Aircraft Landing Problem

FEUP – MDSE

2022/2023 1st semester

Analytical Decision Support Systems - Practical Assignment

Group: B3

Farzam Salimi (up201007922)

Luís Henriques (up202204386)

Rojan Aslani (up202204382)

Abstract

This paper aimed to design and develop two different optimization models for the aircraft landing problem (ALP) with different approaches, mixed-integer programming (MIP) and constraint programming (CP), and apply these models to the 13 airland datasets. Both models aimed to minimize the penalty costs of landing before or after the target landing time, while satisfying the separation times between each aircraft and time windows for each plane to land. IBM ILOG CPLEX OPTIMIZATION STUDIO was used to implement the models to the sets of data. The results were validated by comparison to previous studies made with these sets of data, in terms of results and running time. The best MIP and CP models were selected to extend the problem to multiple runways. For small datasets, from airland1 to airland8, all models were able to compute the optimal solutions for one runway and multiple runways. For large datasets, the MIP model was able to do a breakthrough in the state of the art, giving optimal solutions for some of the problems with multiple runways in less than 30 minutes, and some of them even in less than a minute. However, none of the models on the big datasets was able to find the optimal solution with one runway in this time span. The improvements made may be partially due to the fact that the hardware and software used has improved significantly over the years, which can impact the model performance.

Keywords: Aircraft Landing Problem, Mixed-integer programming, Constraint programming, Multiple runways.

Introduction

Runways are the primary bottleneck at commercial service airports throughout the world. Efficient runway usage is essential for consistently ensuring safe and on-time passenger travel [1]. Given a set of planes, the task is to assign a runway (when there is more than 1 runway in the airport) and a sequence of landing time for all planes. Each plane must land within its predefined time window, ideally, closest to the target landing time. Moreover, a safety separation time must be maintained between any pair of planes to ensure sage and adequate time between consecutive landings in the same runway. A cost is charged when a plane lands after or before its target landing time. The objective in this problem is to minimize the total cost of deviation from target landing times.

This problem, referred to in the literature as the aircraft landing problem (ALP), and has been studied extensively [2, 3, 1]. The ALP focuses on in-bound flights (landing), but this analysis can be extended to cover both in- and out-bound flights (departure).

This problem can be solved optimally with several algorithms. The objective of this project is to focus on solving the ALP by mixed-integer programming (MIP), and constraint programming (CP), to optimize the penalty cost (minimize).

MIP is a type of linear mathematical optimization problem where some of the variables are constrained to integer values. CP is a type of artificial intelligence technology used to solve complex problems by breaking them down into smaller parts and creating constraints to help guide the search for solutions. Both MIP and CP can be used for planning, scheduling, resource allocation, optimization, and other operations research tasks [4]. In this project, 2 MIP and 1 CP formulations were implemented and tested. Furthermore, the results were compared to the literature in terms of computational cost for small datasets (Airland 1 to 8), and in terms of the best-found solution for the big datasets (Airland 9 to 13) within a limited timeframe (30 minutes).

State of the Art

In [3] the authors present a mixed integer zero-one formulation of the problem as well as a heuristic model. For smaller files the MIP formulation is successfully used. On the other hand, the bigger files are computationally too expensive and optimal solution is not achieved. Hence, they proposed a heuristic model to solve the bigger data files (higher number of planes). The problem is solved optimally with a linear programming-based tree search algorithm.

In [5] the authors present a comparison of several models and heuristics. They study MIP and an integer linear program using time discretization introduced by [3]. They also compare two heuristic algorithms.

Another exact method used for single and multiple static ALP is described in [6]. The proposed exact algorithm receives a feasible landing sequence on single or multiple runways and solves the landing time assignment part of the problem only. In [7] the authors used constraint programming to solve the ALP. The model is well suited for small instances. However, the quality of the provided solutions is not as good as for larger instances.

Materials

To test the models, 13 datafiles were acquired from OR-Library website[1]. The database was created and initially used in [3] by J.E. Beasley. The files contain data about a set of planes in a certain airport that need to land, with the least penalty cost, while keeping the separation time between every plane in mind. The text files have data in the following format:

- ✓ ***number*** *of planes*
- ✓ ***freeze*** *time*

For each plane:

- ✓ ***appearance*** *time*
- ✓ ***earliest*** *landing time*
- ✓ ***target*** *landing time*
- ✓ ***latest*** *landing time*

[1] people.brunel.ac.uk/~mastjjb/jeb/info.html

- ✓ *penalty **cost** per unit of time for **landing before** target landing time*
- ✓ *penalty **cost** per unit of time for **landing after** target landing time*
- ✓ ***separation time** between any two pairs of planes:* the minimum amount of time each plane must wait before landing if scheduled to land directly after another plane on the same runway.

where time is represented as a discrete set of time points. It seems that landing later is penalized heavier than landing earlier (Figure *1*), but that might not always be the case. The number of aircrafts ranges from 10 to 500. While the entire dataset is of size 1.3 MB, the largest datafile has a size of 800 KB approximately (airland13).

