A slot scheduling mechanism at congested airports which incorporates efficiency, fairness and airline preferences

Jamie Fairbrother[1], Konstantinos G. Zografos[1], and Kevin Glazebrook[1]

[1]Centre for Transport and Logistics, Lancaster University

May 29, 2019

A Proofs of Lemmas and Theorems

To prove Proposition 1 we make use the following lemma.

Lemma A.1. *Let $\nu(\mathcal{M})$ denote the optimal solution value for problem (1)–(9) with respect to the set of request series $\mathcal{M}$. If m' is a request series such that $m' \notin \mathcal{M}$ then $\nu(\mathcal{M} \cup \{m'\}) \geq \nu(\mathcal{M})$.*

Proof. It suffices to show that for every feasible solution with respect to $\mathcal{M} \cup \{m'\}$ there is a corresponding feasible solution for the set $\mathcal{M}$ whose objective is no greater than the first solution. Let $\tilde{\mathbf{x}}$ be a feasible schedule with respect to the requests $\mathcal{M} \cup \{m'\}$. Then let $\bar{\mathbf{x}}$ be the restriction of this solution to $\mathcal{M}$. This clearly is a feasible solution for the slot allocation with respect to $\mathcal{M}$ and we have:

$$\sum_{t \in \mathcal{T}} \sum_{m \in \mathcal{M} \cup \{m'\}} f_m^t \tilde{x}_m^t = \sum_{t \in \mathcal{T}} \sum_{m \in \mathcal{M}} f_m^t \bar{x}_m^t + \sum_{t \in \mathcal{T}} f_m^t \bar{x}_{m'}^t.$$

Since $f_{m'}^t \geq 0$ for all $t \in \mathcal{M}$ we therefore must have

$$\sum_{t \in \mathcal{T}} \sum_{m \in \mathcal{M} \cup \{m'\}} f_m^t \tilde{x}_m^t \geq \sum_{t \in \mathcal{T}} \sum_{m \in \mathcal{M}} f_m^t \bar{x}_m^t,$$

as required. □

We now prove Proposition 1.

Proof of Proposition 1. We prove the result for the case of arrival sensitivity. The proof for the case of departure sensitivity is directly analogous. Our proof is via negation, namely we shall demonstrate that if a period is not arrival saturated with respect to some optimal solution for the base problem then it is not arrival sensitive.

Let $\mathbf{x}^*$ be an optimal solution to the base slot allocation problem for the set of request series $\mathcal{M}$. Suppose that under $\mathbf{x}^*$ the period (d', t') is not saturated. Let m' be a new arrival request series with $\mathcal{D}_{m'} = \{d'\}$, $t_{m'} = t'$. It is enough to show that $\nu(\mathcal{M}) = \nu(\mathcal{M} \cup \{m'\})$ where as above, we use $\nu(\cdot)$ for the optimal value of the base problem with the set of request series $\cdot$. It will then follow that the period (d', t') is not arrival sensitive which will conclude the proof.

By Lemma A.1 we have that $\nu(\mathcal{M}) \leq \nu(\mathcal{M} \cup \{m'\})$ so it is enough to construct a feasible solution for the base problem with solution value $\nu(\mathcal{M})$. We construct a new feasible solution $\tilde{\mathbf{x}}$ for the base slot allocation problem with respect to $\mathcal{M} \cup \{m'\}$ as follows:

$$\tilde{x}_m^t = \begin{cases} x_m^{t*} & \text{for all } m \in \mathcal{M},\ t \in \mathcal{T} \\ 1 & \text{for } m = m',\ t = t_{m'} \\ 0 & \text{otherwise.} \end{cases} \tag{1}$$

We now show that $\tilde{\mathbf{x}}$ is a feasible solution for the base slot allocation problem with respect to $m \in \mathcal{M} \cup \{m'\}$.

