Airport Slots Allocation in Ground Delay Programs

Ken C. Ho† Alexander Rodivilov‡

April 15, 2017

Abstract

This paper proposes a new mechanism to allocate landing slots in Ground Delay Program (GDP). We argue the current mechanism does not respect property rights over slots that are owned by airlines before a GDP starts. In our model, the core is not a subset of the Pareto set. The proposed mechanism respects property rights, selects outcomes from the intersection of the core and the Pareto set, and it is strategy-proof. The mechanism reduces to the “You request my house - I get your turn” mechanism (Abdulkadiroglu & Sönmez (1999)) under certain conditions.

JEL: C78, D47, D82, L93, L98, P14, R41.

Keywords: Slots allocation, Strategy-proofness, Mechanism design

---

*We are thankful for the helpful comments from Quan Wen, Yoram Barzel, and participants in UW Microeconomics seminar.

†Department of Economics, University of Washington, Seattle. E-mail: kh89@uw.edu

‡Department of Economics, University of Washington, Seattle. E-mail: arodivil@uw.edu
1 Introduction

Weather causes about 30% of all flight delays in the United States.\(^1\) During thunderstorms, snow, or dense fog that reduced visibility, arrival rate of flights at the affected airport must be reduced. Although landing schedules are made in advance, these unpredictable conditions may lead to reallocations of landing slots among airlines.\(^2\) The reduction of the arrival rate gives more time to land a plane, but it also requires the airport to reconfigure its landing schedule. This paper proposes a mechanism to accomplish this goal.

In United States, the Federal Aviation Administration (FAA) implements ground delay programs to reallocate landing slots. When adverse weather is forecasted (typically hours in advance), FAA declares a GDP is in effect, where the duration of the GDP is also specified (usually several hours). In a GDP, aircraft departing from a airport in the contiguous U.S. and Canada to the affected airport are assigned delayed departure times at their origin airport (while the aircraft are still on the ground).\(^3\)

There is a 2-step procedure to carry out the reallocation of landing slots. The first step is to assign slots to airlines. The current algorithm is called Ration-by-Schedule (RBS). The RBS algorithm orders flights by increasing original scheduled time of arrival, and then assign slots sequentially, that is, the first flight to the first available GDP slot, the second flight to the second available GDP slot, etc.\(^4\) RBS may assign slots to flights that have been canceled or delayed and consequently can not use their assigned slot. The airlines can adjust their schedule by substitutions and cancellations, but flight cancellations and delays may create vacant slots in the landing schedule.

To utilize these new resources, the second step is to reassign these vacant slots to airlines that can use them. In this reassignment step, airlines report their relevant information such as feasible arrival time of its flights and cancellations to a centralized mechanism. Such mechanism then uses the reported information to exchanges slots among airlines to produce a new landing schedule. The current mechanism is called Compression. Essentially, when an airline can not feasibly use a slot, the algorithm exchanges it with another slot that is owned by some airline that can feasibly use it. This algorithm moves flights up in the schedule to fill those vacant slots.\(^5\)

Schummer & Vohra (2013); Schummer & Abizada (forthcoming) show that the Com-

---


\(^2\)A slot at an airport is essentially a time interval that allows an airline to land its plane.

\(^3\)This is in Section 9 of the Facility Operation and Administration at https://www.faa.gov/regulations_policies/orders_notices/index.cfm/go/document.information/documentID/1028577.

\(^4\)The OAG schedule is considered to be the original schedule in the industry.

\(^5\)Flights will not be moved down in the Compression algorithm. For more details, see Schummer & Vohra (2013) or Vossen & Ball (2006a).
pression mechanism produces outcomes that are Pareto-efficient but might be not in the core. The Compression mechanism solicits feasible arrival times and cancellations, and these are private information of the airlines. They also show that Compression is manipulable by flight delay, slot destruction and postpone flight cancellation.\textsuperscript{6} Moreover, it does not respect a form of property rights after a GDP starts. By contrast, the mechanism we propose, Multiple Trading Cycle (MTC), produces outcomes that are not only Pareto-efficient but also in the core; furthermore, it is strategy-proof; that is, it does not suffer from manipulations.

Before RBS was adopted, the FAA used a mechanism called Grover-Jack, which assigns slots based on feasible departure times reported by the airlines. Feasible departure times are private information of the airlines. Suppose a flight needs to delay its departure for 1 hour due to mechanical issues. If a ground delay program (with Grover-Jack) is implemented in which the flight would be delayed for another hour, then its total delay would be 2 hours. This is known as the "Double Penalty." If the airline withhold the information, this flight would have been assigned a slot 1 hour earlier, which it can feasibly use in this example. As a result, airlines would intentionally withhold information to avoid double penalties, and this might lead to some slots go unused even they could be used by other airlines. To avoid such disincentive, the RBS uses originally scheduled times of arrival instead of reported feasible departure times to allocate slots.

The MTC also makes use of the information contained in the original schedule but in a different way. We argue that RBS does not respect a form of property rights before a GDP starts. Slots of different lengths are different objects. A GDP converts original slots into GDP slots, but such conversion is just a re-division of time intervals. Under RBS, owning an early original slot gives the airline an early GDP slot, which is not the same object it has at the beginning while such time interval (of the GDP slot) might be entirely owned by another airline before the start of a GDP. In contrast to RBS, the MTC endows such type of GDP slots to the airlines that own these time intervals entirely before the start of a GDP.

Airlines have lexicographic preferences in our model, in which each airline has an importance ranking of its flights. Based on this ranking, the MTC minimizes the expected delays for each airline lexicographically, i.e., for each airline, it minimizes the expected delay most important flight then minimizes the expected delay for the second most important flight, and so on.\textsuperscript{7}

\textsuperscript{6}The three strategies corresponds to report later feasible arrival times, hide a cancellation to "destroy" a vacant slot, and hide a vacant slot but then retrieve it for consumption in later a reassignment step.

\textsuperscript{7}RBS minimizes the delay for each flight in a lexicographic order. It is easy to see it minimizes the delay for the first flight then minimizes the delay for the second flight, and so on. Vossen & Ball (2006a) show RBS algorithm lexicographically minimize the maximum delay with respect to the original schedule. Their formulation is different but the intuition is similar.
2 Related Literature

The two papers most related to ours are Schummer & Vohra (2013); Schummer & Abizada (forthcoming). Both papers take RBS outcomes as initial endowments and focus on the reassignment step. In Schummer & Vohra (2013), the preferences are incomplete, and they propose a Trading Cycle (TC) mechanism, which is a variant of the Top Trading Cycle mechanism. In Schummer & Abizada (forthcoming), the preference domain is larger than ours since they allow airlines to put weights on flights, while lexicographic preference assume such weight of a flight is infinitesimal compared to the weight of a more important flight; they propose a mechanism called Deferred Acceptance with Self Optimization (DASO). DASO is not Pareto efficient, but it is not manipulable by slot destruction and postpone flight cancellation, while it is still manipulable by flight delay. By contrast, our mechanism achieves full strategy-proofness and Pareto efficiency in a smaller domain.