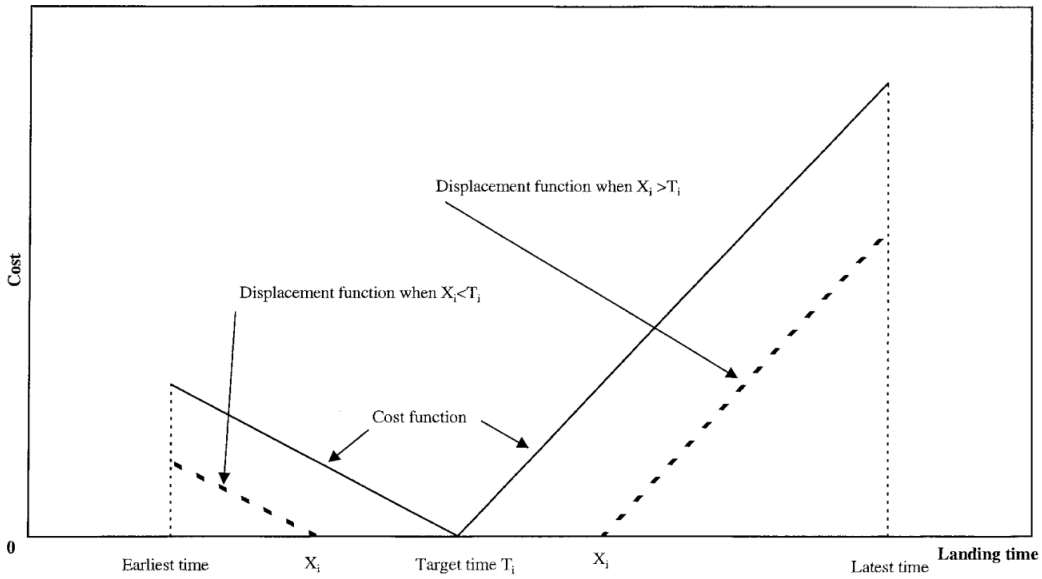

Figure 1. Cost function and displacement function. Xi represents the chosen landing time and Ti represents the target landing time for plane i. Figure obtained from [2].

Implementation of the models was done in IBM ILOG CPLEX Optimization Studio[2]. The formulation was written in a model (.MOD) file. For each Airland file, one data file (.DAT) was made to import the data to CPLEX from Excel, as well as a Run Configuration. The results of the model for each data file were automatically exported to an excel file for further analysis.

Data Preparation

All datafiles were organized and transformed to excel format using Python[3] (see annex files in folder *convertTXTtoXLSX*). This was done for an automatized upload of the data to CPLEX studio, where the MIP and CP models were developed and tested.

The data files created in CPLEX import data from the respective *airlandX.xlsx* file, according to the cell numbers. After the run configuration is completed, the results of the selected variables (decision variables and the objective function) are copied automatically to an excel sheet. For more details on the code see annex file *airland1.mod* in the *Windows//Data Files* folder as an example.

[2] www.ibm.com/products/ilog-cplex-optimization-studio
[3] www.python.org

Regarding the data points themselves, those being the parameters of the models developed, they were already in data types that CPLEX can work with, those being integers and floats. Hence, data transformation was not required.

Mathematical Programming Model

As mentioned earlier, two distinct MIP formulations were programmed and tested. Initially a formulation was proposed by the authors (MODEL1). Next, the formulation proposed by [2] was studied, implemented, and tested (MODEL2004). Both models were initially implemented considering a single runway. Moreover, MODEL1 was also tested with multiple runways.

MIP Design

1- MODEL1

This model was created by developing the simple idea of if the planes can land on their target time, they will impose no cost, and if not, there will be cost. The model was inspired by a combination of the studies in the literature, and in some aspect, they are very close to each other [3, 6, 5].

To create this model initially it was assumed that there is no separation time and as expected, in the absence of the separation time the objective function is equal to zero. The objective function (Equation 1.1) was implemented aiming to minimize cost, with the idea that the plane either lands before or after the target landing time. Each one of those two cases have a different cost associated to it.

In terms of constraint, one of the important assurances that is made is the fact that for each plane at least one of the two main decision variables (Landing *earlier* or *later* than the target landing time) is equal to zero since a plane cannot land both before and after the target time. This assurance has been done by defining a binary variable and utilizing of a big constant, M, to activate only one of two constraints at a time. The other constraints for the base model are related to the threshold of the landing and they will be explained briefly.

After all these considerations and tests to the base model on the different data sets, to assure it works properly, the separation time constraints were added to the model. This addition was inspired by the binary variable used in article [2], since it was a good fit for this model. The idea of the separation time constraint is that first we try to recognize that between two different planes which one is landing first, and the fact that we should assure only one plane land before another one. Adding the separation time constraint (also by using a big constant, M) the model is complete (Equation 2 - 9).

