.20	53%	−27%
12	71	18	170.19	514.80	77.72	475.20	54%	8%
13	97	23	386.69	903.00	174.56	1247.60	55%	−38%
14	61	15	195.54	644.60	86.58	612.80	56%	5%
15	97	20	234.52	569.00	97.04	662.20	59%	−16%
Average results			**234.76**	**588.85**	**120.71**	**803.96**	**48%**	**−31%**

are computed with respect to all flights (in seconds), and averaged over 5 runs (with different uncertainty scenarios). The percentage gains of DLS (compared to FCFS) are given in the two last columns (a negative value indicates a better performance for FCFS).

We can see that DLS can significantly reduce the average delays (almost 50%). Interestingly, the improvement is somewhat increasing with the difficulty of the instance (i.e., with N and n_{max}), but further investigations are required to understand the benefit of DLS with respect to the instance characteristics. FCFS is often better regarding the maximum delay. This makes sense as FCFS guarantees optimality for minimizing the maximum delay (but not the average delay) for single-machine job-scheduling contexts. However, DLS can sometimes do better even for the maximum delay, as it reacts to uncertainties whereas FCFS does not.

6 Conclusion

A lot of work has been done on *Aircraft Landing Planning* (ALP), from exact methods (mainly branch-and-bound) to metaheuristics (e.g., numerous genetic algorithms). These methods, however, entail two main weaknesses. First, they are usually static (i.e., assume that all data is well-known), whereas the problem presents many uncertainties (e.g., weather, traffic, interaction with other flights). Second, the existing approaches generally do not match the quickness of the decision environment in which the decision makers have to work.

ALP is a challenging problem as the runway capacity is the bottleneck of many airports. In collaboration with EUROCONTROL, this study proposes a quick and efficient descent-based solution method for minimizing delays. Indeed, solutions can be obtained within seconds (which is appropriate for real-world implementation) and the average delay is reduced by almost 50%. Possible future works include the joint consideration of various objectives (for instance, in a lexicographic fashion as in [25]), and the development of refined algorithms and other techniques (e.g., speed adjustments, detours) to make the flights meet their landing times in order to reduce the over-the-airport traffic.

Acknowledgments This study was partially financed by EUROCONTROL (SESAR2020 program). The authors would like to thank Mr. Raphaël Christien for his availability and advices.

References

1. Balakrishnan, H., Chandran, B.G.: Algorithms for scheduling runway operations under constrained position shifting. Operations Research **58**(6), 1650–1665 (2010)
2. Bennell, J.A., Mesgarpour, M., Potts, C.N.: Airport runway scheduling. 4OR Q. J. Oper. Res. **9**(2), 115–138 (2011)

3. Bennell, J.A., Mesgarpour, M., Potts, C.N.: Dynamic scheduling of aircraft landings. Eur. J. Oper. Res. **258**(1), 315–327 (2017)
4. Bianco, L., Dell'Olmo, P., Giordani S.: Scheduling models for air traffic control in terminal areas. J. Sched. **9**(3), 180–197 (2006)
5. Bierlaire, M., Thémans, M., Zufferey, N.: A heuristic for nonlinear global optimization. INFORMS J. Comput. **22**(1), 59–70 (2010)
6. Beasley, J.E., Krishnamoorthy, M., Sharaiha, Y.M., Abramson, D.: Displacement problem and dynamically scheduling aircraft landings. J. Oper. Res. Soc. **55**(1), 54–64 (2004)
7. Brentnall, A.R., Cheng, R.C.H.: Some effects of aircraft arrival sequence algorithms. J. Oper. Res. Soc. **60**(7), 962–972 (2009)
8. Du, J., Leung, J.Y.T.: Minimizing total tardiness on one machine is NP-hard. Math. Oper. Res. **15**(3), 483–495 (1990)
9. Erzberger, H.: Design Principles and Algorithms for Automated Air Traffic Management. AGARD Lecture Series No. 200: Knowledge-Based Functions in Aerospace Systems, Madrid, Paris, and San Francisco (1995)
10. EUROCONTROL.: Performance Review Report 2015. Technical report, 2016
11. Faye, A.: Solving the aircraft landing problem with time discretization approach. Eur. J. Oper. Res. **242**(3), 1028–1038 (2015)
12. Heidt, A., Helmke, H., Liers, F., Martin, A.: Robust runway scheduling using a time-indexed model. In: Proceedings of the 4th SESAR Innovation Days, Madrid, pp. 1–8 (2014)
13. Hertz, A., Schindl, D., Zufferey, N.: Lower bounding and tabu search procedures for the frequency assignment problem with polarization constraints. 4OR **3**(2), 139–161 (2005)
14. Krishnan, S., Barton, G.W., Perkins, J.D.: Robust parameter estimation in on-line optimization - part I. Methodology and simulated case study. Comput. Chem. Eng. **16**(6), 545–562 (1992)
15. Moser, I., Hendtlass, T.: Solving dynamic single-runway aircraft landing problems with extremal optimisation. In: Proceedings of the 2007 IEEE Symposium on Computational Intelligence in Scheduling, Honolulu, pp. 206–211 (2007)
16. Nikoleris, T., Hansen, M.: Queueing models for trajectory-based aircraft operations. Transportation Science **46**(4), 501–511 (2012)
17. Pinedo, M.: Scheduling: Theory, Algorithms, and Systems. Springer (2016)
18. Respen, J., Zufferey, N., Wieser, P.: Three-level inventory deployment for a luxury watch company facing various perturbations. J. Oper. Res. Soc. **68**(10), 1195–1210 (2017)
19. Russell, R.A., Urban, T.L.: Horizon extension for rolling production schedules: Length and accuracy requirements. Int. J. Prod. Econ. **29**(1), 111–122 (1993)
20. Samà, M., D'Ariano, A., Pacciarelli, D.: Rolling horizon approach for aircraft scheduling in the terminal control area of busy airports. Transp. Res. E Logist. Transp. Rev. **60**, 140–155 (2013)
21. Samà, M., D'Ariano, A., Pacciarelli, D.: Scheduling models for optimal aircraft traffic control at busy airports: tardiness, priorities, equity and violations considerations. Omega **67**, 81–98 (2017)
22. Schindl, D., Zufferey, N.: A learning tabu search for a truck allocation problem with linear and nonlinear cost components. Naval Res. Logist. **62**(1), 32–45 (2015)
23. Tavakkoli-Moghaddam, R., Yaghoubi-Panah, M., Radmehr, F.: Scheduling the sequence of aircraft landings for a single runway using a fuzzy programming approach. J. Air Transp. Manag. **25**, 15–18 (2012)
24. Thevenin, S., Zufferey, N., Widmer, M.: Order acceptance and scheduling with earliness and tardiness penalties. J. Heurist. **22**(6), 849–890 (2016)
25. Thevenin, S., Zufferey, N., Potvin, J.-Y.: Makespan minimisation for a parallel machine scheduling problem with preemption and job incompatibility. Int. J. Prod. Res. **55**(6), 1588–1606 (2017)