Another related paper is Abdulkadiroğlu & Sönmez (1999). Indeed, when (i) each airline owns at most one flight and (ii) the number of flights equals to number of slots, our model degenerates to the housing allocation with existing tenants model, and the MTC reduces to the “You request my house- I get your turn (YRMH-IGYT)” mechanism (with random orderings) in that paper. Given (i) and (ii), the model reduces to a house allocation problem (Hylland & Zeckhauser (1979)) when no airline owns a GDP slot initially, and the MTC reduces to random serial dictatorship (RSD). In addition, given (i) and (ii), the model becomes a house market (Shapley & Scarf (1974)) when every slot is owned by an airline, and the MTC reduces to Top Trading Cycle.

Konishi et al. (2001) also generalize the house market. In their model, multiple types of indivisible goods are traded. They show the core may be empty and there is no Pareto efficient, individually rational, and strategy-proof mechanism. In our context, we obtain positive results on these properties.

In the transportation literature on GDP, optimization is the main focus. Vossen (2002) proposes a “proportional random assignment method,” in which each flight is entitled to an equal share of each slot it can use. Balakrishnan (2007) uses the house market model by treating flights as agents. These two papers do not take airlines’ incentives into account. Ball et al. (2010) propose an algorithm called Ration-by-Distance that assigns slots to flights based on distance.

The rest of the paper is organized as follows: Section 3 introduces the model; Section 4 illustrates the mechanism; Section 5 shows properties of the mechanism; Section 6 discusses subsequent reassignments; and Section 7 concludes the paper. All proofs are in Appendix.

---

8See Vossen & Ball (2006a,b); Bard & Mohan (2008); Ball et al. (2009); Glover & Ball (2013).
8.1, and a summary of properties is in Appendix 8.2.

3 Model

There is a finite set of airlines $A = \{a_1, a_2, \ldots, a_{|A|}\}$ and a finite set of controlled flights $F^o = \cup_{a \in A} F^o_a = \{f_0, f_1, \ldots, f_{|F^o| - 1}\}$, where the $F^o_a$ denotes all flights owned by airline $a$. Some flights might be canceled during or before the start of GDP; we use $F \subseteq F^o$ to denote the set of non-canceled flights and $F_a \subseteq F^o_a$ to denotes all flights owned by airline $a$ that are not canceled. There is a set of initial slots $S^o = \{s^o_0, s^o_1, \ldots, s^o_{|L|-1}\}$, where the length of each slot is normalized to one unit of time. Note that $|F^o|$ of the $|L|$ original slots were owned by some airlines. Let the set of available GDP slots be $S = \{s_0, s_1, \ldots\}$, where the length of each slot is $l > 1$ unit of time. For $n = 0, 1, \ldots$, slot $s_n$ is the time interval $[nl, (n + 1)l]$. We use $s_n$ to represent $nl$ on the time line.

There is an earliest feasible arrival time $e_f \in S$ for each flight $f \in F$; each flight $f$ can be feasibly assigned to slot $n$ only if $e_f \leq s_n$. Let $e = (e_f)_{a \in F} \in \mathbb{R}^{|F|}$ be the vector of all earliest feasible arrival times. A landing schedule is an injective function $\Pi : F \rightarrow S$. Let $\mathcal{M}$ be the set of all landing schedules. A landing schedule $\Pi$ is feasible if $\forall f \in F, \Pi(f) \geq e_f$. A landing schedule $\Pi$ is non-wasteful if $\not\exists f \in F$ and $s \in S$ such that $\Pi^{-1}(s) = \emptyset$ and either $\Pi(f) = \emptyset$ or $e_f \leq s < \Pi(f)$.

An initial landing schedule is an injective function $\Pi^o : F \rightarrow S^o$. Given some initial landing schedule $\Pi^o$, one can infer slot $\Pi^o(f)$ is initially endowed to some airline $a$ such that $f \in F^o_a$. Let $\Phi(A, S') : A \rightarrow 2^{S'}$ be a slot ownership function such that $a \neq a' \implies \Phi(a, S') \cap \Phi(a', S') = \emptyset$, where $S' \subseteq \{S^o, S\}$ and $\Phi(a, S')$ is the set of slots owned by airline $a$. $\Phi(A, S')$ is consistent with $\Pi$ if $\forall a \in A, \forall f \in F_a$, $\Pi(f) \in \Phi(a, S')$. A pair $(\Pi, \Phi(A, S'))$ satisfying this consistency condition is an assignment. Given initial assignment $(\Pi^o, \Phi(A, S^o))$, the set of initial slots owned by airline $a$ is $\Phi(a, S^o)$.

---

9A flight is “controlled” means it is included in the GDP program.

10There is a single runway in our model, but our results can be extended to a model with multiple runways, in which there are multiple slots available at a time.

Note that available GDP slots do not have to be adjacent since there are exempted flights in GDP (for examples, international and airborne flights). The number of arrivals an airport can accept each hour is called Airport Acceptance Rate/Airport Arrival Rate (AAR). If 1 unit of time is 1 minute and the AAR during the GDP is 30, then $l = \frac{AAR \times \text{unit of time}}{30} = \frac{30}{60} \times 1 = 2$.

11Following the literature, we call $e_f$ the earliest feasible arrival time for $f$, but strictly speaking, $e_f$ is the earliest feasible arrival slot for $f$. 

6
3.1 Preferences

We assume each $a \in A$ has an importance ranking of its flights.\footnote{We consider an importance ranking at least partially reflect the economic values of flights.} Formally, let $R_a$ be a strict total order over $F_a$; we interpret $R_a$ as $a$’s importance ranking. If $f \in F_a$ is more important than $f' \in F_a$, we write $fR_a f'$. Given $F_a$ and $R_a$, let $e_a = (e_{f_a,1}, e_{f_a,2}, ..., e_{f_a,|F_a|}) \in \mathbb{R}^{F_a}$ be a vector of earliest feasible arrival times such that $f_{a,i} R_a f_{a,i+1}$ for $i \in \{1, ..., |F_a|\}$. Let $R = (R_a)_{a \in A}$ be the importance ranking profile.

