



ELSEVIER

Available online at www.sciencedirect.com

SCIENCE @ DIRECT®

European Journal of Operational Research 171 (2006) 439–462

EUROPEAN
JOURNAL
OF OPERATIONAL
RESEARCH

www.elsevier.com/locate/ejor

O.R. Applications

Scatter Search and Bionomic Algorithms for the aircraft landing problem

H. Pinol^a, J.E. Beasley^{b,*}

^a *Tanaka Business School, Imperial College, London SW7 2AZ, UK*

^b *Department of Mathematical Sciences, Brunel University, Uxbridge UB8 3PH, UK*

Received 24 February 2004; accepted 3 September 2004

Available online 8 December 2004

Abstract

The problem of deciding how to land aircraft approaching an airport involves assigning each aircraft to an appropriate runway, computing a landing sequence for each runway and scheduling the landing time for each aircraft. Runway allocation, sequencing and scheduling for each aircraft must ensure the scheduled landing time lies within a predefined time window and meet separation time requirements with other aircraft. The objective is to achieve effective runway use.

In this paper, the multiple runway case of the static Aircraft Landing Problem is considered. Two heuristic techniques are presented: Scatter Search and the Bionomic Algorithm, population heuristic approaches that have not been applied to this problem before.

Computational results are presented for publicly available test problems involving up to 500 aircraft and five runways showing that feasible solutions of good quality can be produced relatively quickly.

© 2004 Elsevier B.V. All rights reserved.

Keywords: Metaheuristic; Genetic algorithm; Population heuristic; Transportation; Aircraft landing

1. Introduction

Before actually landing at an airport, an aircraft must go through an approach stage directed by air

traffic controllers. When entering the airport radar range, the aircraft's flight number, altitude and speed are transmitted to controllers within the air traffic control tower. Based on this information, controllers give instructions relating to the approach corridor to use and to the speed and altitude of the aircraft in order to align it with the allocated runway. During peak hours, controllers must handle safely and effectively landings of a continuous

* Corresponding author. Tel.: +44 1895 266219; fax: +44 1895 269732.

E-mail addresses: h.pinol@imperial.ac.uk (H. Pinol), john.beasley@brunel.ac.uk (J.E. Beasley).

flow of aircraft entering the radar range onto the assigned runway(s). The capacity of runways is highly constrained and this makes the scheduling of landings a difficult task to perform effectively. Because of environmental, political and geographical constraints, capacity cannot be easily increased by building new airports or runways. Therefore there is a need to develop effective decision support tools that provide useful help to controllers.

The decision problem that air traffic controllers face repeatedly over time is: which aircraft should land next and when should it land? As some airports have several runways, it is possible that more than one runway is available for landings. In that case, air traffic controllers face an additional decision: which runway should be allocated to the next incoming aircraft?

In the single runway situation the landing sequence adopted for incoming aircraft is often decided in a first-come first-served manner. The first aircraft that enters the radar range must land first, the second aircraft that enters lands second and so on. When multiple runways are present, the first-come first-served manner is also often used such that aircraft land on their allocated runway, in the same order they appeared in the radar range. The scheduling of an appropriate landing time for each aircraft is constrained by its characteristics. When an aircraft enters the radar range, a target (preferred) landing time is defined as the time the aircraft could land if it flies straight to the runway at its cruise (most economical) speed. This target landing time is bounded by an earliest landing time and a latest landing time. The earliest landing time is determined as the time the aircraft could land if it flies straight to the runway at its fastest speed with no holding. The latest landing time is determined as the time the aircraft could land if it is held for its maximal allowable time before landing. The time that elapses between an aircraft's earliest time and latest time is called its time window. Hence, scheduled landing times as decided by air traffic controllers must lie within aircraft time windows.

The flow of incoming aircraft is not homogeneous, it contains different aircraft types. All aircraft in flight create wake vortices at the rear of the aircraft. These vortices have a chaotic evolution and can cause serious turbulence to a closely

following aircraft, even to the extent of a crash [30]. In order to maintain an aircraft's aerodynamic stability, separation distances based on the preceding aircraft types must be respected during landing. In our research separation distances are dealt with by converting them into separation times using a fixed landing speed depending on the aircraft type. These separation times are the main limiting factors on runway usage and these constraints mainly apply for aircraft landing on the same runway. If several runways are available for landing the application of such constraints for aircraft landing on different runways depends upon the relative positions of the runways.

