

Assignment of Arrival Slots

James Schummer* Rakesh V. Vohra†

January 6, 2012

Abstract

Inclement weather forces the FAA to reassign airport landing slots. Industry participants agree that reassignment procedures should satisfy incentives and property rights conditions. While mentioned in existing literature, we appear to be first in rigorously defining such conditions. We show that the FAA's Compression Algorithm is incentive compatible, but fails to guarantee a form of property rights. This is significant since the motivation for introducing Compression a decade ago included incentives and property rights. Next we show that an alternative mechanism does satisfy the conditions. Though this model is distinct, the mechanism has the flavor of Top-trading Cycle variants of Papai (2000) and Abdulkadiroğlu and Sönmez (1999). Finally, both mechanisms may fail to provide the incentive to vacate unusable landing slots.

When inclement weather strikes an airport, the airport's landing schedule must be reconfigured. First, the allowed arrival rate of flights is reduced, creating a shortage of resources (arrival slots). Even after these slots are reallocated, subsequent flight cancelations and delays, made by airlines themselves, free up some of these slots, resulting in a situation of *excess* resources (newly vacated landing slots). The efficient reallocation of these excess resources requires airlines to report certain information to a centralized planner (e.g. the Federal Aviation Administration (FAA) in the U.S.), who uses it to arrive at a final reassignment of flights to landing slots. This environment yields a matching problem that has been studied from an operations and transportation perspective, but has been less studied from the perspective of formal mechanism design.

*Kellogg School of Management, MEDS Department, Northwestern University. Email: schummer@kellogg.northwestern.edu.

†Kellogg School of Management, MEDS Department, Northwestern University. Email: r-vohra@kellogg.northwestern.edu.

What makes this problem especially relevant to such analysis is agreement by both the FAA and the airlines that mechanisms (algorithms) used for slot reallocation should satisfy a particular set of *attributes*. Aside from the obvious goal of efficiency, this set of attributes includes ideas related to airlines’ incentives, and to the respect of property rights that reflect an airline’s “ownership” of assigned landing slots. In fact, it is well-documented (see Section 1) that the algorithms currently in use in the U.S. are the result of an historic redesign intended to achieve these attributes.

Our goals in this paper are to formalize such attributes, and use them in evaluating landing slot reassignment mechanisms. What we find sheds an interesting light on the degree to which the algorithm currently used by the FAA achieves these aforementioned objectives. First, under suitable definitions of the above criteria, we show that the currently used mechanism satisfies an incentives condition relating to the reporting of feasible arrival times, but fails to satisfy a certain interpretation of property rights. This is significant in as much as both of these attributes motivated the implementation of the current mechanism.

Second, we provide an example of a mechanism that does satisfy both of these attributes. Our mechanism has the flavor of certain generalizations of the top-trading cycle algorithm (Shapley and Scarf (1974)) that have appeared in related but distinct models (e.g. Papai (2000), Abdulkadiroğlu and Sönmez (1999)). Despite the similar flavor of these models to ours, there are key differences which create the need for a new analysis. In particular, we provide a novel argument to prove the main property rights result.

Third, we formalize another of the incentives attributes mentioned in FAA correspondence, motivating each airline to promptly release any slot that becomes unusable by that airline. Though this condition seems similar to the previous incentives condition, we provide examples showing that neither our mechanism nor the one currently in use is immune to manipulation by airlines who can retain landing slots that they cannot use. This shows the importance of formalizing and distinguishing various incentives concepts.

Our paper connects work done in the transportation literature with work done in the economic matching literature. In Section 1 we describe the FAA’s general problem of reassigning flights to landing slots during inclement weather, and provide reference to the objectives mentioned above. In Section 1.3 we contrast our application to existing models in the matching literature, relating and summarizing our results.

1 THE RATIONING AND REASSIGNMENT OF LANDING SLOTS

When an airport in the U.S. is about to be subject to inclement weather that reduces visibility and/or the number of usable runways, the FAA institutes a ground delay program (GDP), typically hours in advance. Flights that were to arrive at the affected airport during the bad weather are issued delayed departure times at their origin. Furthermore, flights' departure times are spread out so as to reduce the arrival rate during the inclement weather.¹ The specifics of how a GDP reconfigures the landing times of flights is explained in the following 2-step process, the second step being the focus of this paper.

First, a GDP reduces the number of landing slots per hour at the affected airport. For example, suppose an airport can normally land sixty flights per hour, but under a GDP is limited to landing thirty flights per hour. Then thirty 2-minute slots are created where there used to be sixty 1-minute slots, and these scarce slots are rationed to the airlines who held the original 60 slots. This *rationing step* is presently done on a first-scheduled, first-served basis using a procedure known as *Ration-by-Schedule* (RBS).²

Second, after the rationing step is completed, further changes in the status of certain flights—known only to the airlines at the time—may mean that a slot rationed to a particular airline can no longer be used by that airline. This happens, for example, when a flight becomes further delayed due to delays at other airports, crew or mechanical problems, or outright cancellation by the airline. Hence, what began as a problem of scarce resources (slots) becomes, in the interim, one of excess resources (newly vacated slots). These subsequent cancellations leave gaps in the landing schedule that otherwise could be used by other airlines in order to get their flights on the ground earlier.³ This necessitates the *reassignment step* of the GDP: to efficiently fill these new gaps in the schedule. To perform the reassignment step, the FAA must use an algorithm that feasibly moves flights to earlier slots in the schedule, removing gaps in the landing schedule. Since 1998, the FAA has used the Compression Algorithm defined in Section 3 to accomplish this

¹Prior to 1981, the FAA did not impose this constraint. If there was congestion at the destination, aircraft were placed in holding patterns until they were able to land or were directed to an alternative airport if their fuel was spent.

²Roughly speaking, the flight assigned the k th slot in the original schedule is assigned by RBS to the k th slot in the new, reduced schedule. In this paper, that step is taken as given, and hence not relevant to our analysis. Therefore we refer the reader to Ball, Hoffman, and Vossen (2002) for further details on RBS.

³An airline is free to shift another of its own flights into its own vacant slot; inefficiency would come about if it has no other flight that could use the vacated slot.

step.⁴

1.1 INDUSTRY OBJECTIVE: INCENTIVES

A variety of mechanisms could be used to perform the Rationing and Re-assignment steps of a GDP. The way we approach this mechanism choice problem is to (i) list the objectives one may wish to obtain in landing slot reallocation procedures, (ii) formalize these objectives, and then (iii) search for a mechanism that satisfies them.

