Modelling and Solving a Bi-Objective Airport Slot Scheduling Problem

Konstantinos N. Androutsopoulos[a*], Eleftherios G. Manousakis[a], Michael A. Madas[b]

[a] Department of Management Science and Technology, School of Business, Athens University of Economics and Business,76 Patission Str., Athens 104 34, Greece.

[b] Department of Applied Informatics, School of Information Sciences, Information Systems and e-Business (ISeB) Laboratory, University of Macedonia, 156 Egnatia Str., 546 36 Thessaloniki, Greece.

Abstract

The strategic airport slot allocation problem concerns the scheduling of airlines' requests for landings and take-offs at congested airports for a series of days within a given scheduling season. Relevant scheduling models dealing with the strategic airport slot allocation problem have employed various combinations of the total schedule displacement criterion with several variations of acceptability metrics. However, most variations of schedule displacement pursued in existing literature do not thoroughly capture the real-world scheduling practice, and, most importantly, do not guarantee the allocation of acceptable/tolerable or viable displacement among competing airlines' slot requests. In this paper, we propose the formulation of the strategic airport slot allocation problem as a bi-objective resource constrained project scheduling problem with partially renewable resources and non-regular objective functions. We employ two non-regular performance criteria: i) the total earliness-tardiness and ii) a dispersion measure aiming to alleviate over-displaced requests. A novel hybrid heuristic algorithm integrating the Objective Feasibility Pump (FP) algorithm with the Large Neighborhood Search technique (LNS) is proposed.We generate a set of new problem instances originating from the patterns of a data set of actual slot requests for a Greek Regional Airport (GRA) to assess the performance of the algorithm. The computational results indicate that the proposed algorithm is reasonably accurate, and it has the capability to approximate the entire efficient frontier of the problem.

Keywords*: heuristics; transportation; multiple criteria analysis; project scheduling; airport slot allocation*

[*]Corresponding author: kandro@aueb.gr

1. Introduction

The strategic airport slot allocation problem involves the scheduling of airlines' requests for landings and take-offs at congested (i.e., schedule coordinated) airports within a given scheduling season (e.g., approximately six months). It usually pertains to airports subject to rules and regulations issued by IATA (IATA, 2014) setting up a structured scheduling framework. The daily scheduling period of a coordinated airport is divided into time intervals of a fixed length (e.g., 5 minutes) called *slots*. An airline's slot request involves the permission to use the airport for landing or taking-off within a requested time interval (slot) for a series of days within the scheduling season. The schedule coordination authority (usually an independent national committee responsible for slot allocation at all schedule coordinated airports) collects all slot requests (i.e., requested slots) and attempts to provide a viable allocation of slots (i.e., initially allocated slots) to the airlines' requests without violating certain regulatory rules and operational constraints. As part of this initial allocation process, the appointed schedule coordinator either confirms or comes up with a revised (rescheduled) proposal of the airlines' requests. Further refinements or adjustments of the initial allocation outcome are pursued through biannual worldwide scheduling conferences, where airlines, airports and schedule coordination authorities seek to achieve some convergence on their global airport network schedules by means of slot exchanges or re-timings.

The size of the problem - over 30,000 arrivals and departures per scheduling season for a medium-sized coordinated airport (Ribeiro et al., 2018; Zografos et al., 2018) - is further complicated by a set of scheduling rules and priority guidelines setting different priority classes for slot requests pertaining to historic slot usage rights, change-to-historic, new entrants etc. As a result, the main challenge for the respective decision maker (i.e., schedule coordination committee / coordinator) is to propose a master schedule for all competing airlines that will be feasible and operationally viable. A time interval allocated to a slot request can be considered viable if its deviation from the requested time interval can be effectively incorporated into the master flight schedule of the corresponding airline, subject to connecting flights, turnaround times and airport capacity restrictions. Hence, the airport slot allocation problem resembles a scheduling problem, henceforth called Airport Slot Scheduling Problem, which aims to determine a viable schedule of the airlines' slot requests.

A review of scheduling models dealing with the strategic airport slot allocation problem can be found in (Zografos et al., 2017). A typical criterion that has been widely used to express the viability of a schedule of slot requests is total schedule displacement for a single airport (Zografos et al., 2012, Ribeiro at al., 2018; Zografos et al., 2018) or a network of airports (Castelli et al., 2012; Pellegrini et al., 2012; 2017; Benlic, 2018). The schedule displacement of a slot request is

defined as the (absolute value of the) deviation of the allocated time interval from the corresponding requested time interval. Hence, the total schedule displacement defines the sum of displacements imposed on all slot requests. A criticism on the proposed displacement metric is that it does not take into account the number of days that each slot request appears regularly within the scheduling season (Zografos et al., 2017). As a result, a 15-minute displacement of a slot request is counted only once in the total displacement even if several instantiations of the specific slot request apply in multiple days of the scheduling season.

The minimization of total schedule displacement, as a single optimization criterion, may not be sufficient to build a viable schedule of slot requests. As a matter of fact, a solution ensuring minimum total displacement does not necessarily guarantee the allocation of reasonable, in the sense of acceptable/tolerable, fair or operationally viable displacement among competing airlines' slot requests. For example, a portion of slot requests may be over-penalized with exceptionally high displacement, far more than the acceptance tolerance levels that the corresponding airline may be willing or able to accommodate. In response to this problem, more recent research capitalized on and expanded existing scheduling models with additional objectives expressing operational efficiency metrics (e.g., operational/queuing delays) (Corolli et al., 2014), fairness (Zografos and Jiang, 2016), as well as acceptability considerations (Pyrgiotis and Odoni, 2015; Jacquillat and Odoni, 2015; Ribeiro et al., 2018; Zografos et al., 2018). The latter category (i.e., acceptability), which falls within the main scope of our research, involves basically models combining total schedule displacement with other variations of acceptability metrics. Some new acceptability-related criteria introduced in the airport slot scheduling problem are the maximum displacement associated to the requests of a given schedule (Pyrgiotis and Odoni, 2015), the number of displaced or rejected slots (Ribeiro et al., 2018), as well as the number of unacceptably displaced slots (Zografos et al., 2018).

The new class of acceptability-oriented scheduling models basically aims to lexicographically minimize the total and maximum schedule displacement (or the number of displaced slots) metrics. Zografos et al. (2018) proposed a bi-objective single-airport slot scheduling model for studying the trade-off between total and acceptable schedule displacement, with the latter being alternatively modeled as either the maximum schedule displacement or the number of slot requests assigned with "unacceptably" high displacement beyond maximum acceptable thresholds (called *violated slot assignments*). The authors have demonstrated that substantial improvements in slot acceptability and subsequently the actual slot utilization in a small coordinated airport can be achieved without "sacrificing" a lot in terms of total schedule displacement as a measure of the overall scheduling efficiency. However, the computation of this metric requires a deep insight into the actual tolerance limits of airlines per slot request which is a rather unrealistic assumption. In

other words, certain assumptions are essentially required in order to determine the maximum acceptable thresholds above which allocated slots are considered “unacceptable” or “violated” as compared to their originally requested slot times. Besides, these acceptability thresholds may vary with airlines or even per request (for the same airline), which renders the specification of all these airline-specific or request-specific information rather difficult. In a similar context, Ribeiro et al. (2018) proposed a multi-objective, single-airport, slot scheduling model fully complying with the complex set of priority classes envisaged by the IATA slot allocation framework. The model involves four weighted objectives, namely the number of rejected slots, the maximum and total schedule displacement, and finally the number of displaced slots. It has been shown that it is possible to accommodate all slot requests even at a medium-sized airport, while simultaneously reducing the maximum and total schedule displacement objectives, as well as the number of displaced flights. Both recent research efforts discussed above have successfully proven that there are ample and beneficial opportunities for integrating scheduling efficiency objectives with acceptability considerations, while substantial gains in various acceptability metrics may come at reasonable cost in terms of total schedule displacement.

In this paper, we propose a bi-objective single-airport slot scheduling model with simultaneous scheduling efficiency and acceptability considerations. We show that the proposed model is a resource constrained project scheduling problem with partially renewable resources and non-regular objective functions. First, we use an enhanced scheduling criterion called “aggregate displacement” and measured as the product of the displacement of a given slot request with the number of days in which the request regularly applies. This new variation of total schedule displacement that was introduced by Ribeiro et al (2018) is more realistic on the grounds that it “penalizes” the entire series of slots or movements associated to a given request. As a result, a multiplier effect will be applied on the displacement of requests involving multiple daily or weekly instantiations, a fact that better resembles the real-world scheduling practice. The second scheduling criterion involves a new acceptability measure that can be employed even in the absence of microscopic airline-specific or request-specific information about maximum tolerance limits. In particular, we incorporate the *squared aggregate displacement* metric, expressing the square of the former scheduling criterion. The underlying motivation is that the total aggregate displacement acts as a measure of central tendency that cannot actually eliminate the possibility of having few but exceptionally highly displaced slot requests. In that respect, the squared displacement metric will complement the objective function with a dispersion/variability measure that will penalize the assignment of large displacements. The latter has significant managerial implications. More specifically, over-displaced slots cannot be effectively incorporated into airline’s master flight schedules, hence being underutilized or even not utilized at all. As a result, extremely scarce resources/slots (especially in congested airports) remain unused or underutilized

due to misallocation problems. The squared displacement criterion introduced in our model alleviates the over-displacement of slot requests, hence promoting airline's acceptability, operational viability, while simultaneously increasing the actual utilization of scarce airport slots. Furthermore, we propose a novel hybrid heuristic algorithm that integrates the Objective Feasibility Pump (FP) algorithm into with the Large Neighborhood Search technique (LNS) for approximating Pareto optimal solutions. The proposed solution algorithm is tested on various problem instances generated from a data set of actual slot requests for a Greek Regional Airport (GRA). The computational results indicate that the proposed algorithm is reasonably accurate, and it has the capability to approximate the entire efficient frontier of the problem.

The remainder of this paper is structured into five sections. Section 2 defines the problem and describes its main characteristics and the underlying model formulation. Section 3 discusses the relevant literature review on addressing relevant scheduling problems. Section 4 presents the proposed solution approach, while Section 5 elaborates on the generated test problems and presents the computational results of the proposed methodology. Finally, Section 6 provides the concluding remarks of the paper and discusses some future research directions.

2. Problem Definition and Formulation

2.1 Problem Definition

The planning horizon of the proposed scheduling problem lasts for a series of calendar days denoted by D (usually six months in practice). The use of the airport for an arrival or departure at a specific time interval on a given day is called a *movement*. Each calendar day is divided into time intervals (named *coordination time intervals*) of fixed and equal length denoted by $T = \{0, 1, \dots, n-1\}$. For instance, if the airport is open on a daily basis for 20 hours (e.g., from 04:00 to 24:00) and the length of the coordination time intervals is five minutes, then the daily time horizon involves 240 coordination time intervals denoted by $T = \{0, 1, \dots, 239\}$, where 0 represents time interval [04:00,04:05), 1 represents [04:05,04:10), etc. Each *slot request* r is associated to: i) a series of movements M_r (either arrivals or departures) each one requested on a different day but on the same time interval, ii) the requested arrival (or departure) time $\tau_r \in T$, and iii) the set of calendar days $D_r \subseteq D$ of the associated movements M_r. Let $a_{\delta r}$ be a parameter that takes value 1 if request r is applicable on day $\delta \in D$ (i.e., $\delta \in D_r$), and zero otherwise. It is worth underlining that the requested coordination interval τ_r of a slot request r is applicable to all calendar days in D_r, i.e., the airline requests the same coordination interval for all movements in M_r. The set of slot requests of all airlines is denoted by R_0. The subset of requests that involve arrivals is denoted by $R_1 \subseteq R_0$ and the subset of requests involving departures by $R_2 \subseteq R_0$. Scheduling slot requests in R_0 involves assigning each one of them to a coordination time interval in T. Since all movements $m \in$

M_r of request r are scheduled at the same coordination time interval, assigning a request r to a coordination time interval τ (as close as possible to the one requested) implies that all movements $m \in M_r$ are assigned to time interval τ.