Aside from the above constraints the problem also includes some objective relating to making effective use of fixed runway capacity. In this paper we consider the static (or off-line) version of the problem, where the set of aircraft that are waiting to land is known. This contrasts with the dynamic (or on-line) case where decisions about each aircraft must be made as time passes, aircraft land and new aircraft appear.

In this research, we want to consider other landing sequences than the common first-come first-served sequence. Because this implies searching through a wide solution space, usual exact solution techniques would require too much time regarding the considered problem to get to the solution. Population based heuristics mixed with problem specific knowledge are techniques able to produce solutions of good quality in a reasonable amount of time when the solution space is very large. Scatter Search is a well-known such technique and has already been applied with success to various scheduling problems. Therefore we are interested in observing how it performs for the Aircraft Landing Problem. We also apply the Bionomic Algorithm, which is less well-known than Scatter Search and uses more structured procedures, to compare with Scatter Search results.

Section 2 presents a formulation of the Aircraft Landing Problem as a mixed-integer zero-one program. Section 3 reviews some recent approaches for the Aircraft Landing Problem and discusses previous work involving population based heuristics to address this problem. The rationale of population heuristics is explained in Section 4 along with an overview of Scatter Search and the Bio-

conomic Algorithm. In Section 5 we present algorithmic details for these two heuristics. Results for both algorithms when applied to problems involving up to 500 aircraft and five runways are presented and discussed in Section 6. Conclusions are given in Section 7.

2. Formulation

This section presents a mixed-integer zero-one formulation of the static multiple runway Aircraft Landing Problem that is based on the formulations presented in [6–8].

2.1. Notation and decision variables

We define

- P the number of aircraft waiting to land,
- R the number of runways available for landing,
- E_i, T_i, L_i the earliest, target, latest landing time (resp.) for aircraft $i, i \in \{1 \dots P\}$,
- $S_{ij} \geq 0$ the separation time between aircraft i and j where i lands before j on the same runway, $(i, j) \in \{1 \dots P\}^2, i \neq j$,
- $s_{ij} \geq 0$ the separation time between aircraft i and j where i lands before j on a different runway, $(i, j) \in \{1 \dots P\}^2, i \neq j$.

As we deal with the static case, the number of aircraft P is fixed and known beforehand together with all information about each aircraft. In this notation, separation times S_{ij} and s_{ij} are not aircraft type dependent, they are uniquely defined for each pair of aircraft. Practically, this enables other possible time requirements depending on flight characteristics and approach corridors used to be taken into account in addition to the aircraft type based separation time. In our formulation below, it is assumed that the separation times s_{ij} that apply when aircraft i and j land on different runways, are independent of the runways' configuration.

We use the decision variables

$$z_{ir} = \begin{cases} 1 & \text{if aircraft } i \text{ lands on runway } r, \\ & i \in \{1 \dots P\}, r \in \{1 \dots R\}, \\ 0 & \text{otherwise.} \end{cases}$$

$$\gamma_{ij} = \begin{cases} 1 & \text{if aircraft } i \text{ and } j \text{ land on the same} \\ & \text{runway, } (i, j) \in \{1 \dots P\}^2, i \neq j, \\ 0 & \text{otherwise.} \end{cases}$$

$$\delta_{ij} = \begin{cases} 1 & \text{if aircraft } i \text{ lands before aircraft } j, \\ & (i, j) \in \{1 \dots P\}^2, i \neq j, \\ 0 & \text{otherwise.} \end{cases}$$

$x_i \geq 0$ the scheduled landing time for aircraft $i, i \in \{1 \dots P\}$.

2.2. Constraints

2.2.1. Time window constraints

For each aircraft waiting to land, its scheduled landing time must lie within its time window. The formulation is straightforward:

$$E_i \leq x_i \leq L_i \quad \forall i \in \{1 \dots P\}. \tag{1}$$

Actually, a more algorithmically convenient formulation is used in this paper that makes use of a proportion of the time window. We define y_i the proportion of the time window that elapses before aircraft i lands, $i \in \{1 \dots P\}$. This proportion is related to the corresponding scheduled landing time via:

$$x_i = E_i + y_i(L_i - E_i). \tag{2}$$

The time window constraints are now formulated as:

$$0 \leq y_i \leq 1 \quad \forall i \in \{1 \dots P\}. \tag{3}$$

Hence, if all the y_i are always between zero and one, the time window constraints are automatically satisfied.

