MINIMIZING DEVIATION FROM SCHEDULED TIMES IN A SINGLE MIXED-OPERATION RUNWAY

A. Rodríguez-Díaz[a], *B. Adenso-Díaz*[b#], *P.L. González-Torre*[b]

[a] INDRA SISTEMAS, Air Traffic Control

[b] Escuela Politécnica de Ingeniería de Gijón, Universidad de Oviedo, Spain

[#] Corresponding author: adenso@epsig.uniovi.es

Fax: +34 985 182010
Phone: +34 609 848525

A. Rodríguez-Díaz, B. Adenso-Diaz, P. González-Torre (2017) "Minimizing deviation from scheduled times in a single mixed-operation runway", COMPUTERS & OPERATIONS RESEARCH, 78, 193-202.

Abstract

The dynamic nature of airports demands the development of scheduling algorithms that are computationally efficient and therefore amenable to replanning when new traffic events occur. The main objective of this research is to design an algorithm with very low computational times able to minimize delays in the scheduled times of arrival and departure flights in an airport with a mixed-operation runway, under wake vortex separation and Constrained Position Shifting restrictions. The simulated annealing algorithm obtains a 95% improvement on time delays in less than one second of computation for the test instances generated, which means that it can be used online for high-demand scenarios to reduce delays. It has also been tested in a public testbed as well as in a real environment, showing an improvement of 30% in the time delays of real operations at London Gatwick airport.

Keywords: scheduling, wake vortex separation, mixed-operation runway, constrained position shifting, first-come first-served, simulated annealing

1. INTRODUCTION

Some 3.1 billion passengers made use of the global air transport network for their business and tourism needs in 2013. The annual passenger total was up by approximately 5% compared to 2012 and is expected to reach over 6.4 billion by 2030, based on current projections (ICAO, 2014). The number of aircraft departures reached 33 million globally last year, establishing a new record and surpassing the 2012 departure figure by more than one million flights. Scheduled passenger traffic grew at a rate of 5.2% (expressed in terms of revenue passenger-kilometres or RPKs) (ICAO, 2014).

Therefore, one of the central challenges facing the aviation industry is air traffic demand growth, which results in congestion in many airports – primarily hubs (Flores-Fillol, 2010). The economic cost of delay is enormous and will worsen as traffic demand increases. In 2010, the annual cost of US delays was $32.9B (Ball *et al.*, 2010).

The efficiency and effectiveness of Air Traffic Management (ATM) is enabled by air traffic flow management (ATFM). It contributes to the safety, efficiency, cost effectiveness and environmental sustainability of an ATM system. ATFM aims to enhance safety by ensuring the delivery of safe densities of traffic and by minimizing traffic surges. Its purpose is to balance traffic demand and available capacity. ATFM relies on the clear definition of capacities (i.e. number of flights that can be handled by an airport or in a route sector), as well as on the analysis of forecasted traffic flows (number of traffic flows that are expected in an airport or in an en route sector). ATFM therefore relies on the exchange of information related to flights, airspace availability and capacity. With ATFM, the various system stakeholders collaborate to

reconcile system resource constraints with economic and environmental priorities (Bertsimas *et al.*, 2011).

The dynamic nature of the terminal area necessitates the development of scheduling algorithms that are computationally efficient and therefore amenable to replanning when new traffic events occur (Mukherjee and Hansen, 2009), such as when a new aircraft enters the centre boundary or when data updates are obtained (Balakrishnan and Chandran, 2010). The challenge lies in simultaneously achieving safety, efficiency and equity, which are often competing objectives, and doing so in a reasonable amount of time (Anagnostakis *et al.*, 2001). Safety is achieved by maintaining separation between aircraft and by satisfying downstream metering constraints; efficiency is equivalent to achieving high throughput and/or low average delay; and equity is modelled by limiting the deviation from a nominal order or by minimizing variance in delay.

However, few solution approaches have been able to simultaneously model all three components and optimally solve the runway scheduling problem in a computationally tractable manner. One reason for this computational hurdle is that most runway scheduling models are, from a theoretical perspective, inherently hard to solve (Beasley *et al.*, 2000).

Lieder *et al.* (2015) mention that no efficient methods have been proposed in the literature for the arrival-landing problem (ALP) that are capable of solving large problem instances. The most common solution approaches are dynamic programming (DP) approaches (Dear, 1976); branch-and-bound (B&B) algorithms (Abela *et al.*, 1993); mixed-integer programming (MIP) formulations (Beasley *et al.*, 2000), which are solved with a standard solver; and heuristic solution approaches (Pinol & Beasley,

2006). However, these implementations resort to heuristic or approximate approaches that produce "good" solutions in short computational times (not in real time) but are not suitable for large problems and therefore, not useful to apply in real environments.

Managing take-offs and landings of any airport is a complex problem that plays an important role in airport management. Runways and air controllers are limited resources, so air traffic needs to be planned carefully in order to limit peak demand and satisfy as many airlines' requirements as possible. However, unpredictable delays make it almost impossible to schedule planes with precision and in advance (Artiouchine *et al.*, 2008). Indeed, the initial schedule needs to be reorganized when planes are close enough to the airport, i.e. when they approach the TRACON (Terminal Radar Approach Control Facilities – between 5 and 50 miles from the airport), or are on the ground (delays on the ground cost half as much as they do in the air) (Inniss and Ball, 2004).

Hence, congestion delays materialize either on the ground, where aircraft have to wait before accessing a runway, or during the flight, where they are deviated from their intended trajectory. Delays may also propagate through the whole transportation network when the schedule buffers are tight. Congestion delays can be managed at a strategic level (by runway expansion or shorter separation standards), a pre-tactical level (by splitting flows and sectors) or a tactical level (by sequencing and re-sequencing aircraft) (Gwiggner and Nagaoka, 2014).

Our work deals with the fact that aircraft scheduling needs fast algorithms that help air traffic controllers to take real time decisions. These algorithms need to be able to process large amounts of data in a very short time. In this paper, we will evaluate a simulated annealing (SA) algorithm designed to calculate the landing and/or departure

times, minimizing the total delay from the estimated landing time (ELDT) or from the estimated time of departure (ETD), subject to wake vortex separation (WVS) requirements and constrained position shifting (CPS). Following the Gwiggner and Nagaoka (2014) classification, this research is placed at the tactical level and focused on airports with single, mixed-operation runways that need to optimize their resources to deal with capacity problems. We have considered three factors of interest for the algorithm evaluation: number of aircraft, wake-turbulence category and number of shifts of the aircraft from their initial position in the sequence. An important condition that we propose to cover is to obtain online results as requested in real operations. To our knowledge, no previous work has addressed this scenario.

The paper is organized as follows: Section 2 describes the problem of the runway bottleneck and different approaches that have been studied in this field. In Section 3, the proposed solution is presented and the SA algorithm defined. Section 4 provides a description of the experimental tests carried out in order to analyse the behaviour of the algorithm. Section 5 presents the numerical results obtained, not only from the experimental tests but also from the results of the algorithm in the real environment of Gatwick airport. Finally, a short summary is given in Section 6, together with suggested topics for future research in this area.

2. PROBLEM DESCRIPTION AND PREVIOUS APPROACHES

The runway system, as a resource shared by all aircraft, creates a significant flow "bottleneck" that increases delays. The flow of aircraft entering the airport radar range is not very orderly (Soomer and Franx, 2008), but a regular, balanced supply of arrival traffic is essential for the successful planning of dense arrival flows (Post and de

Jonge, 1997). The main objective of this research is to explore the possibility of a time-efficient metaheuristic to develop an algorithm that minimizes deviations from the scheduled times of arrival and departure of flights in an airport. A scenario with a single, mixed-operation runway will be considered, the goal being to deliver schedules within a few seconds.

Based on the level of airport and air traffic control (ATC) system impact, as a result of a constraint violation, two types of constraint can be distinguished (Anagnostakis *et al.*, 2001):

- **Hard constraints**. Being inviolable because they affect safety, they must be satisfied by all generated solutions. One of the most limiting factors for the take-off and landing frequency in airports is the danger of wake turbulence. Wake vortex effects are generally proportional to aircraft weight and the lighter the following aircraft, the more it suffers from wake vortex effects, therefore demanding greater separation from the leading plane (Artiouchine *et al.*, 2008). The required separation times between successive aircraft (the WVS) depend on the types of the two planes involved; the order in which aircraft land/take-off plays an important role in the capacity of the runway (Bäuerle *et al.*, 2007).
- **Weak constraints** can be violated but the smaller the violation the better the solution quality. An example is the scheduled take-off times (or slots). A slot is the scheduled time of departure or arrival available, or allocated to, an aircraft movement on a specific date at the so-called capacity-constrained airports (also referred to as slot-controlled, slot-restricted, slot-constrained, or slot-coordinated airports). In flight scheduling, flights have an optimal schedule time to respond to time-dependent demand and the requirement of frequency plans, of available fleets

and of aircraft routings, among others. However, the capacity limit of runways could mean that some flights cannot operate at their expected time. Hence, it is acceptable to modify the schedule times of some flights from their optimal times (Cao and Kanafani, 2000).

