A NOTE ON OPTIMIZATION MODELLING OF PIECEWISE LINEAR DELAY COSTING IN THE AIRLINE INDUSTRY

PRABHU MANYEM

College of Science
Nanchang Institute of Technology
Nanchang, Jiangxi 330099, P.R. China

(Communicated by Ruhul Sarker)

Abstract. We present a mathematical model in an integer programming (I.P.) framework for non-linear delay costing in the airline industry. We prove the correctness of the model mathematically. Time is discretized into intervals of, for example, 15 minutes. We assume that the cost increases with increase in the number of intervals of delay in a piecewise linear manner. Computational results with data obtained from Sydney airport (Australia) show that the integer programming non-linear cost model runs much slower than the linear cost model; hence fast heuristics need to be developed to implement non-linear costing, which is more accurate than linear costing. We present a greedy heuristic that produces a solution only slightly worse than the ones produced by the I.P. models, but in much shorter time.

1. Introduction. Motivation. Delay costing is an important issue in the airline industry. Delays cost airlines millions of dollars every year. These include passenger inconvenience, rebooking passengers in different flights, meal vouchers, hotel accommodation, crew overtime — the list goes on. When the delay is very high, the “delay cost” is equal to the cost of cancelling the flight, which includes the cost of accommodating passengers in other flights. Given its importance to the airline industry, it is crucial and critical to estimate the delay costs as accurately as possible.

Hence assuming that the delay cost increases linearly with delay is a naive approach. Cook et al have studied the issue in a series of publications [3, 5, 6]. As we can observe in Figure 8 of [6], we can broadly identify three distinct phases of an S-shaped curve: a gradual increase at the beginning, a steep increase next, and a gradual increase later. Ferguson et al [7] compare US airline delay costing with the European model.

Complete tables for various aircraft types and for different scenarios in cost increase (or decrease) between 2010 and 2014 were published in 2015 [4].

In our previous studies of Sydney airport in Australia, we have used the (simpler) linear delay cost model [8, 9, 13]. Linear cost models have also been used in [2] and [15].

2010 Mathematics Subject Classification. Primary: 90C10; Secondary: 90B06.

Key words and phrases. Air traffic management, Non-linear delay costing, Mathematical modelling, Discrete (Combinatorial) optimization, Integer programming.

The author was supported by Grant No. GJJ161113 (2017-2019) from the Education Department of Jiangxi Province, P.R. China.
Optimization of non-linear costing can be found in both discrete and continuous optimization. A common example is convex cost functions. For examples of the discrete case, see [1, 12].

For a simulation based optimization of flight delay costs in a multi-airport setting, see [10]. In [14], the author considers non-linear costing for flights in a stochastic model for single airport ground holding, but he uses Linear Programming to obtain solutions.

A good example of a non-linear cost function in continuous optimization in aviation is [11]. This publication deals with aircraft trajectory optimization.

For an important application of non-linear delay costing in the trucking industry which is very recent, see [16].

1.1. **Discrete versus continuous time models.** When we attempt to develop a mathematical model, the other issue at play here is the following: Do we wish to use a continuous optimization model or a discrete optimization model? Discrete optimization models have been widely used in the aviation industry. We have used it in our previous work for Sydney airport.

In discrete optimization, an integer programming (I.P.) framework is frequently used to state the problem.

**Discrete time model:** In a discrete model, time is discretized into periods (or intervals) of 15 or 30 minutes, typically. We assume that within each interval, the delay cost increases linearly. Hence the delay cost for each flight increases in a piecewise linear manner as time goes by. Each linear range of the cost function could cover one or more (discretized) time intervals.

**Airport arrival capacities.** Let us define *arrival capacity* at an airport as the maximum number of aircraft that can arrive in a time interval whose duration is \( \phi \).

Aircraft come in different sizes. Due to this, the minimum separation required between aircraft varies. For example, if a small aircraft is following a large one to land on the same runway, the separation should be significantly higher than the case when a large aircraft tails a small one. When we consider airport arrival capacities, \( \phi \) should be sufficiently large to accommodate these effects.

For instance, setting \( \phi \) to be 30 seconds (or less) makes no sense, because the minimum separation needed between two consecutive landing aircraft may be more than a minute. In such a case, \( \phi \) should be significantly larger than a minute.

Therefore, a discrete time model is a natural choice when we study models with arrival and/or departure capacities. It is convenient to set the duration of each time-step in our optimization model to be \( \phi \).

At Airservices Australia, they use a value of 10 minutes for \( \phi \).

1.2. **Our contribution.** In this paper, we present the I.P. constraints necessary to model non-linear delay costing in a piecewise linear manner. To our knowledge, such discretized optimization modelling of delay-costing has not been carried out before, especially with respect to airport disruption recovery.

**Ground holding.** We illustrate by applying this to a ground holding model. In *ground holding*, also known as *ground delay programs* (GDP), flights are held on the ground at their origin due to capacity reductions at their destination; once such flights are cleared for take-off, they arrive at their destination without any additional delay.
We then present a heuristic algorithm, since solving the I.P. model with an I.P. solver is too slow, too inefficient. We compare our heuristic with the I.P. solver computationally with data obtained from Sydney airport.