The feasibility of the solution $\mathbf{x}^*$ with respect to the request series $\mathcal{M}$ immediately implies the assignment, turnaround and binary constraints hold for $\tilde{\mathbf{x}}$ with respect to $\mathcal{M} \cup \{m'\}$. The addition of

an arrival request will also not affect the departure capacity constraints, and so it only remains to verify the arrival and total capacity constraints still hold.

Let $\mathbf{y}^*$ and $\tilde{\mathbf{y}}$ represent the aggregate schedules corresponding to $\mathbf{x}^*$ and $\tilde{\mathbf{x}}$ respectively. Note that by (1) we have the following relation:

$$\tilde{y}^A_{dt} = \begin{cases} y^{*A}_{dt} + 1 & \text{for } d = d',\ t = t_{m'}, \\ y^{*A}_{dt} & \text{otherwise.} \end{cases} \tag{2}$$

Given that the aggregate number of arrivals only changes for the period (d', t'), we only need to consider arrival capacity constraints which involve aggregate allocations for this period. For constraints of duration $c \in \mathcal{C}$, the affected constraints are those with date index $d = d'$, and time index $t \in [\max\{t' - c + 1, 0\}, t']$. But since the period (d', t') is not saturated with respect to $\mathbf{x}^*$, we have that

$$\min_{t \in [max(t'-c+1,0),t']} \left(\alpha^{d't}_c - \sum_{s=t}^{t+c-1} y^{*A}_{d's} \right) \geq 1.$$

Hence, for all $c \in \mathcal{C}$ and $t \in [\max(t' - c + 1, 0), t']$ we have that

$$\begin{aligned} \alpha^{d't}_c - \sum_{s=t}^{t+c-1} \tilde{y}^A_{d's} &= \alpha^{d't}_c - \sum_{s=t}^{t+c-1} y^{*A}_{d's} - 1 \\ &\geq 0. \end{aligned}$$

Hence, all arrival capacity constraints hold for $\tilde{\mathbf{x}}$. It can be similarly be shown that all total capacity constraints still hold for $\tilde{\mathbf{x}}$ and so we can conclude that $\tilde{\mathbf{x}}$ is a feasible solution for the slot allocation with respect to the set of request series $\mathcal{M} \cup \{m'\}$.

It now remains to evaluate the objective function value for $\tilde{\mathbf{x}}$:

$$\begin{aligned} \sum_{m \in \mathcal{M} \cup \{m'\}} \sum_{t \in \mathcal{T}} f^t_m \tilde{x}^t_m &= \sum_{m \in \mathcal{M}} \sum_{t \in \mathcal{T}} f^t_m \tilde{x}^t_m + \sum_{t \in \mathcal{T}} f^t_{m'} \tilde{x}^t_{m'} \\ &= \sum_{m \in \mathcal{M}} \sum_{t \in \mathcal{T}} f^t_m x^{t*}_m \qquad \text{since } f^{t_{m'}}_{m'} = 0 \\ &= \nu(\mathcal{M}) \end{aligned}$$

as required. □

The following is a counter-example to the converse of Proposition 1.

Example A.2. *Suppose we have a set of two requests $\mathcal{M} = \{m_1, m_2\}$ which must be scheduled for a single day, for both requests we have $t_{m_1} = t_{m_2} = t$ and there is a total capacity limit of one for each time period.*

An optimal solution for this problem would be to displace m_2 to time $t + 1$. This is shown on the left-hand side of Figure 1. Time $t + 1$ is saturated with respect to this optimal solution. However, as is shown on the right-hand side of Figure 1, an extra request m' such that $t_{m'} = t + 1$ can be scheduled for $t + 1$ without increasing the total displacement, and so time $t + 1$ is not sensitive.