Controllers have limited flexibility in reordering aircraft, and requirements in scheduling solutions are fairness and safety. In this study, WVSs (measured in minimum time separations between flights) will be considered as hard constraints, and time of arrival/departure and sequence order as weak constraints.

To tackle the latter, CPS is taken into account. The CPS approach is based on a fundamental underlying principle that involves the specification of a parameter, which limits the maximum number of position shifts (forward or rearward) that any aircraft will receive with respect to its first-come, first-served (FCFS) position (Balakrishnan and Chandran, 2010). Dear (1976) observed that CPS increases the runway throughput rate, treats individual aircraft equitably and fits well within the capabilities of today's computers because updating the solution avoids “global” re-sequencing, amongst other characteristics.

Psaraftis (1980) was the first to develop a polynomial-time algorithm for scheduling under CPS. This algorithm relied on all aircraft of the same type being identical, which did not accommodate time-window restrictions on aircraft or precedence relationships among aircraft, thus effectively scheduling all aircraft of a certain type in FCFS order. Trivizas (1998) proposed a search-based algorithm. His model also failed to account for time-window restrictions and precedence constraints. The difficulty of incorporating all operational constraints within a CPS framework even led to a conjecture by Carr (2004) that, in general, runway scheduling under CPS had

exponential complexity.

Bianco *et al*. (2006) consider two parameters, namely the maximum position shifting (MPS) to prevent an aircraft from being excessively delayed, and the relative position shifting (RPS) to limit the pilots' and controllers' workloads during aircraft resequencing.

Malaek and Naderi (2008) described a new procedure for the real time dynamic scheduling of arrival aircraft via computing optimal sequences, which eliminates most of the shortcomings in k-CPS (k being the maximum number of position shifts) while respecting its optimal nature to minimize the makespan and mean delay time. The new approach, referred to as dynamic position shifting (DPS), allows operational considerations to be easily implemented. However, the complexity of computations increases linearly with respect to the total number of aircraft and runways, due to the elimination of recursive computations and the maximum number of allowable position shifting being computed dynamically as a function of traffic flow. Their algorithm behaves very similarly to that of FCFS in the normal traffic while it acts similarly to that of k-CPS methods in heavy traffic. However, it is only valid for arrivals as it is dependent on the transition times in the TRACON and the times required for aircraft to travel from the meter fixes to the runway threshold.

There is an interesting stream of research on models and algorithms for the control of a terminal manoeuvring area (TMA) that is based on job shop scheduling as discussed in Bennell *et al.* (2013). From this point of view, the TMA can be viewed as a single-machine (Bianco *et al*., 1999) or as a job shop scheduling problem (Bennell *et al.*, 2013). Bianco *et al*. (1999) show that the scheduling problem is equivalent to the Cumulative Travelling Salesman Problem with Ready Times. The problem becomes

more complex if both rescheduling and rerouting are implied in balancing the runway workload and minimizing delay propagation. When taking these two problems together into consideration, job shop methodology is a useful way of modelling the problem. Therefore, heuristic algorithms are required to compute good quality solutions in a short computation time (Samà *et al*., 2017). D'Ariano *et al*. (2012) use a truncated branch and bound algorithm to compute aircraft schedules with fixed routes which is then incorporated in a tabu search (TS) scheme for aircraft rerouting. Also Samà *et al*. (2014) develop and compare different models for simultaneous aircraft scheduling and routing, including strong traffic disturbances.

However, none of them can offer online solutions with their approaches. D'Ariano *et al*. (2015) developed three formulations based in graphs that offer online solutions by assigning to each aircraft the start time from the fixed, and all relevant points in such a way that all aircraft conflicts are resolved. Samà *et al*. (2013) make use of alternative graphs, which consist of dividing the problem into multiple steps, enabling the dynamic management of aircraft for large time horizons. Graphs are also used by Samà *et al*. (2016) to examine the trade-off between some classical performance indicators (tardiness, priority, throughput, and number of deadline violations) in a very complete and interesting study.

Other methodologies are present in the literature covering this scheduling problem. For example, Tavakkoli-Moghaddam *et al*. (2012) examined ways of landing aircraft with the least waiting time in time windows under critical conditions using a fuzzy programming approach and an estimator for the landing sequence of planes. However, the degree of satisfaction for more than 20 planes in the sequence is less than unity, which makes this approach not particularly useful in real environments.

Ernst *et al.* (1999) presented a specialized simplex algorithm, which evaluates the landing times, and then a problem space search heuristic is used as well as a B&B method for both single- and multiple-runway problems. Their objective is the landing problem meeting the separation criteria between all pairs of planes (not just successive ones) and where each plane has an allowable time window. However, the scenarios described consider no more than 50 planes.

Metaheuristics approaches are also present in the literature, including hybrid genetic algorithms (GAs). Ghizlane *et al.* (2013) studied the multiple runway case of the aircraft landing problem (MRALP), through four hybrid algorithms that use two computational heuristic search techniques, namely, TS and GAs, offering competitive solutions in terms of quality and robustness. However, in the best-case scenario (instances of fewer than 50 planes), results are achieved in more than 50 seconds. Genetic search methods were previously studied by Hansen (2004), with the purpose of investigating the utility of the genetic search approach using characteristic sets of TMA problems instead of particular solutions for certain airports, and also by Hu and Di Paolo (2009) who designed a GA with uniform crossover to tackle the aircraft arrival sequencing and scheduling (ASS) problem in multi-runway systems. Other than scheduling, Liu (2010) developed a solution procedure based on a genetic local search (GLS) algorithm for solving the runway dependent Airport Layout Plan (ALP) for determining the runway allocation, sequence and landing time for arriving aircraft. However, these algorithms are evaluated with small sets of planes and without considering their use in real environments.

Other metaheuristics that deal with more realistic scenarios were also considered. Pinol and Beasley (2006) in their study presented the scatter search and bionomic algorithm applied to landing problems involving up to 500 aircraft and five runways.

None of these authors deals with the mixed-operation runway scheduling problem in real time and with a large number of flights in the sequence. One of the main purposes of this research is to develop an algorithm that can find good results in real time, in order to be applied in real situations.

3. PROPOSED APPROACH

As stated above, the scenario considered is a sequence of flights in a single mixed-operation runway subject to CPS. The scope under research here is to develop a suitable model that finds a schedule as similar as possible to the optimal one in a very short time. The optimal schedule is a sequence of flights with target times satisfying the minimum safety separations that minimize the total delay time of the sequence considered. Note that we have not considered programming aircraft before their estimated time as some other, more schedule-oriented researches consider (e.g. Pinol & Beasley, 2006; Salehipour *et al.*, 2013), since this procedure is not always possible in the TMA (as it implies manoeuvring operations, taxiing, etc.). This objective allows not only reducing delays, but also maximizing runway capacity, thus reducing congestion at airports.

According to the review of the literature, both exact and heuristic algorithms have been developed for scheduling problems. Given the complexity of the problem, exact methods do not perform as required for medium-sized instances (Salehipour *et al.*, 2013); researchers are using heuristic algorithms as solution approaches for the problem. Although these algorithms do not guarantee optimal solutions, their performance in delivering competitive schedules in short periods of time makes them very attractive. When looking for a fast, meta-heuristic model, SA is on many

occasions the chosen alternative, given its performance as well as its simplicity (Bertsimas & Tsitsiklis, 1993; Salehipour *et al.*, 2013).

The objective is thus to minimize delays in scheduled times – in other words, to minimize the total cost (delay in the target times) considering the two following constraints: a safety interval between successive operations determined by the WVS and the runway capacity of the airport; and forcing each aircraft to comply with the CPS restriction, which means that there is a limit to the maximum number of position shifts (forwards or backwards) that any aircraft will receive with respect to its FCFS position.

Figure 1 illustrates a pseudocode for the SA algorithm, where S_{act} is the actual solution and S_{cand} is the candidate solution to be compared with S_{act}. The parameters used for developing the SA algorithm are:

T_o	initial temperature
α	lowering rate of the temperature
T_f	final temperature
L	number of times that the algorithm tries to find new solutions before decreasing the temperature

As usual, when implementing SA metaheuristics an initial solution is defined as the actual one S_{act}, to start exploring the solution space looking for a better candidate. To allow scape from local optima, with a certain probability $U(0,1) < e^{(-\delta/T)}$, SA accepts worse solutions during the search, allowing a more extensive exploration.

The variables used in the following model are:

N	total number of flights in the sequence

WTC_i wake turbulence category of flight i (which can be light, medium or heavy)

WVS_{ij} wake vortex minimum separation between flights i and j

p_i flight in position i in the sequence solution

v_i position occupied by flight *i* in the sequence solution

e_i estimated time of landing/arrival of flight i

t_i target time of landing/arrival of flight i, computed taking into account the refined planning times and WVS requirements

δ Difference between the cost (S_{cand}) and cost (S_{act})

c_i Cost penalty for unit of delay of flight i

========== Figure 1 ==========

Table 1 shows the values considered in this paper for the separation times between pairs of aircraft. These are average values based on real information from different airports.