The first natural example of such an objective involves incentives. Both the airlines and the FAA are aware that incentives matter. In fact, the current use of RBS in the Rationing Step is due to the very issue of incentives. Before the FAA switched to using RBS and the Compression Algorithm, landing slots were reallocated under a procedure known as the Grover–Jack Algorithm. Unlike RBS, Grover–Jack performed the rationing step by giving priority to flights based on their *feasible departure times* (as opposed to their original landing schedule order), which is private information that is reported to the mechanism by the airlines themselves. The airlines well understood that this would lead to undesirable outcomes; even the FAA’s own description of RBS acknowledges this:⁵

“If an airline reported a mechanical delay on a flight, the system would re-project its arrival time. If a GDP [using Grover–Jack] were run at that time, that flight would likely receive an additional delay on top of its mechanical delay. These effects were known as the ‘Double Penalty’ issue. . . The airlines would simply not send in information that would produce clear adverse economical consequences [to themselves]. RBS removes this disincentive.”

So not only was there a perception of inequity due to the Double Penalty, but more importantly there was concern about *incentives*, and the resulting adverse impact on efficiency from misreporting delays.

⁴In reality, the reassignment step may need to be applied iteratively or dynamically, as airlines continuously update their information. For instance, the Compression Algorithm may be applied as often as once every 15 minutes, if conditions warrant it, though typically hourly. Since our objective is to highlight the impact of objectives on mechanism choice, we abstract away from such dynamics by assuming that reassignment needs to be performed only once.

⁵The quote is from the FAA’s online description of RBS, available at <http://cdm.fly.faa.gov/ad/rbs.html> (last retrieved May 2011).

The concern over incentives extends even into the Reassignment Step of a GDP. Vossen and Ball (2006a) explain part of the motivation for the Compression Algorithm as follows.

“Once arrival slots have been allocated, the Compression algorithm performs schedule updates by an inter-airline slot exchange, which aims to provide airlines with an incentive to report flight cancelations and delays.”

That quote highlights two incentives conditions, involving the reports of cancelations and of delays. It is worth noting that there is an actual concern that airlines could fail to report (or, in practice, postpone to report) the cancelation of a flight. This is demonstrated in the following quote from a 1996 U.S. Department of Transportation internal memo, expressing concern over the possibility that airlines could fail to report flight cancelations by simply sitting on vacant slots.

“If an airline sits on a slot that it is not planning to use, is there any way for ATMS to detect this and to take this slot away from the airline? Should this be done?”⁶

The “incentive to report cancelations” and the “incentive to report delays” look like similar concepts; one could reasonably suspect that a flight cancelation is something like a long delay. Once formalized, it turns out that these are in fact two different incentives requirements with different implications. We show that the algorithms we discuss satisfy only one of these two conditions (failing the former; see Section 5).

1.2 INDUSTRY OBJECTIVE: PROPERTY RIGHTS

In addition to defining and satisfying incentives requirements, another concern often expressed in this environment pertains to property rights and fairness. For instance, the FAA’s *Facility Operation and Administration* handbook⁷ gives explicit permission for an airline to swap landing slots amongst any of its own flights. As a special case, if an airline cancels a flight, then it may move another into the vacated slot. This reflects a form of property rights: an airline has “usage rights” over any slot assigned to it.

⁶The June 18, 1996 memo is available at <http://cdm.fly.faa.gov/ad/compdocs.html>. The same memo even addresses the possibility that airlines could create dummy flights in order to occupy otherwise vacated slots, discussing “the incentive for an airline to create extra flights that it could use to abuse the system.”

⁷Available at <http://www.faa.gov/atpubs>.

Making an even stronger statement, Vossen and Ball (2006a) write (with our emphasis):

“[An airline] can reassign its flight-slot assignment using the cancellation and substitution process. It should be emphasized that *this notion of slot ownership is one of the main tenets of the CDM paradigm*: there is a general consensus among airlines that this is indeed a fair method of rationing arrival capacity.”

Since the CDM paradigm⁸ is the result of collaboration between the FAA and airlines, we view this as an endorsement of the normative idea that airlines have a degree of property rights associated with some degree of “ownership” of the slots assigned to them by the RBS procedure.

In fact, our interpretation of the industry’s position on property rights can be further bolstered by pointing out another element of flight scheduling currently available to airlines: Slot Credit Substitutions (SCS). During a GDP, an airline has the right to ask for a bilateral slot trade, i.e. to trade one of its landing slots for one belonging to another airline. We have two points to make about SCS’s. First, the right to trade away one’s slots reinforces our position that an airline’s landing slots are a form of “endowment” toward which they have property rights.

Secondly, the SCS process is unwieldy. Its description in Section 4.1.5 of the FAA document “Interface Control Document for Substitutions during Ground Delay Programs, Ground Stops, and Airspace Flow Programs” shows how unwieldy a SCS can be. It requires an airline to go through a separate process to initiate such a trade, requires a complementary airline to be actively participating in SCS trades, which in turn requires that airline to avoid making any intra-airline trades of its own flights within its own slots. Robyn (2007) elaborates on the inefficacy of this protocol in practice. This justifies the idea that the implementation of property rights should happen at the Reassignment step of a GDP, rather than in a separate, cumbersome process.

Because these ideas are central to this paper’s motivation, we summarize our position on property rights in this environment as follows. Airlines already have the rights (i) to reassign their own flights within their own parts of the landing schedule; and (ii) to trade with each other. This implies that, based on an initial *endowment* of landing slots, airlines are entitled to property rights in the form of *core allocations* of landing slots. In Section 2.2

⁸Collaborative Decision Making is a broader framework designed to push decision making down to those with information: the airlines. See Chang et al. (2001) for more.

we explore and define the notion of the core, which enforces a degree of property rights to the extent that airlines can either *use their own slots as they wish, or trade them to their best advantage*. The main observation in this paper is that the Compression algorithm does not respect this definition of property rights, yet we provide another algorithm (Section 4) that does.

It is worth noting that our motivation for using the concept of the core differs from that in most of the matching literature. In most other work, the core relates to the notion of stability. Mechanisms that do not choose core outcomes are vulnerable to unraveling or collapse: coalitions of participants have an incentive to depart the market and improve their outcomes. In this application, market departure is *infeasible*, since an airline cannot unilaterally form its own landing schedule. Our motivation, instead, rests with the above observation that airlines have already been granted the level of well being that comes with core allocations.⁹

1.3 RELATED LITERATURE

A block of airport landing slots comprises a set of indivisible goods indexed by time. A GDP's Rationing step assigns these goods to flights. After some airlines cancel flights, the Reassignment step of the GDP reallocates the goods to the remaining flights. Phrased this way, the problem of reassigning flights from an initial endowment of slots evokes what is known as the *house allocation problem*. This model is introduced by Shapley and Scarf (1974), who examine the core of a game in which each of n traders initially owns a house, but has preferences over all traders' houses.¹⁰ Rather than speaking of traders and houses, one can merely label these as flights and landing slots.