1.3. Notation. Our model is a discrete time model. It is based on the Navazio and Romanin-Jacur (NRJ) model [15]. We use the following notation in our models:

- \( T \) set of time periods, \( T = \{1, \ldots, t, \ldots, |T|\} \)
- \( Z \) set of airports, \( Z = \{1, \ldots, z, \ldots, |Z|\} \)
- \( F \) set of all flights, \( F = \{1, \ldots, f, \ldots, |F|\} \)
- \( F_z \) set of flights arriving at airport \( z \)
- \( F_{z,t} \) set of flights scheduled to arrive at airport \( z \) in period \( t \)
- \( K_{z,t} \) arrival capacity of airport \( z \) in period \( t \) (that is, the maximum number of aircraft that can land during period \( t \))
- \( \phi \) length of the time interval to measure airport arrival capacity, as well as the duration of each time step in the set \( T \)
- \( r_f \) planned arrival period of flight \( f \) (\( r_f \in T \))
- \( c_{i,f} \) cost per period of delay in Range \( i \) for flight \( f \)
- \( d_{i,f} \) a break-point interval that separates Range \( i \) from Range \((i+1)\); \( d_{i,f} \) is the last interval in Range \( i \)
- \( \delta_{i,f} \) equal to one if the active range is \( i \) or higher, and zero otherwise (see Table 2)
- \( \mu_{i,f} \) equal to zero if the active range is less than \( i \), and equal to \((\Delta_f - d_{i-1,f})\) otherwise
- \( D_f \) Total delay cost for flight \( f \)
- \( \Delta_f \) delay experienced by flight \( f \), in periods
- \( A_f \) Actual arrival period of flight \( f = r_f + \Delta_f \)
- \( \Delta_{\text{max}} \) maximum allowed delay on any flight, in periods
- \( L_f = r_f + \Delta_{\text{max}} \)
- \( T_f \) set of time intervals in which flight \( f \) may land = \([r_f, L_f]\)

2. Methodology: Modelling of piecewise linear delay costs. We first demonstrate this with an example for three cost ranges before presenting the general case of \( n \) cost ranges. A cost range is a sequence of intervals (or periods) that are adjacent to each other, during which the delay cost per interval remains the same.

2.1. Example: Three cost ranges. For example, if the planned arrival interval of a flight \( f \) is 20 and if the maximum allowed delay \( \Delta_{\text{max}} \) is 11 intervals, then the first cost range could be intervals 20, 21, 22, 23 and 24, the second cost range could be intervals 25, 26 and 27, and the third cost range could be intervals 28, 29, 30 and 31. The cost coefficient (that is, the delay cost in dollars per interval) in these three ranges are identified as \( c_{1,f}, c_{2,f} \) and \( c_{3,f} \) respectively.

If flight \( f \) suffers a delay of 9 intervals, for instance, its delay cost = \( 4c_{1,f} + 3c_{2,f} + 2c_{3,f} \) dollars. For this example, we have used \( c_{1,f} = 2, c_{2,f} = 4 \) and \( c_{3,f} = 1 \).

We identify two break-points. At the end of a “break-point” interval, the cost coefficient changes its value. The first and second break points are \( d_{1,f} = 4 \) and
That is, the cost coefficients in the three delay ranges (a) \(1 \leq \Delta \leq 4\), (b) \(5 \leq \Delta \leq 7\) and (c) \(\Delta \geq 8\) are different from each other. See Figure 1 for a plot of this example.

\[
\begin{align*}
\delta_1, f & = \text{integer}\quad \text{(given value)} \\
\delta_2, f & = \text{integer}\quad \text{(given value)} \\
\delta_3, f & = \text{integer}\quad \text{(given value)} \\
\mu_2, f & = \text{integer}\quad \text{(given value)} \\
\mu_3, f & = \text{integer}\quad \text{(given value)} \\
D_f & = \text{integer}\quad \text{(given value)} \\
\Delta_f & = \text{integer}\quad \text{(given value)}
\end{align*}
\]

Figure 1. \(S\)-shaped piecewise linear curve for Cost vs Delay; the curve is convex first and concave later.

2.1.1. Integer programming constraints for three cost ranges. Assume that there are three cost ranges with respect to \(\Delta_f\), the delay experienced by flight \(f\). Recall that \(\Delta_f\) is a non-negative integer. The cost coefficient \(c_f\) is the cost per time interval of delay.

\(d_{1,f}, d_{2,f}, d_{3,f}, c_{1,f}, c_{2,f}\) and \(c_{3,f}\) are constants (given values), whereas \(\delta_{1,f}, \delta_{2,f}, \delta_{3,f}, \mu_{2,f}, \mu_{3,f}\) and \(\Delta_f\) are variables in the optimization model. These are explained in Table 2.

Observe that the first time interval in Ranges 2 and 3 are \((1 + d_{1,f})\) and \((1 + d_{2,f})\) respectively. Also note that \(\delta_{1,f}\) is always one, since \(\Delta_f\) is required to be non-negative.

Active range: We say that Range \(k\) is active when the delay \(\Delta_f\) falls in that range.