========== Table 1==========

Some authors (e.g. Bennell *et al*., 2013; Chandran & Balakrishnan, 2007) assume that the separations satisfy the triangle inequality, that is:

$$WVS_{ij} \leq WVS_{ik}+WVS_{kj} \text{ for all aircraft types i, j, k} \quad (1)$$

However, Balakrishnan and Chandran (2010) prove that the triangle inequality does not necessarily hold when both arrivals and departures are scheduled simultaneously. Since we are considering mixed-operation runway airports, it is not possible for us to assume triangle inequality and therefore the target times are calculated for each flight i as:

$$t_i = \max\{e_i; t_j + WVS_{ji}\} \qquad \forall j = 1, \ldots, i-1 \qquad (2)$$

The objective function can then be defined as:

$$\min C_\sigma = \min (\Sigma_{i=1..N} (|t_i - e_i|*c_i) \qquad (3)$$

One of the main challenges when designing SA algorithms is how to guarantee the generation of feasible solutions. In our particular case, a solution is feasible when its flights in the sequence fulfil the CPS condition, and in addition the wake turbulence category (WTC) separations are met.

Figure 2 shows an example of the process of generating a feasible solution regarding the CPS constraint (note that WTC is easier to guarantee by just delaying the flight times accordingly).

========== Figure 2 ==========

In order to generate feasible solutions, our procedure randomly chooses a position n and defines the set Ω_n={n-CPS, n-CPS+1, …, n-1, n+1,…, n+CPS-1, n+CPS} containing 2Δ flights *"CPS-compatible"* with flight n. Here Δ=(2×CPS)-1 represents the number of flights that are potentially exchangeable with flight n.

Then, it is necessary to find an element n' $\in$ Ω_n fulfilling (4) and (5).

$$v_{n'} \in \{n\text{-CPS},\ldots, n\text{+CPS}\} \qquad (4)$$

$$p \in \{n'\text{-CPS},\ldots, n'\text{+CPS}\} \qquad (5)$$

In order to look for n', the different elements of Ω_n are randomly evaluated until an n' is found that meets both conditions. If both conditions are met, it is guaranteed that the exchange (or swap) of n and n' will lead to a new feasible solution S_{cand} which will be

evaluated and compared with S_{act}. Note that it will always be possible to find a feasible solution around S_{act}, although not for any pair (n, n') will it be possible. For example, let us suppose an initial sequence {1,2,3} with CPS =1. If we obtain a random position n=2 and swap it with n'=3, the sequence then becomes {1,3,2}; if the next random position number is n=1, then we cannot find a CPS-compatible n' with which to swap flight 1. However, there will always be a feasible solution, which is returning to the initial sequence {1,2,3}.

If the cost of S_{cand} is lower than the cost of S_{act}, then the new current solution would be S_{cand}. In order to escape from a local minimum, the SA algorithm allows some worse solutions S_{cand} to become S_{act} with a probability $U(0,1) < e^{(-\delta/T)}$.

4. EXPERIMENTAL FRAMEWORK

Once the SA algorithm has been defined, a set of tests has to be conducted in order to validate the behaviour of the algorithm. The idea is to generate a large number of random instances for which we know the optimal result and compare them with the solutions found by the algorithm.

Three factors have been identified in the previous literature review as potentially influencing the results obtained. In order to quantify how relevant these factors are, we have introduced them in the procedure for the generation of the instances:

- **F1.** *Wake turbulence category of the flights* in the sequence. We have considered two different situations: one in which all the flights have WTC = Medium (F1.1.) and another where all flights have a random WTC (F1.2.) between the three categories considered (Heavy, Medium and Light). This allows us to analyse whether having planes of the same WTC (i.e. each of them needs to satisfy the

same separation between them, so the triangle inequality is satisfied) is relevant or not for the performance of the algorithm.

- **F2.** *Constrained position shifting*. Five levels have been chosen for CPS = {1; 2; 3; 4; 5}, bearing in mind that controllers always wish to keep the sequence as similar as possible to the FCFS sequence received.
- **F3.** *Number of flights*. In order to have a realistically managed size sequence of planes in an airport with a single runway that can be held for a short period of time (couple of hours), we have studied sequences of 50 (F3.1), 100 (F3.2), 150 (F3.3) and 200 (F3.4) flights. Although it is not realistic for a real airport to consider so large a sequence in a mixed-operation runway (the maximum throughput capacity for a homogeneous fleet mix on a single runway is 60 flights per hour (Sherry, 2009), this will allow us to check the ability of the algorithm to manage extremely large instances.

4.1. Instance generation

As previously outlined, once the factors have been identified, it is necessary to generate a set of instances for which we know the optimal solution. This means generating instances of the different number of flights considered (F3) with cost 0 that fulfil the WTC considerations. These instances are then randomly shuffled in respect of the five levels of CPS considered (F2). The instances generated as a result are then considered as the input FCFS sequences for the algorithm.

Regarding the total number of instances to generate in our experiments, 50 replications were generated for each of the 2x5x4 factor level combinations of the three factors considered, giving a total of 40×50 = 2,000 instances. We have considered that the

penalty cost for unit of delay of flight plan i is one in the generation of these instances.

The procedure followed to generate such feasible instances with cost 0 is the following:

1. Let us suppose the flight sequence <1, 2, 3, 4, …> is the optimal solution
2. The optimal estimated times of the flights in the sequence are calculated, taking into account their WTC separation restrictions.
3. In order to shuffle the flights according to the CPS limitation, a data structure is generated. Each column represents the position of a flight in the sequence and contains all the possible flights that can be in that position respecting the CPS limitation.

========== Figure 3 ==========

Figure 3 shows an example of the last step for CPS = 2. For example, flight 4 can be moved to positions (columns) {2, 3, 4, 5, 6} respecting the CPS = 2 limitation. In order to create the random sequence that fulfils the CPS condition, we follow the next steps:

i. Starting in the first column, we randomly choose a value among those possible (for example 2 in Figure 3).
ii. The flight selected in the previous step is deleted from all the following columns in order not to select it again (since it is not feasible to have a repeated flight in the sequence).
iii. Go to the next column. If the flight in that column is its last chance to be selected, it is automatically chosen (in Figure 3, flight 1 of the third column). This happens when we are evaluating a column j where there is

still available the flight j-CPS (it has not been deleted in previous steps). If not, repeat the previous step for the successive columns.

4.2. SA Parameters tuning

The SA parameters are, as mentioned above, the initial (T_o) and final (T_f) temperatures; the cooling rate (α); and the number of iterations at a certain temperature (L), which was fixed at one.

In order to determine the optimal values for T_o, α, T_f, three levels of each parameter were considered based on previous tests and SA literature (Aarts & Korst, 1989; Dowsland & Adenso-Díaz, 2001; Van Laarhoven & Aarts, 1988): $T_o \in \{200, 600, 1{,}000\}$; $\alpha \in \{0.99, 0.999, 0.9999\}$; $T_f \in \{0.01, 0.001, 0.0001\}$. Choosing five instances out of the 50 generated for each of the 2x5x4 combinations of levels, makes a total of $(40 \times 5) \times 3^3 = 5{,}400$ runs of the algorithm for testing the SA parameters.

Given that the data collected do not fulfil the normality assumptions (Kolmogorov-Smirnov p = 0.000), in order to study the influence of each of the factors in the minimization of the cost (or maximize the improvement percentage), a Kruskal-Wallis analysis seems to be the most appropriate. The results of this non-parametric analysis confirm that there are highly significant differences between the factors considered for both levels of α (p = 0.000) and T_o (p = 0.043), but not for T_f (p = 0.567). Level 3 of both significant factors shows the best results (Table 2). Therefore, α = 0.9999, T_o = 1,000 and T_f = 0.01 are the values chosen for the experiments to be carried out.

========== Table 2 ==========

5. RESULTS

In this section we present the results obtained by the algorithm proposed with the SA parameters tuned according to Section 4.2. We have used the testbed of 2,000 instances generated in Section 4.1, the 12 instances publicly available from OR-Library (Beasley, 1990) involving from 10 to 500 aircraft, and finally a real sequence of flights from Gatwick airport. The results are measured in terms of percentage of improvement,

$$\% \text{ improvement} = (C_{FCFS} - C_{solution}) / C_{FCFS} \quad (6)$$

where C_{FCFS} and $C_{solution}$ are the costs of the FCFS solution and the found solution, calculated as described in equation (3).

5.1. Testbed

As can be seen in

Figure 4, for 828 out of 2,000 instances (41.4% of the total of instances generated), the algorithm was able to find the optimal solution (which means finding the sequence of cost 0); a 98.65% of instances achieved more than a 95% improvement in the cost of the sequence.

========== Figure 4 ==========

A Kruskal-Wallis test was performed in order to test the importance of the three

factors considered. Results show that F3 (i.e. number of flights) is not as significant (p = 0.214) in the explanation of the improvement as are F1 (p = 0.001) and F2 (p = 0.000). This means that our algorithm's efficiency does not depend on the number of aircraft in the sequence considered. As shown in Table 3, factor F1.2 (different types of plane) and factor F2.1 (CPS=1) were found to be the most difficult to manage for the SA algorithm developed. These results are reasonable as factor F1.2 implies more complexity since the triangle inequality is met and when CPS=1 the number of possible reassignments of the aircraft in the sequence to find their best position is smaller. On the other hand, and as would be expected, the best results are achieved for high values of CPS (F2) and only one type of plane (F1.1).

