Pruning rules for optimal runway sequencing with airline preferences

Geert De Maere [1] and Jason A.D. Atkin [1]
[1] Automated Scheduling, Optimisation and Planning Group (ASAP),
School of Computer Science, The University of Nottingham, Nottingham, UK
{Geert.DeMaere, Jason.Atkin}@nottingham.ac.uk

Proc. ICAOR 2015
Vienna, Austria

Keywords:
Airline preferences
Machine scheduling
Pruning rules
Runway sequencing
Sequence dependent setup times

Abstract

This paper presents a new pruned dynamic programming approach to solve real world runway sequencing problems in which delay cost differentiation between the different aircraft is applied. The pruning rules are based on more generic dominance rules and properties of the objective function that enable to significantly reduce the size of the state space. Optimal results for publically available benchmark instances are reported, and obtained at an extremely low computational cost.

Introduction

Ever increasing air travel and traffic loads result in increased pressure on airport and airspace infrastructure. This results in the need to more efficiently manage airport and airspace operations, which can be achieved through the use of highly innovative decision support systems. This increasing demand is part of what has driven high profile research programs such as the Single European Sky ATM Research project (SESAR) in Europe and NextGEN in the United States. Unfortunately, such decision support systems are highly complex and require sophisticated search/optimisation methodologies to underpin them.

The focus of this paper is on decision support for runway operations. Traditionally, research on airport operations has focussed on solving the key operational problems in isolation. Examples of such problems include runway sequencing, gate assignment, ground movement and baggage handling. More recent work in this field has focussed on integrated approaches that tackle multiple problems simultaneously. For instance, Atkin *et al*. (2013) focus upon integrating the runway and pushback sequencing problem at London Heathrow airport, and thereby explicitly recognise that pushback contention at the stands can compromise the quality and feasibility of the runway sequence. Neuman *et al*. (2014) integrate the ground movement and gate assignment problem and strategically de-conflict the gate assignment solution to avoid contention during the pushback operations. Other examples of integrated approaches include Lee and Balakrishnan (2012), Deau *et al*. (2008), and Keith *et al*. (2008).

The solution methods for the integrated problems are often underpinned by methods to solve the individual models. For instance, Atkin *et al*. (2013) solve the runway sequencing problem as the "master problem" (using a branch and bound algorithm within a rolling window) subject to the existence of a feasible pushback sequence, verified in the "sub problem". The feasible pushback times for the resulting runway sequence are subsequently optimised in a post processing stage. This illustrates the importance of fast and accurate solution methods for the individual sub-problems (including the method presented here) when considering integrated problems.

A second trend in decision support for airport operations is to enable airlines and other airport stakeholders to exert increased influence on the way in which the airport is managed on the day of operation, allowing them to have more influence on, for example, the runway sequence. Historically, fairness between different aircraft and airlines was encouraged by using the same cost function (e.g. for delay) for all aircraft, irrespective of the aircraft type, operator, or the specific flight. Such an approach does not recognise the fact that different aircraft/flights may have different costs associated with them (including delay costs and operational costs such as fuel burn). Hence, airlines may want to prioritise some of their flights in the runway sequence relative to their other flights (subject to fairness between the different operators). Soomer *et al*. (2008) apply delay cost differentiation and present results for proprietary problem instances. Equity between airlines is maintained in their work by giving every airline an equal and proportional "equity budget" that they can allocate between their own flights. Furini *et al*. (2012) apply a similar approach based on cost differentiation (without explicitly considering equity) and use a MIP based rolling window search to sequence the aircraft. The authors present results for publicly available benchmark instances from Milan Airport.

The use of advanced decision support systems in airport operations is gaining momentum since the emergence of Airport Collaborative Decision Making (A-CDM, http://www.euro-cdm.org/). The key idea behind A-CDM is that airports and airport stakeholders (e.g. airlines, ground handlers, air traffic control) share accurate operational information with one another. This facilitates better informed decision making, more efficient use of resources, reduced delays, and reduced environmental impact. The implementation and acceptance of A-CDM systems creates a data rich environment that provides an excellent "playground" for the operational research scientist to develop decision support methodologies, e.g. for runway sequencing.