Objective function. The objective function is as follows:

\[
\text{Minimize } \sum_{f \in F} D_f,
\]

where \(D_f\) is given below. The following constraints model the requirements in Table 2:

\[
\begin{align*}
c_{1,f} \Delta_f + (c_{2,f} - c_{1,f}) \mu_{2,f} + (c_{3,f} - c_{2,f}) \mu_{3,f} & = D_f \\
\delta_{2,f} & \geq \delta_{3,f} \\
M \delta_{2,f} & \geq \mu_{2,f} \\
\mu_{2,f} & \geq \Delta_f - d_{1,f} \\
\Delta_f - d_{1,f} \delta_{2,f} & \geq \mu_{2,f}
\end{align*}
\]
Table 2. Delay ranges and corresponding costs. (The letters (A) to (F) identify the various columns.)

| Range (A) | Delay range (B) | Cost (C) coefficient (D) |
|-----------|----------------|--------------------------|
| 1         | $0 \leq \Delta_f \leq d_{1,f}$ | $c_{1,f}$ | $c_{1,f} \Delta_f$ |
| 2         | $d_{1,f} < \Delta_f \leq d_{2,f}$ | $c_{2,f}$ | $c_{1,f}d_{1,f} + c_{2,f}(\Delta_f - d_{1,f})$ |
| 3         | $\Delta_f > d_{2,f}$ | $c_{3,f}$ | $c_{1,f}d_{1,f} + c_{2,f}(d_{2,f} - d_{1,f}) + c_{3,f}(\Delta_f - d_{2,f})$ |

(A) $\delta$ values (E) Requirements (F)

1. $\delta_{1,f} = 1$ and $\delta_{2,f} = \delta_{3,f} = 0$ $\mu_{2,f} = \mu_{3,f} = 0$
2. $\delta_{1,f} = \delta_{2,f} = 1$ and $\delta_{3,f} = 0$ $\mu_{2,f} = (\Delta_f - d_{1,f})$ and $\mu_{3,f} = 0$
3. $\delta_{1,f} = \delta_{2,f} = \delta_{3,f} = 1$ $\mu_{2,f} = (\Delta_f - d_{1,f})$ and $\mu_{3,f} = (\Delta_f - d_{2,f})$

\[
\begin{align*}
M\delta_{3,f} &\geq \mu_{3,f} & (7) \\
\mu_{3,f} &\geq \Delta_f - d_{2,f} & (8) \\
\Delta_f - d_{2,f}\delta_{3,f} &\geq \mu_{3,f} & (9) \\
\delta_{2,f}, \delta_{3,f} &\in \{0, 1\} & (11)
\end{align*}
\]

M is a sufficiently large positive number.

For the general case (any number of cost ranges), the Constraints (19-24) are on Page 7.

When the delay $\Delta_f$ is in Range 1, $\delta_{1,f}$ will be one, and $\delta_{2,f}$ and $\delta_{3,f}$ will be zero. When it is in Range 2, $\delta_{2,f}$ will be one and $\delta_{3,f}$ will be zero. In Range 3, all three $\delta_{i,f}$ ($1 \leq i \leq 3$) will be one.

In Range 1, when $\delta_{2,f} = 0$: Inequality (4) becomes $0 \geq \mu_{2,f}$. Since it is also required that $\mu_{2,f} \geq 0$, we let $\mu_{2,f} = 0$. This satisfies (5) since $\Delta_f \leq d_{1,f}$. Inequality (6) is also satisfied since $\Delta_f - d_{1,f}\delta_{2,f} = \Delta_f \geq 0 = \mu_{2,f}$.

In Ranges 2 and 3, when $\delta_{2,f} = 1$: Inequality (6) becomes $\Delta_f - d_{1,f} \geq \mu_{2,f}$. Together with (5), we obtain $\Delta_f - d_{1,f} = \mu_{2,f}$. This satisfies (4) since $M\delta_{2,f} = M$ has a large positive value.

In a similar manner, using inequalities (7-9), it is straightforward to see that $(\delta_{3,f} = 0) \Rightarrow (\mu_{3,f} = 0)$ and that $(\delta_{3,f} = 1) \Rightarrow (\mu_{3,f} = \Delta_f - d_{2,f})$.

Hence we have proved that Constraints (3-9) satisfy the requirements in Table 2.

We can also show that the delay cost definition in (2) satisfies the delay costs mentioned in Table 2.

In Range 1, $\mu_{2,f} = \mu_{3,f} = 0$ and hence from (2), $D_f = c_{1,f} \Delta_f$.
When Range 2 is active, $\mu_{2,f} = \Delta_f - d_{1,f}$, $\mu_{3,f} = 0$ and hence from (2),

\[
\begin{align*}
D_f & = c_{1,f} \Delta_f + (c_{2,f} - c_{1,f})\mu_{2,f} = c_{1,f} \Delta_f + (c_{2,f} - c_{1,f})(\Delta_f - d_{1,f}) \\
& = c_{1,f} \Delta_f + c_{2,f} \Delta_f - c_{1,f} \Delta_f - c_{2,f} d_{1,f} + c_{1,f} d_{1,f} \\
& = c_{2,f} \Delta_f - c_{2,f} d_{1,f} + c_{1,f} d_{1,f} = c_{1,f} d_{1,f} + c_{2,f}(\Delta_f - d_{1,f})
\end{align*}
\]
which is what we have for the delay cost for Range 2 in Column (D) of Table 2.