Table 3 also shows the average percentage GAP as defined in Pinol & Beasley (2006). The values of the GAPS obtained show that our algorithm is able to find the optimal solution in all the cases. Lower bound values show that the worst case is a 90% improvement for some cases.

========== Table 3 ==========

As mentioned in the objectives, computational time is very important in this study.

Figure 5 and Figure 6 show that really good results are obtained in less than one second. As would be expected, results improve as the algorithm makes more interactions (i.e. as we allow longer computational times). As can be seen in

Figure 5, in the 15,000 iterations, the results reached average improvements of 80% in just 0.15 seconds while the 100% average improvement (i.e. solution sequence cost is 0) is reached in 0.25 seconds (in the 30,000 iteration).

========== Figure 5 ==========

========== Figure 6 ==========

Figure 6 shows the evolution of the improvement for the 2,000 instances in different iterations. It can be seen that in the last iteration (50,000) almost all the instances have an improvement in the cost of the sequence higher than 95%.

5.2. OR-Library Instances

For the sake of comparison with some publicly available sets of instances, we have considered the 13 aircraft landing data files from the OR-Library (Beasley, 1990). In order to be able to process those 13 instances, we needed to perform some necessary

adjustments, given that those instances were not designed exactly for the problem we are considering:

- We have considered that the order of the flights in the files is the FCFS sequence.
- We have considered as the estimated time of landing the target landing time.
- Although no data about the WTC of each flight or the aircraft type is provided by the OR-library, they consider a matrix of separation time required between pairs of flights, what we can use directly.
- Since the CPS concept is not present in the OR-Library, we have computed the instances for a range of CPS values, starting at 1 and up to 49 for all the instances.

========== Table 4, 5 ==========

Results are compared to those obtained by Pinol & Beasley (2006) and Salehipour *et al.* (2013) in Tables 4 and 5. Table 4 shows a comparison between the percentage of improvement of the best solution found both by Pinol & Beasley (2006) and Salehipour *et al.* (2013), and our SA, calculated related to the cost of the FCFS sequence. As expected, the majority of the results obtained by these two sets of authors are slightly better than the SA approach since their algorithms allow flights to land earlier than their estimated time, while ours does not consider this possibility based on usual airport operations. In spite of that, the percentage of improvement is not very different in most cases. The Airland 6 case deserves a special mention as our algorithm is able to improve on the results offered by the other authors. We must say that Airland 6 is the only case in which the data do not allow us to consider an earlier time of landing since the earliest landing time is equal to the ELDT. This is the

situation for which the SA algorithm was designed, which makes it reasonable to claim that the percentage improvement of the rest of the cases is obtained from considering earlier times of landing.

The advantage regarding computational times of the SA algorithm as pursued (see Table 5) must be stated, given the necessity of obtaining online support. Therefore, the SA algorithm is able to obtain competitive improvements in the cost of the sequences (in most cases in less than a second), even for those instances which are not exactly aimed at the problem they tackle. Figure 7 shows a comparison between the percentage of improvement of our algorithm and the optimal results of the instances. It also reflects the difference in processing time, denoting as the processing time of Pinol & Beasley (2006) and Salehipour *et al.* (2013), the minimum obtained by their algorithms.

========== Figure 7==========

5.3. Case study

Once the validity of the proposed approach has been established using the generated testbed, it is interesting to see what the algorithm behaviour is when using real data. For this purpose, Gatwick airport was chosen because it is the UK's second largest airport and is the busiest single runway commercial airport in the world (Gatwickairport, 2015).

Flight information was collected from 7th September 2015 from 6:00 am to 9:00 am (FlightRadar24, 2015). Unfortunately, FlightRadar provides only scheduled times and not real times at the runway for landing and taking-off. Therefore, in order to compute this real flight information, the following assumptions have been made:

- The FCFS sequence has been created by arranging the flights according to their "ready time" to land or take-off. Our algorithm considers the time that the plane is at the runway, i.e. "ready time" represents the time where the plane is at the runway ready to take-off or touching the runway at landing. Taking this in mind:
    - The "ready time" for a take-off has been calculated as its scheduled time plus its real taxi time from the stand to the runway.
    - The "ready time" for a landing has been calculated as its scheduled time less its real taxi time from the runway to the stand.

The minimum time separation depending on the WTC of each flight considered is based on Malaek and Naderi (2008). Since the WTC was not provided by FlightRadar24, it has been obtained by searching for the aircraft model (which is provided by FlightRadar24) in the International Civil Aviation Organization's **(**ICAO) database.

CPS is a theoretical parameter, so we have used 49 different values (from 1 to 49) to see how this parameter can influence the results in a real case and thus try to determine the optimal value of CPS. The SA parameter values used were the same as in our testbed experiments, and found to work better.

The real cost of the sequence (calculated as the deviations of the actual landing/take-off times of the flights) is 1,492 min. Our algorithm is able to reorganize the flights to obtain an average cost of 507 min of deviations, which means the total delay is reduced to a third of the real delay.

========== Figure 8 ==========

Figure 8 shows the evolution of the percentage of improvement depending on the CPS after running the SA algorithm. It can be seen that the improvement rises up to 30% for CPS 10. This improvement is calculated as mentioned before, by comparing the result with the cost of the FCFS sequence. This means that allowing flights to move up to 10 positions respect their initial position in the sequence, results in a reduction of the delay of the total sequence; however, allowing flights to move more positions respect the initial FCFS does not mean that the total delay of the sequence continues improving.

Regarding its potential implementation, the algorithm is able to deliver its solutions in less than one second, which is a reasonable computational time to be used in real environments, where controllers are used to obtaining online solutions.

It is worth noting that comparing the behaviour of the algorithm using real data has certain limitations. Flightradar24 (2015) only provides the scheduled times of flights, which are the ones we consider as estimated times. However, flight operations suffer from unexpected delays (related to technical problems or weather conditions for instance) that we cannot predict and take into account in the tests performed with our algorithm. However, knowing the last update of real estimated times, the algorithm is able to find better solutions by incorporating these delays in the calculation of the

optimal target times for the sequences considered.

When implementing this algorithm in real environments it would be necessary to agree the value of the CPS parameter with the controllers in order to obtain the best performance of the algorithm that satisfies their requirements of minimizing the movements of flights in the initial sequence.

6. CONCLUSIONS

Airports have a serious problem of saturation. A good schedule of runways is critical, but not many optimization tools have been implemented on sites that are able to deliver quick and efficient schedules.

In this paper, an SA algorithm for a specific case (considering WVS and CPS as constraints) has been tuned and developed. The analysis of the results, after an extensive experimental framework had been carried out (involving 2,000 instances for validating the algorithm results), shows that for improvements of 95% in the deviation from the target schedule, the algorithm presents good results in most cases, with just 15,000 iterations, in less than a quarter of a second. High CPS and only one type of plane make it easier to solve the problem. Up to 200 planes were considered in our data, with no reduction in efficiency.

In addition, a comparison with other algorithms that used a public library shows that our algorithm obtains competitive results in a significantly less time, fulfilling the objective of its online use. Also, real data from Gatwick airport were used to test the behaviour of the algorithm in a real situation, obtaining around a 33% improvement in the total delay of the sequence considered. Therefore, the algorithm can be used in real

environments since the results are achieved in a very short time (less than one second in most cases) and the improvements in the cost of sequence (without varying the FCFS sequence completely) are valuable.

There are a number of topics for further research. In particular, taking slots (or time windows) into account to make the algorithm more flexible and allowing flights to come in ahead of their scheduled time would add value to this approach. Also, extending the problem to multiple runways would be interesting, as main and busier airports usually have a multiple-runway configuration. Finally, the possibility of considering as an objective not only the total reduction delay but also the priority of flights would be interesting.

Acknowledgements. This research was carried out with the financial support of the Spanish Ministry of Science grant DPI2013-41469-P and the European Regional Development Fund (ERDF). The authors are grateful to the anonymous referees for their valuable comments and suggestions that have contributed to improving the quality of this paper.

7. REFERENCES

Aarts, E.H.L., Korst, J.H.M. (1989), *Simulated Annealing and Boltzmann Machines*, Wiley, Chichester.

Abela, J., Abramson, D., Krishnamoorthy, M., De Silva, A., Mills, G. (1993). "Computing optimal schedules for landing aircraft". In *Proceedings of the 12th National Conference of the Australian Society for Operations Research.* pp.71-90.

Anagnostakis, I., Clarke, J.P., Böhme, D., Völckers, U. (2001). "Runway operations planning and control: Sequencing and scheduling", *Journal of Aircraft*, vol.38, no.6, pp.988-996.

Artiouchine, K., Baptise, P., Dürr, C. (2008). "Runway sequencing with holding patterns", *European Journal of Operational Research*, vol.189, pp.1254-1266.

