Coupling Mathematical Optimization and Machine Learning for the Aircraft Landing Problem

Sana Ikli*, Catherine Mancel*, Marcel Mongeau*, Xavier Olive†, Emmanuel Rachelson‡

*ENAC
Université de Toulouse
Toulouse, France

†ONERA – DTIS
Université de Toulouse
Toulouse, France

‡ISAE–SUPAERO
Université de Toulouse
Toulouse, France

***Abstract*—The Aircraft Landing Problem (ALP) consists in sequencing aircraft on available runways, and scheduling their landing times taking into consideration various operational constraints. It is an NP-hard problem and an ongoing challenge for both researchers and air traffic controllers. A straightforward solution widely used in practice consists in scheduling aircraft using the simple "First-Come First-Served" (FCFS) sequence. However, it rarely provides optimal solutions, especially in large congested airports.**

In this work, we propose a heuristic approach based on optimistic planning to solve the problem. We model the ALP as an environment of *states*, *actions*, *transitions* and *costs*, then explore the resulting search tree so as to identify a near-optimal sequence of actions within a limited time budget. In a previous contribution, we used the "First-Come First-Served" (FCFS) rule in the computation of the costs, to estimate the cost of the cheapest path (sequence of actions) from a given state. Now, we investigate a baseline model based on linear regression, and two different machine learning (ML) models trained on a large number of optimized solutions. These models can quickly and accurately estimate the cheapest-sequence cost, which helps the search to identify a near-optimal branch more efficiently.

Numerical experiments are performed on our publicly available data set, and show that using machine learning models in our heuristic search does not only ameliorate the previous results in terms of percentage improvement, but also reduces the optimality gap within a computation time that is compatible with on-line operations.

Keywords: Aircraft landing problem; Mixed integer linear programming; Optimistic planning; Machine learning

I. Introduction

According to the International Air Transport Association (IATA), air passengers are expected to double by 2036 [1]; this increasing demand on air transportation exposes the available infrastructure to a risk of saturation. Constructing new infrastructures (runways, airports) is a solution to increase the capacity; however, it may not always be the best solution due to its investment costs. The alternative is to optimize the use of current infrastructures, especially runways, which are recognized to be one of the main bottlenecks of the whole Air Traffic Management (ATM) system.

Several research works focus on in the optimization of runway sequences, which correspond in the literature to the following problems:

- The Aircraft Landing Problem (ALP) aims at sequencing arriving aircraft on available runways and scheduling their landing times taking into consideration several operational constraints.
- The Aircraft Take-off Problem (ATP) consists in scheduling take-off slots to departing aircraft.

When scheduling both landings and take-offs, the problem is then called the Aircraft Scheduling (or Sequencing) Problem (ASP).

Several methods are proposed in the literature for the three above-mentioned problems, and can be classified in two main categories:

- Exact approaches, mainly Mixed Integer Programming (MIP) and Dynamic Programming (DP). MIP-based approaches formulate the problem as a MIP model, and solve it using optimization solvers such as CPLEX [2], [3], [4], [5] or GuRoBi [5]. DP approaches usually model the problem as a modified shortest path problem, and solve it using DP techniques [6], [2], [7]. Exact approaches provide optimal solutions, but they may be inefficient to solve large-scale instances in reasonable computation time.
- Metaheuristics. Three types of metaheuristics are proposed in the literature for the ALP: *evolutionary-based algorithms* such as genetic algorithms [8], [9], *swarm-based optimizers* such as ant colony optimization [10], [11], and *local-search based* algorithms such as simulated annealing [12], [13], [14], [15].

In this work, we are interested in scheduling aircraft landings (ALP) at the runway threshold. Each aircraft has a target landing time and a landing time window, expressed as an earliest and a latest acceptable landing times based on fuel considerations. Deviations from the target times incur a cost that depends on the type of each aircraft, and the aim is to minimize the total deviations from target times.

We first review an exact solution approach based on Mixed-Integer Linear Programming (MILP), that takes into consideration safety constraints by means of separation constraints between aircraft imposed at the runway threshold. We then show the limits of such an approach to solve large realistic instances. Therefore, we recourse to a heuristic approach originally introduced in [16], based on Optimistic Planning (OP) algorithm [17], [18]. This approach models the ALP as an environment of *states*, *actions*, *transitions* and *costs*, then explores the resulting search tree so as to identify a near-

optimal sequence of actions within a limited time budget. In the original paper [16], it is the "First-Come First-Served" (FCFS) rule that is used in the computation of the costs to estimate the cheapest path (sequence of actions) cost from a given state. The main contribution of the present work is the investigation of different Machine Learning (ML) models which can quickly and accurately estimate the cost of the cheapest sequence. Using ML models in our heuristic search helps identifying a near-optimal branch more efficiently: this ameliorates previous results in terms of both percentage improvements (with respect to the traditional FCFS solution) and the gap to best-known solutions.