When Range 3 is active, \( \mu_{2,f} = \Delta_f - d_{1,f} \), \( \mu_{3,f} = \Delta_f - d_{2,f} \). Hence from (2),
\[ D_f = c_{1,f}\Delta_f + (c_{2,f} - c_{1,f})\mu_{2,f} + (c_{3,f} - c_{2,f})\mu_{3,f}. \]

Now using the results from (12),
\[ D_f = c_{1,f}\Delta_f + (c_{2,f} - c_{1,f})\mu_{2,f} + (c_{3,f} - c_{2,f})\mu_{3,f} \]
\[ = c_{1,f}\Delta_f + (c_{2,f} - c_{1,f})(\Delta_f - d_{1,f}) + (c_{3,f} - c_{2,f})(\Delta_f - d_{2,f}) \]
\[ = c_{1,f}\Delta_f + c_{2,f}(\Delta_f - d_{1,f}) + c_{3,f}\Delta_f - c_{2,f}\Delta_f - c_{3,f}d_{2,f} + c_{2,f}d_{2,f} \]
\[ = c_{1,f}\Delta_f + c_{2,f}\Delta_f - c_{2,f}d_{1,f} + c_{3,f}\Delta_f - c_{2,f}\Delta_f - c_{3,f}d_{2,f} + c_{2,f}d_{2,f} \]
\[ = c_{1,f}\Delta_f + c_{2,f}(d_{2,f} - d_{1,f}) + c_{3,f}(\Delta_f - d_{2,f}), \quad (13) \]

which is what we have for the delay cost for Range 3 in Column (D) of Table 2.

2.2. Constraints: The general case. The methodology captured by the constraints (2-11) can be extended to any number of cost ranges. The objective function is the same as before (as in Eq. (1)).

\[ D_f = c_{1,f}\Delta_f + \sum_{i=2}^{n}(c_{i,f} - c_{i-1,f})\mu_{i,f} \quad (14) \]

where \( \mu_{i,f} = 0 \) if \( \delta_{i,f} = 0 \) and \( \mu_{i,f} = (\Delta_f - d_{i-1,f}) \) if \( \delta_{i,f} = 1 \).

In the form of the delay cost as provided in Column (D) of Table 2, the delay cost is
\[ DC = c_{1,f}\Delta_f + \sum_{i=2}^{n-1}c_{i,f}(d_{i,f} - d_{i-1,f}) + c_{n,f}(\Delta_f - d_{n-1,f}). \quad (15) \]

We can prove that (14) results in (15) using mathematical induction, similar to how we showed (13) using (12).

**Theorem 2.1.** For every integer \( n \geq 2 \), \((14) \equiv (15)\); that is \( D_f = DC \).

**Proof.** We have shown that (14) leads to (15), that is, \( D_f = DC \) for Range \( n \) where \( 1 \leq n \leq 3 \). Assume that this is true when Range \( n = k \) is active; that is,
\[ c_{1,f}\Delta_f + \sum_{i=2}^{k}(c_{i,f} - c_{i-1,f})\mu_{i,f} \]
\[ = c_{1,f}\Delta_f + \sum_{i=2}^{k-1}c_{i,f}(d_{i,f} - d_{i-1,f}) + c_{k,f}(\Delta_f - d_{k-1,f}). \quad (16) \]

Now let us show this when Range \( n = (k + 1) \) is active. That is, we should prove that
\[ c_{1,f}\Delta_f + \sum_{i=2}^{k+1}(c_{i,f} - c_{i-1,f})\mu_{i,f} \]
\[ = c_{1,f}\Delta_f + \sum_{i=2}^{k}c_{i,f}(d_{i,f} - d_{i-1,f}) + c_{k+1,f}(\Delta_f - d_{k,f}). \quad (17) \]

This is accomplished by expanding the LHS of (17) and substituting the RHS of (16), as shown below:
\[ c_{1,f} \Delta_f + \sum_{i=2}^{k+1} (c_{i,f} - c_{i-1,f}) \mu_{i,f} \]

\[ = c_{1,f} \Delta_f + \sum_{i=2}^{k} (c_{i,f} - c_{i-1,f}) \mu_{i,f} + (c_{k+1,f} - c_{k,f}) \mu_{k+1,f} \]

(now substitute the RHS of (16)

\[ = c_{1,f} \Delta_f + \sum_{i=2}^{k-1} c_{i,f} (d_{i,f} - d_{i-1,f}) + c_{k,f} (\Delta_f - d_{k-1,f}) + (c_{k+1,f} - c_{k,f}) \mu_{k+1,f}. \]

When Range \((k+1)\) is active, \(\delta_{k+1,f} = 1\) and hence \(\mu_{k+1,f} = (\Delta_f - d_{k,f})\).

Substituting this above, we get:

\[ c_{1,f} \Delta_f + \sum_{i=2}^{k-1} c_{i,f} (d_{i,f} - d_{i-1,f}) + c_{k,f} (\Delta_f - d_{k-1,f}) + (c_{k+1,f} - c_{k,f})(\Delta_f - d_{k,f}). \quad (18) \]

Expanding the last two additive terms of (18) to obtain the RHS of (17) is a simple exercise which we omit.

Furthermore, we need to add a few more constraints for every \(2 \leq i \leq n\):

\[ M \delta_{i,f} \geq \mu_{i,f} \quad (19) \]

\[ \mu_{i,f} \geq \Delta_f - d_{i-1,f} \quad (20) \]

\[ \Delta_f - d_{i-1,f} \delta_{i,f} \geq \mu_{i,f} \quad (21) \]

\[ \delta_{i-1,f} \geq \delta_{i,f} \quad (22) \]

\[ \Delta_f, \mu_{i,f} \geq 0 \text{ and integer} \quad (23) \]

\[ \delta_{i,f} \in \{0,1\}. \quad (24) \]

**Concluding this section:** This methodology works regardless of whether the cost function is convex piecewise-linear or otherwise. The S-shaped curve in Fig. 8 of [6] is convex at the beginning and concave later. The inflection point where the curve changes from convex to concave occurs in the mid-section of the curve (the second range in the example of Sec. 2.1).