Balakrishnan, H., Chandran, B.G. (2010). "Algorithms for scheduling runways operations under constrained position shifting", *Operations Research*, vol.58, no.6, pp.1650-1665.

Ball, M., Barnhart, C., Dresner, M., Hansen, M., Neels, K., Odon, A., Peterson, E., Shery, L., Trani, A., Zou, B. (2010). "Total Delay Impact Study: A Comprehensive Assessment of the Costs and Impacts of Flight Delay in the United States". Technical report of the National Center of Excellence for Aviation Operations Research (NEXTOR), USA.

Bäuerle, N., Engelhardt-Funke, O., Kolonko, M. (2007): "On the waiting time of arriving aircrafts and the capacity of airports with one or two runways", *European Journal of Operational Research*, vol.177, pp.1180-1196.

Beasley J. (1990). "OR-library: Distributing test problems by electronic mail", *Journal of the Operational Research Society*, vol.41, pp.1069-1072, Available from: http://mscmga.ms.ic.ac.uk/info.html.

Beasley, J., Krishnamoorthy, M., Sharaiha, Y., Abramson, D. (2000). "Scheduling aircraft landings – the static case". *Transportation Science*, vol.34, no.2, pp.180-197.

Bennell, J.A., Mesgarpour, M., Potts, C.N. (2013). “Airport runway scheduling”. *Annals of Operations Research*, vol. 204, no.1, pp.249-270. doi:10.1007/s10479-012-1268-1.

Bertsimas, D., Lulli, G., Odoni, A. (2011). “An integer optimization approach to large-scale air traffic flow management”. *Operations Research*, vol.59, no.1, pp.211-227. doi:10.1287/opre.1100.0899.

Bertsimas, D., Tsitsiklis, J. (1993). “Simulated annealing”. *Statistical Science*, vol.8, no.1, pp.10-15.

Bianco, L., Dell’Olmo, P., Giordani, S. (1999). “Minimizing total completion time subject to release dates and sequence-dependent processing times”. *Annals of Operations Research*, vol.86, no.1, pp.393-415.

Bianco, L., Dell’Olmo, P., Giordani, S. (2006). “Scheduling models for air traffic control in terminal areas”. *Journal of Scheduling*, vol.9, no.3, pp.223-253. doi:10.1007/s10951-006-6779-7.

Cao, J.M., Kanafani, A. (2000). “Value of runway time slots for airlines”. *European Journal of Operational Research*, vol.126, pp.491-500. doi:10.1016/S0377-2217(99)00304-5.

Carr, F.R. (2004). “Robust Decision-Support Tools for Airport Surface Traffic”, *Massachusetts Institute of Technology*. Massachusetts.

Chandran, B., Balakrishnan, H. (2007). “A Dynamic Programming Algorithm for Robust Runway Scheduling”. *2007 American Control Conference*, pp.1161-1166. IEEE. doi:10.1109/ACC.2007.4282922.

D’Ariano, A., Pacciarelli, D., Pistelli, M. (2012). “Aircraft retiming and rerouting in vicinity of airports”. *IET Intelligent Transport Systems*, vol.6, no.4, pp.433-443. doi:10.1049/iet-its.2011.0182.

D’Ariano, A., Pacciarelli, D., Pistelli, M., Pranzo, M. (2015). “Real-Time Scheduling of Aircraft Arrivals and Departures in a Terminal Maneuvering Area”. *Networks*, vol.65, no.3, pp. 212-227.

Dear, R.G. (1976). “The dynamic scheduling of aircraft in the near terminal area: Technical report”. *Cambridge, MA: Flight Transportation Laboratory, Massachusetts Institute of Technology*.

Dowsland, K.A., Adenso-Díaz, B. (2001). “Diseño de heurísticas y fundamentos del recocido simulado”. *Revista Iberoamericana de Inteligencia Artificial*, no.20, pp.24-52.

Ernst, A.T., Krishnamoorthy, M., Storer, R.H. (1999). “Heuristic and Exact Algorithms for Scheduling Aircraft Landings”. *Networks*, vol.34, pp.229-241.

FlightRadar24: www.flightradar24.com (last enquiry: 7th September 2015).

Flores-Fillol, R. (2010) “Congested hubs”, *Transportation Research-B*, vol. 44, 358-370.

Gatwickairport: www.gatwickairport.com (last enquiry: 7th September 2015)

Ghizlane, B., El Khoukhi, F., Baccouche, M., Abdelhaq Belkadi, D.B. (2013). “Hybrid Algorithms For The Multiple Runway Aircraft Landing Problem”. *International Journal of Computer Science and Applications*, vol.10, no.2, pp.53-71.

Gwiggner, C., Nagaoka, S. (2014). “Data and queueing analysis of a Japanese air-traffic flow”. *European Journal of Operational Research*, vol.235, no.1, pp.265-275. doi:10.1016/j.ejor.2013.10.056.

Hansen, J.V. (2004). “Genetic search methods in air traffic control”. *Computers & Operations Research*, vol.31, no.3, pp.445-459. doi:10.1016/S0305-0548(02)00228-9.

Hu, X.B., Di Paolo, E. (2009). “An efficient genetic algorithm with uniform crossover for air traffic control”. *Computers and Operations Research*, vol.36, no.1, pp.245-259. doi:10.1016/j.cor.2007.09.005.

ICAO (2014), Air Navigation Report. www.icao.int (last enquiry: 13th October 2015).

Inniss, T., Ball, M.O. (2004): “Estimating one-parameter airport arrival capacity distribution for air traffic flow management”, *Air Traffic Control Quarterly*, vol.12, no.3, pp.223-251.

Lieder, A., Briskorn, D., Stolletz, R. (2015). “A dynamic programming approach for the aircraft landing problem with aircraft classes”. *European Journal of Operational Research*, vol.243, no.1, pp.61-69. doi:10.1016/j.ejor.2014.11.027.

Liu, Y.-H. (2010). “A genetic local search algorithm with a threshold accepting mechanism for solving the runway dependent aircraft landing problem”. *Optimization Letters*, vol.5, no.2, pp.229–245. doi:10.1007/s11590-010-0203-0.

Malaek, S.M.B., Naderi, E. (2008). “A New Scheduling Strategy for Aircraft Landings under Dynamic Position Shifting”, *IEEE Aerospace Conference*, pp.1-8.

Mukherjee, A., Hansen, M. (2009) “A dynamic rerouting model for air traffic flow management”, *Transportation Research-B*, vol. 43, no. 1, 159-171.

Pinol, H., Beasley, J.E. (2006). “Scatter Search and Bionomic Algorithms for the aircraft landing problem”. *European Journal of Operational Research*, vol.171, no.2, pp.439-462. doi:10.1016/j.ejor.2004.09.040.

Post, W., de Jonge, H.W.G. (1997): “Free flight in a ground controlled ATM environment”, *Technical Report, National Aerospace Laboratory NLR*, Netherlands.

Psaraftis, H.N. (1980). “A dynamic approach for sequencing groups of identical jobs”, *Operations Research*, vol.28, no.6, pp.1347-1359.

Salehipour, A., Modarres, M., Naeni, L. M. (2013). “An efficient hybrid meta-heuristic for aircraft landing problem”. *Computers and Operations Research*, vol.40, no.1, pp.207-213.

Samà, M., D’Ariano, A., D’Ariano, P., Pacciarelli, D. (2014). “Optimal aircraft scheduling and routing at a terminal control area during disturbances”. *Transportation Research Part C: Emerging Technologies*, vol.47, no.1, pp.61-85.

Samà, M., D'Ariano, A., D'Ariano, P., Pacciarelli, D. (2016). "Scheduling models for optimal aircraft traffic control at busy airports: Tardiness, priorities, equity and violations considerations". *Omega*. doi:10.1016/j.omega.2016.04.003.

Samà, M., D'Ariano, A., Corman, F., Pacciarelli, D. (2017). "Metaheuristics for efficient aircraft scheduling and re-routing at busy terminal control areas", *Transportation Research C: Emerging technologies*, in press, DOI: 10.1016/j.trc.2016.08.012.

Samà, M., D'Ariano, A., Pacciarelli, D. (2013). "Rolling horizon approach for aircraft scheduling in the terminal control area of busy airports". *Transportation Research Part E*, vol.60, pp.140-155.

Sherry, L. (2009), "Capacity of a single runway", *Center for Air Transportation Systems Research*, George Mason University, USA.

Soomer, M.J., Franx, G.J. (2008): "Scheduling aircraft landings using airlines' preferences", *European Journal of Operational Research*, vol.190, pp.277-291.

Tavakkoli-Moghaddam, R., Yaghoubi-Panah, M., Radmehr, F. (2012): "Scheduling the sequence of aircraft landings for a single runway using a fuzzy programming approach", *Journal of Air Transport Management*, vol.25, pp.15-18.

Trivizas, D.A. (1998), "Optimal scheduling with maximum position shift (MPS) constraints: a runway scheduling application", *Journal of Navigation*, vol.51, no.2, pp.250-266.

Van Laarhoven, P.J.M., Aarts, E.H.L. (1988). *Simulated Annealing – Theory and Applications*. Kluwer, Dordrecht.