Nevertheless, the environment considered here is different from the house allocation problem. On one hand, the model here is more limited than the classic housing model due to the added structure in this environment.¹¹ On the other hand, the landing slot problem has other characteristics that make it appear more general than the Shapley-Scarf, as we now explain.

Any slot that is assigned to a *flight* of a given airline, say United Airlines, actually becomes the endowment of that *airline*. For instance, if RBS assigns a United Airlines flight to a slot, but United eventually cancels that flight,

⁹One could object that, since the SCS procedure already exists, there is no need for the FAA to implement core outcomes; the market can sort things out. While this is true in theory, the unwieldiness of the SCS procedure is a market friction that provides a counterargument to such an objection.

¹⁰This connection is also mentioned by Balakrishnan (2007).

¹¹Formally, the preference domain is restricted. Flights always want to arrive as early as they feasibly can.

then United may assign another of its flights to that landing slot. Thus, the endowments determined by the RBS procedure span across multiple flights.

More directly, the airlines are the agents in this setting, not the flights. In the terminology of the house allocation problem, a “trader” (airline) consumes multiple “houses” (slots), so this is not a one-to-one matching problem. Here, this distinction is important in two ways. First, regarding incentives analysis, one must ask: By misrepresenting the preferences (feasible arrival time) of one flight, is it possible for an airline to improve the outcome of another of its flights? Second, this affects the formalization of the concept of the core. The strong core (see Section 2.2) may fail to exist in this model. Nevertheless, we can take advantage of some previous incentive results for one-to-one problems to give a novel proof that the algorithm we provide in Section 4 selects from the *weak* core.

Another distinction between our model and the Shapley–Scarf model is that there are extra “houses” (slots) to be allocated. This generalization has already been made in the literature by, e.g. Abdulkadiroğlu and Sönmez (1999), Papai (2000), and others. However these works endow the extra slots to individual flights (using the language of our paper) (or to no one), as opposed to endowing them to *airlines* (subsets of flights). This difference also has an impact on the analysis.

The one way in which our model restricts the usual house allocation problem is the structure of preferences: earlier is better. It makes little sense to assume that arbitrary preferences over landing slots should be considered on behalf of the airlines. Holding all else constant, it is reasonable to assume that a flight wants to obtain the earliest landing slot possible, as long as it is not earlier than the flight’s feasible arrival time. This restriction impacts the applicability of previous work on a manipulability condition we discuss in Section 5.

In Section 2.1 we formalize our specific preference restriction. Under that model, the main results of this paper are the following.

1. We show that the FAA’s Compression Algorithm is strategyproof, in the sense of giving airlines the incentive to truthfully report their flights’ feasible arrival times. Thus our paper is the first to make precise the notion that the Compression Algorithm satisfies any kind of incentive compatibility.
2. We show that the Compression Algorithm need not return an outcome in the weak core. While work on other matching problems suggests that core outcomes are crucial for the stability of exchanges, it is im-

portant to note that our motivation here is based on one of property rights (slot ownership).

3. We describe a mechanism in Section 4 that is strategy-proof *and* always selects an assignment in the weak core. While our model is distinct from the Shapley–Scarf house allocation problem, our mechanism has the flavor of mechanisms appearing in Abdulkadiroğlu and Sönmez (1999) and Pápai (2000). Pápai’s results are used to prove one of our results on incentives and they contribute to our proof on core selection.
4. We provide examples to show that both the Compression Algorithm and the one we introduce can be manipulated by an airline that fails to vacate a slot (e.g. by filling one of its slots with a “dummy flight”). This directly answers the question in the 1996 DOT memo quoted in Section 1.1.

The last result may not be surprising within the context of the matching literature. In another matching model, Atlamaz and Klaus (2007) show that any efficient mechanism that satisfies a minimal level of property rights (individual rationality) must be manipulable to the destruction of endowments. Since their paper uses a general domain of preferences (as opposed to the preference restriction necessary in our model) their result does not imply ours.¹²

The most related papers to ours are the ones mentioned above by Pápai (2000) and Abdulkadiroğlu and Sönmez (1999) for a related model. Their mechanisms, like ours, generalize the Top Trading Cycle algorithm of Shapley and Scarf, but to a different model. Another related paper is by Balakrishnan (2007), who relates the Top Trading Cycle algorithm and the core to a model closer to ours. The key difference between our papers is that Balakrishnan treats individual flights as agents, bringing the model closer to that of Shapley and Scarf. As discussed earlier, this implies two main technical differences in the analysis, related to incentives and to the core.¹³

¹²Nor would their result be implied by such a result in our model.

¹³ Balakrishnan(2007) does not analyze incentives, so we have no comparison to make. Regarding the core, Balakrishnan proves the existence of core matchings using a very weak definition of the core: for a coalition of airlines to block an allocation, every flight owned by every airline in the coalition must *strictly* gain. This definition is even weaker than what we call “weak core” in this paper. Our existence result would be stronger if we operated on the same domain of preferences, but since her paper models preferences from the perspective of flights (rather than airlines), our results are not comparable.

2 THE MODEL AND DEFINITIONS

There is a finite set of arrival Slots $S = \{1, 2, 3, \dots, |S|\}$. We interpret the labels of these slots as (ordinal) representations of time: for $s, t \in S$, $s < t$ means that slot s is earlier than slot t .

There is a set of airlines \mathcal{A} , and each airline $A \in \mathcal{A}$ has a set of flights F_A . In relation to the discussion in the Introduction, we interpret F_A as the set of flights that airline A has *not* canceled at the time of a ground delay. Denote by $F = \bigcup_{A \in \mathcal{A}} F_A$ the set of all flights. We consider situations in which $|F| < |S|$.

The earliest feasible arrival time (ETA) of flight $f \in F$ is denoted $e_f \in S$. Hence, flight f can be feasibly assigned to slot $j \in S$ only if $e_f \leq j$.

A *Landing Schedule* is formalized as a feasible assignment of flights to slots. Specifically, it is a function $\Pi: F \rightarrow S$ mapping flights to slots such that distinct flights get distinct slots, and flights are assigned to feasible slots. Formally, for any two flights $f, f' \in F$, it is Injective: $f \neq f'$ implies $\Pi(f) \neq \Pi(f')$; and it is Feasible: $\Pi(f) \geq e_f$.