The remainder of this paper is organized as follows. In Section II we describe the ALP, highlighting operational constraints, and we revisit a Mixed-Integer Linear Programming (MILP) formulation that incorporates these constraints. In Section III we explain how machine learning models can be used in our heuristic approach to guide the search for near-optimal solutions. Section IV presents computational results that show the benefit of using ML models in our heuristic search. Finally, Section V summarizes the contributions of this work and suggests some future tracks of research.

II. Problem description and formulation

This section introduces some basic concepts related to the ALP, and highlights the most important constraints that must be taken into consideration when addressing such problems. The MILP formulation introduced in [16], which incorporates these constraints is then recalled. It will be used to evaluate the performance of the ML heuristic introduced in this paper.

A. Basic concepts

Given a set of aircraft near the terminal maneuvering area (TMA) of an airport, the ALP consists in mapping each aircraft to a landing time such that a given criterion is optimized while operational constraints are satisfied. When the airport has more than one runway, a decision with respect to the landing runway has to be made by controllers; the runway assignment depends on several factors such as the airport configuration and the direction of arriving aircraft [19].

A straightforward solution usually implemented by air traffic controllers is the First-Come First-Served rule, where aircraft land according to the order of the estimated times of arrival at the runway: controllers only ensure the minimum separation requirements. However, this FCFS sequence is rarely optimal in terms of runway throughput, especially in congested airports [6]. This motivates the development methods to compute optimal sequences satisfying several operational constraints such as minimum separation, authorized time windows, and constrained-position shifting.

- The minimum separation constraint guarantees that no aircraft is affected by the wake-vortex turbulence generated by a leading aircraft, especially during take-offs and landings. The International Civil Aviation Organization (ICAO) classifies aircraft in three main categories, namely Heavy (H), Medium (M) and Light (L), and the separation requirements are defined depending on the category of both the leading and the following aircraft. Separation requirements between landings are presented in Table I.

TABLE I: Final-approach separation matrix (in seconds) according to ICAO's basic wake-turbulence categories (source [6])

		Following aircraft		
		H	M	L
Leading aircraft	H	96	157	196
	M	60	69	131
	L	60	69	82

- Time-window constraints are defined by an earliest and a latest possible landing times, based on fuel availability or possible speed-ups.
- The Constrained-Position Shifting (CPS) constraint ensures that an aircraft is not deviated from its initial position in the FCFS order by more than a given number of positions called maximum position shifting and denoted by m, which is usually small: $m = 3$ or 4 [6].

The next section presents MILP formulation that incorporates these operational constraints.

B. Mathematical formulation

Runway assignment and aircraft scheduling is formulated as an MILP model, that assigns a landing time to each aircraft, while satisfying the above-mentioned constraints.

Input data

Let $\mathcal{K} = \{1, 2, ..., R\}$ be an index set of available runways, and $\mathcal{A} = \{1, 2, ..., N\}$ be an index set of arriving aircraft. Without loss of generality, we assume that each aircraft index $i \in \mathcal{A}$ represents its position in the FCFS sequence. Then, for each flight $i \in \mathcal{A}$, the given input data of the ALP is:

- T_i : Target landing time
- $[E_i, L_i]$: Landing time window ($L_i > E_i$)
- S_{ij} : minimum separation time (≥ 0) between aircraft i and j, where i lands before j
- c_i^- : Penalty cost (≥ 0) per time-unit for landing *before* the target time T_i
- c_i^+ : Penalty cost (≥ 0) per time-unit for landing *after* the target time T_i

Decision variables

Two types of optimization variables are involved in our model (Table II): binary variables for runway assignment and sequencing, and continuous variables for assigning landing times at the runway threshold.

Formulation

The objective function aims at minimizing the total deviation cost from target times (T_i). The complete MILP formulation is given by (1)–(12).

TABLE II: OPTIMIZATION VARIABLES

Type	Variable	
Binary	$a_{ik} =$	$\begin{cases} 1 & \text{if aircraft } i \text{ is assigned to runway } k, \\ 0 & \text{otherwise,} \end{cases}$
	$\delta_{ijk} =$	$\begin{cases} 1 \text{ if aircraft } i \text{ and } j \text{ are assigned to} \\ \text{runway } k, \text{ and } i \text{ lands before } j, \\ 0 \text{ otherwise,} \end{cases}$
	$y_{ij} =$	$\begin{cases} 1 \text{ if aircraft } i \text{ lands before aircraft } j, \\ 0 \text{ otherwise,} \end{cases}$
Continuous	t_i	landing time
	t_i^-, t_i^+	deviations from the target landing time T_i (before and after T_i, respectively)