WTC Leader	WTC Follower	Separation (seconds)
H	H	1,000
H	M	300
H	L	300
M	H	180
M	M	180
M	L	180
L	H	60
L	M	60
L	L	60

Table 1. Separations considered for each pair of aircraft WTC

Parameters	Levels	Mean	Standard deviation	95% Confidence Interval	
				Lower Bound	Upper Bound
α	0.9900	91.404	0.195	91.020	91.787
	0.9990	95.360	0.195	94.977	95.744
	0.9999	96.806	0.195	96.423	97.190
T_o	200	94.073	0.195	93.690	94.457
	600	94.668	0.195	94.285	95.052
	1,000	94.829	0.195	94.445	95.212

Table 2. Descriptive results for improvement (%) regarding α and T_o parameters in the initial tuning process

Factors	Levels	Mean	Average percentage gap	Lower bound	Std. deviation	95% Confidence Interval	
						Lower Bound	Upper Bound
F1	1	99.877	0	90.36	0.329	99.857	99.897
	2	98.789	0	92.23	1.032	98.725	98.854
F2	1	98.733	0	90.36	1.580	98.573	98.892
	2	99.373	0	90.36	0.791	99.295	99.451
	3	99.564	0	97.81	0.511	99.513	99.614
	4	99.537	0	95.51	0.520	99.486	99.588
	5	99.460	0	94.29	0.607	99.401	99.520
F3	50	99.369	0	90.77	0.952	99.284	99.453
	100	99.320	0	92.63	0.916	99.239	99.401
	150	99.363	0	92.20	0.879	99.285	99.441
	200	99.309	0	90.36	0.994	99.222	99.397

Table 3. Descriptive results for improvement (%) regarding F1, F2 and F3. Percentage GAP means the difference in percentage between the optimal solution and the best solution. Lower bound shows the worst result obtained by the algorithm. The standard deviation shows the amount of variation of the results obtained in these trials.

	%improvement$_{best}$	%improvement$_{SA}$
Airland1	98.26%	97.14%
Airland2	97.26%	96.81%
Airland3	98.58%	97.21%
Airland4	96.81%	94.33%
Airland5	96.44%	94.49%
Airland6	0%	58.42%
Airland7	60.99%	0%
Airland8	98.92%	98.30%
Airland9	84.77%	72.22%
Airland10	77.21%	55.37%
Airland11	81.30%	67.10%
Airland12	80.21%	65.84%
Airland13	74.91%	62.33%

Table 4. Comparative results for improvement (%) related to the FCFS sequence between the Z_{best} result of Pinol & Beasley (2006) instances and the SA solution

	Pinol & Beasley (2006)		Salehipour *et al.* (2013)				SA
	$time_{SS}$	$time_{BA}$	$time_{cplex}$	$time_{SA+VND}$	$time_{SA+VNS}$	$time_{SS}$	$time_{SA}$
Airland1	4	60	0.66	0	0	4	0.42
Airland2	6	90	0.49	1.59	1.38	6	0.55
Airland3	8	99	0.39	1.78	1.73	8	0.71
Airland4	8	95	5.12	1.98	2.85	8	0.73
Airland5	9	100	20.44	1.85	1.89	9	0.70
Airland6	158	274	0.1	2.12	2.14	158	0.92
Airland7	195	79	0.86	2,68	2.65	195	1.3
Airland8	42	287	0.98	7.1	7.31	42	1.46
Airland9	119	554	1000	11.59	10.12	119	2.18
Airland10	227	925	1000	20.12	20.75	227	2.54
Airland11	256	1417	1000	24.17	33.84	256	2.85
Airland12	381	2011	1000	219.03	198.85	381	3.31
Airland13	1237	5852	1000	566.82	528.84	1237	4.59

Table 5. Comparative results of processing time (sec) results for Pinol & Beasley (2006) instances for the following algorithms: SA, BA (Pinol & Beasley, 2006); CPLEX, SA+VND, SA+VNS, SS (Salehipour *et al.*, 2013); SA (the proposed algorithm)

```
INPUT (To, α, Tf , L)
T ← To
Sact ← Generate_initial_solution
WHILE T ≥ Tf DO
                BEGIN
                FOR  cont ← 1 TO L(T) DO
                                BEGIN
                                Scand ← Select_solution_N(Sact)
                                δ ← cost(Scand) – cost (Sact)
                                IF (U(0,1) < e^(-δ/T) OR (δ <0)
                                THEN Sact ← Scand
                                END
                 T ← α(T)
                 END
{OUTPUT: best Sact visited}
```

Figure 1. Simulated annealing algorithm structure

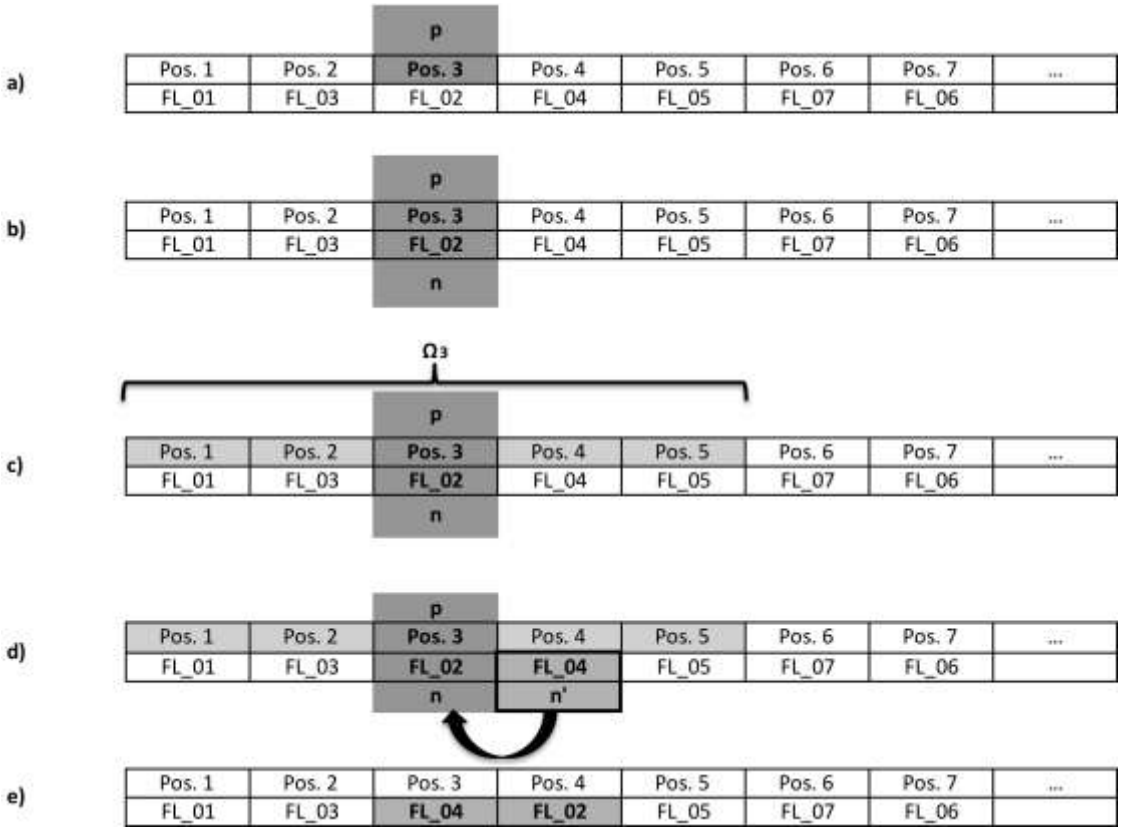

Figure 2. Process for generating a CPS feasible solution (CPS = 2): a) Choose a random position p; b) Find the flight occupying position p; c) Calculate the range of positions (Ω_3) that could be exchanged with flight n; d) Choose a random flight n' from Ω_3 fulfilling (4) and (5); e) Exchange n and n' to obtain a valid solution.