This paper extends our previously introduced pruned dynamic program for exact runway sequencing to allow for cost differentiation between aircraft. The efficiency of our approach is demonstrated on publicly available benchmarks, which were previously considered to be intractable (Furini *et al*., 2012). The results illustrate the excellent performance of our approach, and demonstrate its ability to generate optimal sequences for the given problem instances within milliseconds.

The outline of our paper is as follows. Section 2 describes the runway sequencing problem considered here, its constraints and the objectives that we considered. The pruning rules that underpin our dynamic program are introduced in Section 3, together with an outline of how they were integrated in the dynamic program itself. Computational results for the benchmark instances presented by Furini *et al*. (2012) are presented in Section 4. We conclude with a summary of the main contributions of our work and the future directions in the final section of this paper.

The Runway Sequencing Problem

Given a set of aircraft denoted by S, the runway sequencing problem aims at finding a sequence s that meets all constraints imposed upon the take-off/landing times of the individual aircraft, and gets an acceptable or, in this case, optimal value for a pre-specified (set) of preferences.

Different variants of the runway sequencing problem are characterised by the number of runways that are considered, the interaction between the runways (i.e. interacting or independent), and the mode in which the individual runways are operated (either segregated or mixed). In segregated mode, the individual runway(s) are used for either arrivals or departures only, whereas in mixed mode the individual runway(s) are used for arrivals and departures simultaneously. If multiple runways are considered and they are interacting, the separations between arrivals/departures on one runway can be influenced by operations at the other runway. Based on the above characteristics, one can distinguish between runway sequencing problems with single or multiple runways, for which the runway(s) are operated in segregated mode, mixed mode, or a combination thereof. In this paper, we focus on single runway operations in segregated mode.

Constraints

Runway Separations: The runway sequencing problem considering arrivals only, departure only, or mixed mode operations is often considered to be similar in the academic literature. Whilst this is true from the perspective of the underlying mathematical model (all can be modelled as machine scheduling problems with asymmetric sequence dependent setup times), in practice, the runway separations for departures are considerably more complex than for arrivals. Let the minimum required runway separation between two arbitrary aircraft i and j (with i preceding j) be denoted by δ_{ij}. In the case of departures, the value of δ_{ij} is determined by the weight class, speed class, and the standard instrument departure route (SID) of aircraft i and j. In the case of arrivals, the value of δ_{ij} is usually determined only by the weight classes of aircraft i and j.

The observation above makes the departure sequencing problem significantly more difficult to solve than the arrival sequencing problem. For instance, in the case of departures for Heathrow Airport, up to 12 different departure routes, 5 different weight classes, and 3 different speed classes may have to be considered at any given point in time, resulting in 180 different aircraft types and 32400 possible combinations. In contrast, in the case of arrivals only, as few as 5 different weight classes have to considered, resulting in only 25 possible combinations. In addition, departure separations do not necessarily satisfy the triangle inequality ($\delta_{ij} + \delta_{jk} \leq \delta_{ik}$ for aircraft i then j then k taking off). I.e., the separation of an arbitrary aircraft k can be influenced by multiple preceding aircraft. This explains why approaches in the literature that are applied to arrival sequencing problems do not necessarily scale well to departure sequencing problems.

In contrast to the sequence-dependent separations for the generic machine scheduling problem, the separations for the runway sequencing problem are usually well structured. Their structure can be understood by considering that larger aircraft, in general, cause more wake turbulence, to which smaller aircraft are more sensitive. In addition, faster aircraft are more likely to catch up with slower aircraft before their routes diverge. Hence, the separation before a fast small aircraft will never be less than the separation for a slow large aircraft on the same route, assuming that both aircraft are added to the end of the same sequence of preceding aircraft. Taking advantage of this structure is key to the research presented in this paper. The separation constraints for an arbitrary sequence s are modelled by Equation 1, in which ts_x and ts_i denote the absolute positions of aircraft x and i in sequence s. Since the separations are asymmetric, resequencing the aircraft by deviating from the first come first served order enables the exploitation of the presence of the asymmetric separations and may result in better sequences with reduced delay.