$$\min_{a,\delta,y,t} \sum_{i\in\mathcal{A}} c_i^- t_i^- + c_i^+ t_i^+ \tag{1}$$

$$t_i = T_i - t_i^- + t_i^+ \qquad i \in \mathcal{A} \tag{2}$$

$$E_i \le t_i \le L_i \qquad i \in \mathcal{A} \tag{3}$$

$$y_{ij} + y_{ji} = 1 \qquad i,j \in \mathcal{A} :\ i < j \tag{4}$$

$$\sum_{k\in\mathcal{K}} a_{ik} = 1 \qquad i \in \mathcal{A} \tag{5}$$

$$\sum_{k\in\mathcal{K}} \delta_{ijk} + \delta_{jik} \le 1 \qquad i,j \in \mathcal{A} :\ i < j \tag{6}$$

$$\delta_{ijk} + \delta_{jik} \ge a_{ik} + a_{jk} - 1 \quad i,j \in \mathcal{A} : i < j, k \in \mathcal{K} \tag{7}$$

$$2(\delta_{ijk} + \delta_{jik}) \le a_{ik} + a_{jk} \quad i,j \in \mathcal{A} : i < j, k \in \mathcal{K} \tag{8}$$

$$t_j \ge t_i - M_1(1 - y_{ij}) \qquad i,j \in \mathcal{A} : i \ne j \tag{9}$$

$$t_j \ge t_i + S_{ij} - M(1 - \delta_{ijk}) \quad i,j \in \mathcal{A} : i \ne j \tag{10}$$

$$i - m \le N - \sum_{j\in\mathcal{A}, j\ne i} y_{ij} \le i + m \qquad i \in \mathcal{A} \tag{11}$$

$$\delta_{ijk}, y_{ij}, a_{ik} \in \{0,1\} \qquad i,j \in \mathcal{A} : i \ne j,\ k \in \mathcal{K} \tag{12}$$

Constraints (2) are introduced to linearize the piecewise-linear objective function. Constraints (3) represent the time window restrictions. Constraints (4) enforce the order precedence relationship between flights i and j at landing; constraints (5) ensure that an aircraft is assigned to exactly one runway. Constraints (6) enforce the order precedence relationship between flights landing on the same runway. Constraints (7) and (8) translate the logical relationship between δ_{ijk} and a_{ik}. Constraints (9) relate precedence relationships between landings and landing time-order. Constraints (10) ensure the separation requirements between aircraft landing at a same runway. Constraints (11) impose the CPS constraint, and constraints (12) stipulate the binary restrictions of our discrete variables.

Several techniques can be used to improve this MILP formulation, such as the variable-fixing strategies explained in [5], which consists in fixing the order of some aircraft that belong to a same *class* (*e.g.* a wake-vortex category), based on their time windows. *Valid inequalities* can also be added to strengthen (improve) MILP formulations. We refer the interested readers to [20] for details on these techniques.

III. SOLUTION APPROACHES

After reviewing the exact solution approach used to solve the ALP, this section explains how ML models are trained and used in our heuristic search, so as to guide efficiently the search for near-optimal solutions.

We compute exact solution by solving the MILP formulation (1)–(12) with IBM CPLEX 12.8. Results of implementing this approach are reported in Section IV-A.

Given the complexity of the problem (this problem is NP hard), exact approaches may fail to solve large instances within reasonable computing times. For this reason, we recourse to heuristic approaches.

We originally introduced a heuristic approach in [16], in which we proposed a new model for the ALP, inspired from Markov Decision Process (MDP). The ALP is modeled as an environment defined by *states*, *transitions*, *actions*, and *costs* where:

- a ***state*** is a partition of the set of aircraft $\mathcal{A}$ into two subsets: aircraft having already landed, denoted $\bar{I}$, and aircraft which have not landed yet, denoted I;
- an ***action*** is an aircraft index $i \in I$ that we decide to land;
- a ***transition*** is an update of the two sets $\bar{I}$ and I, which generates a unique next state, denoted I', where $I' = I \setminus \{i\}$, and $\bar{I}' = \bar{I} \cup \{i\}$ (aircraft i landed);
- the estimated ***cost*** c of the the new state is defined by:

$$c(\bar{I}, I) = f(\bar{I}') + g(I'),$$

where $f(\bar{I}')$ is the delay cost of the (landed) sequence $\bar{I}'$, and $g(I')$ is an estimate of the lowest cost among all possible ordering of I' that satisfy the CPS constraints.

This model results in a tree, in which nodes are labeled by states and arcs are labeled by actions and costs. In [16] we propose a tree-search method, based on Optimistic Planning (OP) algorithm [17], [18]. This method starts from the initial state where the set $\bar{I}$ is empty, and $I = \mathcal{A}$. At each iteration, it seeks which aircraft to land based on an optimistic evaluation of the state cost (c). This evaluation involves two functions f and g defined above. The first one computes the delay of the landed sequence, *i.e.*, the cost of the path from the initial state to the current state; the second function estimates the lowest cost among all possible sequences of the subset of aircraft that remain to land. The algorithm keeps exploring the tree guided by the costs c, until a stopping criteria is met: all aircraft are landed or a time limit is reached.