A schedule of the slot requests in R_0 is denoted by σ and it is defined as the image of set R_0 to T,$\sigma : R_0 \to T$. Hence, the image $\sigma(r)$ of $r \in R_0$ is a time interval in T. A partial schedule $\sigma^{R'}$ includes the time intervals assigned only for a subset of requests $R' \subseteq R_0$. Scheduling slot requests involves two basic categories of constraints: i) airport capacity and ii) slot requests precedence constraints. Airport capacity constraints account for the so-called declared capacity levels aiming to control (or limit) the number of slots available per unit of time at schedule coordinated airports. These are expressed in the form of the number of aircraft movements that can be accommodated by the runway system, often representing the most constraining factor of the airport system. Other capacity determinant factors such as terminal or apron capacity may be equally binding and can enrich the scope and measurement basis of declared capacity, but are not addressed in our model. The proposed scheduling problem involves three types of capacity constraints, each one denoted by $c \in C = \{0, 1, 2\}$, where 0 denotes the set of capacity constraints on the total number of movements, 1 on arrivals, and 2 on departures. Any capacity constraint of type $c \in C$ imposes a maximum number ($u_c^{\delta\tau}$) of requested movements of type c that can be scheduled within time period $T_c^{\tau} :=[\tau, \tau + t_c)$ of fixed length $t_c > 0$ starting at time τ, on calendar day δ. Note that each of these capacity constraints must be satisfied for any $\tau \in [0, n - t_c + 1)$ throughout all calendar days of D. The emerging constraints are called Rolling Capacity constraints, since the fixed length time period t_c (known in advance, e.g., 60 min) over which the maximum number of movements is imposed, rolls throughout the entire scheduling period of a day. Request r consumes b_{cr} units of the airport capacity, for $r \in R_0$. If r refers to an arrival, then b_{cr} is equal to 1 for $c = 0$ and $c = 1$ and zero otherwise. If r refers to a departure, then b_{cr} is equal to 1 for $c = 0$ and $c = 2$ and zero otherwise.

The precedence constraints of the proposed scheduling problem emerge from the existence of pairs of linked requested movements (arrival-departure): i) operated by the same aircraft or ii) used by common passengers (connected flights). Thus, an arrival slot request r_1 is considered linked to a departure slot request r_2 when either the corresponding arrival and departure movements are connected (i.e., passengers of each of the arriving flights associated to r_1 are bound to travel with the corresponding departing movement r_2) or they are operated by the same aircraft on the same day. We denote with $\wp \subseteq R_1 \times R_2$ the set of pairs of linked requests $(r_1, r_2),\ r_1 \in R_1,\ r_2 \in R_2$. A minimum time difference ($t_{r_1 r_2}$) is imposed between the time intervals assigned to any pair of linked requests r_1 and r_2. It should be noted that the precedence constraints due to connecting flights are not particularly applicable to our case, but would be certainly useful in large hub

airports, where connecting flights may variously affect the feasibility of the schedule. For example, assume the case of an aircraft (or passengers) arriving as a "feeder traffic" from several domestic "satellite" airports to a national hub airport and then departing for an international destination. Upon availability of data related to connecting flights, additional passenger or aircraft connectivity constraints merit further investigation. In a simpler form of a regional airport typically without connecting flights (in our case), precedence constraints pertain exclusively to the so-called turnaround time constraints. In that respect, the time interval assigned to a departure request should have a minimum time lag from the corresponding slot assigned to the linked arrival request to account basically for the aircraft preparation (upon arrival) for the subsequent departure flight. The following condition must hold for every pair of linked requests (r_1,r_2) sharing the same aircraft: $\sigma(r_1) + t_{r_1 r_2} \leq \sigma(r_2)$.

Furthermore, additional constraints emerge from IATA slot scheduling rules providing higher level of priority to certain classes of slot requests (e.g., historic usage rights or grandfathered slots, new entrants) or secondary allocation criteria taking into consideration the type of service (e.g., scheduled, charter), the type of route/flight (short-haul, long-haul), the aircraft size (e.g., narrow-body, wide-body), the length of slot request (e.g., extending to year-round operations) or even social mobility criteria (e.g., connecting underserved small communities). The existence of these constraints creates a series of classes of slot requests with different scheduling priorities. Each class of slot requests constitutes a slot allocation problem. This means that our algorithm can be used sequentially to separately optimize each of these classes with respect to their priority. Therefore, with no loss of generality, we do not incorporate these constraints into our model.

Ideally, each request r would be scheduled at the corresponding requested time interval τ_r. However, due to the airport capacity and precedence constraints, such an ideal scheduling would be rather impossible for congested airports. Hence, part of the requests is inevitably assigned to time intervals different from those requested. If a request $r \in R_0$ is scheduled at the coordination time interval $t \in T$ (i.e., $\sigma(r) = t$), then the *displacement* of r (i.e., the difference $|t - \tau_r|$ between the requested time interval τ_r from its schedule time $t \in T$) is denoted by f_{rt}. This metric expresses the deviation of the allocated time from the requested only once although this deviation is actually encountered for every movement in M_r. The first criterion used in this work is *aggregate displacement*, which accounts for the total displacement of all movements associated to a slot request. The second criterion used in this formulation is the *squared displacement* which is expressed by the square of f_{rt}. This criterion aims to penalize the assignment of large displacements to slot requests, hence leading to a more uniform distribution of displacement to slot requests.

In conclusion, the objective of the proposed airport slot scheduling model is to assign each request to a single time interval with view to the minimization of both total aggregate and squared displacement, subject to the rolling capacity constraints and the precedence constraints among interdependent slot requests.

Tables 1 and 2 provide a concise presentation of the problem notation that will be thereafter employed in our model formulation in the subsequent section.

Table 1. Sets notation.

Set	Definition	Index
D	Set of calendar days (planning horizon)	δ
$T = \{0, 1, \dots, n-1\}$	Set of coordination time intervals of fixed and equal length of a day $\delta \in D$	t
R_0	Set of all slot requests	r
$R_1 \subseteq R_0$	Subset of requests involving arrivals	r
$R_2 \subseteq R_0$	Subset of requests involving departures	r
M_r	Set of movements (either arrivals or departures) associated with slot request r	m
$D_r \subseteq D$	Subset of calendar days associated with slot request r	δ
$C = \{0, 1, 2\}$	Set of movement types (0 denotes all movements, 1 arrival movements and 2 departure movements)	c
$\wp \subseteq R_1 \times R_2$	Set of pairs of linked requests(r_1, r_2), $r_1 \in R_1$, $r_2 \in R_2$	(r_1, r_2)

Table 2. Problem notation.

Symbol	Definition
$\tau_r \in T$	Requested coordination time interval for arrival (or departure) of slot request r (same for all $\delta \in D_r$)
$\sigma : R_0 \to T$	A schedule of the slot requests in R_0
$\sigma^{R'}$	A partial schedule including time intervals assigned for a subset of requests $R' \subseteq R_0$
$T_c^\tau = [\tau, \tau + t_c)$	A time period starting at time τ and ending after a fixed length $t_c > 0$
$u_c^{\delta\tau}$	Maximum number of movements of type c that can be scheduled within time period T_c^τ on calendar day δ
$t_{r_1 r_2}$	The minimum number of coordination time intervals allowed between the $\sigma(r_1)$ and $\sigma(r_2)$
f_{rt}	The schedule displacement of r (i.e., the difference $\lvert\sigma(r) - \tau_r\rvert$ between the requested time interval τ_r and the allocated time interval $\sigma(r)$)
$a_{\delta r}$	1 if request r is applicable on day $\delta \in D$, and 0 otherwise
b_{cr}	Units of capacity of type c required for $r \in R_0$

2.2 Mathematical Formulation

We define a set of binary variables $x_{rt} \in \{0,1\}, r \in R_0,\ t \in T$, that take value 1 if slot request r is assigned to time interval t, and zero otherwise. The mathematical formulation of the problem is provided through (1)-(6). Constraints (3)-(6) are identical to those used by Zografos et al. (2012).

(P1) $\boldsymbol{Min}(Z_1, Z_2)$

$$Z_1 = \sum_{r \in R_0} \sum_{t \in T} |M_r| f_{rt} x_{rt} \tag{1}$$

$$Z_2 = \sum_{r \in R_0} \sum_{t \in T} (f_{rt})^2 x_{rt} \tag{2}$$

Subject to:

$$\sum_{t \in T} x_{rt} = 1, \qquad r \in R_0 \tag{3}$$

$$\sum_{r \in R_0} a_{\delta r} b_{cr} \Big(\sum_{t \in T_c^\tau} x_{rt} \Big) \leq u_c^{\delta\tau}, \quad c \in C, \qquad \delta \in D, \qquad \tau \in T \tag{4}$$

$$\sum_{t \in [0,k)} x_{r_1 t} + \sum_{t \in [k - t_{r_1 r_2}, n)} x_{r_2 t} \leq 1, \quad (r_1, r_2) \in \wp, \qquad k \in [t_{r_1 r_2}, n) \tag{5}$$

$$x_{rt} \in \{0,1\}, r \in R_0,\ t \in T \tag{6}$$

Objective function (1) represents the total aggregate displacement, while objective function (2) expresses the total squared displacement. Constraints (3) imply that every slot request must be assigned to exactly one time interval in T. Constraints (4) impose the rolling capacity constraints throughout the entire planning horizon D, while constraints (5) state that every pair of linked slot requests must be scheduled at times which are at least $t_{r_1 r_2}$ time intervals apart (i.e. turnaround time).

2.3 Problem Characteristics

The proposed problem can be reduced to a Generalised Assignment Problem (GAP) known to be NP-Hard (Sahni and Gonzalez, 1976). By setting the turnaround time equal to zero (i.e., deactivating constraint (5)) and the length of the time periods T_c^τ equal to one coordination time interval, the model ($\boldsymbol{P1}$) transforms to a bi-objective GAP. This finding implies that real-world sized instances of the problem can only be solved by heuristic algorithms.

A major feature of the problem is that both objective functions (1) and (2) are based on the displacement metric, which essentially constitutes an earliness-tardiness performance measure. Hence, both objective functions are *non-regular* (a *regular* function exhibits a non-decreasing property with respect to the start-times of activities).

Another significant feature of the proposed scheduling model relates to the use of the rolling capacity constraints. We show below (Figure 1) that using the conventional form of capacity constraints instead of the rolling capacity constraints creates unrealistic capacity measures (Zografos et al., 2012). Assume an airport slot scheduling example where 20 slot requests must be scheduled for a single day on the daily time horizon 1-18. The capacity constraints allow up to 4 requests per 5 minutes time period and 10 requests per 15 minutes time period. The first row of cells in Figure 1 represents the time scale of the daily scheduling horizon (1-18), each one representing a 5-minute time period. For simplicity, we assume that no precedence constraints exist. The second row of cells represents a solution to the specific problem, where each cell includes the number of slot requests scheduled at each 5-minute time period of the scheduling horizon (empty cells correspond to zero requests). It can be derived from Figure 1 that although the solution satisfies the capacity constraints of the problem for the 5-minute time periods, it fails to satisfy the capacity constraints for the 15-minute time periods (e.g., 5th and 6th 15-minute intervals).

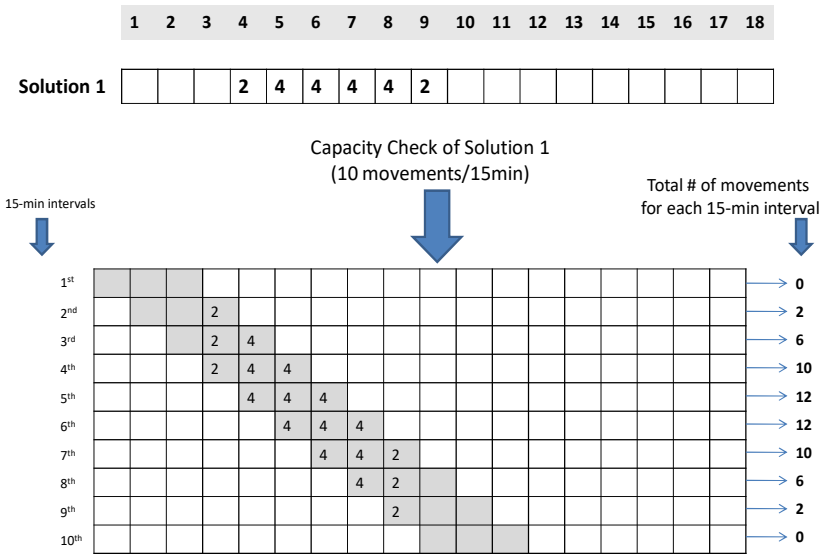

Figure 1. Schedule and capacity check for the solution of the example.

The proposed airport slot scheduling problem belongs to the category of Resource Constrained Project Scheduling Problems with multiple partially renewable resources. While a renewable resource is available with a certain capacity per time unit, a partially renewable resource is available with a total capacity defined over a period of time. Hence, the usage of a renewable resource on a given time unit does not affect the resource availability on any other time unit of the scheduling period. On the contrary, the usage of a unit of a partially renewable resource implies that this specific resource unit cannot be reused in any other time unit. In a formal context, a partially renewable resource ρ is available for a given set of time periods $\Pi(\rho) = \{P(\rho, \pi),\ \pi = 1, .., \mathcal{M}\}$ and is defined by the total amount of resource units $\kappa_{\rho\pi}$ that can be consumed within each

of these time periods (i.e., the total amount of resource units that can be used within time period $P(\rho, \pi)$ is equal to $\kappa_{\rho\pi}$).