With that in mind, there is one little detail in the last two separation constraints (Equation 8 and 9), and that has to do with the index order of the separation times. The separation time matrix is not symmetric, meaning that plane j must land at least a certain time after plane i, however plane i may or may not be obligated to land in a longer or shorter time period after plane j. So, with that in mind, independent of plane i landing before or after plane j, the separation time considered at a time should be the same. However, that is not the case in the formulation below. That index swap was done because the solver (CPLEX) gave some constraint conflicts for some data files, which was solved by the addition of this detail. A different way found to remove those conflicts was removing constraint in Equation 8, but the final model took the index swap approach. The reason why this works is because the search tree will consider all possible constraints, therefore keeping the context intact in the process.

Parameters:

p: $number\ of\ planes$

elt_i: $earliest\ landing\ time\ of\ plane\ i$

tlt_i: $scheduled\ landing\ time\ of\ plane\ i$

llt_i: $latest\ landing\ time\ of\ plane\ i$

lbt_i: $penalty\ cost\ per\ unit\ of\ time\ for\ landing\ before\ target\ landing\ time\ for\ plane\ i$

lat_i: $penalty\ cost\ per\ unit\ of\ time\ for\ landing\ after\ target\ landing\ time\ for\ plane\ i$

st_{ij}: $separation\ time\ between\ plane\ i\ and\ j$

Decision variables:

a_i: $the\ amount\ of\ time\ unit\ the\ aircraft\ land\ after\ the\ scheduled\ time\ (delay)$

b_i: $the\ amount\ of\ time\ unit\ the\ aircraft\ land\ before\ the\ scheduled\ time\ (delay)$

x_i: $landing\ time\ for\ plane\ i$

$$\gamma_i = \begin{cases} 1, if\ plane\ i\ lands\ before\ the\ target\ landing\ time \\ 0, otherwise \end{cases}$$

$$\delta_{ij} = \begin{cases} 1, if\ plane\ i\ lands\ before\ plane\ j \\ 0, otherwise \end{cases}$$

Objective function:

$$\min Z = \sum_{i=1}^{p} (lat_i \cdot a_i + lbt_i \cdot b_i) \qquad (1.1)$$

Subject to:

$$a_i \leq (1 - \gamma_i) \cdot M, \qquad \forall i \in \{1, \dots, p\} \qquad (2)$$

$$b_i \leq (\gamma_i) \cdot M, \qquad \forall i \in \{1, \dots, p\} \qquad (3)$$

$$x_i \geq elt_i, \qquad \forall i \in \{1, \dots, p\} \qquad (4)$$

$$x_i \leq llt_{i,} \qquad \forall i \in \{1, \dots, p\} \qquad (5)$$

$$x_i = tlt_i + a_i - b_i, \qquad \forall i \in \{1, \dots, p\} \qquad (6)$$

$$\delta_{ij} + \delta_{ji} = 1, \qquad \forall i, j \in \{1, \dots, p\},\ i \neq j \qquad (7)$$

$$x_i - x_j \geq st_{ji} - \delta_{ij} \cdot M, \qquad \forall i, j \in \{1, \dots, p\},\ i \neq j \qquad (8)$$

$$x_j - x_i \geq st_{ij} - (1 - \delta_{ij}) \cdot M, \qquad \forall i, j \in \{1, \dots, p\},\ i \neq j \qquad (9)$$

2- MODEL2004

As mentioned earlier, MODEL2004 is the model proposed in [2]. This model was implemented to serve as a comparison tool to MODEL1. It was important to implement and test the model by us to make sure that the running conditions were the same for both models (most importantly, the computing power).

In this model, like in MODEL1, the objective function (Equation 1.2) minimizes the total penalty cost across all planes. The first/second maximization terms in the objective function account for aircraft that land before/after target landing time. This cost function is illustrated diagrammatically in Figure *1*.

The constraints are presented in Equation 10 to 13. The first two constraints (Equation 10 and 11) ensure that the landing time is within the time window of earliest and latest landing times. Constraint 3 (Equation 12) ensures that each plane lands before another one. Lastly, constraint 4 (Equation 13) enforces the separation time between aircrafts. Adding a constraint for the landing time to be greater than or equal to the appearance time would be irrelevant because of the earliest landing time constraint (earliest landing time is always greater than appearance time).