3. **Computational testing.** The integer programming models of the two optimization problems that we solved are described in the appendix.

We ran our programs on a computer running a 64-bit Linux kernel version 4.15.0-34 with 4 GB memory and 4 GB Swap space. No other significant programs ran on the computer during our experiments, thus allowing the CPU to be entirely at our programs’ disposal. We used the GLPK Integer Programming (I.P.) software with GMPL as the modelling language.

We used actual arrival-departure information\(^1\) collected from Sydney airport in Australia. There are 517 flights, 261 of which arrive in Sydney and the rest depart from Sydney. More information is provided in Table 3. “Perfect Capacities” in Table 3 refers to the maximum possible capacity, such as when the sky is clear at mid-day and visibility is the highest possible.

Points to be noted from the experiments are as follows:

\(^1\)Readers can obtain the data and the computer program source code used in this research by contacting the author.
Table 3. Parameters used in the testing of the non-linear cost model

| Parameter                        | Description                                                                 |
|----------------------------------|-----------------------------------------------------------------------------|
| Number of flights                | 261 arrivals, 256 departures                                                |
| Max. delay allowed               | Eight periods (four hours)                                                  |
| Perfect capacities               | 20 arrivals, 20 departures (per 30-minute interval)                         |
| Time intervals                   | 30 minutes each                                                             |
| Cost break points                | $d_1 = 2$ units of delay, $d_2 = 6$ units of delay                          |
| Cost coefficients                | $c_{1,f} = c_f$, $c_{2,f} = (1.5)c_f$ and $c_{3,f} = 0.5c_f$. $(c_f$ is the coefficient in the single-range model) |

1. Even for this simple optimization model not involving more complicated phenomena such as connecting flights or flight cancellations, the 3-range (non-linear) cost I.P. model runs much slower than the single-range (linear) cost model. This indicates that approximation heuristics are necessary to handle non-linear costing.

2. The unit of measurement for $c_f$ is assumed to be “$ per period”.

3. Airport arrival capacity profile: The capacity begins dropping from 2:30 AM as the weather worsens. Between 3:30 AM and 8:30 AM, the capacity is zero (no arrivals allowed). Then the weather and visibility begin to improve gradually; and by 11:30 AM, perfect arrival capacity is restored.

4. We ran the model with both versions of the variable $x_{f,t}$. In the first version, $x_{f,t} = 1$ if flight $f$ arrives during or before period $t$ and zero otherwise. In the second version, $x_{f,t} = 1$ if flight $f$ arrives during period $t$ and zero otherwise.

   In both versions, the single-range cost model ran much faster than the 3-range cost model (which is to be expected). The single range model obtained an optimal solution in less than two minutes, whereas the 3-range cost model was unable to reach optimality even after 48 hours.

5. The results are tabulated in Table 4 (Page 9). From Row 3 of this table, we observe that computation time increases rapidly. Hence we decided not to proceed with finding an optimal solution.

6. The objective function (26) is to be minimized. For an instance $I$ and a feasible solution $S$ for $I$, its approximation ratio $A_S$ can be defined as:

\[
A_S = \frac{\text{Value of Solution } S}{\text{Value of an optimal solution to } I} = \frac{\text{Value}(S)}{\text{Opt}(I)}.
\]  

(25)

However, since we did not find optimal solutions to instances, we can only estimate $A_S$ or find an upper bound to $A_S$. The closer $A_S$ is to one, the better the quality of the solution $S$ obtained.

7. The denominator in (25): A lower bound on a feasible solution to $I$, which we call $LB(I)$, is also a lower bound on $Opt(I)$.

   The GLPK software uses branch and bound (B&B) to solve I.P. problems. $LB(I)$ is continuously updated as the program moves from one B&B node to another. If a better (i.e. higher) value is found, $LB(I)$ is replaced with the new value.

8. The numerator in (25): For $Value(S)$, GLPK uses the value of the best integer feasible solution encountered so far in the B&B search.
9. The three different 3-range models (A, B and C in the table) refer to the tolerance limits mentioned in Row 2. The 3-range model found a feasible solution with \( A_S \leq 1.36 \) in 97 minutes. However, to find a feasible solution with \( A_S \leq 1.33 \), it took more than 6 hours. To find a feasible solution with \( A_S \leq 1.32 \), it spent more than 38 hours.

10. Although the total delay is the same in all cases (518 periods), this is distributed more fairly (i.e. over more flights) in the 3-range case. A multi-range cost model is able to better distribute delays among different flights than the single-range model.

11. From Rows 7 and 8 of the Table 4, note that the single-range model returned an integer optimal solution even when solved as a Linear Program, that is, when the integrality constraints were relaxed.

4. **A Greedy Heuristic for a single airport problem.** Since solving the I.P. model using an I.P. solver appears to be impractical, we developed a heuristic in order to produce a good approximate solution. This is for a “single airport” problem; that is, capacity constraints are imposed at only one airport in the system, and no restrictions on the number of arrivals at other airports. The airport with restricted capacities is said to be the “first” airport, with index \( z = 1 \).