Landing schedules do not specify the ownership of vacant slots. Therefore we introduce the concept of a *slot ownership* function, which is a function $\Phi: S \rightarrow \mathcal{A}$ that satisfies consistency with some Π : for all $f \in F$, $f \in F_A$ implies $\Phi(\Pi(f)) = A$. An *assignment* is a pair of functions (Π, Φ) that jointly satisfy the above conditions.

An *instance* (or economy) is summarized by $I = (S, \mathcal{A}, (F_A)_{A \in \mathcal{A}}, e, \Pi, \Phi)$, which satisfy the above assumptions. Interpret (Π, Φ) as an *initial* assignment which, due to the location of vacant slots, may be inefficient.

Our objective is to discuss methods for creating more efficient assignments, so we introduce the following concept. A *matching function* (or algorithm) takes an instance as input and produces a (new) Landing Schedule as output: $\varphi(I) = \Pi'$.

2.1 PREFERENCES

Airlines wish to get each flight to its next destinations as early as possible, subject to feasibility. For a single flight, this means it is better to be assigned to slot e_f , second-best to be assigned to slot $e_f + 1$, etc. Being assigned any slot $s < e_f$ is unacceptable since the flight cannot arrive in time to use it.

To express an airline's preference over landing schedules is a more complicated affair since it involves the landing times of multiple flights. As a first approach in this paper, we suppose that an airline is made better off only if it moves a flights up in the schedule while no others move down.

To formalize this, consider two Landing Schedules Π and Π' . We say that airline A *strictly prefers* Π to Π' if $\Pi(f) \leq \Pi'(f)$ for every flight $f \in F_A$, and $\Pi(f) < \Pi'(f)$ for at least one $f \in F_A$. That is, airlines prefer assignments that never move a flight ‘down’, i.e., later and move their flights ‘up’, i.e., earlier. On the other hand, if an assignment Π' adjusts Π by simultaneously moving some of A ’s flights to later positions and moving others earlier, then we assume that Π' is preference-incomparable with Π .

This specification of preferences may at first appear to be narrow. For example, it rules out situations where an airline would consider it profitable to sacrifice the arrival time of one its flights (e.g. a small plane with few passengers) in order to advance the arrival time of another of its flights (e.g. a large plane with many connecting passengers). Nevertheless, there are reasons why, as a first approach, we are comfortable with using this preference assumption to define our incentives and core conditions in this paper. The most significant is that an airline already has the right to swap its own flights within its own set of slots. This partially mitigates the need for an airline to manipulate any kind of algorithm to achieve such a swap, as they are permitted to do so directly. A second reason is that, in any situation where the previous argument does not apply, there would be significant complexity in determining how an airline should properly misreport the status of one flight (guaranteeing its delay) in order to *potentially* benefit another flight. Finally, as is known in the matching literature, difficulties arise in many-to-one problems when arbitrary preferences (tradeoffs) are permitted in the side of the market that matches to “many.” Our objective to relate the industry’s objectives to different mechanisms can be accomplished even on this simple specification of preferences, so we leave a more general model to future work.

Given our preference domain, we say that an algorithm is *manipulable* if, by misreporting the feasible arrival times of its flights, an airline can obtain a landing schedule it strictly prefers to the one obtained by reporting true feasible arrival times.

2.2 CORE SCHEDULES

Before a GDP goes through the Reassignment step, the (previous) Rationing step specifies an initial assignment (Π^I, Φ^I) . We think of this assignment as specifying an *endowment* to each of the airlines. In order to implement the property rights discussed in Section 1.2, we define the concept of core assignments as a function of (Π^I, Φ^I) .

The concept commonly used in this literature is the *strong* core: an

assignment is in the strong core when no coalition of agents could have instead traded only amongst themselves, using their initial endowment, to make themselves weakly better off, with one agent strictly better off. Elegantly, this concept defines a unique assignment in the Shapley–Scarf housing market problem. In our terminology, this would be when airlines have one flight each and there are no vacant slots; even if airlines have arbitrary strict preferences over slots. But in the airline landing slots problem, it is straightforward to see that the set of strong core assignments can be empty.¹⁴

Therefore we instead focus on the *weak* core. With respect to an initial assignment (Π^I, Φ^I) , a landing schedule Π is a (*weak*) *core schedule* if no subgroup of airlines could reallocate their initial slots (from Φ^I) to each other in order to make themselves better off than in Π . Formally, Π is in the core if there exists no other landing schedule Π' and set of airlines $\mathcal{B} \subset \mathcal{A}$ such that (i) for all $f \in \cup_{A \in \mathcal{B}} F_A$, $\Phi^I(\Pi'(f)) \in \mathcal{B}$, and (ii) each airline $A \in \mathcal{B}$ strictly prefers Π' to Π . For the remainder of the paper, we refer to the weak core simply as the “core.”

We give an example to show that the core can be multi-valued. Suppose $S = \{1, 2, 3, \dots, 12\}$, and that the initial assignment is as shown below. Each row lists a slot, the flight that is initially assigned it, the airline owning that flight, and the flight’s earliest arrival time.

EXAMPLE 1 The initial assignment (Π^I, Φ^I) is described by the following table.

¹⁴This non-existence is well-known to happen in matching models with indifference. In our model the non-existence can be driven by the fact that a vacant slot may be useless to its airline, which creates a form of indifference. The simplest example involves an airline with no flights and a vacant slot, along with two other airlines, both of whom want that vacant slot.

Slot	Flight	Airline	feasible arrival time e_f
1	vacant	A	
2	f_2	B	1
3	f_3	C	1
4	f_4	A	2
5	vacant	B	
6	f_6	C	5
7	f_7	A	5
8	f_8	B	6
9	vacant	C	
10	f_{10}	A	9
11	f_{11}	B	9
12	f_{12}	C	10

For instance, slot 1 is vacant and is owned by airline A (i.e. $\Phi^I(1) = A$). Slot 2 is owned by airline B and is occupied by B 's flight $f_2 \in F_B$ ($\Pi^I(f_2) = 2$). Flight f_2 could feasibly arrive in slot 1 ($e_{f_2} = 1$).

The following landing schedule Π is a core schedule with respect to the initial assignment in Example 1.

$$\begin{array}{lll}
\Pi(f_2) = 1 & \Pi(f_6) = 5 & \Pi(f_{10}) = 9 \\
\Pi(f_3) = 3 & \Pi(f_7) = 7 & \Pi(f_{11}) = 11 \\
\Pi(f_4) = 2 & \Pi(f_8) = 6 & \Pi(f_{12}) = 10
\end{array}$$

To see that Π is in the core, consider the coalition of airlines $\mathcal{B} = \{A, B\}$. They do not initially own slot 9 (i.e. $\Phi^I(9) = C$). Hence any schedule they attempt to construct on their own must make flight f_{10} arrive later than it does under Π , hence airline A cannot be made better off. A symmetric argument applies to any other pair of airlines, and the remaining arguments are trivial.