Airline $a$’s preference over landing schedules is induced by $R_a$ and $e_a$. All else being equal, airline $a$ prefers flight $f \in F_a$ with $e_f$ to land as early as possible. Given a landing schedule $\Pi \in \mathcal{M}$, we define the \textit{delay} for $f$ by $d_f(\Pi) = s_f - e_f$, where $s_f$ is the slot assigned to $f$, and if $s_f = 0$ or $s_f < e_f$, the we set $d_f(\Pi) = M$, where $M$ is a sufficient large positive number such that $M > s - e_f$ for all $s \in S$ and $e_f \in T$.\footnote{Note that this delay is respected to $e_f$ but not its original slot $s_f$.}

For all landing schedules $\Pi$ and $\Pi'$, airline $a$ lexicographically prefers $\Pi$ to $\Pi'$ if and only if the first non-zero coordinate of $x_a = (x_1, x_2, ..., x_{|F_a|})$ is positive, where for $i \in \{1, ..., |F_a|\}$ and $f_{a,i} R_a f_{a,i+1}$, $x_i = d_{f_a,i}(\Pi') - d_{f_a,i}(\Pi)$, and we write $\Pi \succ_a \Pi'$\footnote{Lexicographic preference does not rule out an airline prefers an infeasible landing schedule to a feasible landing schedule.}. Conversely, if the first non-zero coordinate of $x_a$ is negative, it prefers $\Pi'$ to $\Pi$. If airline $a$ is indifferent between $\Pi$ and $\Pi'$, we write $\Pi \sim_a \Pi'$; this will happen when all coordinates of $x_a$ equal to 0, which will only happen when $\Pi_a = \Pi'_a$, where $\Pi_a : F_a \to \Phi(a, S)$ is a partial landing schedule for $a$. Since airlines only care about their own flights, we will also use $\succeq_a$ to compare partial landing schedules for $a$.

A \textit{schedule lottery} is a probability distribution over the set of all landing schedules $\mathcal{M}$. Let $\Delta \mathcal{M}$ denote the set of all schedule lotteries. We denote a schedule lottery by $\mathcal{L} = \sum p_\Pi \cdot \Pi$ where $p_\Pi \in [0, 1]$ is the probability weight of landing schedule $\Pi$ and $\sum_\Pi p_\Pi = 1$. We now extend an airline’s preference to allow it to compare schedule lotteries. Given a schedule lottery $\mathcal{L} \in \Delta \mathcal{M}$, the \textit{expected delay} for $f$ is $d_f(\mathcal{L}) = \sum_\Pi p_\Pi \cdot (s_f - e_f)$. For all schedule lotteries $\mathcal{L}$ and $\mathcal{L}'$, $\mathcal{L} \succeq_a \mathcal{L}'$ if and only if the first non-zero coordinate of $x_a = (x_1, x_2, ..., x_{|F_a|})$ is positive, where for $i \in \{1, ..., |F_a|\}$ and $f_{a,i} R_a f_{a,i+1}$, $x_i = d_{f_a,i}(\mathcal{L}') - d_{f_a,i}(\mathcal{L})$; other cases are the same as above.

Let $\succeq = (\succeq_a)_{a \in A}$ be the preference profile of all airlines. An \textit{instance} of an airport slots allocation problem is a 6-tuple $I = (S, A, F, R, e, \Phi^o)$. 
4 The Mechanism

In an instance \( I = (S, A, F, R, e, \Phi^o) \), \( S, A, F, \Phi^o \) are fixed, and only \( R \) and \( e \) will be reported by the airlines. Denote the set of strict total order over \( F_a \) by \( \mathcal{R}_a \). Recall \( e_a \in \mathbb{R}_{\mathcal{F}_a}^+ \). The strategy space for \( a \in A \) is therefore \( \mathcal{R}_a \times \mathbb{R}_{\mathcal{F}_a}^+ \).

A schedule mechanism \( \varphi : (R, e) \rightarrow \mathcal{M} \) is a mapping that selects a landing schedule for every strategy profile \( (R, e) \). Let \( \varphi_f(R, e) \) be the slot that is assigned to \( f \), and \( \varphi_a(R, e) \) be the partial landing schedule for \( a \). A lottery mechanism \( \phi : (R, e) \rightarrow \Delta \mathcal{M} \) is a mapping that selects a schedule lottery for every strategy profile \( (R, e) \).

Let \( S_a = \{ s_n \in S | s_n = [nl,(n+1)l] \subseteq \bigcup_{a \in \Phi(a,S^o)}[n,n+1] \} \) be the set of available GDP slots that their time intervals are entirely owned by airline \( a \) before the GDP starts. For \( A' \subseteq A \), let \( S_{A'} \equiv \cup_{a \in A'} S_a \). Given a set \( A \), an ordering is a bijective function \( z(A) : \{1,2,\ldots,|A|\} \rightarrow A \). Let \( Z(A) \) be the set of orderings with image \( \{1,2,\ldots,|A|\} \). Obviously, \( |Z(A)| = A! \).

Our Multiple Trading Cycle Mechanism employs the following algorithm to produce a landing schedule for each input.

According to \( \Phi^o \), construct \( S_a \) for each \( a \in A \). For each \( a \in A \), creates \( |F^o_a| \) surrogates of \( a \), name them \( a(1),a(2),\ldots,a(|F^o_a|) \). Denote the set of surrogates by \( \mathcal{A} \). Randomly select an ordering \( z(A) \) with uniform distribution over \( Z(A) \). For each \( a \), rearrange its surrogates to their positions in the ordering such that they are in the order of \( a(1), a(2), \ldots, a(|F^o_a|) \). Denote the resulting ordering \( z \).

Pre-competition Stage (allocation of non-scarce resources):

Set \( F^{0-0} = F \) and \( e = e^{0-0} \).

(i) Use the smallest element of \( e^{0-0} \) to identify the earliest slot \( s \in S \) that is demanded, eliminates all slots that are earlier than \( s \), update \( S \) to \( S^{0-0} \).

(ii) If the earliest slot can be used by only one flight \( f \in F^{0-0} \), assign the slot to this flight, remove the last surrogate of its owner from \( z \). If the earliest slot can be used by multiple flights from a single airline \( a \), assign the slot to the most important flight among these flights, remove the last surrogate of \( a \) from \( z \). Update \( S^{0-t} \) to \( S^{0-(t+1)} \), \( F^{0-t} \) to \( F^{0-(t+1)} \) and \( e^{0-t} \) to \( e^{0-(t+1)} \) for \( t = 0, 1, \ldots \).

(iii) Repeat (i) and (ii) until the earliest slot is demanded by more than one airlines.

Denote the resulting sets \( S^1 \) and \( F^1 \). Remove flights \( F \setminus F^1 \) from \( R \) accordingly; denote the resulting ranking profile \( R^1 \).

Main Stage (allocation of scarce resources):

Let \( a(i) \) represents the \( i \)th important flight of airline \( a \) according to \( R^1 \) or a dummy flight.
if \( a \) has no more flight at some step.\(^{15}\) If \( S_a \cap S^1 \) is non-empty, endow the set of slots to \( a \).

**Step 1** - Without lost of generality, let \( a(1) \) be the first flight in \( z \).

(i) If \( a(1) \) is a dummy flight, go to the next step.

Otherwise, let \( a(1) \) pick the earliest slot in \( S^1 \) such that it can feasibly use it.

(ii) If it picks an empty slot or a slot endow ed to it, \( s' \), go to the next step.

(iii) If it demands some slot \( s' \) that is endow ed to some other airline \( b \), and \( b \) has a remaining flight in \( F^1 \), modify \( z \) by inserting \( b(1) \) in front of \( a(1) \). If \( b \) has no remaining flight, go to the next step. If a cycle forms, it is formed by the remaining most important flights and slots that are owned by their airlines. Each of these flights requests a slot that owned by the airline that is next in the cycle. For each \( a \in A \), let \( s_a \) be some slot in \( S_a \cap S^1 \).