In [16], we use the FCFS rule as the estimation g. In the present work, we propose training different ML models on a large number of previously solved instances of the ALP, so as to obtain an estimate of the lowest cost of the subset of aircraft that remain to land, which we shall use as the estimation g. A similar idea is used in [21] to predict the optimal production of an offshore wind park.

We describe below how training and testing sets are generated, and we detail the inputs and outputs of these ML models.

A. Generation of data

Training and testing sets are generated from four data sets of [22]: `alp_7_11.csv`, `alp_11_15.csv`, `alp_15_19.csv`, and `alp_19_23.csv`. The first file (`alp_7_11.csv`) is used to generate the testing set; the remaining files are used to generate the training set.

B. Definition of features

The optimal cost of an instance depends on several factors such as the congestion of the instance, and the mixture of the different types of aircraft. To be able to use ML models, it is important to select significant features, in order to predict accurately optimal costs. In Fig. 1, the relation between some characteristics of an instance and the optimal (lowest) cost is plotted.

First, we observe that the optimal cost increases with the number of heavy aircraft (Fig. 1a). A possible explanation is related to the fact that deviating Heavy aircraft cause larger costs than deviating Light or Medium aircraft.

Moreover, the optimal cost depends on the number of conflicts (*i.e.*, the number of aircraft pairs that violate the separation requirements presented in Table I) in an instance (Fig. 1b). Indeed, as the number of conflicts increases, more aircraft are to be deviated from their target times, which results in an increased cost. On the other hand, the optimal cost decreases when the ratio between instance length and the total number of aircraft increases (Fig. 1c).

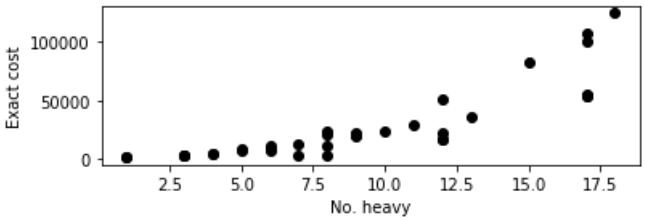
The optimal cost is also correlated to the FCFS-sequence cost (Fig. 1d). Indeed, if an instance is congested and have a large number of Heavy aircraft, both the FCFS sequence and the optimal solutions will have high cost values.

Based on these observations, we select the following features to be passed as inputs to the ML models:

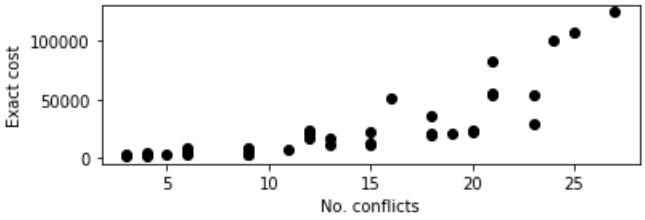
- the total number of aircraft of type Heavy;
- the total number of conflicts in the instance;
- the ratio between the length of the instance (in seconds) and the total number of aircraft: this captures how congested the instance is;
- the cost of the FCFS sequence.

C. Machine learning models

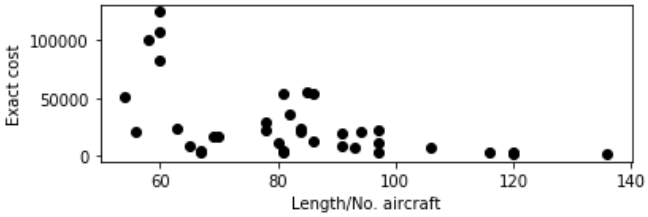
We use Linear Regression (LR) as a baseline model and two other machine learning models, namely Neural Networks (NN) and Support Vector Regression (SVR). The four attributes defined above are given as an input vector, denoted x, to these models. The output is a lowest-cost estimation denoted $\hat{g}$, which is modeled by a function denoted f, that depends on some unknown weights w (to be determined), such that $\hat{g} = f(x, w)$ approximate g as much as possible in the least-square sense. In other words, these weights are learned by the ML models during the training phase, by minimizing the Root Mean Squared Error (RMSE) over the training set: $\{(x_1, g_1), .., (x_{n_t}, g_{n_t})\}$, where x_i is an attribute vector and g_i is the corresponding optimal cost to be estimated, and which is obtained using the MILP model (1)–(12) and CPLEX.