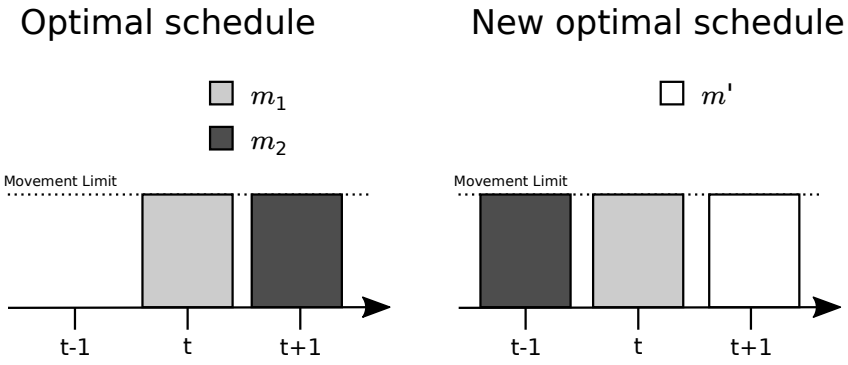

Figure 1: Counter-example to converse of Proposition 1

The following proof is for Proposition 2 which relates to the satisfiability of fairness constraints.

Proof of Proposition 2. The proof is by contradiction. Suppose there exists a movement $m_1 \in \widetilde{\mathcal{M}}$ such $\tilde{x}_{m_1}^{t_{m_1}} \neq 1$, that is, where the request m_1 is displaced. Consider the case where m_1 is an arrival, and there exists a departure $m_2 \in \mathcal{M}$ such that $(m_1, m_2) \in \mathcal{P}$. Define a new schedule as follows:

$$\hat{x}_m^t = \begin{cases} x_m^{t*} & \text{if } m \notin \{m_1, m_2\}, \\ 1 & \text{if } m = m_1 \text{ and } t = t_{m_1}, \\ 1 & \text{if } m = m_2 \text{ and } t = t_{m_2}, \\ 0 & \text{otherwise.} \end{cases}$$

By assumption (i) this schedule satisfies turnaround constraints. The periods (d, t_{m_1}) are off-peak for all $d \in \mathcal{D}_{m_1}$ since $r_a = 0$, and by assumption (ii) the periods (d, t_{m_2}) are off-peak for all $d \in \mathcal{D}_{m_2}$. Hence, the movements m_1 and m_2 can be rescheduled at t_{m_1} and t_{m_2} respectively without breaking any capacity constraints, and so this new schedule is feasible. However, this schedule has smaller total displacement than (x_m^{t*}) which contradicts the fact that it is an optimal solution to the base model. The case where m_1 is a departure can be similarly shown to yield contradictions. □

B Minimum Insertion Displacement

We describe in this appendix how the minimum insertion problem (24)–(30) can be solved efficiently by enumerating over pairs of time intervals for the arrival and departure slots.

In particular, we enumerate possible arrival-departure time intervals. To feasibly allocate a pair of arrival-departure slots, this pair must satisfy turnaround time constraints, and the allocation of the additional slots must not break any of the capacity constraints.

Turnaround constraints can be ensured by restricting the enumeration to pairs slots which satisfy these. Arrival and departure capacity constraints can be efficiently verified by using saturation indicators. On the other hand, the total capacity constraints must be verified by explicitly checking all affected constraints. Algorithm 1 below uses these ideas to efficiently calculate the minimum insertion displacement of a request.

input : $\tilde{\mathbf{y}}$ feasible aggregate schedule, (m_1, m_2) request pair, arrival, departure and total capacities α, β, γ, and capacity durations $\mathcal{C}$
output: $B_{m_1 m_2}$ minimum insertion displacement