Take-off/landing time constraints: In addition to runway separations, an aircraft *i*'s take-off/landing time, denoted by t_i, is further constrained by the earliest time that it can reach the runway, denoted by $r_{i,}$, and the start time of any runway usage time windows that may be imposed, denoted by $[ec_i, lc_i]$. A CTOT or Calculated Take-Off Time imposes a 15 departure time-window during which the aircraft should take-off. The aircraft cannot take-off before its start (i.e. ec_i is a hard constraint), and should not violate the end of its CTOT window (i.e. lc_i is a - severely penalised - soft constraint). Equations 1-4 provide a summary of the constraints upon an aircraft *i*'s take-off/landing time.

$$t_x + \delta_{xi} \leq t_i \ \forall x \in s, ts_x < ts_i \tag{1}$$

$$r_i \leq t_i \tag{2}$$

$$et_i \leq t_i \leq lt_i \tag{3}$$

$$ec_i \leq t_i \tag{4}$$

Objectives

The objective function considered in this research is given by Equations 5 – 6, in which ω_1, ω_2, ω_3, and ω_4 denote constants. The elements of Equation 5 model runway utilisation, non-linear delay with cost differentiation, and slot compliance, respectively. Runway utilisation is defined by the take-off time of the last aircraft in the sequence, i.e. the makespan. Delay cost differentiation is implemented through the linear factor W_i and the exponent α_i (assumed to be greater than or equal to 1). If $\alpha_i > 1$ holds, larger delays are penalised more severely, to prefer an equal delay distribution (subject to delay cost differentiation) across all aircraft. The third component models slot compliance as a linear piecewise discontinuous function. We note that objective function used here differs from the one used in De Maere *et al*. (2015) and Atkin *et al*. (2013) by considering delay cost differentiation through the value of W_i. Our objective function reduces to the one used by Furini *et al*. (2012) when ignoring runway utilisation (i.e. the first component) and setting the values of α_i and P equal to 1 and 0, respectively.

$$F(s) = (\max_{i \in s} \ t_i, \sum_{i \in s} (W_i(t_i - r_i)^{\alpha_i} + P \times C(t_i, lc_i))) \tag{5}$$

$$C(t_i, lc_i) = \begin{cases} 0 & if \quad t_i \leq lc_i \\ \omega_1(t_i - lc_i) + \omega_2 & if \quad lc_i < t \leq lc_i + 300 \\ \omega_3(t_i - lc_i) + \omega_4 & if \quad t_i > (lc_i + 300) \end{cases} \tag{6}$$

Approach

A dynamic program is used to solve the problem. The key contribution of our approach is to exploit structural characteristics of the runway sequencing problem and the objective function above (Equation 5) to prune the state space of the dynamic program, making the solution method feasible even for larger, realistic problem instances. A number of alternative objective functions have been used for runway sequencing in the past, such as runway utilisation, delay, and non-linear delay. The characteristics that are exploited are usually generic and apply to a large number of real world problem instances and objective functions. The resulting dynamic program is able to solve problem instances which were previously considered to be intractable to optimality at an extremely low computational cost.

Every state in the state space of our dynamic program corresponds to a partial sequence *s*, and is defined by the set of aircraft it contains, its objective values (Equation 5), and the set of aircraft that may influence the separations (and hence objective values) of the aircraft remaining to be added to *s*. Our dynamic program expands the state space by adding one aircraft at every stage of the algorithm. I.e., a state in stage *n* of the algorithm will contain exactly *n* aircraft. The outline of our dynamic program is provided below, together with an explanation of the dominance rules it uses to prune the search space.

```
Initialise previousStateSpace with a single empty state
Initialise currentStateSpace to be empty
for each state in previousStateSpace
 for each ordered set of cost-separation identical aircraft
  a = first aircraft in the set that is not in s, null if none are left
  if a != null
   if appending a to s will not result in idle time
    if appending a to s will not violate conditional orders
     expand s by adding a, resulting in sNew
     add sNew to currentStateSpace
     check for relaxed dominance in currentStateSpace
    end if
   end if
  end if
 end for
end for
previousStateSpace = currentStateSpace
Initialise currentStateSpace to be empty
```

