Minimizing Delays in Aircraft-Landing Scheduling

Marie-Sklaerder Vié[1] Nicolas Zufferey[1] Roel Leus[2]

[1] GSEM - University of Geneva, Switzerland
marie-sklaerder.vie@unige.ch, n.zufferey@unige.ch
[2] Faculty of Economics and Business, KU Leuven, Belgium, roel.leus@kuleuven.be

Keywords: aircraft scheduling, heuristic, delay minimization.

1 Introduction

In collaboration with EUROCONTROL (*European Organization 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 seconds). 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).

Different pointers on ALP can be found in the literature (Avella *et al.* 2017, Bennell *et al.* 2017, Furini *et al.* 2015, Vié *et al.* 2018). 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. The paper is organized as follows. The problem is formally introduced in Section 2. Next, the proposed optimization method is designed in Section 3. Results are given in Section 4, followed by conclusions in Section 5.

2 Problem Formulation

Each instance covers a 3-hour planning horizon, which allows capturing the peak period of most airports. A rolling planning window $H^t = [t, t + w[$ (with $w = 45$ minutes) is associated with the current time t, with time steps of $\Delta t = 30$ seconds. 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 $L = 15$ minutes, 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 minutes). 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). FCFS used to be the most employed current-practice approach (Erzberger 1995), and it is an optimal rule for the single-machine job-scheduling problems when the maximum tardiness has to be minimized (Pinedo 2016) (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 $s_{j,j'} \in \{90, 120, 150, 180, 210, 240\}$ seconds, depending on the involved plane types.

We have two types of decision variables: (1) 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); (2) 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): $f = \sum_{j \in J^t} \max\{C_j^t - d_j, 0\}$. Constraints (1) impose that two flights are not scheduled in the same position. Constraints (2) capture the separation constraints. Constraints (3) determine the expected landing times. Constraints (4) are the domain constraints.

$$\Pi_j^t \neq \Pi_{j'}^t \qquad \forall j, j' \in J^t \tag{1}$$

$$C_{j'}^t \geq C_j^t + s_{j,j'} \qquad \forall j, j' \in J^t \text{ such that } \Pi_j^t + 1 = \Pi_{j'}^t \tag{2}$$

$$C_j^t = t + p_j^t + W_j^t + L \qquad \forall j \in J^t \tag{3}$$

$$\Pi_j^t \in \{1, \ldots, n\}, C_j^t \geq 0, W_j^t \geq 0 \qquad \forall j \in J^t \tag{4}$$

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 (Du and Leung 1990).

3 Optimization Method

Algorithm 1 presents how to roll the planning window H^t over the full 3-hour planning horizon. In Step 2, the landing positions Π^t of the new flights are computed with the following insertion rules used in practice: (1) 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); (2) 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_j^t 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, 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 $\Pi_{j_1}^t < \Pi_{j_2}^t < \ldots < \Pi_{j_n}^t$. Next, for $k = 1$ to n, we perform steps (S1) and (S2).

(S1) $C_{j_k}^t = \max\{C_{j_{k-1}}^t + s_{j_{k-1},j_k}, t + p_{j_k}^t + L\}$ (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).

(S2) $W_{j_k}^t = C_{j_k}^t - (t + p_{j_k}^t + L)$ (i.e., the flight turns over the airport if it is too early with respect to the planned landing time).

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 $t \geq C_j^t - l$), 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 $p_j^t = p_j^{t-\Delta t} - \Delta t \cdot (1 + u_j^t)$.
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 $H^t = [t, t + w[$.

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 ($T = \Delta t = 30$ seconds), 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_j^t 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 (it 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 (Balakrishnan and Chandran 2010). 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.

4 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). 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 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. It 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.

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%**

5 Conclusion

The *Aircraft Landing Planning* 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 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.

Acknowledgements. Study partially financed by EUROCONTROL (SESAR2020 program). Many thanks to Mr. Raphaël Christien for his availability/advices.

References

Avella P., Boccia M., Mannino C., Vasilyev I., 2017, "Time-Indexed Formulations for the Runway Scheduling Problem", *Transportation Science*, Vol. 51 (4), pp. 1031-1386.

Balakrishnan H., Chandran B.G., 2010, "Algorithms for scheduling runway operations under constrained position shifting", *Operations Research*, Vol. 58 (6), pp. 1650-1665.

Bennell J.A., Mesgarpour M., Potts C.N., 2017, "Dynamic scheduling of aircraft landings", *European Journal of Operational Research*, Vol. 258 (1), pp. 315-327.

Du J., Leung J.Y.T., 1990, "Minimizing total tardiness on one machine is NP-hard", *Mathematics of Operations Research*, Vol. 15 (3), pp. 483-495.

Erzberger H., 1995, "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*, Vol. 7 (2).

Furini F., Kidd M.P., Persiani C.A., Toth P., 2015, "Improved rolling horizon approaches to the aircraft sequencing problem", *Journal of Scheduling*, Vol. 18, pp. 435-447.

Pinedo M., 2016, "Scheduling: Theory, Algorithms, and Systems", Springer.

Vié M.-S., Zufferey N., Leus R., 2018, "Aircraft landing planning: past, present and future", *Proceedings of the 19th ROADEF Conference*, Lorient, France.