So which \( x_{f,t} \) should be set to one first? We should start filling up the airport capacity (the “arrival slots”) from the earliest time slot. If we fill Time Period 2 first

---

**Table 4. Results of comparing single-range versus three-range cost model (Note. The “$” used in Rows 7 and 8 refers to a generic monetary unit, not actual amount in US dollars or Australian dollars or any other currency.)**

| Model → | Single-range | 3-range A | 3-Range B | 3-Range C |
|---------|--------------|-----------|-----------|-----------|
| (1) Solution how far from optimal (\( A_S \) value) | 1.0 | \( \leq 1.36 \) | \( \leq 1.33 \) | \( \leq 1.32 \) |
| (2) Number of delayed flights | 85 | 117 | 122 | 122 |
| (3) Time taken to find solution | 0.18 seconds | 97 mins | 6.5 hours | 38 hours |
| (4) Sum of the delays of all flights (periods) | 518 | 518 | 518 | 518 |
| (5) Average delay (periods) over all flights | 1.002 | 1.002 | 1.002 | 1.002 |
| (6) Average delay (periods) only over delayed flights | 6.094 | 4.427 | 4.2459 | 4.2459 |
| (7) Objective function value ($) | 32960 | 38975 | 37830 | 37830 |
| (8) Optimal value of the Linear Programming relaxation ($) | 32960 | 7762 |
and Period 1 later, a flight $f_1$ scheduled to arrive in Period 1 could become delayed, in which case $f_1$ may have to be delayed even further, to Period 3 or beyond! Hence we start filling the arrival slots from the earliest period.

Note that for a given period $t$, since $t$ remains fixed, comparing $tc_f$ for different flights is the same as comparing their $c_f$.

4.1. A snapshot of the heuristic. The algorithm works as follows. We have skipped some details in this snapshot, such as infeasibility of a particular flight, but here is a general idea about the heuristic algorithm:

1. First choose time period 1 (set $t = 1$) and airport $z = 1$.
2. To minimize the costs, first choose the flight $f$ (say $f_1$) with the highest cost coefficient $c_{f_1}$ and schedule its arrival time $A_{f_1}$ as $A_{f_1} = t$.
3. Next, choose the flight with the second highest cost coefficient (say $f_2$) and schedule its arrival time $A_{f_2}$ as $A_{f_2} = t$.
4. Continue this until all flights scheduled to arrive at Period $t$ at airport $z = 1$ have been scheduled, or, if the arrival capacity $K_{1,t}$ has been reached, whichever occurs first.
5. Rolledover flights. If the capacity at the first airport is insufficient in period $t$, that is, if there are still unscheduled flights when the capacity $K_{1,t}$ is reached, these remaining flights (or leftover flights or rolledover flights) are moved to the following time period ($t + 1$). They are considered for arrival scheduling in period ($t + 1$).
6. Repeat the above process (from Step 2 onwards) for every time period $t$ until all flights scheduled to arrive at Airport 1 have been processed (that is, assigned an arrival time).
7. Finally, process flights scheduled to arrive at other airports. (Recall that this is a single airport problem. That is, only Airport 1 suffers from arrival capacity restrictions.)

See Section 1.3 for notation. The complete method is provided in Algorithm 1 below.

**Marginal costs.** In this algorithm, we have used the marginal cost approach to choose which flights to land and which flights to delay, in a given time period $t$. Marginal cost considers the following: if a flight is delayed by one time period, what is the effect on the objective function? We choose the flight with a higher cost coefficient over a flight with a lower cost coefficient, even if the total accumulated delay cost of the latter (up to $t$) has exceeded that of the former.

**Algorithm 1** Heuristic for ground holding for flights arriving at a single airport

1: procedure **Main**
2: for $t = 1$ to $T$ do Flt[$t$] = $F_{1,t}$;
3: for $t = 1$ to $T$ do
4: RemCap = $K_{1,t}$; // Remcap is the remaining capacity
5: while (Flt[$t$] ≠ ∅ & RemCap > 0) do
6: Let $f$ = the most expensive flight in Flt[$t$], with cost $c_f$;
7: If $g$ ∈ Flt[$t$] so that its last possible arrival time is $t$, set $f = g$.
8: If $g$ ∈ Flt[$t$] with successor $h$ and $L_h = t + \sigma_{g,h} + \theta_h$, set $f = g$.
9: $x_{f,t} = 1$ and $A_f = t$; // Schedule $f$ to arrive at $t$
10: RemCap = RemCap −1;
11: Flt[$t$] = Flt[$t$] \ {$f$};
end while
13: if (RemCap = 0 & Flt[t] ≠ ∅ & t = T) then
14: declare infeasibility and exit;
15: // Flights in Flt[t] can never be scheduled to land,
16: // hence the infeasibility.
17: end if
18: if (RemCap = 0 & ∃ f ∈ Flt[t] with t = L_f) then
19: declare infeasibility and exit;
20: // (since f can never be scheduled to land)
21: end if
22: if (RemCap = 0 & Flt[t] ≠ ∅ & t < T) then
23: // Notice that for every f ∈ Flt[t], t < L_f
24: Flt[t + 1] = Flt[t + 1] ∪ Flt[t];
25: end if
26: end for
27: // Now schedule arrivals of flights at airports other than the main airport:
28: for every flight f ∈ F_z where z ≠ 1 do
29: if any of f’s predecessors have not been scheduled to arrive then
30: declare infeasibility and exit;
31: end if
32: Let t_min = earliest time that f could arrive, considering the actual
33: arrival times of all of f’s predecessors, as per successor constraint (32).
34: If t_min > L_f, declare infeasibility and exit;
35: x_f,t_min = 1 and A_f = t_min; // Schedule f to arrive at t_min
36: end for
37: end procedure

Rolledover flight. In Line 22 of Alg. 1, if Flt[t] is non-empty, these flights are
considered for arrival in the next time period. Such flights are known as rolledover
flights of period t.