In general, a resource constrained project scheduling problem aims to schedule each activity so that the capacity and precedence constraints are satisfied, while simultaneously optimizing a certain scheduling criterion (e.g., the makespan of the project). In the airport slot scheduling problem, we treat each slot request as a project activity with duration equal to a single time unit (coordination time interval). Hence, a slot request scheduled at coordination time interval t implies that the service of the corresponding movements starts at time t and finishes at time t+1. The airport constitutes the physical resource of the problem. However, it is modeled as multiple different resources depending on the type of movement and the calendar day $\delta \in D$ of the planning horizon of the problem. Hence, we define a set of resources $\Re = \{\rho_{c\delta},\ c \in \{0,1,2\},\ \delta \in D\}$. Servicing a slot request r consumes one unit from each of the resources $\rho_{0\delta}$ and either $\rho_{1\delta}$ or $\rho_{2\delta}$ with $\delta \in D_r$ depending on the type of movement that r refers to (1 for arrival or 2 for departure). The rolling capacity constraints of the problem express the resource availability over periods of time (rather than per unit of time) and thus, the resources of the problem can be characterized as partially renewable. Assume, for instance, that the airport capacity on day δ of the planning season is defined for periods of 30 min (i.e., six 5-minutes coordination intervals) and it is given as follows: 10 arrivals, 12 departures, and 15 movements in total. The airport is available on a daily basis from 04:00-24:00. Hence, in every possible 30-minute time period of day δ, the number of scheduled arrivals, departures and total movements that can be accommodated should not exceed 10, 12, and 15 movements, respectively. Representing these airport capacity constraints as partially renewable resource constraints, we define resources $\rho_{c\delta}$ for $c = \{0, 1, 2\}$ each one defined on the set of time periods $\Pi(\rho_{c\delta}) \coloneqq \{P(\rho_{c\delta}, j), j = 0,1,2,..,234\}$, where each time period $P(\rho_{c\delta}, j)$ corresponds to a time period of 6 coordination time intervals starting at the j^{th} coordination time interval of the daily time horizon. Figure 2 illustrates the relevant time periods $P(\rho_{c\delta}, j)$. The quantity of available units $\kappa_{0\delta}$, $\kappa_{1\delta}$, $\kappa_{2\delta}$ of resources $\rho_{c\delta}$ per time period $P(\rho_{c\delta}, j)$ are equal to 15, 12, and 10 respectively. Hence, airport capacity constraints are equivalent to the corresponding constraints on the partially renewable resources $\Re = \{\rho_{c\delta},\ c \in \{0,1,2\},\ \delta \in D\}$.

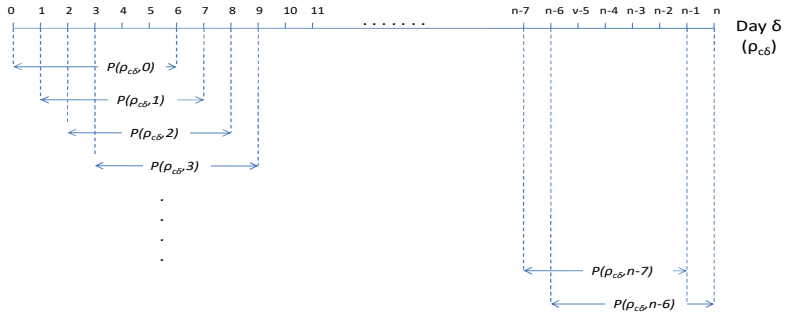

Figure 2. Time periods $\Pi(\rho_{c\delta})$ (i.e., of six time intervals length) within a day that the resource is available.

3. Literature Review

The proposed scheduling problem involves two major features: i) two non-regular criteria, and ii) partially renewable resource constraints. A review of the research on multi-criteria resource constrained project scheduling (Multi-criteria RCPSP) can be found in (Ballestin and Blanco, 2011). In general, one can find various different formulations for the multi-criteria RCPSP depending on the objective functions used (e.g., the project makespan; the project earliness or lateness; the total weighted start time of the activities; the number of tardy activities) and the type of resources (renewable vs. non-renewable). In multi-criteria RCPSP with regular objective functions, the set of Pareto optimal solutions is contained within the set of active schedules (Ballestin and Blanco, 2011), hence rendering the problem tractable by any heuristic working with priority lists and serial schedule generating schemes. However, searching in active schedules is not sufficient to find the efficient frontier of a RCPRP with at least one non-ROF (Ballestin and Blanco, 2011). Various multi-objective meta-heuristics have been developed to tackle the multicriteria RCPSP with ROFs or the combination of ROFs (e.g., makespan) with at least one non regular objective function (non-ROF) (Viana and Souza, 2000; Al-Fawzan and Haouari, 2005; Kazemi and Tavakkoli-Moghaddan, 2008; Ballestin and Blanco, 2011; Gomez et al., 2014).

Other previous related work includes research on: i) the minimum total/weighted earliness-tardiness (i.e., single non-regular objective function) project scheduling problem with renewable resources (Vanhoucke et al., 2001; Ballestinand Trautman, 2008) and ii) the minimum makespan (i.e., single regular objective function) project scheduling with partially renewable resources (Bottcher et al., 1999; Schirmer and Drexl, 2001; Alvarez-Valdes et al., 2015). To the best of the authors' knowledge, no research work is available on project scheduling problems with non-ROFs

and partially renewable resources. In what follows, we provide an overview of the literature on both types of scheduling problems and highlight the contribution of this paper.

3.1 Partial Renewable Resource Constrained Project Scheduling

The partially renewable resources were introduced and studied by Bottcher et al. (1999). The resource constrained problems with partially renewable resources that have been studied in the literature, involve the minimization of the project makespan (RCPSP/π). Bottcher et al. (1999) adapted two alternative solution approaches for the RCPSP in order to solve the RCPSP/π: i) the depth-first Branch and Bound algorithm of Talbot and Patterson (1978) and ii) the Greedy Randomized Adaptive Search Procedure (GRASP). In the former solution approach, each node of the branch and bound tree corresponds to a partial solution (i.e., partial schedule) of the problem. Branching involves selecting and scheduling one of the activities (that can be feasibly scheduled) extending the current partial schedule to a new feasible (in terms of resource and precedence constraints) partial schedule. This procedure is further enhanced with two new feasibility bounds associated to the estimation of the minimum resource consumption of the remaining activities.

The GRASP algorithm (Bottcher et al., 1999) involves a randomized multi-pass serial scheduling scheme. A multi-pass serial scheduling scheme involves iterating a (semi-randomized) serial scheduling algorithm to generate different solutions. A serial scheduling algorithm performs a number of iterations in which a single activity is selected and scheduled at the earliest possible start time such that precedence and resource constraints are satisfied. Given that more than one activity may be feasibly scheduled at each iteration, the serial scheduling algorithm involves a deterministic priority rule with which the choice of a single activity is performed. The GRASP algorithm proposed in (Bottcher et al., 1999) constructs a feasible schedule by iteratively selecting and scheduling an activity j^* at a time period τ^*. This choice is made probabilistically from a set of pairs of activities and their alternative schedule times (j, τ) that satisfy all precedence and resource constraints. The selection of (j^*, τ^*) from the decision set of feasible alternative choices (j, τ) is performed through a randomized process called regret-based biased random sampling (Drexl and Grunewalt, 1993). Bottcher et al. (1999) employed and tested various priority rules including eight rules designed especially for assessing capacity feasibility for partially renewable resources.

Alvarez-Valdez et al. (2006) propose a scatter search algorithm for the RCPSP/π. Their algorithmic approach involves three phases. The first phase aims to reduce the solution region of the problem by applying certain filters (e.g., eliminating idle and non-scarce resources, narrowing the set of alternative finish times for a given activity). The second phase of the algorithmic approach aims to build an initial set of feasible solutions called Reference Set through a GRASP routine. The third phase of the algorithm applies the Scatter technique. Two or more of the solutions in the Reference

Set are iteratively selected and combined in order to produce new solutions of higher quality. In a more recent work, Alvarez-Valdez et al. (2006) proposed an algorithmic approach which involved only the first two phases, while employing a path relinking technique in the third phase.

3.2 Minimum (weighted) earliness-tardiness resource constrained scheduling

Existing work on the weighted earliness-tardiness resource constrained project scheduling problem is very limited. Vanhoucke et al. (2001) developed a two-phase exact recursive procedure that solves the resource unconstrained weighted earliness-tardiness project scheduling problem. At the first phase, a forward pass is performed through the activity network, scheduling each activity at the requested due time or later. At the second phase, a recursive process is applied in which the initial start time of certain activities is left-shifted in order to reduce the weighted earliness-tardiness objective function. This recursive procedure was integrated in a branch and bound algorithm dealing with the weighted earliness-tardiness RCPSP (Vanhoucke et al., 2001). The role of the proposed recursive procedure was to determine a lower bound to each node of the branch and bound tree.

Ballestin and Trautmann (2008) proposed a two-phase Iterated Local Search heuristic for the weighted earliness-tardiness resource constrained project scheduling problem with renewable resources. The first phase involves a multi-pass heuristic algorithm that schedules the project activities at times as close as possible to the relevant due dates taking into account only temporal/precedence constraints (ignoring resource capacity constraints). Although the emerging solutions are expected to violate one or more resource constraints, they tend to have low weighted earliness-tardiness value. Each of the emerging solutions is used to build a new priority list of the activities based on their starting time (in non-decreasing order). The activities are then rescheduled based on the new priority list (i.e., as close as possible to the initial schedule) considering both temporal/precedence and resource constraints. The newly formed solutions are further processed through four local search procedures aiming to reduce the weighted earliness tardiness by either delaying early activities (scheduled before due date) or left-shifting late activities (scheduled after due date).

The airport slot scheduling problem modeled in this paper introduces a new class of project scheduling problems that combines non-ROFs and partially renewable resources. To the best of the authors' knowledge, there is no research work conducted on this category of multi-criteria project scheduling problems with non-ROFs and partially renewable resources. This class of scheduling problems requires innovative efficient and effective solution approaches capable of addressing multiple non-ROFs and the scheduling implications arising due to partially renewable resources at

the same time. The aim of this paper is to fill in this gap in relevant literature by proposing a new trustworthy solution approach for the airport slot scheduling problem.

4. Solution Approach

Solving the proposed bi-objective scheduling problem involves determining the alternative Pareto-optimal solutions. We propose a solution approach, henceforth called ***Feasibility Pump Guided ε-Constraint Method*** (***FPεCM***), that: i) uses the ε-Constraint Method (Haimes 1973) to decompose the bi-objective problem to a series of constrained single objective airport slot scheduling problems and ii) a novel hybrid heuristic algorithm called ***Feasibility Pump-Guided Hybrid Algorithm*** (***FPHA***) to address each of the emerging single objective constrained scheduling problems.

The proposed FPHA has the structure of a Large Neighborhood Search (LNS) algorithm. The search starts from a feasible solution determined by a scheduling algorithm called ***Feasibility Pump-Guided Schedule Generating Heuristic*** (***FPSGH***). The emerging initial solution is used as the starting point of the local search performed by the ***Feasibility Pump–Guided Large Neighborhood Search*** (***FPLNS***). A strong novel feature of the FPHA is that it incorporates the Feasibility Pump technique (FP) in both the Schedule Generating Heuristic (SGH) and the LNS components. The FPLNS used the FP technique to reconstruct the destroyed part of the solution, while the FP integration in the SGH aims to strengthen its capability to determine not only feasible but also high-quality solutions for hard constrained instances of the problem. This ensures a better starting solution for the FPLNS component. We now proceed to the exposition of ***FPεCM*** framework and its components.

4.1 Applying the ε-Constraint Method

In this paper, the ε-Constraint Method is used in *FPεCM* to generate the Pareto frontier heuristically. Moreover, it is also used as an exact method to generate the Pareto frontier for the instances that were created in order to benchmark *FPεCM* against Pareto optimal solutions. We note that only small instances of this NP-hard problem may be solved to optimality and this is the motivation for the FPεCM approach.

The application of the ε-Constraint Method for the proposed bi-objective airport slot scheduling problem ($P1$) involves the following steps:

i) Determine the Z_1-optimum solution by solving ($P1$) lexicographically using Z_1 as the primary objective function and Z_2 as the secondary objective function. The emerging lexicographic problem is denoted by ($P1'$). The Z_1 and Z_2 values of the emerging Z_1-optimal solution are denoted by Z_1^* and Z_2^o, respectively.