Parameters

p : $number\ of\ planes$

elt_i : $earliest\ landing\ time$

tlt_i : $target\ landing\ time$

llt_i : $latest\ landing\ time$

lbt_i : $penalty\ cost\ per\ unit\ of\ time\ for\ landing\ before\ target\ landing\ time\ for\ plane\ i$

lat_i : $penalty\ cost\ per\ unit\ of\ time\ for\ landing\ after\ target\ landing\ time\ for\ plane\ i$

st_{ij} : $separation\ time\ required\ after\ one\ plane\ lands\ before\ another\ plane\ can\ land$

Decision Variables

x_i: $landing\ time\ of\ the\ plane\ i, \quad i = 1, 2, \dots, p$

$$\delta_{ij} = \begin{cases} 1, & if\ plane\ i\ lands\ before\ plane\ j \\ 0, & otherwise \end{cases}, \quad i = 1,2,\dots,p, \quad j = 1,2,\dots,p, \quad j > i$$

Objective function

$$\min Z = \sum_{i=1}^{p} (lat_i * \max[0,\ tlt_i - x_i] + lbt_i * \max[0,\ x_i - tlt_i]) \qquad (1.2)$$

Constraints

$$elt_i \leq x_i, \qquad \forall i \in \{1, \dots, p\} \qquad (10)$$

$$llt_i \geq x_i, \qquad \forall i \in \{1, \dots, p\} \qquad (11)$$

$$\delta_{ij} + \delta_{ji} = 1, \qquad \forall i, j \in \{1, \dots, p\}, \qquad j > i \qquad (12)$$

$$x_j \geq x_i + st_{ij} \cdot \delta_{ij} - (llt_i - elt_j) \cdot \delta_{ji} \qquad \forall i, j \in \{1, \dots, p\},\ i \neq j \qquad (13)$$

Constraint Programming Model

In the previous section the ALP was solved with MIP approach and a model was designed with objective function and multiple constraints. In this section the model was transformed to CP and this methodology was deeply explored.

CP Design

To design the CP model many different approaches and aspects were explored. In this section too, the software used is IBM ILOG CPLEX. One of the very basic and first approaches we had was to test the MIP model to be solved in CP. Applying this model is done in CPLEX and the model was working properly in CP mode. While the process was much more time consuming in comparison to MIP model, the optimum results were achieved and maintained the same for the data of airland 1 to 8. For the airland 9 to 13 the model was not able to close the gaps properly, as the size of the data is bigger and understandably, they take more time.

Next, different tools and commands available in CP mode which they are not accessible in MIP mode in CPLEX were explored. Two of the most interesting aspects of CP is the time intervals and non-linearity. As in the ALP we are faced with different time spans for different planes, initially the model was tested by solely changing the decision variables for the time span to an interval variable with start, end, and duration, then solved the problem with getting the size of each variable. This model was extremely time consuming and behaved poorly in closing the gaps. In fact, this model is quite similar to the MIP model and the advantages of CP were not used in extension.

To improve the model, a slightly different approach was taken. The landing time of each plane is considered the decision variable, like in MIP formulation, with the difference that this will be the only non-binary decision variables. Since CP programming allows non-linear programming, the number of decision variables decrease in comparison with MODEL1, and the performance of the model improved.

Moreover, the big M constraint is no longer needed, since the constraint can be written directly in a non-linear format. This too can help extensively in closing the gaps, being the model more compact and understandable. These actions will be done by directly multiplying the binary variables with the non-binary variables, which decreases the size of the model that is being solved. In addition, for the objective function (1.3), instead of having multiple non-binary variables that made the size of the model bigger, this CP model is written by multiplying the binary variables into nonbinary variables.

The issue in the previous version of the models was that we did not know if the aircrafts land before or after the target time, and each of them had different costs. For this reason, in the CP model, we define a binary variable with the size of the number of the aircrafts, which is equal to 1 when the aircraft lands before the target time and is equal to zero when the aircraft lands after the target time. Therefore, in the objective function, if we multiply the differences of the landing time and the target landing time by that binary variable, it automatically eliminates the redundancies and correctly calculate the optimum cost.

Another aspect of the previous model that changed slightly here is the separation times. Although here we have the same binary variable with two dimensions, the interpretation in writing the model is slightly different, since here we assume that if the plane j lands after plane I, then the model should start to check the separation time limitations from the lower half of the matrix instead of the upper half. This slight change helped the solver to remove any sort of conflicts and confusion in solving the problem. All constraints of the model are presented in Equations (14) to (18).

CP allows the use of filter and search methods. One of the primary observations of the proposed CP model is that in all the cases the model found the optimum solution fast, but it continued to search other branches. This issue can be solved by two different approaches, one is to set the limits for the failures, and the other is changing the search methods. The fastest and easiest way to solve this issue in the ALP data was to set the limit for the failures.