A cycle in step 1 is an ordered list \( (s_a, a(1), s_b, b(1), ..., s_y, y(1)) \) of slots and flights where \( a(1) \) demands \( s_b, ..., y(1) \) demands \( s_a \). Remove all flights in the cycle by assigning them the slots they demand. Go to the next step.

Denote the resulting sets \( S^2 \) and \( F^2 \).

**Step \( n \geq 2 \)** - Without lost of generality, let \( a(i) \) be the next flight in line.

(i) If \( a(i) \) is a dummy flight, go to the next step.

Otherwise, let \( a(i) \) pick the earliest slot in \( S^n \) such that it can feasibly use it.

(ii) If it picks an empty slot or a slot endow ed to it, \( s' \), go to the next step.

(iii) If it demands some slot \( s' \) that is endow ed to some other airline \( b \), and \( b \) has a remaining flight in \( F^n \), modify \( z \) by inserting \( b(j) \) in front of \( a(i) \), where \( b(j) \) is \( b \)'s remaining most important flight. If \( b \) has no remaining flight, go to the next step. For each \( a \in A \), let \( s_a \) be some slot in \( S_a \cap S^n \). If a cycle \( (s_a, a(i), s_b, b(j), ..., s_y, y(k)) \) forms, where \( b(j), ..., y(k) \) are the remaining most important flights of the corresponding airlines. Remove all flights in the cycle by assigning them the slots they demand. Go to the next step.

Denote the resulting sets \( S^{n+1} \) and \( F^{n+1} \).

The main stage stop when \( F^k = \emptyset \) for some \( k \geq 1 \).

**Supplemental Stage:**

Let \( V^1 \) be the set of vacant slots. If the earliest vacant slot was in some \( S_a \) and \( a \) has a dummy flight, assign it to \( a \). Otherwise, assign it to the dummy flight with the highest order in \( z \). Denote the resulting set \( V^2 \). Repeat until there is no remaining dummy flight.

This algorithm stops within in \( |F^0| \) steps. Note that two types of slots are conditionally treated as empty slots in the mechanism: slots endowed to an airline yet they can be used by only one airline (possibly different form the one owns it) and slots endowed to an airline but at the steps they are being assigned, the airline has no flight remaining. We refer those slots

\(^{15}\)We use \( a(i) \) but not \( f_{a,i} \) since \( i \)th important flight in \( R^1_a \) might be different from \( i \)th important flight in \( R_a \).
as empty slot as well if no confusion arises. There will always be feasible slots for flights, yet they might be late and outside the GDP time window. \( V^1 \) includes vacant slots that are not assigned in previous steps.

An airline \( a \)'s probability of getting the first position in \( z \) is \( \frac{|F_a^o|}{\sum_{b\in A} |F_b^o|} \), and as it gets the first position, the probability of getting the second position declines to \( \frac{|F_a^o| - 1}{|F_a^o| - 1 + \sum_{b\in A \setminus a} |F_b^o|} \). \( F_a^o \) includes the flights that are canceled; otherwise, there is incentive for an airline to hide its cancellations.

4.1 An Example

Example 1:

Table 1: Initial Slots Allocation

<table>
<thead>
<tr>
<th>initial slot ( s^o \in S^o )</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flight</td>
<td>( f_1 )</td>
<td>( f_2 )</td>
<td>( f_3 )</td>
<td>( f_4 )</td>
<td>( f_5 )</td>
<td>( f_6 )</td>
<td>( f_7 )</td>
<td>( f_8 )</td>
<td>( f_9 )</td>
<td>( f_{10} )</td>
</tr>
<tr>
<td>Owner</td>
<td>( c )</td>
<td>( b )</td>
<td>( a )</td>
<td>( a )</td>
<td>( b )</td>
<td>( b )</td>
<td>( a )</td>
<td>( c )</td>
<td>( c )</td>
<td>( a )</td>
</tr>
<tr>
<td>( e_f \in S )</td>
<td>-</td>
<td>-</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>4</td>
<td>6</td>
</tr>
<tr>
<td>Rank</td>
<td>-</td>
<td>-</td>
<td>1</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>4</td>
<td></td>
</tr>
</tbody>
</table>

\* “-” stands for canceled.

Suppose the realized \( z = a(1), a(2), a(3), c(1), b(1), a(4), b(2), c(2), b(3), c(3) \). Then in the pre-competition stage, since \( s_0 \) can be used by \( f_8 \) only, \( f_8 \) is assigned this slot and \( c(3) \) is removed from \( z \). \( s_1 \) is demanded by more than 1 airline now, so the main stage starts. In step 1, \( a(1) \) will pick \( s_1 \) for \( f_3 \). In step 2, \( a(2) \) points to \( s_2 \) for \( f_7 \). Since it is owned by \( b, b(1) \) will be inserted in front of \( a(2) \), and \( b(1) \) picks \( s_2 \) for \( f_6 \). In step 3, now \( a(2) \) picks \( s_3 \) for \( f_7 \). In step 4, \( a(3) \) picks \( s_4 \) for \( f_4 \). In step 5, \( c(1) \) picks \( s_5 \) for \( f_9 \). In step 6, \( a(4) \) picks \( s_6 \) for \( f_{10} \). In step 7, \( b(2) \) picks \( s_7 \) for \( f_5 \). Since all non-canceled flights got an slot, the supplement step starts. \( c(2) \) and \( b(3) \) will get \( s_9 \) and \( s_{10} \), respectively.

Table 2: GDP Slots Allocation \((l = 2)\)

<table>
<thead>
<tr>
<th>GDP slot ( s \in S^o )</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>( s \in S_c )</td>
<td>( a )</td>
<td>( b )</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RBS+Comp.</td>
<td>( f_8 )</td>
<td>( f_5 )</td>
<td>( f_3 )</td>
<td>( f_4 )</td>
<td>( f_6 )</td>
<td>( f_7 )</td>
<td>( f_9 )</td>
<td>( f_{10} )</td>
<td>( c )</td>
<td>( b )</td>
</tr>
<tr>
<td>MTC</td>
<td>( f_8 )</td>
<td>( f_3 )</td>
<td>( f_6 )</td>
<td>( f_7 )</td>
<td>( f_4 )</td>
<td>( f_9 )</td>
<td>( f_{10} )</td>
<td>( f_5 )</td>
<td>( b )</td>
<td>( c )</td>
</tr>
</tbody>
</table>

In this example, \( s_1 \in S_a \) but \( f_3 \) of \( a \) cannot use \( s_1 \) under the current GDP, while \( e_{f_3} \leq s_1 = 2 \). Also, the current GDP outcome is not efficient since \( f_7 R_a f_4 \) and \( e_{f_7} < \Pi(f_4) < \Pi(f_7) \).
5 Properties of the Mechanism

A schedule mechanism \( \varphi \) is regular if for any strategy profile \( (R, e) \), \( \varphi(R, e) \) is part of an assignment. That implies the induced ownership function \( \Phi^{\varphi(R,e)}(A, S) \), a by product of the mechanism, is consistent with \( \varphi(R, e) \). A schedule mechanism \( \varphi \) is feasible (non-wasteful) if for any strategy profile \( (R, e) \), \( \varphi(R, e) \) is feasible (non-wasteful).