$(\boldsymbol{P1'})$ $\boldsymbol{lexmin}\ (Z_1(x),\ Z_2(x))$ (7)

Subject to:

Constraints (3)-(6)

ii) Determine the Z_2-optimum solution by solving $(P1)$ lexicographically once more by using this time Z_2 as the primary objective function and Z_1 as the secondary objective function. The emerging lexicographic problem is denoted by $(P1'')$. The Z_1 and Z_2 values of the emerging optimal solution are denoted by Z_1^o and Z_2^*, respectively.

$(\boldsymbol{P1''})$ $\boldsymbol{lexmin}\ (Z_2(x),\ Z_1(x))$ (8)

Subject to:

Constraints (3)-(6)

iii) We form a series of single objective constrained airport slot scheduling problems which aim to minimize objective function Z_2 under constraints (3)-(6) and constraining the objective function Z_1 to take values below a parameter $\varepsilon \geq 1$. The emerging family of single objective problems is denoted by $(P2)$. We create and solve these problems sequentially, starting with ε equal to Z_1^o. The solution of $(P2)$ is denoted by $x^*_{(P2)}$. Then, we set ε equal to $Z_1(x^*_{(P2)})$ and we solve $(P2)$ iteratively until $Z_1(x^*_{(P2)})$ becomes equal to Z_1^*.

$(\boldsymbol{P2})$ $\boldsymbol{Min} Z_2(x)$ (9)

Subject to:

Constraints (3)-(6)

$$\sum_{r\in R_0}\sum_{t\in T}|M_r|f_{rt}x_{rt} \leq \varepsilon - 1 \quad (10)$$

We now elaborate on implementation issues regarding the application of the above process. It is worth noting that problem $(P1')$ in step (i) is solved in two stages. At the first stage, we solve problem $(P3)$ below:

$(\boldsymbol{P3})$ $\boldsymbol{Min} Z_1(x)$ (11)

Subject to:

Constraints (3)-(6)

The optimal solution of $(P3)$ is denoted by $x^*_{(P3)}$. Then, at the second stage, we solve problem $(P4)$ described below:

$(\boldsymbol{P4})$ $\boldsymbol{Min} Z_2(x)$ (12)

Subject to:

Constraints (3)-(6)

$$\sum_{r\in R_0}\sum_{t\in T}|M_r|f_{rt}x_{rt} = Z_1\left(x^*_{(P3)}\right) \tag{13}$$

The optimum solution of $(P4)$ is denoted by $x^*_{(P4)}$. Hence, we set $Z_1^* = Z_1(x^*_{(P3)})$ and $Z_2^o = Z_2(x^*_{(P4)})$.

A similar approach is followed for addressing problem $(P1'')$, in which problems $(P5)$ and $(P6)$ are solved.

$(\boldsymbol{P5})$ $\boldsymbol{Min}Z_2(x)$ (14)

Subject to:

Constraints (3)-(6)

$(\boldsymbol{P6})$ $\boldsymbol{Min}Z_1(x)$ (15)

Subject to:

Constraints (3)-(6)

$$\sum_{r\in R_0}\sum_{t\in T}(f_{rt})^2x_{rt} = Z_2\left(x^*_{(P5)}\right) \tag{16}$$

The optimum solution of $(P6)$ is denoted by $x^*_{(P6)}$. Therefore, we set $Z_2^* = Z_2(x^*_{(P5)})$ and $Z_1^o = Z_1(x^*_{(P6)})$.

It is evident that problems $(P2)$-$(P6)$ share a common structure. In more detail, it can be verified that $(P2)$-$(P6)$ may emerge from the following generic problem $(P7)$.

$(\boldsymbol{P7})$ $\boldsymbol{Min}Z_i(x)$ (17)

Subject to:

Constraints (3)-(6)

$$Z_j \le z \tag{18}$$

where $i, j \in \{1,2\}$, $i \neq j$ and $z > 0$.

Problem $(P7)$ may take the form of any of the problems $(P2)$-$(P6)$ by setting appropriate values to indices i, j and parameter z. For instance, $(P7)$ takes the form of $(P2)$ by setting i to 2, j to 1, and z to $(\varepsilon - 1)$. Hence, an approach capable of solving problem $(P7)$ is adequate to tackle all problems $(P2)$-$(P6)$.

The proposed *FPεCM* applies the above steps (i)-(iii) and solves the emerging problems $(P2)$-$(P6)$. Based on our last observation regarding the association of $(P2)$-$(P6)$ with $(P7)$, FPHA has been designed to solve the generic constrained airport slot scheduling problem expressed by $(P7)$. The remainder of this section is devoted to the presentation of the proposed methodology in full detail.

It is worth noting that since we use a heuristic algorithm to deal with problem $(P2)$ in step (iii), we terminate this step when either $Z_1(x^*_{(P2)})$ becomes equal to the approximated Z_1^* or a predefined number of iterations is completed (it is possible that the heuristic algorithm cannot find $Z_1(x^*_{(P2)})$).

A strong feature of the *FPεCM* framework is that it is guided to search for Pareto optimal solutions throughout the entire Pareto frontier. In addition, it has the capability to focus its search for solutions on a specific part of the efficient frontier. In particular, the proposed solution approach may facilitate the determination of Pareto optimal solutions for which $Z_1(x)$ lies between a pair of predefined values ε_1 and ε_2 $(\varepsilon_1 < \varepsilon_2)$. This can be attained by skipping steps (i) and (ii), initializing $\varepsilon = \varepsilon_2$ and terminate when a solution with $Z_1(x)$ below or equal to ε_1 is determined (or a predefined number of step (iii) iterations have been completed).

4.2 FP-Guided Hybrid Algorithm (FPHA)

FPHA inherits the structure of the LNS technique that iteratively destroys and reconstructs part of a feasible solution (Saw, 1998). We implement this procedure depicted in Figure 3 as follows: i) an initial feasible solution is determined through the FP guided schedule generating routine FPSGH, and ii) a pre-specified number of iterations are executed in which the currently best solution found is partially destroyed (by randomly "unscheduling" selected slot requests) and rebuilt (FPLNS). Both steps involve the application of the Feasibility Pump method customized so as to transform a partial schedule of the problem to a feasible complete schedule. In the FPSGH, a scheduling procedure starts building a solution until it is halted (after a pre-specified number of iterations). Then, the Feasibility Pump routine undertakes the task of transforming the resulting partial solution into a feasible solution. The Feasibility Pump routine also plays the role of the repair operator of the FPLNS. In what follows, we provide a detailed schematic illustration of the FPHA heuristic algorithm and its components (Figure 3).

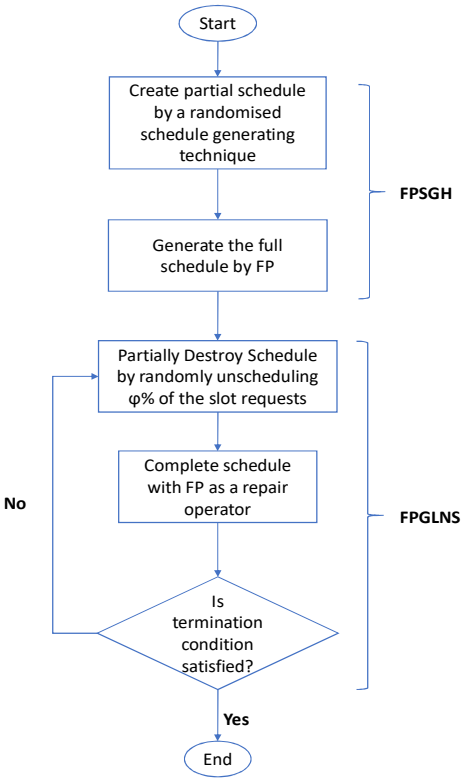

Figure 3. Flowchart of the proposed FPHA for solving problem (P7)

4.2.1 Feasibility Pump Routine

The aforementioned Feasibility Pump (FP) routine plays a significant role in the proposed algorithmic framework. It aims to complement a partial solution σ^{R^S} of problem $(P7)$ in which a predefined number (n_s) of the slot requests (denoted by R^s) have already been scheduled. Hence, the objective of the FP routine is to solve a restricted version of the constrained airport slot scheduling problem $(P7)$ formed by the remaining unscheduled requests (denoted by R^u), the binding precedence constraints and the updated remaining capacities $u_0'^{\delta\tau}$, $u_1'^{\delta\tau}$ and $u_2'^{\delta}$ for $\tau \in T$, $\delta \in D$. If the FP routine returns a feasible partial solution σ^{R^u} for the restricted constrained airport scheduling problem, then a new complete solution σ is formed by concatenating the partial solutions σ^{R^S} and σ^{R^u}.

In general, FP (Fischetti et al., 2005) is a heuristic method determining good quality feasible solutions to Mixed Integer Programming problems (MIPs). The method starts by solving the LP relaxation of the MIP. If the resulting solution x^* does not satisfy the integrality constraints, a sequence of iterations is initiated consisting of two major steps: i) the LP solution x^* (i.e., that

satisfies the functional constraints of the MIP) is rounded to its closest integer solution $\tilde{x}$ and ii) if the emerging integer solution $\tilde{x}$ is not feasible in the MIP (i.e., it does not satisfy the functional constraints), then a new LP solution x^* (as close as possible to $\tilde{x}$) is determined that satisfies the functional constraints of the MIP. Step (ii) is accomplished by solving the linear relaxation of a modified version of the original MIP, formed by: i) the functional constraints of the initial MIP and ii) an objective function that expresses the total deviation from the integer solution $\tilde{x}$. The process terminates when either the integer solution $\tilde{x}$ satisfies the MIP constraints or the LP solution x^* satisfies the integrality constraints. It is worth noting that if this is not achieved after a pre-specified number of iterations, the process is halted. The basic idea behind this technique is to build two converging trajectories of solutions. In this work, we implement the Objective Feasibility Pump heuristic (Achterberg and Berthold, 2007) which guides the procedure to not only feasible but also high-quality solutions. Hence, FP is used as a guide for both FPSGH and FPLNS for attaining feasibility.

We now provide the MIP formulation of the *Restricted Constrained Airport Slot Scheduling Problem* ($RP7$) formed by the unscheduled slot requests R^u on which the FP technique is applied. The ($RP7$) is expressed by (19)-(24).

($RP7$) $$\boldsymbol{Min} \sum_{r\in R^u}\sum_{t\in T} Z_i(x_{rt}) \tag{19}$$

Subject to:

$$\sum_{t\in T} x_{rt} = 1, \qquad r \in R^u \tag{20}$$

$$\sum_{r\in R^u}\sum_{t\in T_c^\tau} a_{\delta r} b_{cr} x_{rt} \le u_c'^{\delta\tau}, \quad c \in C, \qquad \delta \in D, \qquad \tau \in T \tag{21}$$

$$\sum_{t\in[0,k)} x_{r_1 t} + \sum_{t\in[k-t_{r_1r_2},n)} x_{r_2 t} \le 1,\ r_1, r_2 \in R^u,\ (r_1,r_2) \in \wp, \qquad k \in [t_{r_1r_2}, n) \tag{22}$$

$$\sum_{r\in R^u}\sum_{t\in T} Z_j(x_{rt}) \le \varepsilon - 1 \tag{23}$$

$$x_{rt} \in \{0,1\}, \qquad r \in R^u, \qquad t \in T \tag{24}$$

Where $\varepsilon > 1$, $u_c'^{\delta s}$ for $c = \{0, 1, 2\}$ represent the remaining capacity for movements of type c for the time period from time interval τ until $\tau + t_c$ on day δ. Index i can be equal to either 1 or 2. If i is set to 1 then j is set equal to 2, whereas if i is set to 2 then j is set equal to 1.