The following landing schedule Π' is also in the core.

$$\begin{array}{lll}
\Pi'(f_2) = 1 & \Pi'(f_6) = 5 & \Pi'(f_{10}) = 9 \\
\Pi'(f_3) = 2 & \Pi'(f_7) = 6 & \Pi'(f_{11}) = 10 \\
\Pi'(f_4) = 3 & \Pi'(f_8) = 7 & \Pi'(f_{12}) = 11
\end{array}$$

We leave it to the reader to verify this, using similar arguments.

- Step 0 Initialize the current assignment as $\Phi = \Phi^I$, $\Pi = \Pi^I$. Let $V = S \setminus \Pi^I(F)$ denote the set of vacant slots.
- Step 1 If $V = \emptyset$, end the algorithm at the assignment (Π, Φ) . Otherwise pick the earliest vacant slot $s \in V$ and declare it active.
- Step 2 Let $A = \Phi(s)$ denote the airline that owns s . Check whether airline A has a flight $f \in F_A$ that both (i) occupies a later slot $\Pi(f) > s$ and (ii) could feasibly use slot s ($e_f \leq s$). If so, let f be the earliest such flight according to Π , denote its slot $t = \Pi(f)$, and go to Step 4. Otherwise go to Step 3.
- Step 3 Check whether *any* airline has a flight f that both (i) occupies a later slot $\Pi(f) > s$ and (ii) could feasibly use slot s ($e_f \leq s$). If so, let f be the earliest such flight according to Π , denote its slot $t = \Pi(f)$, and go to Step 4. Otherwise remove slot s from V and return to Step 1.
- Step 4 Move flight f from slot t to slot s : set $\Pi(f) = s$ and set $\Phi(s)$ equal to the airline of flight f . Set $\Phi(t) = A$. Remove s from V and add t to V . Return to Step 2 using t as the new active slot s .

Figure 1: The Compression Algorithm

3 THE COMPRESSION ALGORITHM

The Compression Algorithm, currently used by the FAA during the Reassignment step of GDP's, is a significant improvement over previous methods for allocating slots. We refer the reader to Vossen and Ball (2006a) for details, but the primary advantage this algorithm has over previous methods is that it rewards airlines for giving up slots that they cannot use. When an airline gives up a slot it considers useless, it trades the slot for a later one, typically owned by the airline that eventually takes possession of the earlier slot. The Compression Algorithm is formally defined in Figure 3. In the rest of the section we discuss its properties.

Consider an airline who has flight f with earliest feasible arrival time of e_f . It may be plausible for the airline to falsely announce it to be $e'_f \neq e_f$ in an attempt to achieve a better outcome under the Compression algorithm. As the following result shows, however, the Compression algorithm cannot

be manipulated in this way.¹⁵

THEOREM 1 *The Compression Algorithm cannot be manipulated by misreporting the feasible arrival time of its flights.*

PROOF: First we provide detailed proof for the case of misreporting a single flight's arrival time, then give a brief general argument for the case of misreporting multiple flights.

Suppose that by reporting e_f honestly, flight f is assigned to slot s . Clearly, $e_f \leq s$. A misreport e'_f could be of three types.

Case S1: $e_f < e'_f \leq s$. The outcome of the algorithm cannot be altered by this misreport. Whenever f is the flight chosen in Steps 2 or 3 when e_f is reported, f would still be chosen when e'_f is reported, because f never moved into a slot earlier than s .

Case S2: $e_f \leq s < e'_f$. Here f would have to end up in a slot strictly later than s , since the Compression Algorithm never places a flight in a slot earlier than its reported earliest arrival time. This makes its airline worse off.

Case S3: $e'_f < e_f \leq s$. The only way this misreport can change the outcome of the algorithm is to assign f to a slot earlier than e_f . By assumption, this makes the airline worse off.

Hence misreporting a single flight's time either puts the flight in a later slot, or does not affect the outcome. In either case, f 's airline cannot gain.

More generally, suppose airline A 's flights have arrival times $(e_f)_{f \in F_A}$, but A misreports them to be $(e'_f)_{f \in F_A}$ ($e'_f \neq e_f$ for at least one f). Consider the first round of the Compression Algorithm in which execution would differ under a report of (e_f) vs. (e'_f) . The difference must occur in Step 2 or Step 3, in which some flight $g \in F_A$ is queried as to whether it can feasibly occupy some slot s . There are two cases.

Case M1: $e_g < e'_g$. In this case, the slot s is deemed feasible under a report of e_g , but is deemed infeasible under e'_g . That is, we must have $e_g \leq s < e'_g$. By (mis-)reporting $(e'_f)_{f \in F_A}$, flight g must end up in some slot $t \geq e'_g$, so $s < t$. By truthfully reporting $(e_f)_{f \in F_A}$, g must end up in some slot $t' \leq s < t$. Hence flight g ends up in a worse slot under the misreport.

Case M2: $e'_g < e_g$. In this case, s is deemed infeasible under e_g , but deemed feasible under e'_g . That is, we must have $e'_g \leq s < e_g$. In this case,

¹⁵It may be considered implausible for an airline to falsely report $e'_f < e_f$: If the flight is awarded an infeasible slot $s < e_f$, it would eventually be discovered when the flight cannot arrive on time to utilize this slot. Fortunately Theorem 1 makes this point moot.

by reporting $(e'_f)_{f \in F_A}$, flight g ends up in an infeasible slot, which makes the airline worse off.

In both cases, the airline cannot gain, regardless of the outcome of the other flights. \square

While the Compression Algorithm satisfies the incentives property of Theorem 1, we now give an example showing that it can return a landing schedule that is not in the core. Consider the following initial assignment of slots to flights.

Slot	Flight	Airline	feasible arrival time e_f
1	vacant	A	
2	vacant	B	
3	f_3	C	1
4	f_4	B	1
5	f_5	A	2

In Step 1 of the Compression Algorithm, slot 1 is declared active. Slot 1 belongs to A , but the only flight in F_A (f_5) cannot be assigned to it. Therefore, in Step 3 of the algorithm, flight f_3 is assigned slot 1. This updates the original assignment to the following one.

Slot	Flight	Airline	feasible arrival time e_f
1	f_3	C	1
2	vacant	B	
3	vacant	A	
4	f_4	B	1
5	f_5	A	2

By Step 4 of the Compression Algorithm slot 3 is declared active.¹⁶ Since flight f_5 can feasibly fill it, subsequent steps of the algorithm result in the following assignment.