Parameters:

p: $number\ of\ planes$

elt_i: $earliest\ landing\ time\ of\ plane\ i$

tlt_i: $scheduled\ landing\ time\ of\ plane\ i$

llt_i: $latest\ landing\ time\ of\ plane\ i$

lbt_i: $penalty\ cost\ per\ unit\ of\ time\ for\ landing\ before\ target\ landing\ time\ for\ plane\ i$

lat_i: $penalty\ cost\ per\ unit\ of\ time\ for\ landing\ after\ target\ landing\ time\ for\ plane\ i$

st_{ij}: $separation\ time\ between\ plane\ i\ and\ j$

Decision variables:

x_i: $the\ landing\ time\ of\ the\ plane\ i.$

$$\delta_{ij} = \begin{cases} 1, & if\ plane\ i\ lands\ after\ the\ target\ time \\ 0, & otherwise \end{cases}, \quad i = 1,2,\dots,p, \quad j = 1,2,\dots,p$$

$$\mu_{ij} = \begin{cases} 1, & if\ plane\ j\ lands\ after\ plane\ i \\ 0, & otherwise \end{cases}, \quad i = 1,2,\dots,p, \quad j = 1,2,\dots,p$$

Objective function:

$$\sum_{i=1}^{p}\left((1-\delta_i)\cdot lbt_i\cdot(tlt_i - x_i) + \delta_i\cdot lat_i\cdot(x_i - tlt_i)\right) \qquad (1.3)$$

Constraints:

$$allDifferent\ (x) \qquad (14)$$

$$(1-\delta_i)\cdot x_i \le tlt_i \qquad \forall i \in \{1,\ \dots,\ p\} \qquad (15)$$

$$x_i \ge \delta_i\cdot tlt_i \qquad \forall i \in \{1,\ \dots,\ p\} \qquad (16)$$

$$(x_i - x_j)\cdot(1-\mu_{ij}) \ge st_{ji}\cdot(1-\mu_{ij}) \qquad \forall i,j \in \{1,\ \dots,\ p\},\ i \ne j \qquad (17)$$

$$(x_j - x_i)\cdot\mu_{ij} \ge st_{ij}\cdot\mu_{ij} \qquad \forall i,j \in \{1,\ \dots,\ p\},\ i \ne j \qquad (18)$$

As mentioned earlier, this model aims to minimize the total cost of deviation from target landing time. As we can see in the designed objective function, if δ_i = 1 (meaning the plane lands after the target time), lat_i is used as cost, and this cost is being multiplied by the differences between the landing time and the target

time. Likewise, when the δ_i = 0 (meaning the plane lands before the target time), lbt_i is used as cost and is multiplied by the differences between the target time and landing time.

The second and third constraints of this model are related to making sure the relation between the x_i, tlt_i, and δ_i is meaningful, (15) and (16). In these two constraints, when the δ_i = 1, the first constraint will be eliminated, and the second constraint will ensure that in this situation the landing time happens after the target landing time. As well, when the δ_i = 0, the second constraint will be ineffective, and the first constraint will make it sure that the landing time is happening before the target landing time.

(14) is a global constraint that ensures that all landing times are different. Notice that regardless of having this constraint or not, the landing times wouldn't be equal, however in constraint programming the addition of more (even redundant) constraints improves the CP models because decreases the solution space. Also, the usage of global constraints is highly recommended to remove any symmetries or other relations between variables that single constraints combined couldn't find.

The fourth and fifth constraints, (17) and (18), are related to the separation times. If plane i is landing after plane j, then μ_{ij} = 0, and it makes the fourth constraint active which means the separation time should be met, and if μ_{ij} = 1, the fifth constraint activates.

This model was one of the many different models that we developed in the CP format, and uses some abilities of this way of programing as well as having the lower number of variables that can help the solver to solve the problem faster than the other developed models for CP.

Problem Extension: Multiple Runways

So far, the models were developed to find the optimal solution for landing several planes in only one runway. In this section, a new challenge was added to the model, being that the airport has several runways for landing. In this context not only the separation times, but also the runway on which the plane is landing must be considered. In other words, separation times can be violated if the respective planes land on different runways. This was also the approach done previously by other authors [2, 3]. This problem extension was done to the best MIP model considering the initial results for one runway, MODEL1, and for the CP model.

1- MODEL1

In this final version of the MIP model, one parameter and two new decision variables must be added to the model's formulation. The new parameter is the number of runways, added manually. The first decision variable added, a binary variable, tells if one plane lands in a specific runway within the range of runways available. Then, a second binary variable, tells if a plane lands in the same runway as another plane.

The first constraint that was added ensures that a plane only lands in one runway (Equation 19). The second constraint (Equation 20) ensures that the two decision variables that have as indexes the same planes have the same value; in other words, if plane i and plane j land on the same runway, so does planes j and i. The third constraint ensures that if the variables τ_{ir} and τ_{jr} are equal to one for the same runway, then $\varphi_{ij} = 1$, meaning that the two planes land on the same runway. In the last two constraints (Equation 20 and 21) notice that they are only considered for $j > i$; that's the case for symmetry reasons. Now, this is not sufficient because it is necessary to define when the separation time between two planes should be considered or not. To do that, we just need to multiply st_{ij} by φ_{ij}. So, the final MODEL1 consists of equations (1.1), (2), (3), (4), (5), (6), (7), (8.1), (9.1), (19), (20) and (21).