The FP routine developed for ($RP7$) involves the following steps. First, the LP relaxation of the problem is solved. The emerging LP solution x^* is rounded to the closest integer solution $\tilde{x}$ as indicated in (25).

$$\tilde{x} = \lfloor x^* + 0{,}5 \rfloor \tag{25}$$

The next step is to check whether the rounded solution $\tilde{x}$ is feasible to the $(RP7)$. If the solution $\tilde{x}$ is feasible, the procedure terminates. If not, then a modified version of $(RP7)$ denoted by $(RP7')$ is formulated and solved. The optimal solution of $(RP7')$ is used as the new x^*. The target value C^* depends on the best solution found so far (by FPHA). Hence, if σ_B is the currently best solution, then C^* is given by following formula (26).

$$C^* \coloneqq (1+\mu)\left|Z_i(\sigma_B) - Z_i(\sigma^{R^S})\right| \tag{26}$$

where $\mu \in (0,1)$. The formulation of $(RP7')$ is provided below.

$$(RP7') \quad \boldsymbol{Min} \sum_{r\in R^u}\sum_{t\in T} d_{rt} \tag{27}$$

Subject to:

Constraints(20)-(23)

$$\sum_{r\in R^u}\sum_{t\in T} Z_i(x_{rt}) \le C^* \tag{28}$$

$$d_{rt} \ge \tilde{x}_{rt} - x_{rt}, \;\; t \in T, \qquad r \in R^u \tag{29}$$

$$d_{rt} \ge x_{rt} - \tilde{x}_{rt}, \;\; t \in T, \qquad r \in R^u \tag{30}$$

$$0 \le x_{rt} \le 1, \;\; r \in R^u, \quad t \in T \tag{31}$$

Constraint (28) of $(RP7')$ aims to guide the FP process to construct a new complete feasible solution with an objective function that does not exceed the objective function value of the currently best solution by a factor of μ. This procedure is repeated until we find a feasible solution or a certain number of iterations is performed. In order to prevent the algorithm from solution cycling, we compare every new rounded solution with all rounded solutions found in previous iterations. If the new solution is identical with a previous one, a perturbation mechanism is triggered to randomly change some of the solution elements of the current solution.

4.2.2 FP-Guided Schedule Generating Heuristic (FPSGH)

The goal of the *FPSGH* is to build a feasible schedule for problem $(P7)$. This is attained through a series of iterations involving two stages. The first stage of the scheduling procedure involves a semi-randomized schedule generating routine which iteratively selects and schedules a slot request r (details are given below). However, the scheduling process is halted when a given (pre-defined) number of slot requests have been scheduled resulting to a partial solution of the problem. Then, at the second stage, the FP routine is called in order to extend the partial solution to a complete feasible solution. The integration of FP plays a significant role in the scheduling routine since it almost ensures the determination of feasible and high-quality solutions which are otherwise hard to achieve with a plain construction heuristic for such a constrained problem. The construction heuristic alone is fast but often incapable of constructing high-quality solutions, while running a

plain FP routine is time consuming. With an abstraction, we suggest that the FP acts as a guide right before the scheduling heuristic gets trapped into an infeasibility obstacle. FP manages to find a way to overcome this obstacle while preserving the quality of the solution constructed so far. This highly rewarding cooperation is the reason that we chose to combine the specific two algorithms.

The construction heuristic applied at the first stage of FPSGH iteratively selects an unscheduled slot request r and schedules it at the feasible time interval $t^*(r)$ which is closest to the corresponding requested time interval τ_r. The routine terminates when a pre-defined number of requests (n_s) have been scheduled. The value of (n_s) is randomly selected within the integer values of the interval [n_s^{min},n_s^{max}] before the initiation of the routine. Note that upon termination of the scheduling routine, the control is transferred to the FP routine which takes care of scheduling the remaining unscheduled requests.

Selecting and scheduling a slot request is performed probabilistically (roulette wheel selection). The selection is based on a scheduling metric $K(r)$ defined for any unscheduled request r by (32).

$$K(r) = \frac{G(r, t^*(r))}{\lambda(r, t^*(r))} \tag{32}$$

where $G(r, t^*(r))$ is a *regret metric* for not assigning time interval $t^*(r)$ at request r, and $\lambda(r, t^*(r))$ is a metric that expresses the remaining aggregate capacity at time interval $t^*(r)$. The term *aggregate capacity* refers to the sum of the remaining capacities at time $t^*(r)$ over all calendar days of the planning period. The regret metric $G(r, t^*(r))$ is expressed by the difference between the displacement for scheduling request r at the closest feasible time interval $t^*(r)$ and the corresponding displacement if r is scheduled at the second closest feasible time interval denoted by $t^{**}(r)$. If no such time exists, then $t^{**}(r)$ is set to a very large number.

$$G(r, t^*(r)) = |\tau_r - t^{**}(r)| - |\tau_r - t^*(r)| \tag{33}$$

In general, metric $\lambda(r, \tau)$ is given by (34):

$$\lambda(r, \tau) = \sum_{\delta=1}^{|D|} a_{\delta r} u_0^{'\delta\tau}, \quad \tau \in T \tag{34}$$

where $u_0^{'\delta\tau}$ is the remaining capacity (for both types of movements, i.e., $c = 0$) within interval $[\tau, \tau + t_0)$. Hence, the proposed metric $K(r)$ gives priority to scheduling slot requests which: i) they tend to increase substantially the schedule displacement of the solution or ii) the capacity around the area of their most promising schedule time tends to diminish. $K(r)$ is a simple but intelligent metric which greatly boosts the chances of finding a feasible solution.

The probabilistic procedure for selecting and scheduling a slot request involves a roulette wheel selection in which each unscheduled request r has a probability $p(r)$ of being selected given by the following formula:

$$p(r) = \frac{K(r)}{\sum_{r' \in R^u} K(r')} \tag{35}$$

The probabilistic selection aims to achieve diversification of the solution approach and search solution areas with completely different structures.

4.2.3 FP–Guided LNS

Each iteration of the proposed FPLNS routine takes the currently best schedule found (starting with the one determined by *FPSGH*) and selects randomly $\varphi\%$ of the slot requests in R_0 (where $\varphi\%$ is a pre-defined problem parameter). The selected slot requests are removed from the schedule, leaving a partial solution (destroy operator). Then, a new schedule is rebuilt by rescheduling the removed slot requests through the FP algorithm (repair operator). If the emerging new solution has a lower Z_i value from the currently best, then it becomes the new best solution. In more detail, FPLNS involves the following steps:

A. Assume an initial solution σ_0. σ_B denotes the currently best schedule determined so far. Hence, initially we set $\sigma_B \coloneqq \sigma_0$.

B. For each iteration of a pre-defined number N_{LNS} of iterations, the following steps are performed:

   B1. Randomly select $\varphi\%$ of slot requests (rounded to the closest integer) from R_0 and unassign their allocated time intervals. In addition, their linked slot requests are unscheduled as well. Place all unscheduled requests in R^u. The requests that remain scheduled are denoted by R^s. The remaining partial schedule is denoted by $\sigma_B^{R^s}$.

   B2. Update the remaining capacity parameters $u_c'^{\delta\tau}$ for $c \in C$, $\tau \in \{0,1,..,n - t_c + 1\}$, and $\delta \in D$ taking into account the slots and hence the resources that are released after un-scheduling requests in R^u.

   B3. Apply the FP routine described in the previous subsection to solve the emerging restricted airport slot scheduling problem ($RP7$). It is worth noting that C^* is set as follows: $C^* \coloneqq (1-\mu)\left|Z_i(\sigma_B) - Z_i\left(\sigma_B^{R^s}\right)\right|$. Therefore, the objective of the emerging ($RP7$) is to determine a new partial schedule σ^{R^u} for the requests in R^u so that the new complete schedule σ (that emerges directly from concatenating it with partial schedule $\sigma_B^{R^s}$) has an increased potential of having an objective function value lower than $Z_i(\sigma_B)$.

   B4. If $Z_i(\sigma_B) > Z_i(\sigma)$, then set $\sigma_B \coloneqq \sigma$.

It is evident that the computational time of the proposed routine may be controlled by the number of iterations N_{LNS} and parameter φ given that FP is a computationally time-consuming procedure.

5. Computational Tests

The proposed *FPεCM* framework is assessed on the basis of: i) its capability to approximate the Pareto optimal solutions, and ii) its computational time performance. Both types of assessment were applied on a series of test problems that were generated following the patterns of the actual demand of a Greek Regional Airport (GRA). In order to evaluate the performance of the proposed algorithm under the presence of Grandfather Rights constraints, we used the actual dataset of GRA (where we had the information regarding the slot requests having Grandfather Rights) to approximate the corresponding Pareto optimal solutions of the proposed bi-objective slot scheduling problem enhanced with the relevant constraints.

The capability of the *FPεCM* to approximate the Pareto frontier was assessed by measuring the proximity of the solutions determined by the *FPεCM* on a set of benchmark test problems from the corresponding Pareto frontiers (computed by the εCM). The proximity of the *FPεCM* solutions to the Pareto frontier was measured by two widely used metrics in multi-objective optimization: the distance index and the hypervolume index (Zitzler et al., 2003). The experiments were performed under the following capacity levels: i) 6 arrivals, 6 departures, 12 movements in total per hour (6/6/12) and ii) 7 arrivals, 7 departures, 14 movements in total per hour (7/7/14). Due to the huge computational time of the exact εCM, the test problems were solved for half of the actual planning horizon (i.e., for three instead of six months). It is worth noting that the test problems could not be solved by εCM in reasonable time for tighter capacity levels, e.g., (5/5/10).

The computational time performance of the proposed algorithm is assessed by measuring the CPU time required to solve the above stated test problems for the entire planning horizon and for capacity levels (5,5,10), (6,6,12) and (7,7,14). The *FPεCM* algorithm was implemented in Java programming language, whereas Java and CPLEX 12.6 were used for the εCM. The computational experiments were carried out on an Intel Core i7-7700 CPU @ 3.60 GHz with 16 GB RAM x64 Windows 10 machine.

5.1 Actual Data Set and Test Problems Generation

The actual data set contains slot requests made by airlines for the period March-October 2009 (planning horizon). GRA is a small, regional airport located on a Greek island that is designated as schedule coordinated (Level 3), according to IATA scheduling guidelines, only for the summer scheduling season (due to highly seasonal incoming tourist flows). It serves more than 3 million passengers and 19,000 aircraft movements (2018). The data set involves 735 slot requests (placed by 39 airlines) that correspond to 16,065 movements. Figure 4 presents the demand profile on the daily time horizon. Each point on the graph corresponds to a time interval $\tau \in T$ (x-axis) and the

maximum number (over the planning period) of slot requests demanded within any 1 hour time period surrounding time τ (y-axis). The capacity level in which the airport was operating at that time was 10 movements per hour (current declared capacity level).

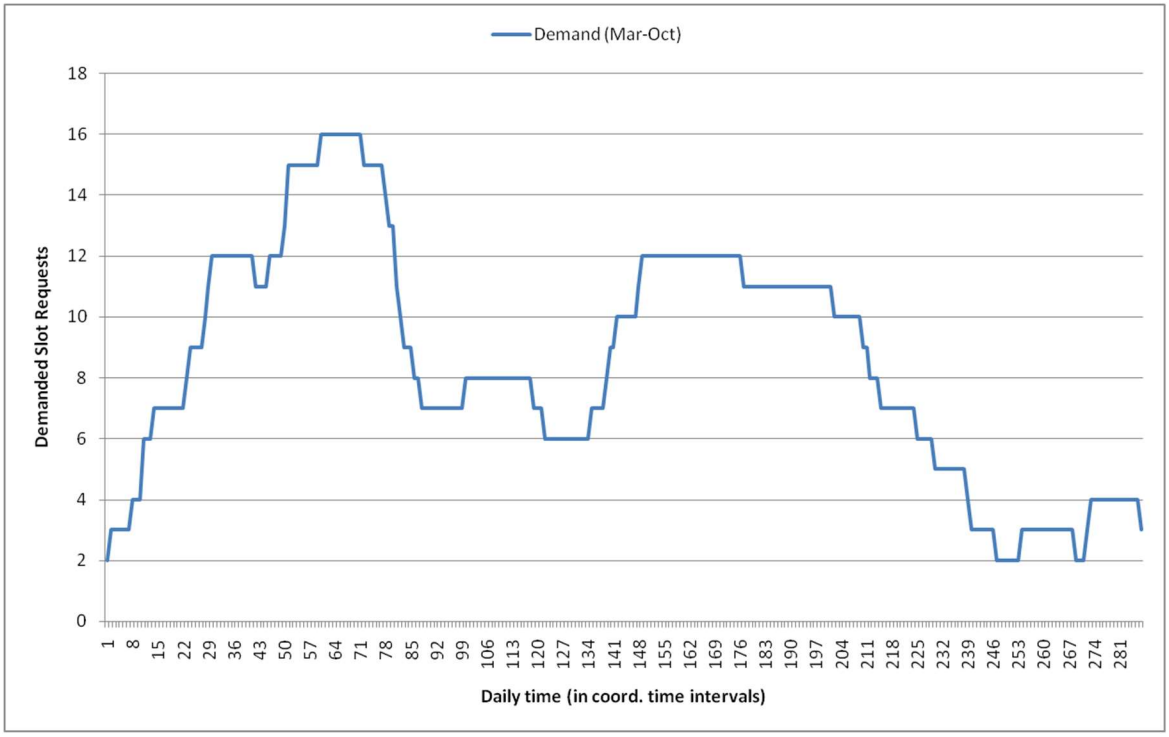

Figure 4. Demand profile in GRA data set for the entire planning horizon.