The dominance rules used by our dynamic program to prune the state space are based on the following observations:

1. Aircraft arrive over time when real world problem instances are considered. I.e., their earliest take-off/landing times, r_i, are distributed over time (not necessarily equally)
2. The individual components in the objective function are monotonic non-decreasing, i.e. delaying an aircraft unnecessarily will not improve its objective value
3. A number of aircraft with similar characteristics are usually present in real world data instances. I.e. belonging to the same weight class, speed class, and following the same departure route after take-off, although there may be a large number of such aircraft groups.

Relaxed Dominance

The presence of sequence dependent separations that do not satisfy the triangle inequality implies that the landing/take-off time (and hence objective values) of the next aircraft may be influenced by multiple preceding aircraft. Two states in a dynamic program may therefore only be comparable if they contain the same set of aircraft, and if the set of separation influencing aircraft at the end of the sequence, otherwise the separations required before later additions to the sequence may differ. It is, however, possible to relax the dominance conditions between states, such that one state can dominate another as long as it contains the same set of aircraft and the take-off times for additions to the dominating sequence will be no later than if the same aircraft is added the dominated sequence. It can be observed that, even if there are many aircraft still being considered for sequencing, many will not be able to reach the runway in time for Equation 1 to be the binding constraint upon their take-off time (instead being constrained by Equation 2). These aircraft do not need to be considered when comparing these states. Dominance can, therefore, by checked by considering only those later aircraft for which Equation 1 is the binding constraint, and a state will be dominated by another sequence containing the same aircraft (and hence can be pruned) if: the total objective value for the aircraft already in the sequence is no better than that for the other sequence; and the take-off times for none of the later aircraft would be any earlier for that sequence than for the dominating sequence.

Orders for Cost-Separation Identical Aircraft

Complete orders: Psaraftis (1980) introduced the presence of sets of separation identical aircraft for total processing cost, in which complete orders can be inferred for the aircraft within the set. Two aircraft *i* and *j* are said to be separation identical if $\delta_{ix} = \delta_{jx}$ and $\delta_{xi} = \delta_{xj}$ for all *x* in *S*. The presence of complete orders within separation identical sets enables a reduction of the sequencing problem to one of interleaving ordered queues (by selecting the first remaining aircraft from the queues). Psaraftis' approach was extended by De Maere *et al.* (2015) to the multi-objective arena, by showing that complete orders can be inferred within sets of separation identical aircraft if a complete order exists for all of the individual objectives. Formal proofs are provided in De Maere *et al.* (2015), showing that this is the case for makespan, delay (without cost differentiation), and hard time when the release times and the end times of the hard time windows (denoted by lt_i and lt_j for aircraft *i* and *j*, respectively) are in order, i.e. $r_i \leq r_j$, $lt_i \leq lt_j$. However, the authors also showed that this is not the case when CTOT compliance is considered. In this case, it may be beneficial for one aircraft to violate its slot more severely to prevent multiple other aircraft from missing their slot (and incurring an additional fixed penalty per slot violation).

The complete orders by De Maere *et al.* (2012) for separation identical aircraft can be extended to separation identical aircraft with cost differentiation. Indeed, all else being equal, complete orders remain valid in the presence of cost differentiation if all aircraft *x* within the individual separation identical sets have the same values for W_x and for α_x, in which case the aircraft are said to be *cost-separation identical*. The presence of cost-separation identical sets enables a reduction in the complexity of the runway sequencing problem from *n*! to $O(N^2(n+1)^N)$, with *N* denoting the number of cost-separation identical sets and *n* the number of aircraft in *S* (Psaraftis, 1980).