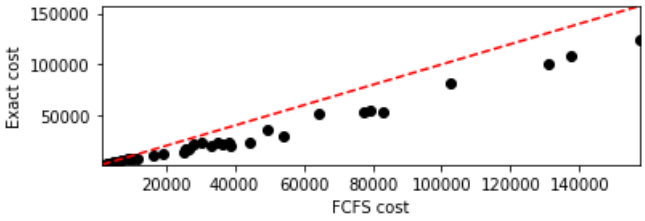
(a) Optimal vs. number of heavy aircraft



(b) Optimal vs. number of conflicts in the instance



(c) Optimal cost vs. ratio length of the instance (seconds) / total number of aircraft



(d) Optimal cost vs. the FCFS-sequence cost

Fig. 1: A visualisation of the relation between the selected features and the exact (optimal) cost. The red dashed line is the line of identity.

The three above-mentioned models are implemented in the *Scikit-learn* Python ML library, and their hyperparameters are obtained through a simple a grid search.

IV. Results and Discussion

In this section we first report computational results obtained when solving MILP formulation (1)–(12). Then, we compare the results obtained with the heuristic search presented in the previous section using:

1) FCFS rule (previous contribution [16])
2) LR, NN, and SVR

as an estimate g of the optimal cost (of the subset of aircraft that remain to land).

TABLE III: MILP APPROACH PERFORMANCE ON THE TEST DATA, FOR MAXIMUM POSITION SHIFTING VALUES $m = 3$ AND $m = 4$

$\|\mathcal{A}\|$	3-CPS		4-CPS	
	% improv (MILP)	CPU (s)	% improv (MILP)	CPU (s)
20	41.81	4.55	50.47	10
22	37.8	31.73	48.4	70.76
24	33.56	830.23	43.73	1398.4
26	31.18	1800	40.91	1800
28	32.43	1800	41.04	1800
30	32.43	1800	40.73	1800
32	32.49	1800	40.52	1800
34	32.68	1800	40.71	1800
36	33.72	1800	43.38	1800
38	34.11	1800	43.33	1800
40	33.71	1800	43.07	1800
min	31.18	4.55	40.52	10
max	41.81	1800	50.47	1800
average	34.17	1387.86	43.29	1443.56

All experiments are run on a personal computer under GNU/Linux operating system, processor Intel(R) Core(TM) i7-4700M with 8 GB of RAM. The MILP model is implemented in DoCplex and solved with CPLEX 12.8.

All instances used in this section are generated from the test data set: `alp_7_11.csv`. Recall that the LR, NN and SVR models were trained on the data sets: `alp_11_15.csv`, `alp_15_19.csv`, and `alp_19_23.csv`.

A. MILP approach

Table III exhibits the performance of the MILP approach on test instances of size $|\mathcal{A}| = 20, 22, ..., 40$, obtained by simply considering the first $|\mathcal{A}|$ lines of the test data set. Columns m-CPS represent results for two values of the maximum position shifting: $m = 3$ and $m = 4$. Results are reported in terms of percentage of improvement with respect to the traditional FCFS solution sequence and of computing time in seconds, with a time limit of 30 minutes (1,800 seconds) under CPLEX.

The percentage of improvement obtained by a method M (here M = MILP) is computed with respect to the FCFS solution as:

$$\%\text{improv (M)} = \frac{C_{\text{FCFS}} - C_{\text{M}}}{C_{\text{FCFS}}} \times 100, \qquad (13)$$

where C_{FCFS} and C_{M} are the costs of the traditional FCFS sequence and of method M, respectively.

Fig. 2 illustrates the growth of the computing time with respect to the size of the instance, for different values of the maximum position shift ($m = 2, 3, 4, 5, 6$). The growth is as expected, exponential due to the complexity of the ALP; the saturation effect observed is due to the time limit (1,800 s).

To summarize, prohibitive computational times required by the exact MILP approach, even for small values of the maximum position shifting, make it unsuited to the dynamic nature of the practical ALP.

B. Heuristic approach

1) Performance of ML models to estimate the cost: Before reporting the results of our heuristic search method, using FCFS, LR, NN and SVR as estimation heuristics, we first visualize results of cost predictions with the baseline model LR and with the two ML models (NN and SVR). Tests are performed on different instances from the test data set, ranging from 5 to 40 aircraft, obtained again by considering the first $|\mathcal{A}|$ lines of this data set.

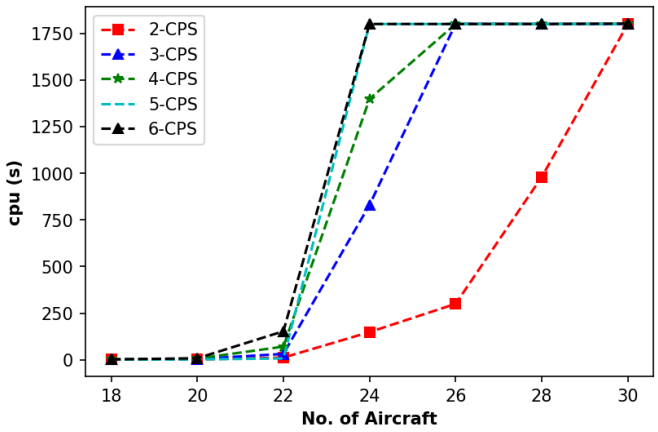


Fig. 2: Computating time of the MILP approach for different maximum position shifting values