Cost coefficient in Line 6. Note that this depends on the delay (t – r[f]) suffered
by flight f up to the current time interval t.

The algorithm works well. It may look simple, unsophisticated, but it does the
job in a satisfactory manner (as explained in Table 5, the results of the heuristic are
slightly worse than that of Integer Program, but the heuristic runs much faster).

4.2. Computational testing of the heuristic. The heuristic was coded in the C
language. We tested it with two data sets; (i) the data set used in Sec. 3 where flight
connectivity was not considered; and (ii) another data set with flight connections
(with 791 flights). In both cases the heuristic produced feasible solutions in less
than 200 seconds.

4.2.1. Case (i): No flight connections. We now compare the heuristic results with
the best 3-range result obtained by the I.P. solver (the last column of Table 4).

As we can observe in Table 5, the average delay over delayed flights and the
objective function value are slightly worse for the heuristic (Algorithm 1), but the
running time of the heuristic is much better (only 60 seconds), which makes imple-
mentation of the heuristic in a real-world situation a very appealing proposition.
Table 5. Results of comparing the Integer Programming solution and the Heuristic solution

| Model/Solver | 3-range C (from Table 4) | Algorithm 1 (Pages 10-11) |
|--------------|--------------------------|---------------------------|
| Number of delayed flights | 122 | 121 |
| Time taken to find solution | 38 hours | 60 seconds |
| Sum of the delays of all flights (periods) | 518 | 518 |
| Ave. delay (periods) over all flights | 1.002 | 1.002 |
| Ave. delay (periods) only over delayed flights | 4.246 | 4.281 |
| Objective function value ($) | 37830 | 39660 |

4.2.2. Case (ii): With flight connections. This data set consists of 791 flights at Sydney airport (393 arrivals and the 398 departures). The number of arriving flights that connect to flights departing from Sydney equals 235. (Each arriving flight connects to at most one departing flight, and every departing flight has to wait for at most one arriving flight.)

Time is discretised into intervals of 15 minutes. Period One is the interval from 12 midnight to 12:15 AM. Similarly the last period of the day, Period 96, is the interval from 11:45 PM to midnight. Arrival capacity in perfect weather is assumed to be 9 flights. In this experiment, capacity is the maximum number of arrivals allowed in a 15-minute period.

Airport capacity profile: The capacity is perfect (9 arrivals) at all times except for the following periods:

| Period | 29-32 | 69-72 | 77-92 |
|--------|-------|-------|-------|
| Time   | 7-8 AM| 5-6 PM| 7-11 PM|
| Capacity| 7     | 7     | 2     |

Table 6. Results of the Heuristic solution with 791 flights and flight connections

| Solver | Algorithm 1 (Pages 10-11) |
|--------|---------------------------|
| 1      | Number of delayed flights | 130 |
| 2      | Time taken to find solution | 35 seconds |
| 3      | Sum of the delays of all flights (periods) | 690 |
| 4      | Ave. delay (periods) over all flights | 0.872 |
| 5      | Ave. delay (periods) only over delayed flights | 5.308 |

The results are encouraging. More than half of the flights in the data set (470 flights) either have a departing flight waiting for them at Sydney or wait for a flight
arriving at Sydney. The heuristic\(^2\) is still able to provide a feasible solution within 35 seconds.

5. Conclusion and further work. Computing flight delay costs using multiple cost ranges (i.e. non-linear costing) is more realistic than assuming a linear cost model. However as we have observed, using an I.P. solver is not the best computational approach to addressing the issue. In response to this, we developed a greedy heuristic that is fast and returns a reasonably good solution. We tested the model with data from Sydney; it should also be tested with data from other airports.

Even for this simple optimization model not involving more complicated phenomena such as connecting flights or flight cancellations, the 3-range (non-linear) cost I.P. model runs much slower than the single-range (linear) cost model. This indicates that approximation heuristics are necessary to handle non-linear costing.

We tested the models with GLPK software. They should be tested with other optimization software such as GUROBI or CPLEX.

Acknowledgments. I thank Andrew Cook and Graham Tanner (University of Westminster, UK) for providing me with sufficient background information. I also thank David Anderson at Airservices Australia for arrival-departure data at Sydney airport. I should thank the reviewers for providing valuable suggestions.

Funding from the following sources is gratefully acknowledged: (a) Education Department of Jiangxi Province, Grant No. GJJ161113 (2017-2019); and (b) An internal university “Yao-Hu Scholar” research startup grant from the Nanchang Institute of Technology (2015-2018).