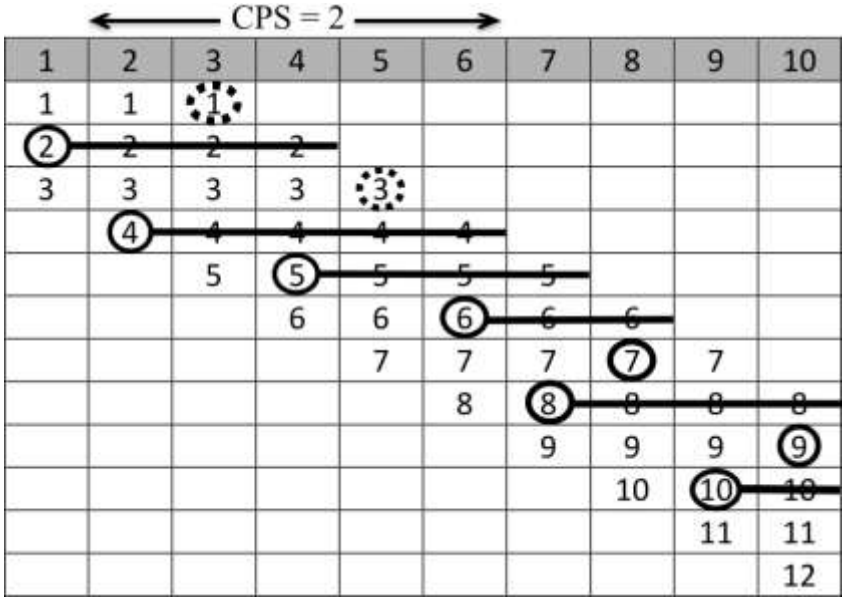

Figure 3. Example of instance generation (CPS = 2). At each iteration (column) a flight not deleted is randomly chosen, avoiding it from being selected later. When it is the last opportunity for a flight, it is forced to be chosen (case of flight 1 in column 3). Final feasible sequence: <2; 4; 1; 5; 3; 6; 8; 7; 10; 9;…>

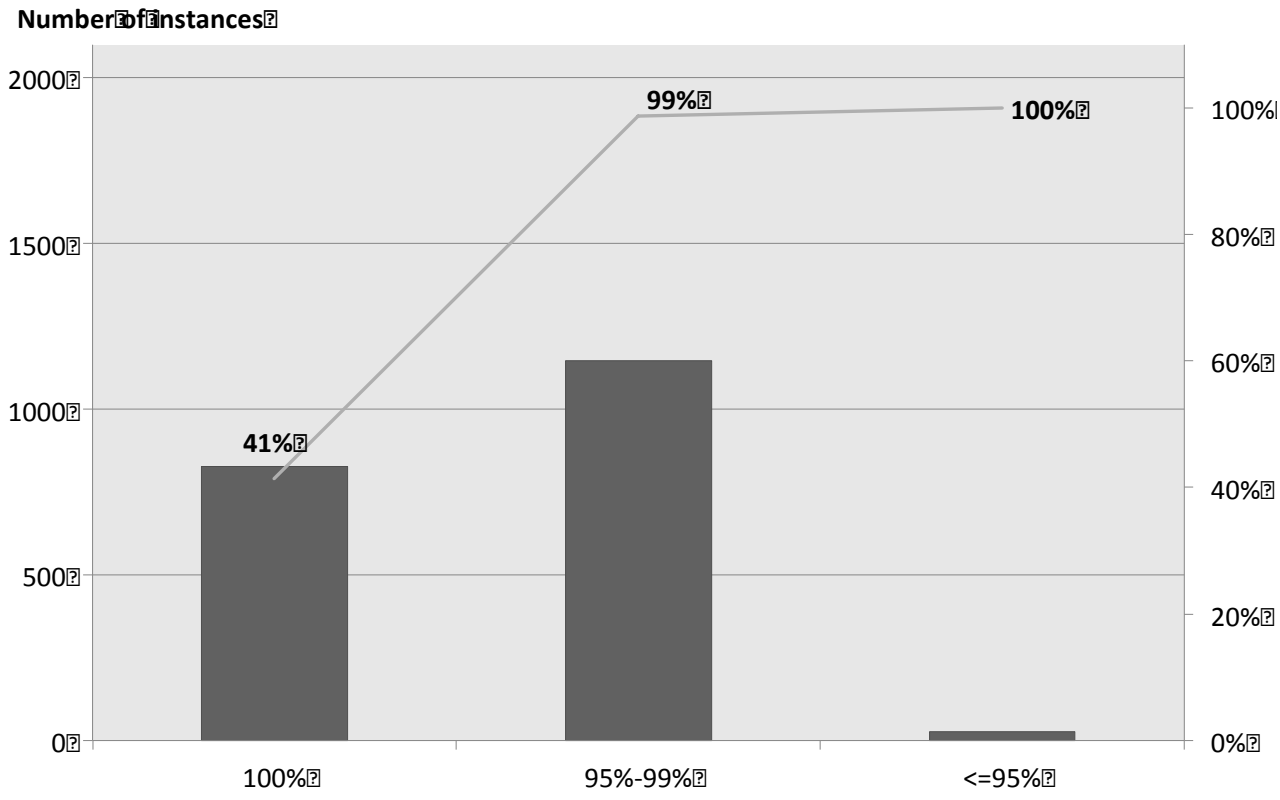

Figure 4. Pareto chart for the improvement obtained for the 2,000 instances generated. The SA algorithm finds the optimal solution for 41.5% of the instances, while only for less than 2% of the cases, the improvement found by the algorithm was smaller than 95%.

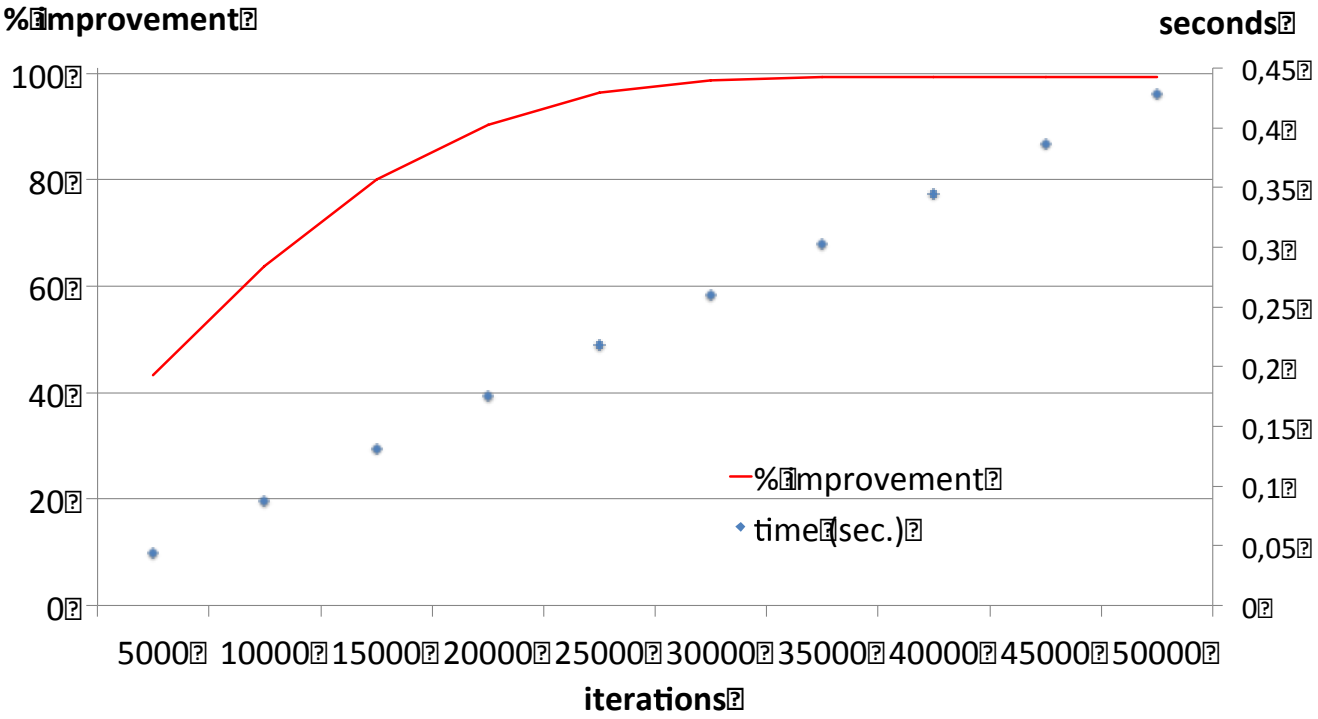

Figure 5. Evolution of the percentage of improvement (continuous line) and computational time average (dots) in seconds for the 2,000 instances generated