2.2.2. Separation time constraints

Separation time constraints are conditioned by the landing order of aircraft and their assigned runway. Considering the landing order, either aircraft i or j must land first:

$$\delta_{ij} + \delta_{ji} = 1 \quad \forall (i, j) \in \{1 \dots P\}^2, i \neq j. \tag{4}$$

Considering the runway allocation, if aircraft i and j land on the same runway, so too must aircraft j and i :

$$\gamma_{ij} = \gamma_{ji} \quad \forall (i, j) \in \{1 \cdots P\}^2, \quad i \neq j. \quad (5)$$

Then separation constraints must ensure that, when aircraft i lands first, aircraft j lands S_{ij} time units after i if they are assigned to the same runway or s_{ij} time units after i otherwise:

$$x_j \geq x_i + S_{ij}\gamma_{ij} + s_{ij}(1 - \gamma_{ij}) - M\delta_{ji} \\ \forall (i, j) \in \{1 \cdots P\}^2, \quad i \neq j, \quad M \gg 0, \quad (6)$$

where the role of M is to ensure that the equation is redundant if j lands before i ($\delta_{ji} = 1$).

2.2.3. Multiple runway constraints

Constraints relating to runway allocation must be considered. First, each aircraft is assigned one and only one runway:

$$\sum_{r=1}^R z_{ir} = 1 \quad \forall i \in \{1 \cdots P\}. \quad (7)$$

Second, when aircraft i and j are assigned to land on the same runway, some constraints must ensure that the runways assigned to aircraft i and j are identical:

$$\gamma_{ij} \geq z_{ir} + z_{jr} - 1 \quad \forall (i, j) \in \{1 \cdots P\}^2, \\ i < j, \quad r \in \{1 \cdots R\}. \quad (8)$$

2.3. Objective

To complete the formulation, it remains to define the objective function. In air traffic management, deviation from target times is a key factor [8]. It is also useful to have the ability to change the objective having regard to air traffic density or weather conditions, for example. Due to these requirements, we consider in this paper two different objectives, both based on deviations from target times.

2.3.1. Non-linear objective

This objective uses a non-linear function and is based on the difference between the scheduled landing time and the target time. We define $d_i = x_i - T_i$ the deviation from the target time for aircraft i , $i \in \{1 \cdots P\}$.

If this deviation is positive, then the aircraft is landing after its target time. This is not an ideal situation and so is penalised, the corresponding con-

tribution to the objective function for this aircraft is arbitrarily set to $-d_i^2$. On the other hand, if the deviation is negative, then the aircraft is landing before its target landing time which is preferred, so the corresponding contribution is set to $+d_i^2$. The objective is to maximise the overall aircraft contribution:

$$\text{maximise } \sum_{i=1}^P D_i, \quad D_i = \begin{cases} -d_i^2 & \text{if } d_i \geq 0, \\ +d_i^2 & \text{otherwise.} \end{cases} \quad (9)$$

The complete formulation with the non-linear objective is to maximise (9) subject to (1), (4)–(8). This formulation of the static multiple runway Aircraft Landing Problem is a non-linear mixed-integer zero–one program whose objective is to make all aircraft land as early as possible. This objective is more likely to be applied when a high traffic density occurs and efficient and effective use of runway capacity must be made.

The nature of this objective is such that, for a given sequence of aircraft, each aircraft should be as close to the preceding aircraft as can be achieved within the time window and separation constraints. We chose to name this property the *close-up* property. As will become apparent below, the advantage of an objective having this property is that, given a landing sequence, it is possible to compute optimal landing times in a polynomially bounded manner.