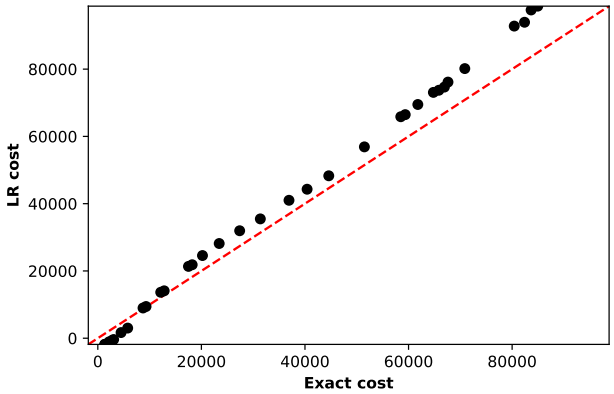
Fig. 3 compares the best-known (exact) cost with the predicted costs from the baseline model (Fig. 3a), neural network (Fig. 3b), and support vector Regression (Fig. 3c). Each point corresponds to a test instance. The red dashed line represents the ideal identity-line.

Fig. 4 displays an alternative visualisation. It shows the best-known cost values (red dashed line), the FCFS-sequence cost (black), and the predicted optimal cost using the LR, NN and SVR models. These plots reveal clearly that the three models (LR, NN, SVR) are better estimates of the best-known cost than the traditional FCFS rule.

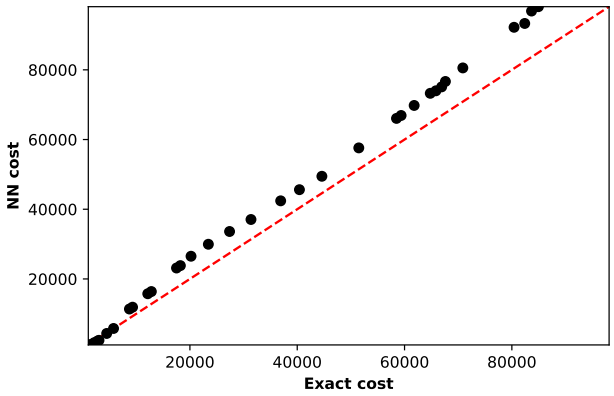
2) Performance of the heuristic with ML models: We report the results of the OP algorithm described in Section III on the same instances used in Table III, involving a single runway, and for a maximum position shifting $m = 3$, and imposing this time a more realistic limited time budget of 5 seconds.

The percentage improvements defined by (13) are reported in Table IV. The first column presents the size of the instance, the second column reports the percentage of improvement of the OP algorithm using the FCFS as estimation heuristic, and the third column shows the percentage improvement using LR, NN, and SVR as estimation heuristics in the OP algorithm. Best improvements are in bold. One observes that, for some instances, the OP algorithm (with FCFS, LR, NN, and SVR as estimation heuristics) significantly ameliorates the traditional FCFS sequence. In particular, SVR gives slightly better improvements on average (27.45%).

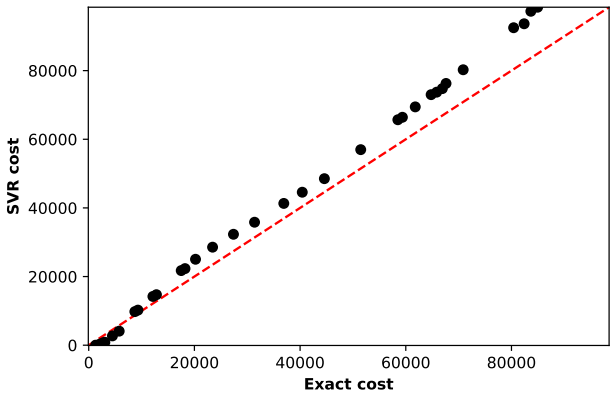
To further assess the quality of the solutions obtained with our OP algorithm using the FCFS, LR, NN and SVR as estimation heuristics, we report results in terms of the gap (in percent) between these solutions and the best-known reference solutions. These best-known solutions are obtained via the



(a) Linear regression



(b) Neural network



(c) Support vector regression

Fig. 3: Comparing the predicted optimal cost (y-axis) with the exact cost (x-axis) using linear regression (Fig. 3a), neural network (Fig. 3b), and support vector regression (Fig. 3c)

MILP approach within a time limit of 30 minutes under CPLEX. The gap is calculated as follows:

$$\%\text{Gap (M)} = \frac{C_{\text{M}} - C_{\text{best}}}{C_{\text{best}}} \times 100, \qquad (14)$$