¹⁶If the algorithm is modified to make slot 2 the next active slot, we would obtain the same result.

Slot	Flight	Airline	feasible arrival time e_f
1	f_3	C	1
2	vacant	B	
3	f_5	A	2
4	f_4	B	1
5	vacant	A	

When slot 5 becomes active, no flight can be assigned to it, so it is discarded. Returning to Step 1, slot 2 is declared active, and since f_4 can fill it, we have the following assignment.

Slot	Flight	Airline	feasible arrival time e_f
1	f_3	C	1
2	f_4	B	1
3	f_5	A	2
4	vacant	B	
5	vacant	A	

Slot 4 has no further use, so the algorithm completes with the above schedule.

However, airlines A and B can achieve the following schedule using only their own resources amongst themselves.

Slot	Flight	Airline	feasible arrival time e_f
1	f_4	B	1
2	f_5	A	2
3	f_3	C	1
4	vacant	B	
5	vacant	A	

Notice that airlines A and B are each strictly better off in this assignment than in the one given by the Compression Algorithm. One can check that this schedule is the unique one in the core of this example.

The intuition behind this example is as follows. The Compression algorithm benefits airlines when, for example, if airline A cannot use a slot A owns, then the algorithm trades it to another airline in exchange for some other slot. However, A cannot decide exactly which other airline it trades

with. In the example above, A would like to trade with B . However, the Compression algorithm does not allow this.¹⁷

In the next section we introduce an algorithm that does take into account A 's implicit preference with whom to trade. Our algorithm has the same incentive properties as the Compression algorithm but has the additional property that it returns an outcome in the core.

4 TRADECYCLE

We describe the TradeCycle algorithm and show below that it returns an allocation in the core. It iteratively adjusts the original landing schedule until any remaining vacant slots are unusable by the present flights. See Figure 2 for the algorithm's description.

A notable distinction from the Compression algorithm is that, in each round of the TradeCycle algorithm, flights moved to new slots are not moved again (having become inactive). In step 2 there are two kinds of cycles that can be formed: those containing at least one vacant slot, and those containing only a flight and the slot it occupies. In either case, flights in such cycles are allocated to their favorite active slot, and have no chance of further improving their position.

Readers familiar with the matching literature¹⁸ may notice that this algorithm has some of the flavor of the Top Trading Cycle algorithm (Shapley and Scarf (1974)). In fact TradeCycle corresponds, at least algorithmically, to an element of the class of *fixed endowment hierarchical exchange* (FEHE) rules described by Pápai (2000).

Pápai's model differs from ours in two ways. First, using our terminology, each flight behaves as an individual strategic agent in Pápai's model. Fortunately, she obtains results on *group* strategyproofness which can easily be adapted to yield individual incentives results in our model. Second, Pápai's model allows general preferences over slots (arbitrary orderings). The structure of the airport landing slots problem implies that not all preference orderings of slots should be considered, as we specified in Section 2.1. But by a standard argument, any strategyproof rule on a larger domain of preferences yields a strategyproof rule when projected to a smaller domain of preferences.

¹⁷This is tangentially related to the issue of the order in which vacant slot are chosen to be active in the Compression algorithm. See section 3.7.2 of Vossen and Ball (2006a), who show that "order matters" when executing Compression.

¹⁸See Roth and Sotomayor (1990) for an introductory survey.

- Step 0 Take as input an initial assignment (Φ^I, Π^I) , and declare all slots and flights to be “active.”
- Step 1 If the set of active flights is empty, the algorithm ends. Otherwise, construct a graph as follows.
- Step 1a Introduce a node for each active slot and each active flight.
 - Step 1b From each flight f , draw a directed edge to the earliest active slot that f can occupy.
 - Step 1c From each occupied slot, draw a directed edge to the flight that occupies it.
 - Step 1d From each vacant slot owned by any airline A , draw a directed edge to (i) the earliest (according to Π^I) active flight in F_A , if one exists; otherwise (ii) the earliest (according to Π^I) active flight in F .
- Step 2 Within any (directed) cycle in the graph: Assign each flight to the slot it points to in the cycle; declare the flight and its assigned slot inactive. Unassigned slots within a cycle become vacant (and remain active). Return to Step 1.

Figure 2: The TradeCycle algorithm.

PROPOSITION 1 *TradeCycle is algorithmically equivalent to a fixed endowment hierarchical exchange rule (Pápai (2000)) in which (i) flights are treated as individual agents, and (ii) each slot has an inheritance structure whereby it prioritizes flights as follows. First is the flight that occupies it in the initial Landing Schedule (if any). Next are the flights of the same airline (if any), prioritized by their order in the initial landing schedule. Finally are the flights of the remaining airlines, prioritized by their order in the initial landing schedule.*

See the Appendix (Section 7) for the formal argument behind this Proposition.

Our algorithm shares the same fundamental incentives property that the Compression Algorithm was shown to satisfy in Theorem 1. The proof is straightforward in light of Proposition 1.

THEOREM 2 *TradeCycle cannot be manipulated by misreporting the feasible arrival time of its flights.*

PROOF: This follows from the theorem in Pápai (2000), stating that FEHE rules for one-to-one assignment problems are group strategyproof even when arbitrary preferences over slots are permitted. Here, each flight announces an ETA e_f , which implicitly represents preferences which rank the slots in the order $e_f, e_f + 1, \dots, s, 1, 2, \dots, e_f - 1$. \square

Our main motivation for introducing TradeCycle is that it chooses an outcome in the weak core. Our proof of this result is simplified by the fact that FEHE rules are group strategyproof.

THEOREM 3 *TradeCycle returns an assignment in the weak core.*

PROOF: Fix an instance $I = (S, \mathcal{A}, (F_A)_{\mathcal{A}}, e, \Pi, \Phi)$, and let Π^T denote the landing assignment yielded by TradeCycle. Suppose, in contradiction to the theorem, that a coalition of airlines $\mathcal{B} \subset \mathcal{A}$ could each strictly gain by relocating their initially scheduled slots amongst themselves. That is, suppose there exists a partial landing assignment $\Pi'_{\mathcal{B}}: \bigcup_{A \in \mathcal{B}} F_A \rightarrow \bigcup_{A \in \mathcal{B}} \Phi(A)$ such that (i) each flight in $\bigcup_{A \in \mathcal{B}} F_A$ arrives weakly earlier in Π' than in Π^T , and (ii) each $A \in \mathcal{B}$ has at least one flight that arrives strictly earlier in Π' than in Π^T .