In order to create realistic test problems, the authors analyzed the actual slot requests data set of GRA and identified the probability distributions of the following problem parameters:

1. Number of weekdays (w) that each slot request applies to (ranging from 1 to 7).
2. Length (ℓ) of the time period D_r (within the six months planning horizon) of a slot request. This parameter is expressed in the form of a percentage (0-100%) over the six-month planning horizon.
3. The requested time (τ) of slot requests (for arrivals and departures) over the daily time horizon (divided in coordination time intervals).

We developed an instance generator in Java that is capable of creating problem instances replicating the demand pattern of GRA (or any other airport). The generator performs a pre-defined number of iterations. The slot requests are generated sequentially by sampling the 4 aforementioned distributions. Each generated slot request carries the following information: i) the requested time of the movement, ii) the weekdays it applies and iii) the period of time within the planning period that the requested movement is applicable. It is worth noting that both in the initial data sets and the generated ones, each arrival is associated (linked) with a departure by at least a

minimum turnaround time. In total, we generated 9 new data sets (TP1 to TP9) each one containing 998 slot requests corresponding to a number of movements ranging from 15,042 to 17,304.

5.2 Pareto Frontier Generation

Due to the high computational time required by the εCM to determine the entire set of Pareto optimal solutions for the entire planning period, we curtailed the test problems (henceforth denoted CTP1 to CTP9) by considering only the first 90 calendar days of the planning period for capacity levels (6/6/12) and (7/7/14) movements per hour. Therefore, each curtailed test problem was obtained by including only those slot requests r that their time period D_r starts within the first 90 calendar days of the planning period. The set D_r of any slot request r that starts within the time period $[0,90]$ is curtailed by removing the calendar days (if any) after the 90th calendar day. Any slot request that its starting calendar day exceeded day 90 was removed from the data set. Table 3 presents the number of requests and movements included for each curtailed test problem and the actual curtailed GRA data set.

Table 3. Number of slot requests and movements for each of the test problems.

	Period: March-June	
Test Problems	#Slot Requests ($\|R_0\|$)	# Movements ($\|M\|$)
Curtailed GRA	629	5983
CTP1	712	6466
CTP2	766	7376
CTP3	748	6952
CTP4	762	6804
CTP5	786	7234
CTP6	742	6620
CTP7	756	7440
CTP8	772	7112
CTP9	774	7842

Figures 5 and 6 present the corresponding demand profiles for the smallest generated data set in terms of the number of movements (i.e., data set CTP1) and the actual data set for the examined period respectively. It is evident that in both cases the demand exceeds the capacity levels under consideration for a significant part of the day.

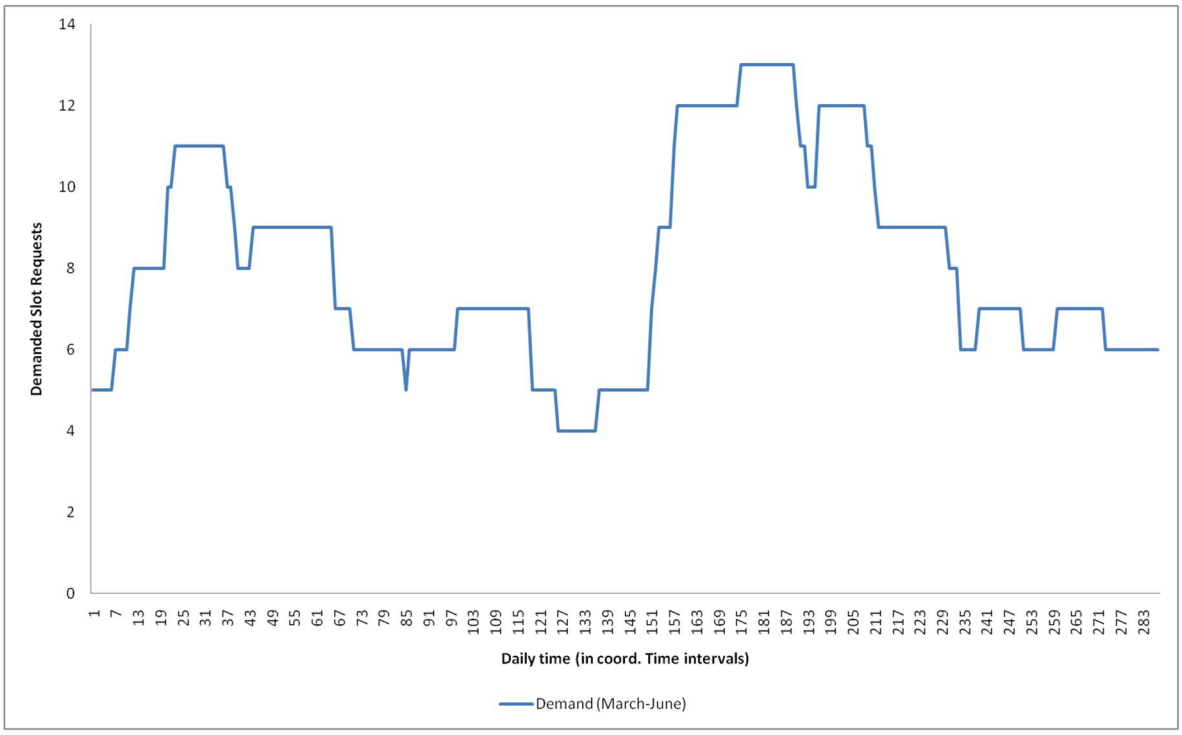

Figure 5. Demand profile emerging from the generated data set CTP1 for the period March-Jun.

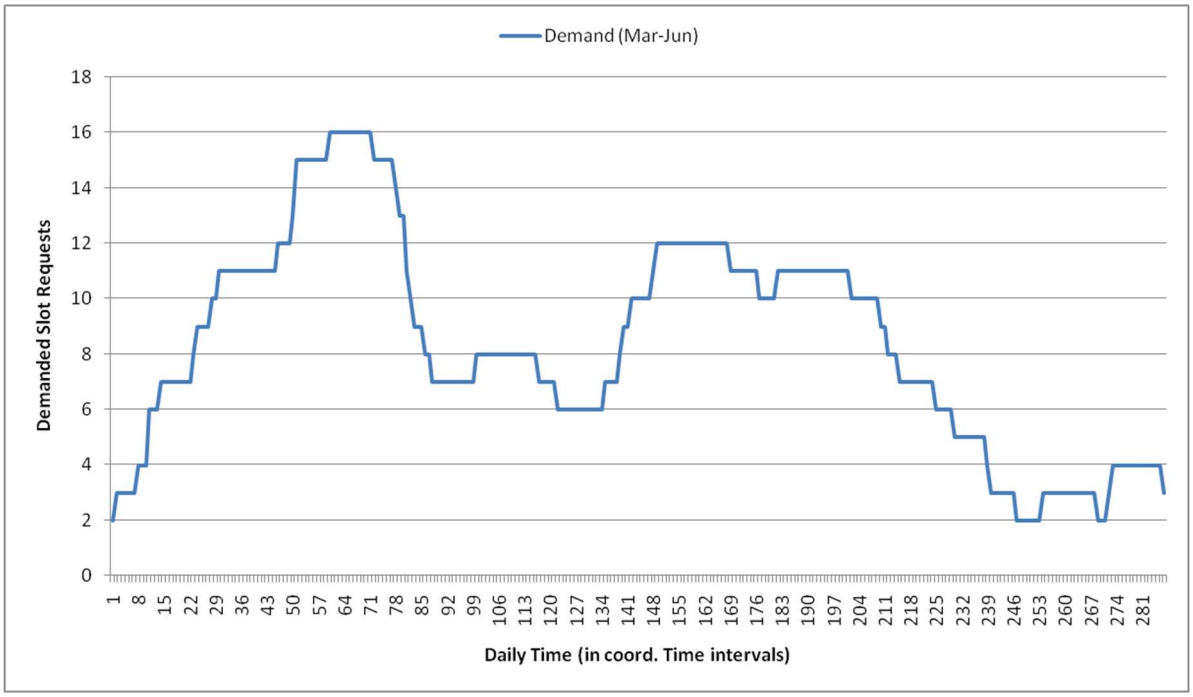

Figure 6. Demand profile of the actual GRA data set for the period March-Jun.

The minimum allowed turnaround time $t_{r_1 r_2}$ was set equal to 30 minutes. All CPLEX settings were set to default.

5.3 Metrics

We denote by L_H and L_E the sets of solutions determined by the $FP\varepsilon CM$ and εCM, respectively. The proximity of L_H to the Pareto optimal solutions in L_E was assessed by calculating the following metrics (Zitzler et al., 2003; Gomes et al., 2014):

i. ***Distance Metric***. This metric expresses a measure of distance of the heuristic solutions from the Pareto optimal solutions. In addition, it assesses the capability of the heuristic algorithm to approximate the entire spectrum of Pareto optimal solutions. The proposed metric defined by (36) is based on the normalized distance between solutions σ and σ' denoted by $h(\sigma,\sigma')$ and defined by (37).

$$\Delta(L_E, L_H) = \frac{1}{|L_E|}\left(\sum_{\sigma \in L_E} \min_{\sigma' \in L_H} h(\sigma,\sigma')\right) \tag{36}$$

where,

$$h(\sigma,\sigma') = \max\left\{\frac{|Z_1(\sigma) - Z_1(\sigma')|}{Z_1^0 - Z_1^*}, \frac{|Z_2(\sigma) - Z_2(\sigma')|}{Z_2^0 - Z_2^*}\right\} \tag{37}$$

Z_1^* is the minimum value for objective function Z_1 and Z_1^0 is the Z_1 value of the solution that minimizes objective function Z_2. Similarly, Z_2^* is the minimum value for objective function Z_2 and Z_2^0 is the Z_2 value of the solution that minimizes objective function Z_1. The lower the value of $\Delta(L_E, L_H)$, the closer is the approximation of L_E by L_H.

ii. ***Hypervolume Indicator***. This metric, denoted by I and defined by (38), aims to compare the dominated area of L_H (denoted by $A(L_H)$) with the corresponding dominated area of L_E (denoted by $A(L_E)$). In order to do so, a reference point (x, y) with x and y being the upper bounds for objectives Z_1 and Z_2 respectively, is used to limit the coverage. The hypervolume indicator I is defined as the ratio of $A(L_E)$ over $A(L_H)$. Since the dominated area covered by L_E is larger or equal (best case) to the dominated area covered by L_H, the closest $I(L_H)$ is to one, the better is the coverage of the dominated area.

$$I(L_H) = \frac{A(L_H)}{A(L_E)} \tag{38}$$

Figure 7 illustrates the definition of the Hypervolume Indicator. The Z_1 and Z_2 values of the solutions in L_E are denoted by circles, whereas the corresponding values of the solutions in L_H are denoted by squared dots. The entire marked area on the right of the circles represents $A(L_E)$, whereas the grey shaded area represents $A(L_H)$.

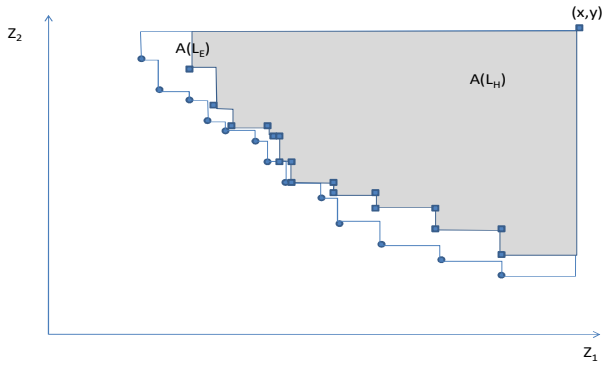

Figure 7. Example of the dominated areas covered by L_E and L_H.

5.4 Preliminary Experiments

The two basic parameters than need to be set for $FP\varepsilon CM$ are the number of LNS iterations (N_{LNS}) and the percentage φ of FPLNS. It is expected that increasing the value of φ would increase the quality of the solutions but also increase the computational time. This is due to the fact that the percentage φ denotes the size of the part of the solution that is destroyed. Hence, the larger the part of the solution destroyed, the higher is the effort needed to repair the solution while the higher is also the possibility that the repaired solution will improve the currently best solution. In a similar reasoning, it is expected that the higher the value of N_{LNS} parameter, the higher is the possibility of finding a repaired solution that improves the currently best found. Both hypotheses have been verified through a set of preliminary runs of the proposed algorithm on a single month planning horizon (i.e., the 1st month of the planning period) of the actual data set of GRA using various values for φ and N_{LNS}. In particular, we used $\varphi \in \{0.1, 0.2, 0.3, 0.4, 0.5\}$ and $N_{LNS} \in \{20, 40, 60\}$. Table 4 presents the results of these preliminary runs for all combinations of the two parameters.