Even though some structure is to be expected in real world problem instances, theoretically, every aircraft *x* in the *S* could have a different combination for α_x and W_x. Hence, the number of cost separation identical sets may be significantly higher than the number of separation identical sets without cost differentiation. The use of additional pruning rules, such as the ones described below, therefore becomes increasingly important to improve and maintain tractability of our approach.

Conditional Orders: For the following section, let *i* and *j* denote two separation identical aircraft which have already been sequenced at times t_i and t_j ($t_i \leq t_j$) in the partial sequence *s*. A sequence will dominate an alternative sequence where the positions of *i* and *j* have been reversed if it can be shown that the total cost of sequencing these aircraft in the reverse order will always increase. Since the aircraft are separation identical, the times for these aircraft will be reversed if the positions are reversed, and the times and minimum separations of the other aircraft in *s* will remain unchanged "(assuming that the earliest times for *i* and *j* are non-binding upon the landing/take-off times)".

In De Maere *et al.* (2012), the authors showed that conditional orders could still be inferred between separation identical aircraft without cost differentiation, even if the exact value of t_j is not yet known, such as when incrementally building up a sequence to which aircraft *j* remains to be added. If aircraft *i* and *j* are not subject to a CTOT window, i.e., they have no cost for slot violation, conditional orders can still be inferred when cost differentiation is considered, for instance, when the incremental cost of delaying aircraft *i* exceeds the incremental cost of delaying aircraft *j*. This is the case if $r_i \leq r_j$, $W_i \geq W_j$ and $\alpha_i \geq \alpha_j$ hold. In this case, the delay for any arbitrary time *t* for aircraft *i* is larger than the delay for aircraft *j* since $r_i \leq r_j$. Moreover, since $W_i \geq W_j$ and $\alpha_i \geq \alpha_j$, the delay cost for aircraft *i*'s increases more rapidly than for aircraft *j*. Similarly, if $r_i \leq t_j$ (aircraft *i* is available for sequencing at time t_j), $r_i \geq r_j$ (aircraft *i*'s delay starts counting after aircraft *j*), $W_i \geq W_j$ and $\alpha_i \geq \alpha_j$ (aircraft *i*'s delay cost increases more rapidly than aircraft *j*'s) hold, conditional orders can still be inferred between aircraft *i* and *j* based on the first derivative of the delay cost. The latter is a measure of the increase in delay cost per time unit. Considering the definition of delay cost in Equation 5, both derivatives for *i* and *j* are monotonically non-decreasing and, for $\alpha_i > 1$, have a single break-even point, after which the increase in delay cost per infinitesimal time unit for aircraft *i* becomes dominant. I.e., the cost of delaying aircraft *i* exceeds the cost of delaying aircraft *j* after this point in time. Hence, any sequence in which *i* is sequenced before *j*, and for which the first derivative of *i* at the corresponding time exceeds the value of the first derivative for aircraft *j* cannot be optimal. Hence, the sequence, and any sequence based on it, can be pruned from the search/state space without compromising optimality.

Pruning Intrinsically Bad Sequences

The first property above implies that the earliest times at which aircraft can make it to the runway for landing/take-off is distributed over time, resulting in earliest landing/take-off times that are distributed over time. This property can be exploited to prune sequences that contain idle time which could be used to sequence an aircraft without delaying any of the subsequent aircraft. For example, any sequence where there is a gap which is sufficiently large to insert into it, without delaying any take-off/landing times of other aircraft, an aircraft which has not yet been sequenced, can be pruned, since the cost for this aircraft can be no worse from doing so, due to observation 1 above. This situation is common when sequences are exhaustively evaluated (or pruned), due to the effect of Equation 2. Note that the state may not be otherwise pruned by standard or relaxed dominance rules, since the states containing the same set of aircraft may not (yet) be present in the state space.