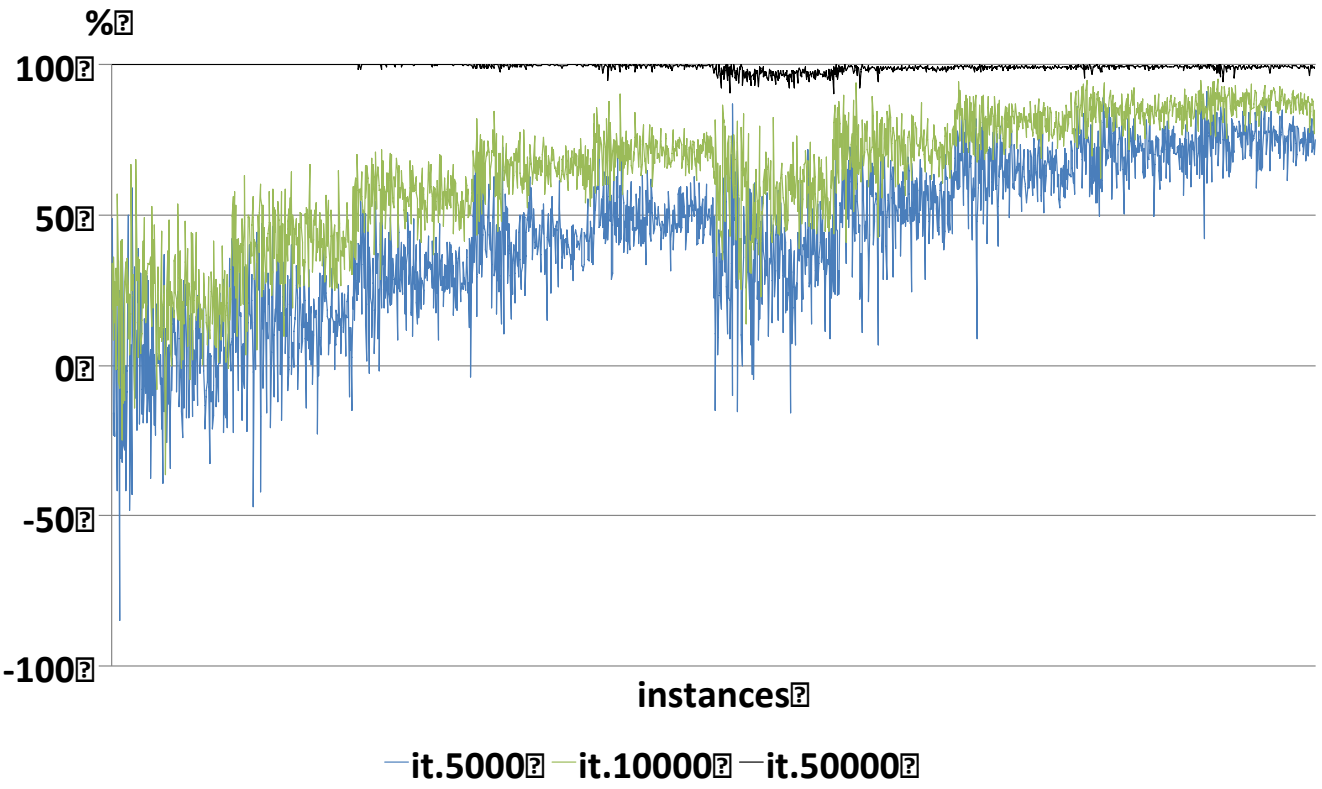

Figure 6. Average of the percentage of improvement for the 2,000 instances, depending on the number of iterations

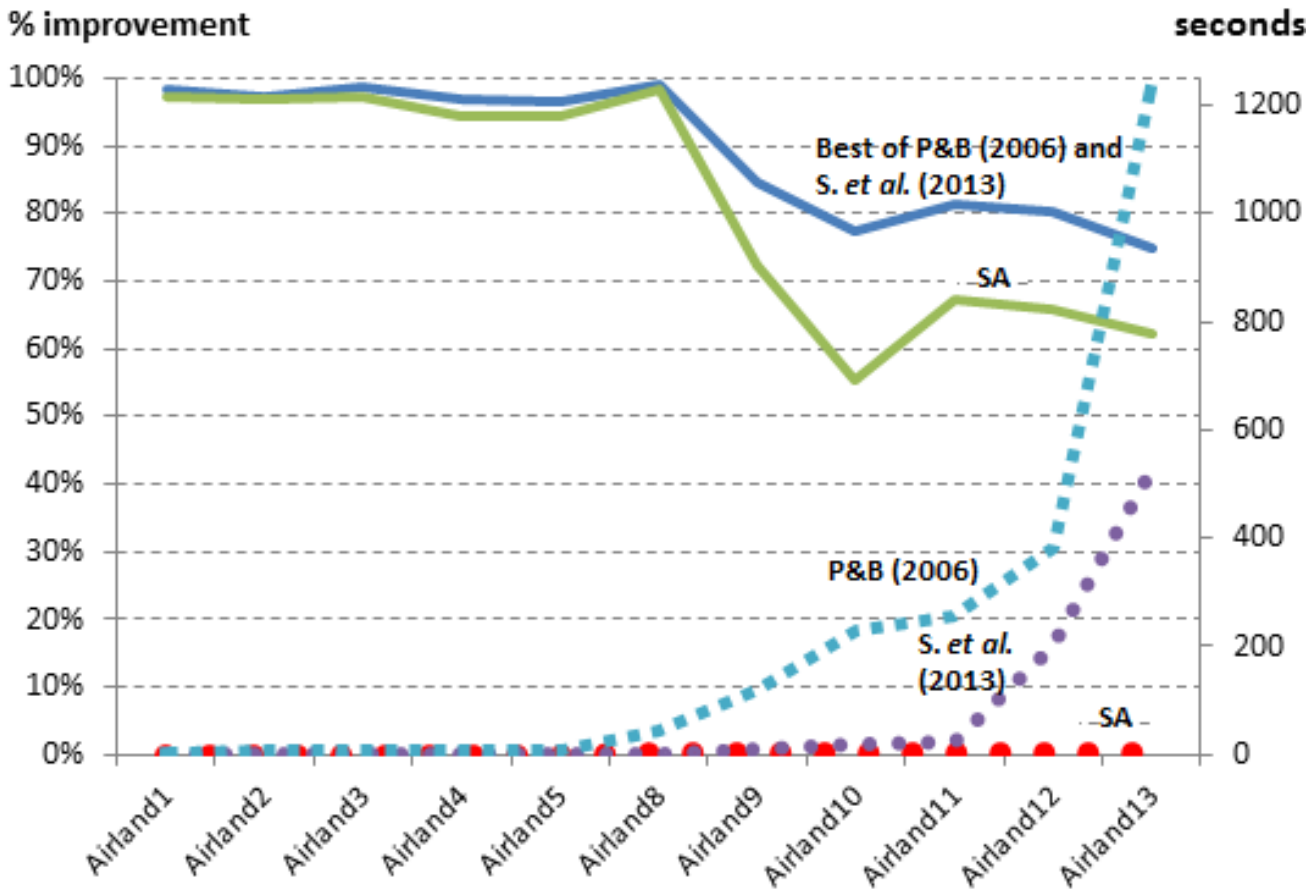

Figure 7. Comparison of different algorithms using the OR-Library instances, regarding percentage of improvement (continuous lines) and computational time in seconds (dotted lines)

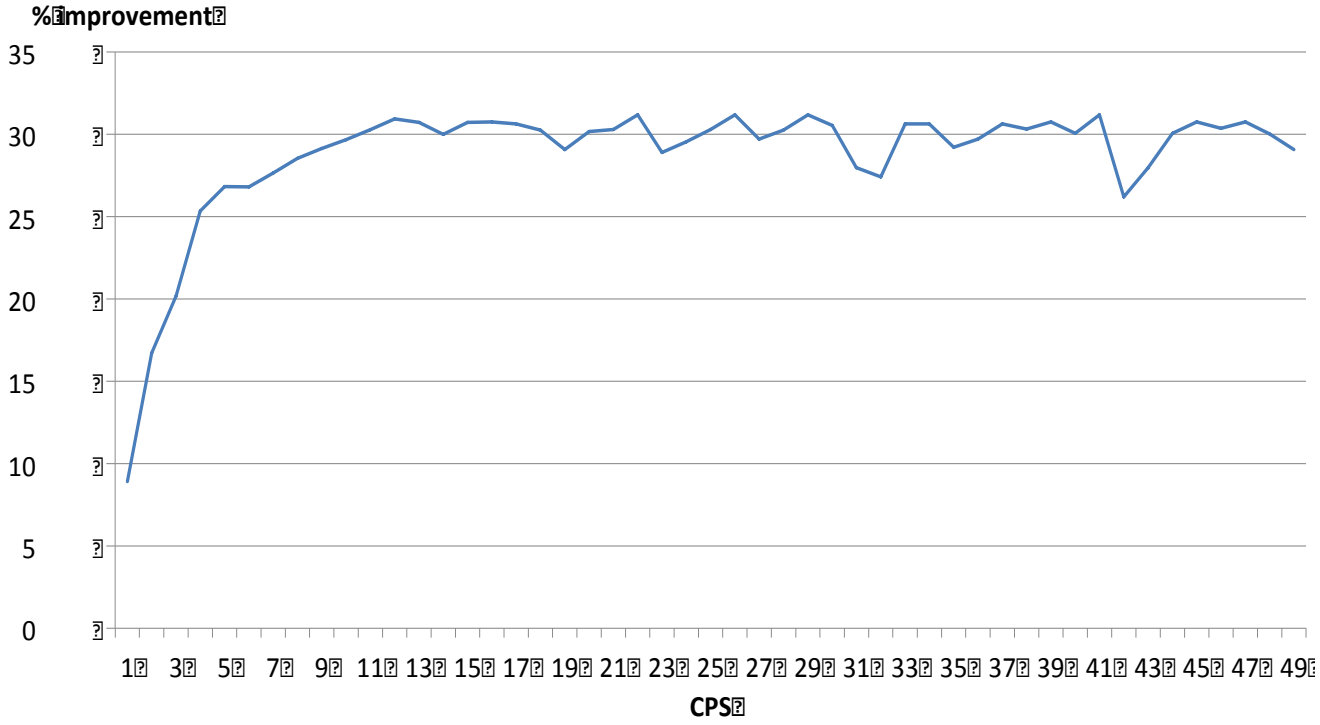

Figure 8. Evolution of the percentage of improvement for Gatwick airport real data, for different CPS values