A landing schedule \( \Pi \) is Pareto efficient if \( \exists \Pi' \) such that (i) \( \forall a \in A, \Pi' \succ_a \Pi \), and (ii) \( \exists a \in A, \Pi' \succ_a \Pi \). The set of Pareto efficient landing schedules is the Pareto set. A schedule mechanism \( \varphi \) is Pareto efficient if for any strategy profile \( (R, e) \), \( \varphi(R, e) \) is Pareto efficient.

A schedule mechanism or lottery mechanism is strategy-proof if truth-telling is a dominant strategy in its induced revelation game.

Let \( \Pi_a^{S'} \) be a partial landing scheduled for \( a \) such that \( \Pi_a^{S'} \succ_a \Pi'_a \) for all \( \Pi'_a \) given \( F_a \) and \( S' \subseteq S \). Construction of \( \Pi_a^{S'} \): Order \( S' \) in increasing order, assign the earliest feasible slot to \( a \)'s most important flight, then assign the earliest feasible slot to \( a \)'s second most important flight.... Repeat until there is no more slot or no more flight.

Let \( \Phi^{\varphi(R,e)}(a, S) \equiv S_a' \), so \( S_a' \) is the set of available GDP slots that are owned by airline \( a \) after a GDP starts. Let \( S_a^{(i)} \in \{S_a, S_a'\} \).

A landing schedule \( \Pi_a \) is individually rational (with respect to \( S_a^{(i)} \)) if \( \forall a \in A, \Pi_a \succ_a \Pi_a^{S_a^{(i)}} \). A schedule mechanism \( \varphi \) is individually rational if for any strategy profile \( (R, e) \), \( \varphi(R, e) \) is individually rational.

A landing schedule \( \Pi \) is in the core (with respect to \( S_a^{(i)} \)) if \( \exists \Pi' \) and \( A' \subseteq A \) such that (i) \( \forall f \in \cup_{a \in A'} F_a, \Pi'(f) \in S_a^{(i)} \), and (ii) \( \forall a \in A', \Pi' \succ_a \Pi \). A schedule mechanism \( \varphi \) is core-selecting if for any strategy profile \( (R, e) \), \( \psi(R, e) \) is in the core.

A schedule mechanism or lottery mechanism respects property rights (with respect to \( S_a^{(i)} \)) if \( \forall s \in S_a^{(i)} \), in the first step that \( s \) is being assigned, \( a \) has the right to (i) use it or (ii) trade it for a better alternative.

A lottery mechanism \( \phi \) is ex post individually rational if it gives positive probability to only individually rational landing schedules. A lottery mechanism \( \phi \) is ex post Pareto efficient if it gives positive probability to only Pareto efficient landing schedules. Other ex post properties are defined analogously.

**Proposition 1:** The multiple trading cycle mechanism \( \phi \) is ex post regular, ex post feasible and ex post non-wasteful.

Regularity here means the mechanism creates a landing schedule and an assignment

---

16 The core defined by weak domination might be empty. For example, when an airline with an GDP slot cancels all of its flights and such GDP slot is demanded by more than 1 airline, the core defined by weak domination is empty.
Proposition 2: The multiple trading cycle mechanism $\phi$ is ex post Pareto efficient.

In a house market, the core is a subset of the Pareto set since for any outcome that is in the core, the grand coalition can not block it. But in our context, the core is not a subset of the Pareto set since $S_A \neq S$; in another word, there are more houses than agents. A landing schedule in the core is not necessary Pareto efficient if some airline can benefit by having a slot that is not endowed to another airline.

Proposition 3: The multiple trading cycle mechanism $\phi$ is ex post individually rational (with respect to $S_a$).

If an airline $a$ only uses its own slots in $S_a$, $\phi$ will produce $\Pi_a^{S_a}$, and if it uses any other slots, $\phi_a(R, e)$ will be preferred to $\Pi_a^{S_a}$.

Proposition 4: The multiple trading cycle mechanism $\phi$ respects property rights (with respect to $S_a$).

The following example show a Pareto efficient landing schedule might not be in the core.

Example 2 (A Pareto efficient landing schedule might be not in the core):

<table>
<thead>
<tr>
<th>Table 3: Initial Slots Allocation</th>
</tr>
</thead>
<tbody>
<tr>
<td>initial slot $s^o \in S^o$</td>
</tr>
<tr>
<td>Flight</td>
</tr>
<tr>
<td>Owner</td>
</tr>
<tr>
<td>$e_f \in S$</td>
</tr>
<tr>
<td>Rank</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table 4: GDP Slots Allocation $(l = 2)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>GDP slot $s \in S$</td>
</tr>
<tr>
<td>$s \in S_{(i)}$</td>
</tr>
<tr>
<td>$\Pi$</td>
</tr>
<tr>
<td>$\Pi'$</td>
</tr>
</tbody>
</table>

$\Pi$ is Pareto efficient, but it is not in the core since $a$ can use slots only from $S_a$ and be better off.

Theorem 1: The multiple trading cycle mechanism $\phi$ is ex post core-selecting (with respect to $S_a$).

According to Theorem 1 and Proposition 4, the MTC selects landing schedules from the intersection of the core and the Pareto Set.

Theorem 2: The multiple trading cycle mechanism $\phi$ is strategy-proof.

When (i) each airline owns at most one flight and (ii) no airline owns a GDP slot initially, MTC reduces to RSD; if we fix the ordering $z$, it further reduces to serial dictatorship.
However, when (ii) holds but not (i), MTC with fixed ordering is different from serial dictatorship. In this context, airlines are agents, so serial dictatorship would allow airline $a$ to pick all slots it wants, then allow airline $b$ to pick all slots it wants (among the remaining slots), etc. It is well-known serial dictatorship is strategy-proof, and the reason is that an airline does not need to manipulate its report to get the best set of slots that is available, but MTC with fixed ordering does not have this feature. We illustrate this in the following example.

**Example 3** (MTC with fixed ordering is not strategy-proof):
Consider a case where (ii) holds but not (i). Fix an ordering $z = a(1), b(1), a(2), b(2)$.

<table>
<thead>
<tr>
<th>Initial Slots Allocation</th>
<th>$s^o \in S^o$</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flight</td>
<td>$f_1, f_2, f_3, f_4$</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Owner</td>
<td>$a, b, a, b$</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$e_f \in S$</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>Rank</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>$\hat{e}_f \in S$</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>$\hat{R}$</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>GDP Slots Allocation</th>
<th>$s \in S$</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\varphi^z(R, e)$</td>
<td>$f_1, f_2, f_3, f_4$</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\varphi^z(\hat{R}<em>a, e_a, (R, e)</em>{-a})$</td>
<td>$f_1, f_3, f_2, f_4$</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\varphi^z(R_a, \hat{e}<em>a, (R, e)</em>{-a})$</td>
<td>$f_3, f_1, f_2, f_4$</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

In this example, by either misreporting its importance ranking or earliest feasible arrival times, $a$ can gain by having $s_2$ (in the later case, it can swap slots for $f_1$ and $f_2$).