1 $B_{m_1 m_2} \leftarrow 2 * T$;
2 **foreach** $t \in \mathcal{T}$ **do** $a_t \leftarrow \bigvee_{d \in \mathcal{D}_{m_1}} A_t^d(\tilde{\mathbf{y}})$;
3 **foreach** $t \in \mathcal{T}$ **do** $d_t \leftarrow \bigvee_{d \in \mathcal{D}_{m_2}} D_t^d(\tilde{\mathbf{y}})$;
4 **foreach** $T_a \in t_{m_1}, \ldots, T - \underline{l} - 1$ **do**
5 **if** a_{T_a} **then** continue;
6 **if** $|t_{m_1} - T_a| > B_{m_1 m_2}$ **then** break;
7 **foreach** $T_D \in T_a + \underline{l}, \ldots, \min\{T - 1, T_A + \bar{l}\}$ **do**
8 **if** d_{T_D} **then** continue;
9 **if** $|T_A - t_{m_1}| + |T_D - t_{m_2}| < B_{m_1 m_2}$ **then**
10 **if** check_insertion$(T_A, T_D, \mathcal{C}, \gamma, \tilde{\mathbf{y}})$ **then**
11 $B_{m_1 m_2} \leftarrow |T_A - t_{m_1}| + |T_D - t_{m_2}|$;
12 **if** $T_D \geq t_{m_2}$ **then**
13 break;
14 **foreach** $T_a \in t_{m_1} - 1, \ldots, 0$ **do**
15 **if** a_{T_a} **then** continue;
16 **if** $|T_a - t_{m_1}| > B_{m_1 m_2}$ **then** break;
17 **foreach** $T_D \in T_A + \underline{l}, \ldots, \min\{T_A + \bar{l}, T - 1\}$ **do**
18 **if** d_{T_D} **then** continue;
19 **if** check_insertion$(T_A, T_D, \mathcal{C}, \gamma, \tilde{\mathbf{y}})$ **then**
20 $B_{m_1 m_2} \leftarrow |T_A - t_{m_1}| + |T_D - t_{m_2}|$;
21 **if** $T_D \geq t_{m_2}$ **then**
22 break;
23 **return** $|\mathcal{D}_{m_1}| B_{m_1 m_2}$

Algorithm 1: Calculate minimum insertion displacement

This algorithm begins by calculating in line 2 a Boolean indicator which tells us for which time periods we can allocate an extra arrival slot without breaking any capacity constraints. This is done by checking the arrival peak indicator for each day the arrival request applies. Similarly, a feasibility indicator is constructed for the departure request series in line 3. In lines 4-13, we search through all pairs of arrival and departure slot times where the arrival slot time is greater than or equal to the request arrival time t_{m_1} for the pair with the smallest feasible displacement. We loop over all pairs of time intervals with feasible turnaround time. The order of pairs of arrival and departure times searched is such that the displacement is strictly increasing when $T_D \geq t_{m_2}$. This means that once a feasible pair of slots is found and $T_D \geq t_{m_2}$, we can break out of the search loop. For each of these pairs, we check that the slots can be allocated without breaking the arrival and departure constraints by using the feasibility indicators calculated in lines 2 and 3. Finally, if these are feasible, and the pair of slots has small displacement than the current best, we check whether the the pairs of slots can be allocated without breaking capacity constraints by using the check_insertion function which is presented in Algorithm 2. This is the most computationally expensive part of the procedure which is why it is the final feasibility check we carry out. Once all pairs with arrival time greater than or equal to t_{m_1} have been searched, we search the remaining possible pairs of requests in lines 14-22. This search is carried out in the same way as in lines 4-12.

input : Arrival and departure times T_A and T_D, capacity constraint durations $\mathcal{C}$, total capacities γ, aggregate schedule $\tilde{\mathbf{y}}$

output: True or False

foreach $c \in \mathcal{C}$ **do**
  if $T_D - T_A| \geq c$ **then** continue;
  foreach $t \in \max\{0, T_D - c + 1\}, \ldots, T_A$ **do**
    if $\sum_{s=t}^{t+c-1} y_{\tilde{d}s} + 2 > \gamma_c^{dt}$ **then return** False;
return True

Algorithm 2: Check whether arrival and departure slots can be allocated to a request pair without breaking any total capacity constraints