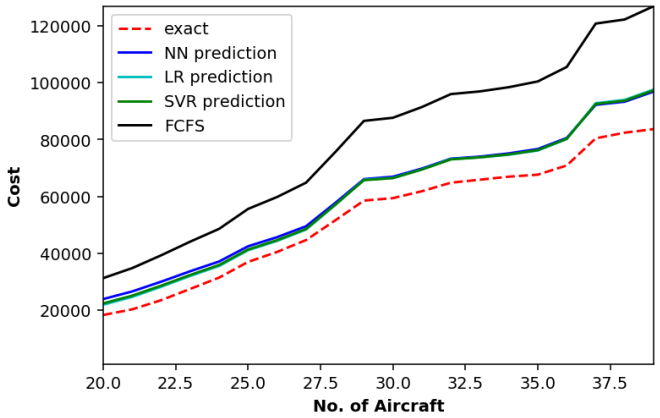


Fig. 4: Comparing the predicted optimal costs (by linear regression and two ML models) with the exact cost. The traditional FCFS-sequence cost is also plotted (black)

where C_{M} and C_{best} are the cost of the solution obtained with method M, and of the best-known solution respectively.

One observes in Table V that, within the time limit of 5 seconds, the gap is always the best with the LR, NN and SVR models, as they involve more accurate cost estimations, which help the search to identify rapidly the most promising nodes.

The computational experiments therefore show that, using LR or ML models as the heuristic estimation function g in an OP algorithm outperforms the simple estimation provided by the FCFS rule, introduced in [16]: LR and ML estimates yield better results in terms of solution quality.

V. Conclusion

The aircraft landing problem consists in sequencing and scheduling arriving aircraft on runways, taking into consideration several operational constraints. It represents an ongoing challenge for both researchers and air traffic controllers, due to its dynamic nature, and the different operational constraints that must be considered.

TABLE IV: Percentage improvement of the different methods (OP algorithm together with an estimate obtained via FCFS, LR, NN or SVR) for a time limit of 5 seconds. Percentages are with respect to using the simple FCFS rule

$\lvert\mathcal{A}\rvert$	**% improv** (OP-FCFS)	(OP-LR)	**% improv** (OP-NN)	(OP-SVR)
20	**38.91**	**38.91**	**38.91**	**38.91**
22	**34.62**	**34.62**	34.30	34.03
24	29.83	29.76	**30.37**	29.76
26	27.82	27.72	**28.45**	28.29
28	27.82	27.72	**28.92**	27.67
30	22.59	24.94	22.93	**25.05**
32	22.13	21.49	22.57	**24.88**
34	19.79	23.15	**24.51**	23.41
36	19.15	22.56	**23.12**	22.75
38	18.96	22.77	23.23	**25.10**
40	18.59	**24.89**	22.01	22.11
min	18.59	21.49	22.01	**22.11**
max	**38.91**	**38.91**	**38.91**	**38.91**
average	25.25	26.90	27.17	**27.45**

TABLE V: OPTIMALITY GAPS YIELDED BY THE DIFFERENT ESTIMATES, FOR A TIME LIMIT OF 5 SECONDS

$\lvert\mathcal{A}\rvert$	**% Gap** (OP-FCFS)	(OP-LR)	**% Gap** (OP-NN)	(OP-SVR)
20	**4.98**	**4.98**	**4.98**	**4.98**
22	**5.11**	**5.11**	6.06	6.06
24	5.61	5.72	**4.80**	5.72
26	4.88	5.03	**3.97**	4.20
28	10.45	10.73	**5.19**	7.04
30	14.56	11.08	14.06	**10.92**
32	15.35	16.29	14.69	**11.27**
34	19.15	14.16	**12.14**	13.77
36	21.98	16.84	**15.99**	16.55
38	22.99	17.21	16.51	**13.67**
40	22.81	**13.31**	17.65	17.50
min	4.88	4.98	**3.97**	4.20
max	22.99	**17.21**	17.65	17.50
average	13.44	10.95	10.54	**10.15**

In this work, we proposed an optimistic planning algorithm relying on machine learning heuristics to address the problem, instead of the simple firs-come first-served rule, proposed in [16]. These heuristics are: a baseline linear regression, and two different machine learning models trained on a large number of optimized solutions, provided by a mixed-integer linear programming method.

Computational experiments show that using ML models in the heuristic search of the optimistic planning algorithm does not only ameliorate results in terms of percentage improvement, but also reduces the optimality gap within a computation time that is compatible with on-line operations (5 seconds).

Future tracks of research include the expansion to the multiple-runway case, and taking into consideration uncertainty on the arrival times.

ACKNOWLEDGMENT

The authors would like to thank Serge Roux, from ENAC, for his assistance with technical support.