There are two sources of the strategy-proofness. The first one is the design of the pre-competition stage. If an airline knows one of the slot can be used by one of its flight only, say the most important one, then it will have the incentive to misreport its ranking such that this flight is the least important one. By doing so, all of its remaining flights will be better off if we run the MTC without the pre-competition stage. By eliminating the last surrogate of an airline when it gets a slot in this stage, the MTC avoids the aforementioned manipulation.

The second one is the randomness introduced in the main stage together with a feature of lexicographic preference that it does not sacrifice the benefit of a flight for the benefit of a less important flight. By truth-telling, the MTC lexicographically minimizes the expected delay for each airline. But if an airline deviates, it might reduce the expected delay of some of its flight, but the expected delays for a more important flight will increase. If the lexicographic
preference assumption is relaxed, the same result might still be obtain if the size of market goes to infinite.

6 Subsequent Reassignment

In the current GDP, the initial assignment is created by running the Ration-by-Schedule algorithm and possible Compression algorithm. But as airlines continuously update their information, the Compression algorithm may be run multiple times.

The MTC creates a landing schedule by making the initial endowment and exchanging slots among airlines at the same time, but it also can be used to perform reassignments. Reassignment mechanism $\psi$:

Run the MTC with $R^{updated}$ and $e^{updated}$ with $S'_a$ in place of $S_a$.

**Corollary 1:** The reassignment mechanism $\psi$ is ex post individual rational, ex post core-selecting and respects property rights (with respect to $S'_a$).

Corollary 1 follows from the proofs of Proposition 3, 4 and Theorem 1 by putting $S'_a$ in place of $S_a$. It is easy to see unless an airline can use some empty slot that is not endowed to another airline, it has to trade some of its slot for another one. Vacant slots obtained from the supplemental stage might become valuable in the next reassignment step as airlines can used them to trade.

7 Conclusion

This paper studies the slots allocation problem in Ground Delay Programs. In particular, when inclement weather strikes an airport, landing schedule has to be reconfigured, as each flight will require more time to land. Some flights have to be postponed, but such postponements might be too costly to airlines, and so the airlines might cancel those flights. Cancellations of flights create vacant slots on the landing schedule, which are new resources to be reallocated. This creates challenges in matching flights with newly created slots. We argue the currently used GDP does not respect property rights of slots both before and after the GDP starts.

The mechanism we proposed solicits private information such as the earliest feasible arrival times and an importance ranking from the airlines. Based on the information, the mechanism produces outcomes from the intersection of the core and the Pareto set. The MTC also respects property rights, and is strategy-proof before and after the initial allocation.
References

Vossen, Thomas, & Ball, Michael. 2006a. Optimization and mediated bartering models for ground delay programs. *Naval Research Logistics (NRL)*, 53(1), 75–90. 5, 7, 8

Vossen, Thomas WM, & Ball, Michael O. 2006b. Slot trading opportunities in collaborative ground delay programs. *Transportation Science*, 40(1), 29–43. 8
8 Appendix

8.1 Omitted Proofs

**Proof of Proposition 1:** Ex post regularity: This is by construction of the mechanism.
Ex post feasibility: For any ordering $z$, at each step, no flight gets an infeasible slot. Ex post non-wastefulness: This is also by construction of the mechanism. If a flight $\exists f \in F$ such that $s \in V^1$ with $e_f < s$, then it must be the case that $\phi_f(R, e) < s$. ■

**Proof of Proposition 2:** For any ordering $z$, flights that leave at the pre-competition stage are already getting the earliest slot they can get, and no slot in $S \setminus S^1$ can be used to make flights leave at later steps better off.

Consider the main stage, any flight that leaves at step 1 is assigned its top choice that is available and cannot be made better off. Any flight that leaves at step 2 is assigned its top choice that is available among those slots remaining at Step 2 and since slots are distinct time intervals, it cannot be made better off without hurting some flight who left at Step 1. Proceeding in a similar fashion, no flight can be made better off without hurting some flight that left at an earlier step.

Moreover, for an airline, a flight left at an earlier step is more important than a flight left later, so it can not make itself better off as well. Therefore, $\phi$ is ex post Pareto efficient. ■

**Proof of Proposition 3:** For any ordering $z$, let $\varphi^z$ be the induced schedule mechanism. Let $x_i = d_{f_{a,i}}(\Pi_a^S) - d_{f_{a,i}}(\varphi^z_a(R, e))$ for $i \in \{1, ..., |F_a|\}$ and $f_{a,i}Raf_{a,i+1}$.

If $x_i = 0$ for all $i \in \{1, ..., |F_a|\}$, then $\Pi_a^S = \varphi^z_a(R, e)$. Otherwise, let $x_j$ be the first non-zero coordinate of $x_a = (x_1, x_2, ..., x_{|F_a|})$. $x_j$ will always be positive since airline $a$ picks $\varphi^z_{f_{a,j}}(R, e)$ instead of $\Pi_a^S(f_{a,j})$, which means in either the pre-competition stage or the main stage, it picks the empty slot $\varphi^z_{f_{a,j}}(R, e)$ or trades a slot in $S_a$ for $\varphi^z_{f_{a,j}}(R, e)$.

Hence, $\forall a \in A, \varphi^z_a(R, e) \succ_a \Pi_a^S$. ■

**Proof of Proposition 4:** For any ordering $z$, at each step an slot $s$ in some $S_a$ is being assigned, if it is not assigned to a flight of $a$, that means the airline trades it for some better slot. In particular, in the pre-competition stage, the airline trades it for some better slot in previous step of this stage or trades an infeasible slot for some feasible slot for some flight in $F_a$ (it might get such slot in some later step or later stage). In the main stage, the airline trades it for some better slot for its remaining most important flight. In the supplemental stage, the airline trades it for some better slot in the previous stages or previous step of this stage.

**Proof of Theorem 1:** For any ordering $z$, let $\varphi^z$ be the induced schedule mechanism. Suppose $\exists \Pi'$ and $A' \subseteq A$ such that (i) $\forall f \in \cup_{a \in A'} F_a$, $\Pi'(f) \in S_{A'}$, and (ii) $\forall a \in A'$, $\Pi' \succ_a \varphi^z(R, e)$. Therefore, $\forall a \in A'$, the first non-zero coordinate of $x_a = (x_1, x_2, ..., x_{|F_a|})$ is
positive for \( i \in \{1, ..., |F_a|\} \), \( f_{a,i} R_a f_{a,i+1} \) and \( x_i = d_{f_{a,i}}(\varphi^z_{\phi_a}(R, e)) - d_{f_{a,i}}(\Pi') \).