Results

The results presented here are for the real world problem instances of Milan Airport introduced by Furini *et al*. (2012). The instances contain 60 aircraft per set and are publically available at http://www.or.deis.unibo.it/research.html. The authors assume the value for α_i to be equal to 1 for all flights, whereas the value of W_i is determined based on the number of seats and the fuel consumption of the aircraft. An excellent description of the problem instances is provided in Furini *et al*. (2012).

Our pruned dynamic program described above was implemented in Java and all experiments were carried out on a standard desktop machine running Windows 7 - 64 bit, Enterprise Edition. A single core of an Intel(R) Core(TM)i7 CPU 950@3.70GHz desktop PC was used in combination with version 6 of Sun's Java(TM) Runtime environment. The results are summarised in Table 1, showing the optimal values for the given benchmark instances. The results were obtained in a total runtime of 64 milliseconds. Comparing the results shown here with the results presented by Furini *et al*. (2012), it can be seen that significant further improvements in solution quality can be obtained over the results reported by the authors. In addition, the results reported here were obtained in an average runtime of just over 5 milliseconds per problem instances, compared to an average runtime of 197 seconds per problem instance reported by Furini *et al*. (2012).

Table 1. Results

Dataset	*Start Time*	*End Time*	*Objective Value*
FPT01	14:55:00	17:32:00	265
FPT02	15:30:00	18:00:00	293
FPT03	15:47:00	18:27:00	255
FPT04	16:14:00	18:47:00	268
FPT05	16:35:00	19:36:00	249
FPT06	14:00:00	16:47:00	167
FPT07	14:32:00	17:08:00	198
FPT08	14:55:00	17:37:00	167
FPT09	15:25:00	18:10:00	183
FPT10	15:55:00	18:45:00	211
FPT11	16:24:00	19:34:00	229
FPT12	16:45:00	20:17:00	207

Conclusions

This paper extends our previously introduced (De Maere *et al*., 2015) pruned dynamic program to airline scheduling with cost differentiation. The key principle behind our approach is the use of more generic dominance rules to prune otherwise non-comparable states, and the use of dominance rules that exploit properties of the objective function that, if violated, compromise the optimality of the current state (corresponding to a partial sequence), and any sequence based on it. The latter rules enable the pruning of states much earlier, even before the dominating sequence is generated. To the best of our knowledge, this is the first paper on runway sequencing to apply such an approach. Results for publically available benchmark instances representing real world scenarios show that our approach is able to obtain optimal sequences at an extremely low computational cost, and is able to generate considerable improvements over previously reported results, both in terms of solution quality and solution times.

References

Atkin, J.A.D., De Maere, G., Burke, E.K., & Greenwood, J.S. (2013). Addressing the pushback time allocation problem at Heathrow Airport. Transportation Science, 47(4), 584-602.

Lee, H., & Balakrishnan, H. (2012). A comparison of two optimization approaches for airport taxiway and runway scheduling. In Digital Avionics Systems Conference (DASC), 2012 IEEE/AIAA 31st (pp. 4E1-1).

Deau, R., Gotteland, J.B., & Durand, N. (2008). Runways sequences and ground traffic optimisation. In ICRAT 2008, International Conference on Research in Air Transportation.

Clare, G.L., & Richards, A.G. (2011). Optimization of taxiway routing and runway scheduling. Intelligent Transportation Systems, IEEE Transactions on,12(4), 1000-1013.

Soomer, M. J., & Koole, G. M. (2008). Fairness in the aircraft landing problem. Proceedings of the Anna Valicek competition, 2008.

Furini, F., Persiani, C. A., & Toth, P. (2012). Aircraft sequencing problems via a rolling horizon algorithm. In Combinatorial Optimization (pp. 273-284). Springer Berlin Heidelberg.

De Maere, G., Atkin, J. A. D. & Burke, E. K. (2015). A pruned dynamic program for optimal runway sequencing. Manuscript submitted for publication.

Neuman, U. M. & Atkin, J.A.D. (2013). Airport Gate Assignment Considering Ground Movement. In Computational Logistics, Lecture Notes in Computer Science 8197, 184-198.