Consider the instance \tilde{I} which is identical to I , except that each $f \in \bigcup_{A \in \mathcal{B}} F_A$ declares an ETA of $e'_f = \Pi'(f)$. Observe that $e_f \leq \Pi'(f) \leq \Pi^T(f)$ for each such f , with the latter inequality being strict for at least

one flight from each $A \in \mathcal{B}$. Furthermore, observe that for any distinct $f, f' \in \bigcup_{A \in \mathcal{B}} F_A$ we have $e'_f \neq e'_{f'}$.

Let $\tilde{\Pi}^T$ denote the landing assignment yielded by TradeCycle applied to \tilde{I} . Strategyproofness and non-bossiness of FEHE rules implies that $\tilde{\Pi}^T = \Pi^T$. (This is a standard strategyproofness argument whose full details we omit. The idea is that if one flight reports a slightly later ETA, it cannot achieve a better arrival time than it originally was assigned, due to strategyproofness. Conversely, if it is assigned a *worse* arrival time, this would violate strategyproofness in the other direction. Nonbossiness allows one to repeat the argument for each flight sequentially, without changing the assignment of *any* flight at each step.)

Therefore \mathcal{B} blocks $\tilde{\Pi}^T$ for precisely the same reason it blocked Π^T .

Under \tilde{I} , consider the first round of TradeCycle in which some $s = \Pi'(f)$ with $f \in \bigcup_{A \in \mathcal{B}} F_A$ is allocated to some flight $g \notin \bigcup_{A \in \mathcal{B}} F_A$. Consider the cycle C (containing s and g) that forms in that round of the algorithm. There must be a slot s' in C pointing to g . However, at this point in the algorithm, all flights and occupied slots corresponding to \mathcal{B} are still pointing to other flights/slots owned by members of \mathcal{B} . Therefore, in order for C to contain a flight outside of \mathcal{B} , there must be a vacant slot in C that is owned by some $A \in \mathcal{B}$.

However, for a vacant slot owned by A to point to a flight owned by a different airline, all of A 's flights must have already been assigned by TradeCycle. By the assumption of the current round of the algorithm, this means that each of A 's flights was assigned its preferred slot (Π'). Thus $\Pi'(A) = \tilde{\Pi}^T$; but this contradicts the fact that A has at least one flight strictly better off under Π' than $\tilde{\Pi}^T$. \square

5 SLOT DESTRUCTION

Another form of manipulation that concerns the industry is the *destruction of vacant slots* (see Section 1.1). Suppose an airline has a flight f initially occupying a slot s , and that the airline intends to cancel the flight. By delaying the cancelation announcement sufficiently long, the airline could effectively destroy the slot's usefulness to other airlines. By failing to announce the cancelation and declaring $e_f = s$, an algorithm would perceive f to be a non-canceled flight, and f would stay in slot s (under both algorithms discussed in this paper, and most other reasonable algorithms).

Therefore we define *destruction of a vacant slot* s to be the insertion of a “dummy flight” f into slot s such that $e_f = s$. The following example shows

that the Compression Algorithm is vulnerable to destruction of a vacant slot.

EXAMPLE 2 Let I be an instance defined by the initial assignment (Π^I, Φ^I) as described in the following table.

Slot	Flight	Airline	feasible arrival time e_f
1	vacant	A	
2	vacant	B	
3	vacant	A	
4	f_4	C	2
5	f_5	B	4
6	f_6	A	4
7	f_7	B	1

When the Compression Algorithm is applied to I , airline B 's flight f_7 is moved into slot 1. In exchange for this trade, airline A receives the useless slot 7. In the next round of the algorithm, airline B receives slot 4 in exchange for f_4 moving into slot 2, after which B 's f_5 moves into slot 4. Finally, A 's f_6 receives slot 5.

However, in the instance obtained from this one by deleting slot 1, denoted I' , it is easy to check that airline A exchanges slot 3 for slot 4; flight f_6 ends up in slot 4. Hence A is better off after the destruction of its slot 1.

The following example shows that TradeCycle also is vulnerable to the destruction of slots.

EXAMPLE 3 Let I be an instance defined by the initial assignment (Π^I, Φ^I) as described in the following table.

Slot	Flight	Airline	feasible arrival time e_f
1	vacant	C	
2	f_2	A	1
3	vacant	A	
4	vacant	D	
5	f_5	B	3
6	f_6	C	5
7	f_7	A	5
8	f_8	D	7

It is easily verified that when TradeCycle is applied to I , airline A 's two flights are moved up to slots 1 and 6. However, in the instance obtained from this one by deleting slot 3, denoted I' , airline A 's flights are assigned by TradeCycle to slots 1 and 5.

In instance I of Example 3, there are only two Pareto-efficient assignments. Efficiency requires that f_2 be assigned to slot 1, f_5 be assigned to slot 3, and f_8 be assigned to slot 7. Flights f_6 and f_7 must be assigned to slots 5 and 6, but efficiency does not dictate their order. Similarly, there are two efficient assignments for I' which are the same, except that f_5 must be assigned to slot 4 (since slot 3 was destroyed).

Interestingly, while TradeCycle assigns f_6 to slot 5 (and f_7 to slot 6) under I , the Compression Algorithm does the opposite. Furthermore, while TradeCycle assigns f_6 to slot 6 (and f_7 to slot 5) under I , the Compression Algorithm again does the opposite of TradeCycle. That is, *in this particular example*, the Compression Algorithm rewards A for the presence of slot 3.

6 CONCLUSION

Inclement weather, combined with flight cancellations, creates a need to reassign vacated airport landing slots. In order to do this, the FAA must elicit schedule and cancellation information from airlines. Setting this up within the framework of a mechanism design problem, we have formalized two corresponding incentive conditions. One—which pertains to the reporting of delays (feasible arrival times)—is satisfied both by the FAA's Compression algorithm and by our proposed alternative, the TradeCycle algorithm. A second condition, pertaining to the reporting of flight cancellations, is satisfied by neither of these algorithms. Based on results in a related matching model, we conjecture that no reasonable algorithm can satisfy such a condition.

We have argued (Section 1.2) that within the industry, there is a perception that airlines are entitled to a degree of property rights based on a sense of initial *endowment* of landing slots. Our argument is most strongly motivated by the following two observations. First, an airline already is given the right to reassign any of its flights within its own part of the landing schedule. Second, airlines are given the right to trade with other airlines through (the cumbersome) SCS procedure. What this means is that, at least in theory, airlines already have the “right” to obtain landing slot assignments that are at least as good as what they could obtain through trade. Put differently, a reassigned landing schedule should be a core allocation.