REFERENCES

[1] The International Air Transport Association (IATA). Accessed: June 18, 2020. [Online]. Available: https://www.iata.org/pressroom/pr/Pages/2017-10-24-01.aspx

[2] D. Briskorn and R. Stolletz, "Aircraft landing problems with aircraft classes," *Journal of Scheduling*, vol. 17, no. 1, pp. 31–45, 2014.

[3] A. Ghoniem and F. Farhadi, "A column generation approach for aircraft sequencing problems: A computational study," *Journal of the Operational Research Society*, vol. 66, no. 10, pp. 1717–1729, 2015.

[7] A. Lieder, D. Briskorn, and R. Stolletz, "A dynamic programming approach for the aircraft landing problem with aircraft classes," *European Journal of Operational Research*, vol. 243, no. 1, pp. 61–69, 2015.

[4] F. Furini, M. P. Kidd, C. A. Persiani, and P. Toth, "Improved rolling horizon approaches to the aircraft sequencing problem," *Journal of Scheduling*, vol. 18, no. 5, pp. 435–447, 2015.

[5] R. Prakash, R. Piplani, and J. Desai, "An optimal data-splitting algorithm for aircraft scheduling on a single runway to maximize throughput," *Transportation Research Part C: Emerging Technologies*, vol. 95, pp. 570–581, 2018.

[6] H. Balakrishnan and B. Chandran, "Scheduling aircraft landings under constrained position shifting," in *AIAA guidance, navigation, and control conference and exhibit*, 2006.

[8] G. Bencheikh, J. Boukachour, A. E. H. Alaoui, and F. E. Khoukhi, "Hybrid method for aircraft landing scheduling based on a job shop formulation," *International Journal of Computer Science and Network Security*, no. 8, pp. 78–88, 2009.

[9] X. B. Hu and E. A. D. Paolo, "A ripple-spreading genetic algorithm for the aircraft sequencing problem," *Evolutionary Computation*, no. 19, pp. 77–106, 2011.

[10] G. Bencheikh, J. Boukachour, and A. E. H. Alaoui, "Improved ant colony algorithm to solve the aircraft landing problem," *International Journal of Computer Theory and Engineering*, vol. 3, no. 2, pp. 224–233, 2011.

[11] Y. Jiang, Z. Xu, X. Xu, Z. Liao, and Y. Luo, "A schedule optimization model on multirunway based on ant colony algorithm," *Mathematical Problems in Engineering*, vol. 2014, 11 pages, 2014.

[12] A. Salehipour, M. Modarres, and L. M. Naeni, "An efficient hybrid metaheuristic for aircraft landing problem," *Computers & Operations Research*, vol. 40, no. 1, pp. 207–213, 2013.

[13] G. Hancerliogullari, G. Rabadi, A. H. Al-Salem, and M. Kharbeche, "Greedy algorithms and metaheuristics for a multiple runway combined arrival-departure aircraft sequencing problem," *Journal of Air Transport Management*, no. 32, pp. 39–48, 2013.

[14] A. Rodriguez-Diaz, B. Adenso-Diaz, and P. L. Gonzalez-Torre, "Minimizing deviation from scheduled times in a single mixed-operation runway," *Computers & Operations Research*, no. 78, pp. 193–202, 2017.

[15] J. Ma, D. Delahaye, M. Sbihi, P. Scala, and M. A. M. Mota, "Integrated optimization of terminal maneuvering area and airport at the macroscopic level," *Transportation Research Part C: Emerging Technologies*, vol. 98, pp. 338–357, 2019.

[16] S. Ikli, C. Mancel, M. Mongeau, X. Olive, and E. Rachelson, "An optimistic planning approach for the aircraft landing problem," in *Proceedings of ENRI International Workshop on ATM/CNS, Tokyo, Japan*, 2019.

[17] J. F. Hren and R. Munos, "Optimistic planning of deterministic systems," *European Workshop on Reinforcement Learning. Springer*, pp. 151–164, 2008.

[18] R. Munos, "From bandits to Monte-Carlo tree search: The optimistic principle applied to optimization and planning," *Foundations and Trends in Machine Learning*, vol. 7, no. 1, pp. 1–129, 2014.

[19] J. A. Bennell, M. Mesgarpour, and C. N. Potts, "Airport runway scheduling," *4OR*, vol. 9, no. 2, pp. 115–135, 2011.

[20] J. E. Beasley, M. Krishnamoorthy, Y. M. Sharaiha, and D. Abramson, "Scheduling aircraft landingsâĂŤThe static case," *Transportation Science*, vol. 34, no. 2, pp. 180–197, 2000.

[21] M. Fischetti and M. Fraccaro, "Machine learning meets mathematical optimization to predict the optimal production of offshore wind parks," *Computers & Operations Research*, vol. 106, pp. 289–297, 2019.

[22] S. Ikli. The aircraft landing problem instances. Accessed: June 18, 2020. [Online]. Available: http://data.recherche.enac.fr/ikli-alp/