Table 4. The distance metric for various combinations of φ and N_{LNS} for the GRA data set.

$\boldsymbol{\varphi}$	Number of LNS iterations ($\boldsymbol{N_{LNS}}$)		
	20	**40**	**60**
0.1	0.077	0.105	0.065
0.2	0.081	0.048	0.066
0.3	0.040	0.032	0.046
0.4	0.034	0.037	0.054
0.5	0.054	0.027	0.046

Due to the fact that the experiment is dependent on the specific data set and the demand of the specific month, we conclude the following:

- The value of 60 iterations seems to provide performance stability in terms of the φ parameter;
- Concerning φ, we should exclude any value below 0.3 where the distance metric is substantially higher. We select value 0.4 for the remaining experiments since, the higher the φ, the more time consuming it is to repair the solutions with FP, but solutions of better quality are expected.

Hence, we select 60 iterations with φ equal to 0.4 as a reasonable parameterization for the presented application in order to keep the FP running times low but also have the ability to alter a big part of the solution.

5.5 $FP\varepsilon CM$ Benchmarking

The solution accuracy of the $FP\varepsilon CM$ was tested on: i) the curtailed test problems CTP1-CTP9 for capacity scenarios (6/6/12) and (7/7/14) and ii) the curtailed slot allocation problem defined on the actual GRA data set. The latter aimed to assess the performance of the algorithm under the presence of scheduling priorities among the slot requests. In more detail, the slot scheduling problem defined on the GRA actual data set was solved in two stages. The first stage aimed to solve the slot scheduling problem considering only for the slot requests with Grandfather rights. The remaining slot requests were scheduled at the second stage.

In both streams of experiments, we used 60 LNS iterations with φ equal to 0.4. The heuristic schedule generating routine (SGH) was halted (and the FP was initiated) after (up to) 10% of the slot requests were scheduled. Tables 5 and 6 present the number of the Pareto optimal and heuristic solutions (2nd and 3rd columns), the distance metric (4th column), the hypervolume indicator (5th column), the average computational time required per Pareto solution (6th column) and the average computational time per $\boldsymbol{FP\varepsilon CM}$ solution (7th column) for each of the test problems under (6/6/12) and (7/7/14). The average value of the distance metric is 0.026 (or 2.6%) for both capacity scenarios. This implies than any Pareto optimal solution of the test problems under any of the two the capacity scenarios may be approximated by a heuristic solution of the proposed algorithm that deviates from the Pareto optimal one by 2.6% (on average) on any of its two objective functions (Z_1 and Z_2). In more detail, the distance metric ranges from 1.0% up to 12.0%.

Table 5. Computational results for CTP1-CTP9 under capacity scenario (6/6/12).

Test Problems	# of Pareto Solutions	# of Heuristics Solutions	Distance Metric	Hypervolume Indicator	Average Comp. Time Per Pareto Solution (sec)	Average Comp. Time Per $FP\varepsilon CM$ Solution (sec)
CTP1	502	186	0.011	0.990	223	152
CTP2	520	183	0.016	0.819	372	272

CTP3	226	123	0.010	0.966	239	158
CTP4	484	147	0.016	0.943	270	252
CTP5	156	31	0.011	0.926	357	242
CTP6	445	187	0.013	0.918	188	159
CTP7	130	84	0.009	0.958	195	195
CTP8	252	147	0.014	0.930	230	341
CTP9	256	53	0.120	0.903	379	253
Average	**330**	**126**	**0.026**	**0.920**	**273**	**225**

Table 6. Computational results for CTP1-CTP9 under capacity scenario (7/7/14).

Test Problems	# of Pareto Solutions	# of Heuristics Solutions	Distance Metric	Hypervolume Indicator	Average Comp. Time Per Pareto Solution (sec)	Average Comp. *Time Per* $FP\varepsilon CM$ *Solution*
CTP1	234	115	0.018	0.975	245	230
CTP2	171	81	0.024	0.822	212	193
CTP3	67	52	0.026	0.902	97	79
CTP4	170	82	0.013	0.956	155	121
CTP5	22	21	0.018	0.903	174	110
CTP6	155	82	0.080	0.514	92	61
CTP7	14	10	0.011	0.617	72	53
CTP8	20	18	0.018	0.728	101	390
CTP9	53	47	0.016	0.990	316	73
Average	**101**	**57**	**0.026**	**0.800**	**163**	**146**

Concerning the hypervolume indicator, it can be concluded that the dominated area formed by the $FP\varepsilon CM$ solutions covers (on average) 92% of the entire dominated area formed by the Pareto optimal solutions for capacity scenario (6/6/12) and (on the average) 80% for capacity scenario (7/7/14). These results indicate that $FP\varepsilon CM$ has the capability to approximate closely the Pareto optimal solutions of the proposed bi-objective airport slot scheduling problem.

It should be pointed out that, in addition to the total aggregate and squared displacement, we have also measured the maximum displacement to any slot request for all Pareto optimal and heuristic solutions. For the solutions that emerged under the capacity scenario (6,6,12), the maximum displacement ranges between 4 and 19 coordination time intervals, i.e., the maximum displacement allocated to any slot request does not exceed 19 time intervals (or 95 minutes). The corresponding values for the maximum displacement of the solutions determined under the (7,7,14) capacity scenario ranges between 3 and 12 time intervals. This result indicates that the proposed model provides alternative solutions without extremely high worst case values of displacement.

An indicative visual representation of the degree of approximation of the Pareto optimal solutions by the heuristic solutions is provided through Figures 8 and 9. The graphs presented in these Figures illustrate the Pareto optimal and solutions on the decision space formed by their values on

Z_1 and Z_2 respectively for the test problem CTP3. Figure 8 refers to capacity scenario 6/6/12 and Figure 9 to capacity scenario 7/7/14.

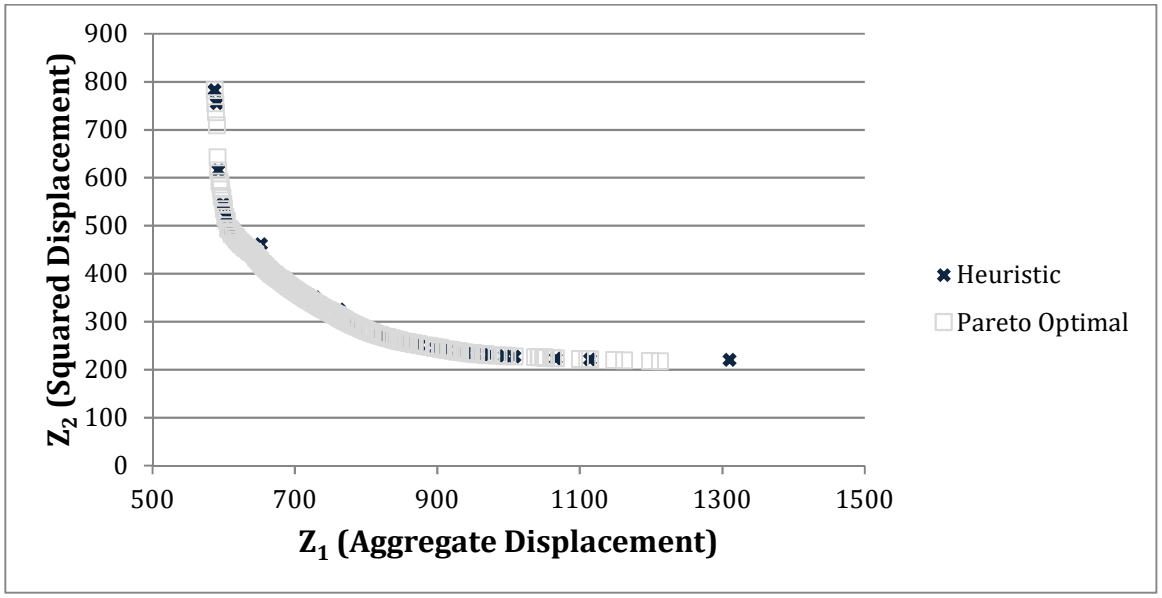

Figure 8. The (Z_1, Z_2) values of the $FP\varepsilon CM$ and the Pareto optimal solutions for CTP3 under capacity scenario (6/6/12).

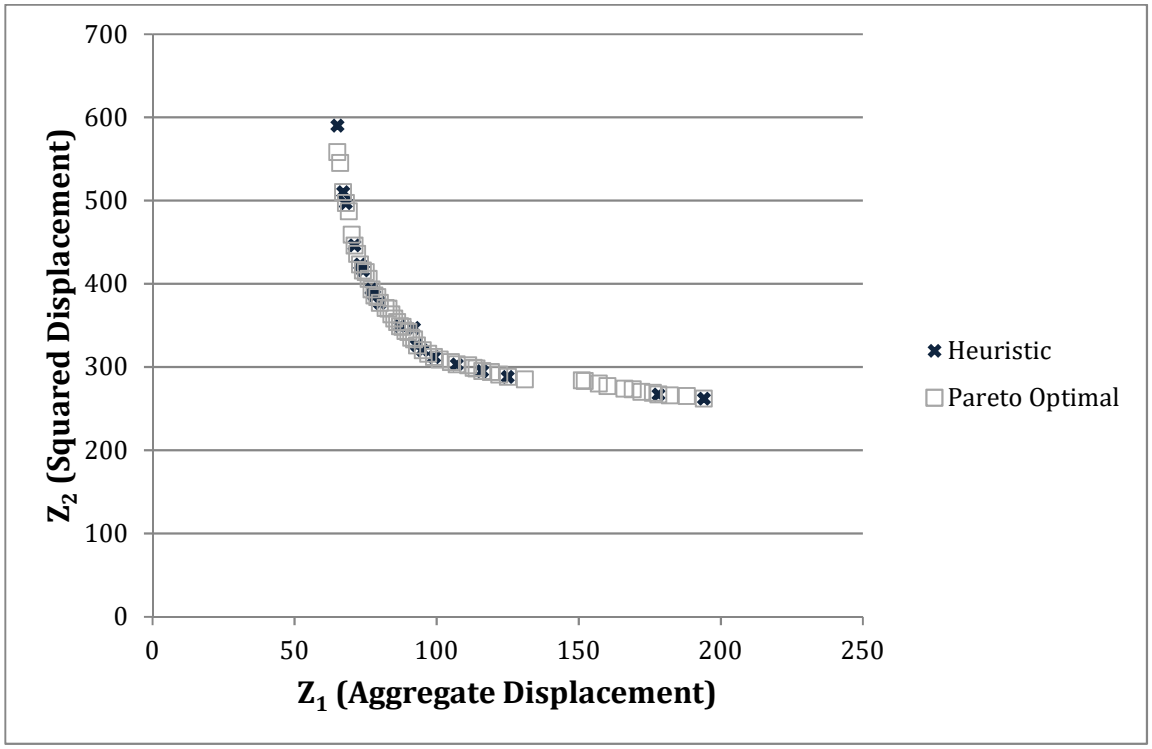

Figure 9. The (Z_1, Z_2) values of the $FP\varepsilon CM$ and the Pareto optimal solutions for CTP3 under capacity scenario (7/7/14).

Table 7 presents the results from GRA that emerged from solving the actual bi-objective slot scheduling problem for GRA taking into account the corresponding Grandfather Rights constraints.

Table 7. Computational results for the GRA for capacity scenarios (6/6/12) and (7/7/14).

Test Problems	# of Pareto Solutions	Distance Metric	Hypervolume Indicator	Average Comp. Time Per Pareto Solution (sec)	Average Comp. Time Per $FP\varepsilon CM$ Solution (sec)
GRA (6/6/12)	328	0.028	0.991	347	190
GRA (7/7/14)	391	0.028	0.634	327	178

5.6 Computational Time Performance

A set of experiments was performed in order to assess the effect of parameters N_{LNS} and φ on the computational time of the algorithm. Each of the test problems TP1-TP9 was solved for the entire planning period (March-October). In order to further evaluate the dependence of the time performance on the capacity levels restriction, we performed experiments under three alternative capacity levels, namely (5/5/10), (6/6/12) and (7/7/14) and the following parameter values:

- the maximum number of iterations of the SGH before the routine is halted was set to 10% of the total number of slot requests;
- each run involved 20 LNS iterations;
- two values of φ were applied: 0.2 and 0.4.