REFERENCES

[1] N. Boland, A. Ernst, C. Goh and A. Mees, Optimal two-commodity flows with non-linear cost functions, *Journal of the Operational Research Society*, 46 (1995), 1192–1207.
[2] L. Brunetta, G. Guastalla and L. Navazio, Solving the multi airport ground holding problem, *Annals of Operations Research*, 81 (1998), 271–287.
[3] A. Cook, G. Tanner, V. Williams and G. Meise, Dynamic cost indexing - Managing airline delay costs, *Journal of Air Transport Management*, 15 (2009), 26–35.
[4] A. Cook and G. Tanner, *European Airline Delay Cost Reference Values*, Report from the University of Westminster UK, Eurocontrol, 2015.
[5] A. Cook, G. Tanner and S. Anderson, *Evaluating the True Cost to Airlines of One Minute of Airborne or Ground Delay: Final Report*, Eurocontrol, 2004.
[6] A. Cook, G. Tanner and A. Lawes, The Hidden cost of airline unpunctuality, *Journal of Transport Economics and Policy*, 46 (2012), 157–173.
[7] J. Ferguson, A. Q. Kara, K. Hoffman and L. Sherry, Estimating domestic US airline cost of delay based on European model, *Transportation Research Part C: Emerging Technologies*, 33 (2013), 311–323.
[8] J. A. Filar, P. Manyem, D. M. Panton and K. White, A model for adaptive rescheduling of flights in emergencies (MARFE), *Journal of Industrial and Management Optimization*, 3 (2007), 335–356.
[9] J. A. Filar, P. Manyem, M. S. Visser and K. White, Air traffic management at Sydney with cancellations and curfew penalties, *Optimization and Industry: New Frontiers*, Appl. Optim., Kluwer Academic Publishers, 78 (2003), 113–140.
[10] W. Gao, X. Xu, L. Diao and H. Ding, Simmod based simulation optimization of flight delay cost for multi-airport system, 2008 *International Conference on Intelligent Computation Technology and Automation (ICICTA)*, 1 (2008), 698–702.
[11] A. Gardi, R. Sabatini and S. Ramassamy, Multi-objective optimisation of aircraft flight trajectories in the atm and avionics context, *Progress in Aerospace Sciences*, 83 (2016), 1–36.

\(^2\)However, we did not compare the solution value with that of an optimal solution, since it could take a much longer time (perhaps several days) to find an optimal solution.
Appendix A. Ground holding optimization model.

A.1. Single range (linear cost) model. The NRJ model [15], on which our model is based, is presented below. Time is discretised into intervals (or periods). Each period is typically of 15 or 30 minutes.

We define that \( x_{f,t} = 1 \) if flight \( f \) arrives at or before time period \( t \), and 0 otherwise.

As explained in Sec. 1.2, this is a ground holding model. All flight delays occur on the ground, at the origins of the flights (departing airports), subject to arrival capacity restrictions at the flights’ destination airports.

The objective function is the sum of the delay costs over all flights. For each flight \( f \), its total cost \( c_f \Delta_f \) increases linearly as its delay \( \Delta_f \) increases:

\[
\text{Minimize} \quad \sum_{f \in F} c_f \Delta_f \quad (26)
\]

subject to the following constraints:

\[
- \sum_{f \in F_z} (x_{f,t} - x_{f,t-1}) \geq -K_{z,t} \quad \forall z \in Z, \quad \forall t \in T \quad (27)
\]

\[
x_{f,t} = 0 \quad \forall f \in F, \quad \forall t \in \{1, \cdots, (r_f - 1)\} \quad (28)
\]

\[
x_{f,t} - x_{f,t-1} \geq 0 \quad \forall f \in F, \quad \forall t \in T_f \quad (29)
\]

\[
x_{f,t} = 1 \quad \forall f \in F, \quad \forall t \in \{\lfloor r_f + \Delta_{\text{max}} \rfloor, \cdots, |T|\} \quad (30)
\]

\[
\Delta_f - \sum_{t \in T_f} t(x_{f,t} - x_{f,t-1}) = -r_f \quad \forall f \in F \quad (31)
\]

\[
\sum_{t=r_f}^{T_f} x_{f,t} - \sum_{u=r_g}^{U} x_{g,u} \geq 0 \quad \forall f \in F, \quad \forall \tau \in T_f, \quad \forall g \in S_f
\]

and \( \forall U \in T_g \) such that

\[
U = \tau + \sigma_{f,g} + \theta_g \quad (32)
\]

\[
x_{f,t} \in \{0, 1\} \quad \forall f \in F, \quad \forall t \in T \quad (33)
\]

Interpretation:

- The objective function (26) is the sum of the delay costs of all flights. For each flight, the delay cost is the product of the number of delay periods and the unit delay cost \( c_f \). We want to minimize the total cost of flight delays.
- The constraints define the feasible region in which the optimal solution is to be found.
• Capacity constraints (27) determine a limit on the number of flights that can land at an airport in each time-period.
• Assignment constraints (28-30) respectively prevent a flight from arriving early, force each flight to land at the airport of destination in only one time-interval, and force a flight to land before the maximum delay for that flight expires.
• Constraint (31) defines the total delay $\Delta_f$ for a flight $f$ as the difference of its actual arrival time $\sum_{t \in T_f} t(x_{f,t} - x_{f,t-1})$ and its planned arrival time $r_f$.
• Coupling constraints (32) force the flight to land at a certain time prior to the departure time of its successor.
• Constraint (33) forces the $x_{f,t}$ to be binary variables.

A.2. Three range (non-linear cost) model. This model is the same as the linear cost model above, except that:
(a) The objective function is now the sum of the delay costs over all flights:

$$\text{Minimize } \sum_{f \in F} D_f,$$

where $D_f$ is given by (2) on Page 4, and
(b) the constraints from (2) to (11) should now be added to the model.

Received February 2019; revised October 2019.

E-mail address: prabhu.manyem@gmail.com