New parameters:

$$R = number\ of\ runways$$

New decision variables:

$$\tau_{ir} = \begin{cases} 1, if\ plane\ i\ lands\ on\ runway\ r \\ 0, otherwise \end{cases}$$

$$\varphi_{ij} = \begin{cases} 1, if\ plane\ i\ lands\ on\ the\ same\ runway\ as\ plane\ j \\ 0, otherwise \end{cases}$$

New constraints:

$$x_i - x_j \geq st_{ji} \cdot \varphi_{ij} - \delta_{ij} \cdot M, \quad \forall i,j \in \{1, \dots, p\},\ i \neq j \tag{8.1}$$

$$x_j - x_i \geq st_{ij} \cdot \varphi_{ij} - (1 - \delta_{ij}) \cdot M, \quad \forall i,j \in \{1, \dots, p\},\ i \neq j \tag{9.1}$$

$$\sum_{r=1}^{R} \tau_{ir} = 1, \quad \forall r \in \{1, \dots, R\} \tag{19}$$

$$\varphi_{ij} = \varphi_{ji}, \quad \forall i,j \in \{1, \dots, p\},\ j > i \tag{20}$$

$$\varphi_{ij} \geq \tau_{ir} + \tau_{jr} - 1, \quad \forall i,j \in \{1, \dots, p\},\ j > i \tag{21}$$

2- Constraint Programming Model

Regarding the constraint programming model for multiple runways, the big M model was used as the CP generalized model. In this case, the equations used were (1.3), (8.1), (9.1), (19), (20), (21), (22) and (23). Equation (1.3) does the same as previous objective functions, but here we have no decision variable for the difference between the target time and the landing time, and here the binary variable γ_i is imbedded in the objective function, and not in a constraint. Equations (22) and (23) allow the activation of the proper limit for the landing time, given the value of γ_i.

New objective function:

$$\sum_{i=1}^{p} ((1 - \gamma_i) \cdot lbt_i \cdot (tlt_i - x_i) + \gamma_i \cdot lat_i \cdot (x_i - tlt_i)) \tag{1.3}$$

New constraints:

$$x_i \geq tlt_i - (1 - \gamma_i) \cdot M \quad \forall i \in \{1, \dots, p\} \tag{22}$$

$$x_i \leq tlt_i + \gamma_i \cdot M \quad \forall i \in \{1, \dots, p\} \tag{23}$$

Results

1- MIP results

Regarding the execution time of the models, MODEL1 has way lower execution times than MODEL2004. For files 1 to 8 they both achieved results similar to that of the state of the art and the running time was always less than 16 seconds per file [2]. On the other hand, for Airland files 9 to 13 the running time grew exponentially, and we were not able to achieve the optimal solution for any of these files using MIP formulation (for one runway at least, and some for more runways). Because the only parameter that changed between the first and the second batch of files is the number of the planes involved, it is evident that this exponential increase in computing time is due to this variable (results presented in Table 1).

Table 1. Results achieved by MIP – MODEL2004 and MODEL1 for small datasets (Airland 1 to 8).

Data	# Planes	# Runways	Optimal Static Solution	Execution Time (sec)	
				MODEL2004	MODEL1
Airland1	10	1	700	8,22	0,22
		2	90	N.A.	0,24
		3	0	N.A.	0,20
Airland2	15	1	1480	8,07	0,18
		2	210	N.A.	0,15
		3	0	N.A.	0,12
Airland3	20	1	820	8,55	1,30
		2	60	N.A.	1,37
		3	0	N.A.	1,25
Airland4	20	1	2520	10,88	1,20
		2	640	N.A.	1,92
		3	130	N.A.	0,51
		4	0	N.A.	0,46
Airland5	20	1	3100	15,73	4,1
		2	650	N.A.	2,63
		3	170	N.A.	1,89
		4	0	N.A.	1,14
Airland6	30	1	24442	8,36	1,36
		2	554	N.A.	1,58
		3	0	N.A.	1,28
Airland7	44	1	1550	9,8	1,75
		2	0	N.A.	1,59
Airland8	50	1	1950	9,76	1,41
		2	135	N.A.	1,41
		3	0	N.A.	1,39

Due to the never-ending running time for Airland 9 to 13 a cutoff time of 30 minutes was defined to identify which MIP model, MODEL2004 or MODEL1, gives the best results within that timeframe, for one runway. Apart from Airland 12, MODEL1 gives better results, meaning lower solution gap and lower integer solution for the objective function (results presented in Table *2*).

Table 2. Results achieved by MIP – MODEL2004 and MODEL1 for big datasets (Airland 9 to 13) – execution time of 30 minutes.