Consider \( f_{a,i} \) where \( x_i \) is the first non-zero coordinate of \( x_a \) for \( a \in A' \). Note that \( \Pi'(f_{a,i}) \in S_{A'} \) is better than \( \varphi^z_{f_{a,i}}(R, e) \), and \( \Pi'(f_{a,i}) \) is not available when \( f_{a,i} \) is picking a slot in \( \varphi^z \). There is a \( \Pi'(f_{a,i}) \) for each \( a \in A' \); let \( S_T \) be the collection of \( \Pi'(f_{a,i}) \) for all \( a \in A' \). \( S_T \) is the set of slots that makes airlines in \( A' \) prefer \( \Pi' \).

(i) If \( a \) is the owner and \( \Pi'(f_{a,i}) \) is used by some \( f_{a,j} \) in \( \varphi^z \), then it must be \( f_{a,j} R_a f_{a,i} \). Since \( x_i \) is the first non-zero coordinate, \( x_j = 0 \), i.e., \( f_{a,j} \) is getting the same slot under \( \Pi' \), a contradiction.

The same argument applies to all airline in \( A' \). Therefore, \( \forall a \in A' \), \( \Pi'(f_{a,i}) \) is coming from some airline \( a' \in A' \) with \( a' \neq a \).

Let \( s_a \in S_T \subseteq S_a \cap S^1 \) endowed to airline \( a \in A' \) be the first slot being assigned to some \( f \in F \) in \( \varphi^z \). \( a \) will pick a slot for its remaining most important flight \( f_{a,j} \) before \( s_a \) is assigned (either it is \( a(j)'s \) turn or \( a(j) \) is being inserted at the top of \( z \)). At this step, all slots in \( S_T \) are available (otherwise it contradicts the way we pick \( s_a \)).

(ii) If \( f_{a,i} = f_{a,j} \), \( a \) picks \( \varphi^z_{f_{a,i}}(R, e) \) but not \( \Pi'(f_{a,i}) \), a contradiction.

(iii) If \( f_{a,i} R_a f_{a,j} \), this means \( \Pi'(f_{a,i}) \) is still available after \( f_{a,i} \) picked a slot in \( \varphi^z \), a contradiction.

(iv) If \( f_{a,j} R_a f_{a,i} \), we have \( \varphi^z_{f_{a,j}}(R, e) = \Pi'(f_{a,j}) \in S_{A'} \).

By (i), \( f_{a,j} \) picks some slot other than \( s_a \). That means airline \( a \) trades \( s_a \) for \( \varphi^z_{f_{a,j}}(R, e) \in S_{A'} \) from some airline \( b \in A' \).

(*) Let \( \varphi^z_{f_{b,j}}(R, e) \) be the slot obtained by \( b \) in this trade. Because all slots in \( S_T \) are still available, the flight \( f_{b,j} \) is more important than \( f_{b,i} \), so \( \varphi^z_{f_{b,j}}(R, e) = \Pi'(f_{b,j}) \in S_{A'} \). If \( \varphi^z_{f_{b,j}}(R, e) \in S_a \) we have a cycle.

If \( \varphi^z_{f_{b,j}}(R, e) \in S_c, c \in A' \) will be the next airline in line of the trade, and the argument (*) applies. Because none of the airline in line gets a slot outside \( S_{A'} \) and \( A' \) is finite, there must exist a cycle contains exclusively airlines in \( A' \). Let \( y \in A' \) be the airlines gets \( s_a \) for \( f_{y,j} \). Recall \( \Pi'(f_{y,i}) \) is not available when \( f_{y,i} \) is picking a slot in \( \varphi^z \). Since all slots in \( S_T \) are still available, \( f_{y,j} \) is more important than \( f_{y,i} \) and therefore \( \varphi^z_{f_{y,j}}(R, e) = \Pi'(f_{y,j}) = s_a \). This contradicts the fact \( s_a \) makes some airline in \( A' \) prefers \( \Pi' \).

Proof of Theorem 2: For any ordering \( z \), let \( \varphi^z \) be the induced schedule mechanism. We consider each stage to see if an airline \( a \) can be better off by misreporting \( R_a \) or \( e_a \), that is, \( \varphi^z(R_a, e_a, (R, e)_{-a}) \succ_{a} \varphi^z(R, e) \). Let \( x_a = (x_1, x_2, ..., x_{|F_a|}) \) be a vector such that for \( i \in \{1, ..., |F_a|\} \), \( f_{a,i} R_a f_{a,i+1} \) and \( x_i = d_{f_{a,i}}(\varphi^z(R, e)) - d_{f_{a,i}}(\varphi^z(R_a, e_a, (R, e)_{-a})) \). Let \( x_j \) be the first non-zero coordinate of \( x_a \). If there is no such \( x_j \), then \( \varphi^z(R_a, e_a, (R, e)_{-a}) \sim_{a} \varphi^z(R, e) \), and we are done. Suppose not. This implies \( e_{f_{a,j}} \leq s = \varphi^z_{f_{a,j}}(R_a, e_a, (R, e)_{-a}) < \varphi^z_{f_{a,j}}(R, e) \) for some \( s \in S_a \). Note that \( s \) is not used by a flight of \( a \) that is more important than \( f_{a,j} \).
since that would contradict \( x_j \) is the first non-zero coordinate of \( x_a \).

In the pre-competition stage, misreporting \( R_a \) will not give \( a \) an extra slot, but misreporting \( e_a \) might. The only possible way for airline \( a \) to get a slot that is feasible for \( f_{a,j} \) (i.e. \( e_{f_{a,j}} \leq s \)) but not being assigned in this stage under \( \varphi^z \) is to make it non-scarce resource such that \( s \) will be assigned to \( f_{a,j} \). Note that \( \varphi^z_{f_{a,k}}(\overline{R_a, e_a, (R, e)} \_a) = \varphi^z_{f_{a,k}}(R, e) \) for all \( f_{a,k}R_a f_{a,j} \).

Case 1: \( s \) is the earliest demanded slot in the main stage.

Since \( s \) is demanded by \( f_{a,j} \) and another airline, there is nothing \( a \) can do to make \( s \) non-scarce resource.

Case 2: \( s \) is not the earliest demanded slot in the main stage.

\( a \) can make \( s \) non-scarce resource only when the following happens:

The earliest demanded slot in the main stage \( s' \) is demanded by \( a \) with \( f_{a,x}, f_{a,x'}, ..., \), where \( f_{a,j}R_a f_{a,x}, f_{a,x'}, ... \) and at most one airline \( b \). Then \( a \) can misreport \( e_{f_{a,x}}, e_{f_{a,x'}}, ... \) (infeasible or later) to give that slot to airline \( b \) (call the flight that gets it \( f_{b,x} \)). For the subsequent slots, there can be 2 cases:

Case 2A: It is demanded by \( a \) with \( f_{a,y}, f_{a,y'}, ..., \) where \( f_{a,j}R_a f_{a,y}, f_{a,y'}, ..., \) and at most one airline \( c \) (it can be airline \( b \) as well), then \( a \) can misreport \( e_{f_{a,y}}, e_{f_{a,y'}}, ... \) to give that slot to airline \( c \) (call the flight that gets it \( f_{c,x} \)).