Table 8 presents the computational time (in hours) of the FPSGH and the total execution time required by *FPεCM* to solve each of the test problems for φ equal to 0.2 and 0.4 (denoted as LNS(0.2) and LNS(0.4) respectively). The results in Table 8 indicate that the worst case computational time for solving the proposed bi-objective problem was 48.2 hours (for TP8 and φ=0.4 under capacity level 6/6/12) which is split to 6.8 hours for the execution of the *FPSGH* and the remaining 41.4 hours for the LNS(0.4) iterations.

Table 8. Computational time performance of $\boldsymbol{FP\varepsilon CM}$ for TP1-TP9 under different capacity levels.

	Capacity Level (5/5/10)			**Capacity Level (6/6/12)**			**Capacity Level (7/7/14)**		
	FPSGH	*LNS(0.2)*	*LNS(0.4)*	*FPSGH*	*LNS(0.2)*	*LNS(0.4)*	*FPSGH*	*LNS(0.2)*	*LNS(0.4)*
TP1	6.6	19.9	28.4	9.0	24.0	32.1	14.9	19.4	36.4
TP2	8.9	27.7	35.3	12.3	28.8	33.8	12.1	24.8	41.3
TP3	11.2	16.7	26.8	16.0	20.1	24.8	15.6	18.6	22.9
TP4	13.5	13.7	36.6	9.7	18.8	42.2	13.1	18.4	25.2
TP5	11.3	21.4	26.5	7.7	17.3	30.1	8.0	19.2	29.5
TP6	5.4	18.0	29.0	8.9	22.6	24.6	10.4	18.4	28.8
TP7	10.3	19.1	30.2	10.5	21.3	32.4	12.5	19.9	31.0
TP8	11.3	21.6	33.6	6.8	23.1	48.2	17.4	26.6	38.1
TP9	11.6	16.5	27.1	9.6	20.3	27.8	10.1	23.6	28.3
Avg.	10.0	19.4	30.4	10.0	21.8	32.9	12.7	21.0	31.3

The computational time performance of the LNS iterations is further analysed through the graphs in Figure 10 presenting the average computational time (in sec) per LNS iteration performed by the *FPεCM* for solving the test problems with φ equal to 0.2 and 0.4.

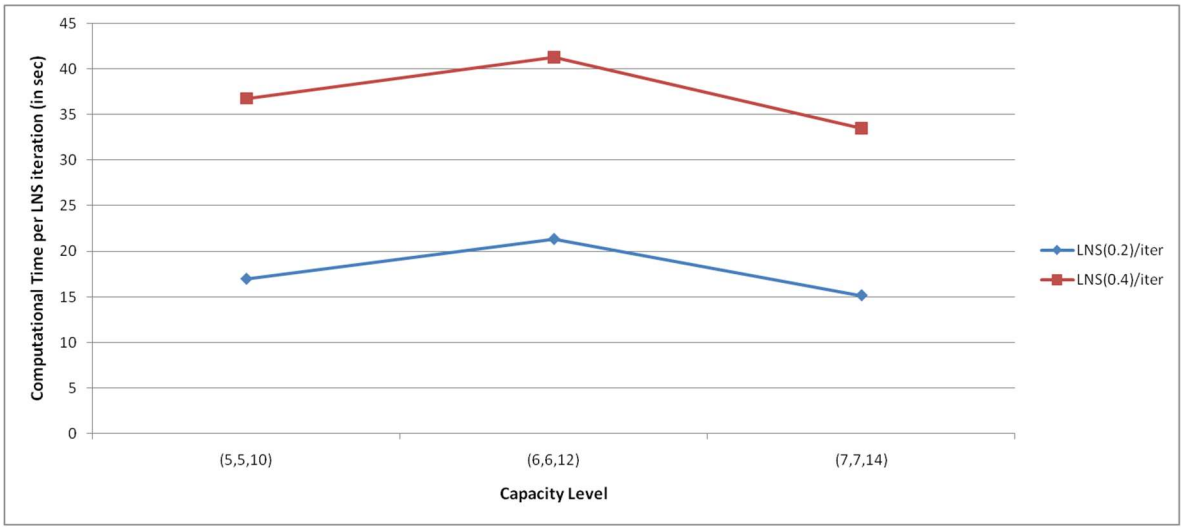

Figure 10. Computational time performance per LNS iteration for φ equal to to 0.2 and 0.4.

In conclusion, the computational performance of the proposed algorithm indicates that despite the large computational times for large values of φ, it still remains applicable to real-world instances.

6. Concluding Remarks

This paper presents a bi-objective formulation for the strategic airport slot allocation problem and provides a novel solution method that achieves a reasonably accurate approximation of the Pareto optimal solutions of the problem. We show that the emerging problem falls into a new class of project scheduling problems which involve multiple non-regular objective functions and partially renewable resources. The proposed solution method, called *FPεCM* is a solution approach that uses *εCM* to generate a series of constrained sub-problems. The novelty is the introduction of a hybrid algorithm called FPHA to heuristically solve the emerging sub-problems enabling us to tackle real-life sized problems. FPHA constructs an initial solution through FPSGH which is a heuristic constructive algorithm that takes advantage of FP to generate high quality solutions. The basic component of FPHA is the FPLNS which is a module based on the co-operation of the meta-heuristic method Large Neighborhood Search (LNS) with Feasibility Pump (FP). The FP–Guided LNS approach capitalizes on the advantages of both methods in order to find high quality solutions. To the best of the authors' knowledge, this is the first application of LNS with Feasibility Pump. Also, a library of new problem instances of 6 months with realistic patterns has been generated and

may contribute in motivating other researchers to get involved with this challenging problem. The computational experiments performed on the new instances provide promising results as the algorithm can efficiently identify a large portion of the Pareto optimal solutions.

The above indicates that similar solution approaches combining LNS with Feasibility Pump as a feasibility ensuring process with high quality solutions can be applied to other bi-objective scheduling problems. In addition, FPHA is important for tackling the difficult εCM sub-problems and could be generalized to other problems as a hybrid algorithm for heuristically generating the Pareto Frontier in acceptable computational times and with satisfactory accuracy. Another promising research direction in terms of algorithmic approach is to conduct experiments to explore the synergy achieved by the combination of FP with other meta-heuristic algorithms (e.g., evolutionary algorithms, local search-based algorithms). Finally, it would be interesting to extend the problem to a multi-objective version including additional objectives capturing other operational constraints with managerial impact (e.g., type of route/flight, aircraft size, length of slot request, safeguarding access to small communities).

Acknowledgments

The presented research work was partially supported by the Research Center of the Athens University of Economics and Business (AUEB-RC) through the projects EP-2638-01 and EP-3002-01.

References

Achterberg, T., & Berthold, T. (2007). Improving the feasibility pump. *Discrete Optimization*, 4, 77-86.

Al-Fawzan, M.A., & Haouari, M. (2005). A bi-objective model for robust resource-constrained project scheduling. *International Journal of Production Economics*, 96, 175–87.

Alvarez-Valdez, R., Crespo, E., Tamarit, J.M., & Villa, F. (2006). GRASP and path relinking for project scheduling under partially renewable resources. *Journal of Heuristics*, 12, 95-113.

Alvarez-Valdez, R., Tamarit J.M., & Villa F. (2015). Minimizing weighted earliness–tardiness on parallel machines using hybrid metaheuristics. *Computers & Operations Research*, 54, 1-11.

Ballestin, F., & Blanco, R. (2011). Theoretical and practical fundamentals for multi-objective optimisation in resource-constrained project scheduling problems. *Computers & Operations Research*, 38, 51–62.

Ballestín, F. & Trautmann, N. (2008). An iterated-local search heuristic for the resource-constrained weighted earliness-tardiness project scheduling problem. *International Journal of Production Research*, 46(22), 6231-6249.

Benlic, U. (2018). Heuristic search for allocation of slots at network level. *Transportation Research Part C - Emerging Technologies*, 86, 488-509.

Bottcher, J., Drexl, A., Kolisch, R. & Salewski, F. (1999). Project Scheduling Under Partially Renewable Resource Constraints. *Management Science*, 45(1), 543-599.

Castelli, L., Pellegrini, P. & Pesenti, R. (2012). Airport Slot Allocation in Europe: Economic Efficiency and Fairness. *International Journal of Revenue Management*, 6(1/2), 28-44.

Corolli, L., Lulli, G., & Ntaimo, L. (2014). The Time Slot Allocation Problem under Uncertain Capacity. *Transportation Research Part C - Emerging Technologies*, 46, 16-29.

Drexl, A. & Grünewald, J. (1993). Non-preemptive multi-mode resource-constrained project scheduling. *IIE Transactions,* 25:5, 74-81.

Fischetti, M., Glover, F. & Lodi, A. (2005). The feasibility pump. *Mathematical Programming*, Ser. A 104, 91–104.

Gomes, H.C., Neves, F.A. & Souza, M.J.F. (2014). Multi-objective metaheuristic algorithms for the resource-constrained project scheduling problem with precedence relations. *Computers & Operations Research*, 44, 92–104.

International Air Transport Association (IATA) (2014). *Worldwide Slot Guidelines*, 6th Edition, Montreal, Canada.

Jacquillat, A & Odoni, A.R. (2015). An Integrated Scheduling and Operations Approach to Airport Congestion Mitigation. *Operations Research*, 63(6), 1390-1410.

Kazemi, F.S., & Tavakkoli-Moghaddan, R. (2008). Solving a multi-objective multi-mode resource-constrained project scheduling with discounted cash flows. In: *Proceedings of the 6th international management conference*. Tehran, Iran.

Pellegrini, P., Castelli, L., Pesenti, R. (2012). Metaheuristic Algorithms for the Simultaneous Slot Allocation Problem. *IET Intelligent Transport Systems*, 6 (4), 453-462.

Pellegrini, P., Bolic, T., Castelli, L. & Pesenti, R. (2017). Sosta: an effective model for the simultaneous optimisation of airport slot allocation. *Transportation Research Part E – Logistics and Transportation Review*, 99, 34–53.

Pyrgiotis, N. & Odoni, A.R. (2015). On the Impact of Scheduling Limits: A Case Study at Newark Liberty International Airport. *Transportation Science*, 50(1):150-165, 2015.

Ribeiro, N.A., Jacquillat, A., Antunes, A.P., Odoni, A.R., & Pita, J.P. (2018). An optimization approach for airport slot allocation under IATA guidelines. *Transportation Research Part B - Methodological*, 112, 132-156.

Sahni, S., & Gonzalez, T.F. (1976). P-Complete Approximation Problems. *J. ACM, 23*, 555-565

Schirmer, A. & Drexl, A. (2001). Allocation of Partially Renewable Resources: Concept, Capabilities, and Applications. *Networks*, 37(1), 21-34.

Shaw, P. (1998). Using Constraint Programming and Local Search Methods to Solve Vehicle Routing Problems. In: M. Maher, J.-F. Puget (eds.), Principles and Practice of Constraint Programming - CP98, Lecture Notes in Computer Science, Springer-Verlag, New-York, 417–431.

Vanhoucke, M., Demeulemeester, E., Herroelen, W. (2001). An Exact Procedure for the Resource-Constrained Weighted Earliness–Tardiness Project Scheduling Problem. *Annals of Operations Research* 102, 179–196.

Viana, A. & Pinho de Souza, J. (2000). Using Metaheuristics in Multiobjective resource constrained project scheduling. *European Journal of Operational Research*, 120, 359-374.

Zitzler, E., Thiele, L., Laumanns, M., Fonseca, C.M., & Fonseca, V.G. (2003). Performance assessment of multiobjective optimizers: an analysis and review. *IEEE Transactions on Evolutionary Computation*, 7(2), 117 – 132.

Zografos, K.G., Salouras, Y., & Madas, M.A. (2012). Dealing with the Efficient Allocation of Scarce Resources at Congested Airports. *Transportation Research Part C – Emerging Technologies*, 21(1), 244-256.

Zografos, K.G. & Jiang, Y. (2016). Modelling and Solving the Airport Slot Scheduling Problem with Efficiency, Fairness, and Accessibility Considerations. *TRISTAN Symposium 2016*, Oranjestad, Aruba, June 13-17.

Zografos, K.G., Madas, M.A., & Androutsopoulos, K.N. (2017). Increasing Airport Capacity Utilisation through Optimum Slot Scheduling: Review of Current Developments and Identification of Future Needs. *Journal of Scheduling*, 20(1), 3-24.

Zografos, K.G., Androutsopoulos, K.N., & Madas, M.A. (2018). Minding the Gap: Optimum Slot Scheduling Considering Total and Maximum Acceptable Schedule Displacement Objectives. *Transportation Research Part A - Policy and Practice*, 114, 203-221.