Data	#Planes	#Runways	MODEL2004		MODEL1		Execution Time (sec)
			Best Objective	Gap	Best Objective	Gap	
Airland9	100	1	5642,60	37,32%	5611,70	28,42%	3600
		2	N.A.	N.A.	444,10*	0%	21,29
		3	N.A.	N.A.	75,75*	0%	14,41
		4	N.A.	N.A.	0*	0%	4,42
Airland10	150	1	13713,59	62,52%	12461,10	53,59%	3600
		2	N.A.	N.A.	1143,70	14,98%	3600
		3	N.A.	N.A.	205,21*	0%	246
		4	N.A.	N.A.	34,22*	0%	19,41
		5	N.A.	N.A.	0*	0%	6,89
Airland11	200	1	13336,42	46,49%	12418,32	38,56%	3600
		2	N.A.	N.A.	1330,91	32,40%	3600
		3	N.A.	N.A.	253,07*	0%	1 521
		4	N.A.	N.A.	54,53*	0%	84
		5	N.A.	N.A.	0*	0%	10,34
Airland12	250	1	16129,3	42,49%	16311,29	43,68%	3600
		2	N.A.	N.A.	1698,90	90,26%	3600
		3	N.A.	N.A.	221,97*	0%	1 655
		4	N.A.	N.A.	2,44*	0%	129
		5	N.A.	N.A.	0*	0%	14,67
Airland13	500	1	60757,8	72,91%	37834,06	50,16%	3600
		2	N.A.	N.A.	3943,62	98,65%	3600
		3	N.A.	N.A.	675,74	89,17%	3600
		4	N.A.	N.A.	89,95*	0%	537
		5	N.A.	N.A.	0*	0%	46,97

* optimal solution

2- CP results

As explained in the previous section the proposed model was able to correctly minimize all the data sets and contained the correct solutions. With the addition of the limit for the failure method, when the model reaches an optimum and after that for the certain number of searches cannot find the better solution, it reports the result. This fast approach helped us to decrease and cut the search for many branches after reaching the optimal solution. However, the flaw of this approach is that we may miss a better result that can be achieved by searching more, but in the case of ALP they were redundant, and we could reach to the global optimum solution with this approach.

The results presented next are only the ones for one runway. Multiple runway's results are not presented given the fact that the results were worse, as we can see from the one runway results for complex problems, and therefore only the MODEL1 has the multiple runway's results.

Table *3* compares the results of the CP model with MODEL1. The execution times for the CP model are reported after adding the necessary Fail Limit. While the execution times for the CP are slightly higher than the MIP model, in all datasets CP model was able to find the optimum solution. There are some interesting observations here, which is for the datasets such as airland6 or airland7. CP model has reached to its optimum almost instantly, however, for the smaller datasets such as airland4 or airland5 it took much longer to reach the global optimum. This could be due to the separation time matrix's shape, that has an important role.

In Table *4* the comparison between the CP model and MODEL1 for the big datasets of airland9 to airland13 are presented. Based on this table, the CP model is performing close to the MODEL1, but it is slightly under performing, since the optimum costs are higher for this model.

Table 3. Comparison of the results achieved by CP model and MODEL1 for small datasets (Airland 1 to 8) with one runway.

Data	#Planes	Optimal Static Solution	Execution Time (min:sec)	
			CP	MODEL1
Airland1	10	700	00:01	00:01
Airland2	15	1480	00:14	00:01
Airland3	20	820	00:21	00:02
Airland4	20	2520	01:17	00:01
Airland5	20	3100	02:29	00:04
Airland6	30	24442	00:02	00:02
Airland7	44	1550	00:02	00:02
Airland8	50	1950	03:43	00:02

Table 4. Comparing the results achieved by CP model and MODEL1 for big datasets (Airland 9 to 13) for one runway – Execution time of 30 minutes.

Data	#Planes	Best Objective	
		MODEL1	CP
Airland9	100	5611	6736
Airland10	150	12461	19697
Airland11	200	12418	18207

In addition, one of the important evaluations is the search type. There are different search types and each of them has its own characteristics. We have tested the different search types such as Restart, MultiPoint, Iterative Diving, and Depth First.

In Table 5 we compare the different search strategies for some of the big datasets and report their results. The reported strategies are the Restart, Multipoint, and Iterative Diving. In addition to these strategies, we also tested the Depth First, but the solver was not able to find any result in the 30 minutes time limit. As we can see in this table, different search strategies lead to different performances in terms of closing the gaps and minimizing the costs for our objective function. For instance, in the airland9, Restart strategy has the best performance and objective function, as well as it has the better results than the default Auto search strategy that CP model uses.

Table 5. Comparing the results achieved by CP model with different search strategy (Airland 9 to 11) – Execution time 30 minutes.