Case 2B: It is demanded by \( a \) with \( f_{a,z}, f_{a,z'}, ... \) (these flights can be more important or less important than \( f_{a,j} \)), and \( a \) will have this slot assigned to the most important flight among \( f_{a,z}, f_{a,z'}, ... \).

At the end, \( s \) must be in case B in which it is demanded by \( a \) with \( f_{a,j}, f_{a,j'}, ..., \) where \( f_{a,j}R_a f_{a,j'}, ..., \) and \( s = \varphi^z_{f_{a,j}}(\overline{R_a, e_a, (R, e)} \_a) = \varphi^z_{f_{a,j}}(R, e) \). This contradicts \( x_j \) is the first non-zero coordinate of \( x_a \) (\( f_{a,j} \) would get \( s \) from the main stage of \( \varphi^z \) because there are sufficient slots to accommodate the only competitors \( f_{b,x}, f_{c,x}, ... \) and \( f_{a,z}, f_{a,z'}, ... \) that are more important than \( f_{a,j} \) before \( s \) is being picked. \( f_{b,x}, f_{c,x} \) will always try to get a slot earlier than \( s \) and flights \( f_{a,x'}, ..., f_{a,y'}, ... \) that are less important than \( f_{a,j} \) will not compete with them before \( f_{a,j} \) gets a slot in the main stage.).

Suppose the \( a \) wants to manipulate the pre-competition to help \( f_{a,j} \) to get a slot in the main stage. The only thing it might manipulate is \( S^1 \). By sacrificing some flights \( f_{a,x}, f_{a,x'}, ... \) that are less important than \( f_{a,j} \), it might shrink the size of \( S^1 \) by giving slots away as described above, or it might enlarge the size of \( S^1 \) by announcing an infeasible \( \overline{e_{f_{a,x}}} \), some slot \( \varphi^z_{f_{a,x}}(R, e) = \overline{e_{f_{a,x}}} \) becomes a scarce resource. But as long as \( z \) is fixed, such manipulations do not work:

Let \( \varphi^z_{f_{a,j}}(\overline{R_a, e_a, (R, e)} \_a) = \varphi^z_{f_{c,x}}(R, e) = s \), where by misreporting earliest feasible arrival times of flights that are less important than \( f_{a,j} \), a shrinks or enlarges \( S^1 \). Note that
$f_{c,x}$ (with $e_{f,c,x} \leq s$) ranks higher than $f_{a,j}$ in $z$. If $s$ is going to $f_{a,j}$, it means $f_{c,x}$ must be getting an earlier slot $s'$ (Shrinking $S^1$ by giving slots away will only help but not hurt flights of other airlines, and enlarging $S^1$ will not force $f_{c,x}$ to get a later slot too since $f_{a,x}$ will not compete with $f_{b,x}$ before $f_{a,j}$ picks a slot); if $s'$ is going to $f_{c,x}$, it means a flight $f_{d,x}$ which ranks higher than $f_{c,x}$ must be getting an earlier slot $s''$, but this contradicts the finiteness of $|F_a|$.

In the main stage, if $a$ wants to manipulate the outcome, it can misreport (i) $R_a$; (ii) $e_a$; or (iii) both.

There is no way to change the ranking of a single flight, and change a flight’s earliest feasible arrival time cannot help the flight itself. So $a$ must use a subset of its flights to help another subset.

Note that there is no way to use less important flights to help $a_j$ as they always pick later than $a_j$. Eventually, there is only 1 channel to improve its outcome: use flights to help flights that are less important than them; that is, use $f_{a,i}, f_{a,i}; ... \in F_{a,I}$ to help $f_{a,j}, f_{a,j}'; ... \in F_{a,J}$, where $f_{a,i}R_{a}f_{a,i}', ... R_{a}f_{a,j}R_{a}f_{a,j}', ...$

(i)' If $a$ misreports $R_a$, in $\varphi^z(R_a, e_a, (R, e)_{-a})$, $f_{a,i}$ will pick later than itself in $\varphi^z(R, e)$. Let $f_{a,x}$ be the flight that takes $f_{a,i}$'s position. It is without loss of generality to assume $e_{f_{a,i}} \neq e_{f_{a,x}}$. Otherwise, we can exclude $f_{a,i}$ from $F_{a,I}$ as it effectively picks a slot for itself since $a$ can always self-optimize at the end.

(ii)' If $a$ misreports $e_a$, $f_{a,i}$ will pick some slot for a less important flight and some less important flight will pick a slot for it, then $a$ can self-optimize given the slots it gets by misreporting.

(iii)' If $a$ misreports both, one of the two cases above must happen.

Note that in all circumstances, $\varphi^z_{a,i}(R, e)$ is the best possible slot $f_{a,i}$ can get under $z$. In (i)', (ii)', and (iii)', $a$ will pick a slot for $f_{a,i}$ strictly later than it would under $\varphi^z(R, e)$. Also, in the main stage, each slot is demanded by more than one airline. Since $z$ is arbitrary, that means there exists some realization such that $\varphi^z_{a,i}(R, e)$ would be picked by some other airline.

Sum up, the probability for $f_{a,i}$ to get an earlier slot is 0, while the probability of getting a latter slot is bounded away from 0. Let $L(\phi(R, e))$ be the schedule lottery induced by the Multiple Trading Cycle Mechanism if $(R, e)$ is reported. We have $d_{f_{a,i}}(L(\phi(R_a, e_a, (R, e)_{-a}))) > 0$, and this means $a$ prefers to report $R_a$ and $e_a$ truthfully.

In the supplemental stage, if $f_{a,j}$ wants a slot that is being assigned in this stage, it will get that slot in the previous stages, a contradiction.
### 8.2 Summary of properties

<table>
<thead>
<tr>
<th></th>
<th>Current</th>
<th>TC</th>
<th>DASO</th>
<th>MTC (fixed ordering)</th>
<th>MTC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Size of domain</td>
<td>-</td>
<td>small*</td>
<td>Large</td>
<td>Medium</td>
<td></td>
</tr>
<tr>
<td>Pareto efficiency</td>
<td>no**</td>
<td>Yes</td>
<td>no</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Core</td>
<td>no</td>
<td>Yes</td>
<td>no</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Flight delay</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>Yes</td>
</tr>
<tr>
<td>Slot destruction</td>
<td>no</td>
<td>no</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Postpone flight cancellation</td>
<td>no</td>
<td>no</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Ex post property right</td>
<td>no</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Ex ante property right</td>
<td>no (because of Ration-by-Schedule)</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>

* Preferences are incomplete as not every pair of landing schedules is comparable.

** No in our preference domain, but yes in the domain of Schummer & Vohra (2013).