Motivated by this observation, we show that the Compression algorithm does not always result in core outcomes, while the TradeCycle algorithm does. An intuition for this difference can be stated in terms of an expression of airline preference over trades. The Compression algorithm compensates an airline for giving up a vacant slot: it implements a trade of the vacant slot for a slot from another airline. When multiple trades are possible, however, that algorithm does not ask the airline which is its *most preferred* trade. The TradeCycle algorithm (indirectly) does, and uses this information to achieve core outcomes.

Aside from property rights, other formalizations of fairness have been considered in the transportation literature. This is especially the case for the Rationing step of GDPs (i.e. RBS or alternatives), which we have not analyzed here. Vossen and Ball (2006a) show that the RBS procedure lexicographically minimizes the maximum delay experienced among all flights relative to their original (pre-GDP) schedule, and so is the “most fair” in that sense. Manley and Sherry (2010) examine various performance and equity measures for RBS and five other methods of performing the Rationing step. Among those methods they show that RBS minimizes a form of inequity among both airlines and passengers. Results like these motivated us to take RBS as a fixed feature of the Rationing step of a GDP, and therefore to analyze only the Reassignment step of a GDP, in which the Compression Algorithm is currently used.

While our main point was to formalize incentives and property rights in terms of the above conditions, we believe that there is much more work to be done in this matching application. For one thing, we have abstracted away from the fact that, in the real world, slot reassignment needs to be performed repeatedly and dynamically, even for a fixed time interval (perhaps hours in advance). How does this impact incentives and/or property rights? Secondly, our negative results are of the typical “proof of concept” type: the Compression Algorithm fails the core by example, and both mechanisms are vulnerable to slot destruction by example. Future research could empirically examine whether, as a practical matter, these problems are common or whether opportunities for manipulation are sufficiently rare.

REFERENCES

- Abdulkadiroğlu, Atila, and Tayfun Sönmez. 1999. “House Allocation With Existing Tenants.” *Journal of Economic Theory*, 88(2): 233–260.

- Atlamaz, Murat, and Bettina Klaus. 2007. "Manipulation via Endowments in Exchange Markets with Indivisible Goods." *Social Choice and Welfare*, 28(1): 1–18.
- Balikrishnan, Hamsa. 2007. "Techniques for Reallocating Airport Resources during Adverse Weather." In Proc. of the IEEE Conference on Decision and Control.
- Ball, Michael, Robert Hoffman, and Thomas Vossen. 2002. "An analysis of resource rationing methods for collaborative decision making." ATM-2002, Capri, Italy.
- Ball, Michael, Thomas Vossen, and Robert Hoffman. 2001. "Analysis of demand uncertainty in ground delay programs." In Proceedings of 4th USA/Europe Air Traffic Management R&D Seminar.
- Chang, Kan, Ken Howard, Rick Oiesen, Lara Shisler, Midori Tanino, and Michael Wambsganss. 2001. "Enhancements to the FAA Ground-Delay Program Under Collaborative Decision Making." *Interfaces*, 31(1): 57–76.
- Kagel, John, and Alvin Roth. 2000. "The Dynamics of Reorganization in Matching Markets: a laboratory experiment motivated by a natural experiment." *Quarterly Journal of Economics*, 115(1): 201–235.
- Manley, Bengi, and Lance Sherry. 2010. "Analysis of performance and equity in ground delay programs." *Transportation Research Part C*, 18(6): 910–920.
- Pápai, Szilvia. 2000. "Strategyproof Assignment by Hierarchical Exchange." *Econometrica*, 68(6): 1403–1433.
- Robyn, Dorothy. 2007. "Reforming the Air Traffic Control System to Promote Efficiency and Reduce Delays." Report to Council of Economic Advisers, The Brattle Group, Inc.
- Roth, Alvin, and Marilda Sotomayor. 1990. *Two-sided matching: A study in game-theoretic modeling and analysis*. Econometric Society Monograph 18. Cambridge, Cambridge University Press.
- Shapley, Lloyd, and Herbert Scarf. 1974. "On Cores and Indivisibility." *Journal of Mathematical Economics*, 1(1): 23–28.
- Vossen, Thomas, and Michael Ball. 2006a. "Optimization and mediated bartering models for ground delay programs." *Naval Research Logistics*, 53(1): 75–90.

- Vossen, Thomas, and Michael Ball. 2006b. "Slot Trading Opportunities in Collaborative Ground Delay Programs." *Transportation Science*, 40(1): 29–43.
- Wambsganss, Michael. 1996. "Collaborative decision making through dynamic information transfer." *Air Traffic Control Quarterly*, 4: 107–123.

7 APPENDIX

In the language of our model, we describe a *fixed endowment hierarchical exchange rule* (FEHE) algorithm, as defined by Papai (2000). We give our description of FEHE in a way that is very close to our description of the TradeCycle algorithm in order to make the proof of Proposition 1 transparent. See Papai (2000) for a description of FEHE as a subclass of Hierarchical Exchange rules, which are the main focus of her paper.

For each slot s , let O_s denote a linear ordering of all the flights in F . Suppose each flight has an (arbitrary) strict preference ordering over slots. With respect to such preferences, and an arbitrary list of orderings $(O_s)_{s \in S}$, FEHE allocates flights to slots as follows.

In round 1 of the algorithm, construct a directed graph in which (1) each slot s points to the first flight in the ordering O_s , and (2) each flight points to its most preferred slot. For each cycle in this graph—at least one must exist—permanently assign each flight in the cycle to the slot at which it points; remove those flights and slots from future rounds of the algorithm.

Subsequent rounds repeat this process among remaining slots and flights: construct a directed graph in which (1) each remaining slot s points to the earliest remaining flight according to the ordering O_s , and (2) each flight points to its most preferred remaining slot. For each cycle in this graph—at least one must exist—permanently assign each flight in the cycle to the slot at which it points; remove those flights and slots from future rounds of the algorithm.

Based on this description, Proposition 1 is straightforward. TradeCycle algorithm runs a FEHE algorithm where orderings and preferences are constructed as follows. For each slot s , O_s lists $\Pi^{-1}(s)$ first, if s is initially occupied by some flight. Next appear each of the flights belonging to the airline that owns slot s , if any exist, ordered by their position in the Initial Landing Schedule. Finally appear the flights of the remaining airlines, also ordered by their position in the Initial Landing Schedule. Preferences are such that each flight f most prefers to arrive at its ETA, e_f ; second best is slot $e_f + 1$, etc., while all slots earlier than e_f can be placed (arbitrarily) at the bottom of the preference ordering.¹⁹

¹⁹It is straightforward to see that a flight f can never be allocated to a slot later than the one it occupies in the original landing schedule; therefore the specification of preferences below this slot are irrelevant in the execution of the algorithm.