Data	#Planes	Restart Objective	Multipoint Objective	Interactive Diving Objective
Airland9	100	6176	7470	7338
Airland10	150	25597	26789	28891
Airland11	200	26956	27695	18469

Discussion

Both MIP and CP approaches solve the problem in different ways, and they use the divide and conquer strategy, where they split the problem into sub-problems. The important difference between these two approaches is in what happens in each node of the tree of the problem. MIP is mostly based on the linear relaxation of the problem, while in CP logical inferences based on the mix nature of each global constraint that is performed.

In terms of the ALP, the MIP model which was inspired by the previous works was able to almost find the solution instantly for the smaller problems, airland1 to airland8, however for the airland9 to airland13 data files it was a time-consuming process, and even after many hours the solver wasn't able to find the optimal solution. However, that is true until a certain number of runways in the formulation. However, because of the changes comparing to the previous works, our model had the better performance in both smaller and larger sets of data. Specifically, our MIP model was able to close the gaps faster than the previously designed models in other papers. Going back to the runways, this project was able to find new optimal solutions that weren't in the literature, therefore improving the state of the art in that regard.

For comparing the two methods of MIP and CP for the ALP, we can say that for the smaller datasets both the MIP and CP models had an acceptable performance comparing to each-other, however the MIP model was slightly better in terms of execution time. The biggest difference in the small datasets of airland1 to airland8 belongs to the airland8 which has the biggest dataset among them, and while in the MIP model it is almost instant, for the CP model it takes up to 4 minutes to reach to the optimum result.

In the bigger datasets, both MIP and CP model were able to solve the problem and since it was taking a long time, we had to set the 30 minutes limit and present the results for this time span. In these datasets also MIP, as expected, was able to reach to the lower objective function than the CP model at the same time. In addition, one interesting aspect of CP models that we tested were the different search types. We did not

find the specific trend in the effect of different search types effect, and in some datasets a search type strategy works better than the other, while in a different dataset it may not be the case.

Conclusions

The goal of this project was to develop and test the MIP and CP models for 13 sets of data related to the ALP, while delay or early landing was implying costs. During this project we primarily developed an MIP optimization model and tested its limitations and compared it to the other studies that has been done in this set of data. The model was successfully able to calculate the optimum results and comparing to the other studies on the big sets of data it showed the better performance in terms of closing the gaps, which can tend to reach to the optimum results in a shorter time.

In the second part of this project, CP was explored. We started with the simple models while keeping the objective function to minimize the costs, the same goal as our MIP model. CP was giving us different abilities such as non-linearly design the model. This way of design helped us to decrease drastically the number of decision variables. In addition to that, in CP there was no need to use the big M method for defining the active constraints. Also, the formulation of the separation times constraints is close to the previous model, but it has a slight change that implemented intuitiveness.

One of the other explored aspects of the CP model was different search types, we implemented the different search types and compare the results of the different strategies, however we could not find a specific trend on this aspect. The CP model has tested in all the 13 datasets and the results could compete in terms of the performance and accuracy with our MIP model, however, the MIP model slightly performing better.

To finish this project, the best MIP model and two CP models, with non-linear equations and with a big constant M, were chosen to being generalized for as much runways as the airport might have. This generalization opened opportunities to check all data files for a different number of runways, enabling a more complete comparison with the literature. With this extension of the main problem, we were able to find new optimal solutions for the large versions of the ALP.

There are several aspects that can be applied to the models for the future research. For instance, the costs of implementation, repair, freeze time, appearance time, and extra staff needed can be considered to understand whether the investment made in increase in the number of runways is worth in the long run.

References

[1] A. Faye, "Solving the Aircraft Landing Problem with time discretization approach," vol. 242, no. 3, pp. 1028-1038, 2015.

[2] J. E. Beasley, M. Krishnamoorthy, Y. M. Sharaiha and D. Abramson, "Displacement problem and dynamically scheduling aircraft landings," *Journal of the Operational Research Society,* vol. 55, pp. 54-64, 2004.

[3] J. E. Beasley, M. Krishnamoorthy, Y. M. Sharaiha and D. Abramson, "Scheduling Aircraft Landings—The Static Case," *Transportation Science,* vol. 34, no. 2, pp. 180-197, 2000.

[4] M. d. Berg, M. v. Kreveld, M. Overmars and O. Schwarzkopf, " Chapter 4: Linear Programming," in *Computational Geometry*, 2000.

[5] T. Fahle, R. Feldmann, S. Götz, S. Grothklags and B. Monien, "The Aircraft Sequencing Problem," in *Computer Science in Perspective*, 2003.

[6] A. Awasthi, K. Oliver and L. Joerg, "Aircraft Landing Problem: Efficient Algorithm for a Given Landing Sequence," *Proceedings - 16th IEEE International Conference on Computational Science and Engineering, CSE 2013,* 2013.

[7] P. v. Leeuwen, H. Hesselink and J. Rohling, "Scheduling Aircraft Using Constraint Satisfaction," *Electronic Notes in Theoretical Computer Science,* vol. 76, pp. 252-268, 2002.