2.3.2. Linear objective

This objective uses a cost for each aircraft linearly dependent on deviation from target time. So it is a linear (or more accurately linearisable) objective. Fig. 1 presents this variation in cost within the time window of an aircraft i . At target time, the cost is zero. If landing occurs before the target time, the corresponding cost of deviation is deduced

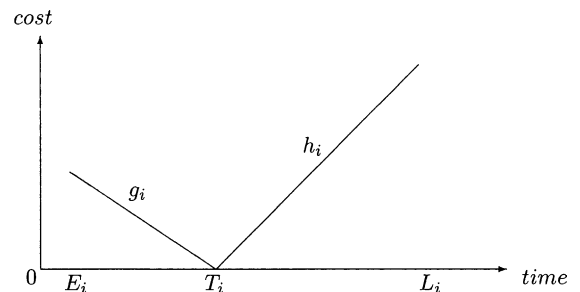


Fig. 1. Variation in cost for an aircraft within its time window.

from slope g_i and similarly, if landing occurs after the target time, the cost is deduced from slope h_i .

The cost presented here (and indeed also for the non-linear objective above) is not the overall cost involving in landing an aircraft but rather the additional cost that occurs when an aircraft is not landing at its target time. We define

$$\alpha_i = \max(0, T_i - x_i) \text{ the deviation before target time for aircraft } i, i \in \{1 \dots P\},$$

$$\beta_i = \max(0, x_i - T_i) \text{ the deviation after target time for aircraft } i, i \in \{1 \dots P\}.$$

The cost corresponding to landing aircraft i is then expressed as $\alpha_i g_i + \beta_i h_i$. If aircraft i lands on target, both α_i and β_i are equal to zero and then the cost assigned to its landing is zero. When aircraft i does not land on target, either α_i or β_i is non-zero and so there is a strictly positive cost incurred by this landing. The objective is to minimise the overall cost:

$$\text{minimise } \sum_{i=1}^P (\alpha_i g_i + \beta_i h_i). \tag{10}$$

Additional constraints are introduced in order to link α_i and β_i to the decision variable x_i :

$$x_i = T_i - \alpha_i + \beta_i \quad \forall i \in \{1 \dots P\}, \tag{11}$$

$$0 \leq \alpha_i \leq T_i - E_i \quad \forall i \in \{1 \dots P\}, \tag{12}$$

$$\alpha_i \geq T_i - x_i \quad \forall i \in \{1 \dots P\}, \tag{13}$$

$$0 \leq \beta_i \leq L_i - T_i \quad \forall i \in \{1 \dots P\}, \tag{14}$$

$$\beta_i \geq x_i - T_i \quad \forall i \in \{1 \dots P\}. \tag{15}$$

The complete formulation with the linear objective is to minimise (10) subject to (1), (4)–(8), (11)–(15). This formulation of the static multiple runway Aircraft Landing Problem is a linear mixed-integer zero-one program, with $O(P^2)$ variables and $O(P^2)$ constraints, where the objective is to make all aircraft land as close as possible to their target time. This objective could be applied when a reasonable traffic density holds.

2.4. Extensions

There are a number of extensions to the above formulation of the Aircraft Landing Problem that

are worth mentioning. All of these extensions relate to introducing runway dependence into the problem.

2.4.1. Runway dependent earliest, target and latest times

In the above formulation we have assumed that the earliest, target and latest times are independent of the runway chosen for landing. In practice this may not be so, in particular, aircraft close to the airport may well have significantly different earliest times for different runways. Extending our formulation to incorporate runway dependence in relation to these times is easily done. We define E_i^r, T_i^r, L_i^r the earliest, target, latest time (resp.) for aircraft $i, i \in \{1 \dots P\}$, on runway $r, r \in \{1 \dots R\}$.

Then the following constraints are introduced to ensure that the earliest, target, latest times previously used are correctly associated with the appropriate earliest, target, latest time corresponding to the runway used:

$$E_i = \sum_{r=1}^R E_i^r z_{ir}, \quad T_i = \sum_{r=1}^R T_i^r z_{ir}, \quad L_i = \sum_{r=1}^R L_i^r z_{ir}$$

$$\forall i \in \{1 \dots P\}. \tag{16}$$

2.4.2. Runway dependent separation times

In the above formulation we have assumed that the separation time S_{ij} for two aircraft i and j landing on the same runway is also independent of the runway chosen. In practice this may not be so, because for example, noise abatement procedures may require a different separation time between i and j if they land on one runway than if they land on another one. Extending our formulation to incorporate runway dependence in relation to separation times is easily done. We define $S_{ij}^r \geq 0$ the separation time between aircraft i and j where i lands before j on runway $r, (i, j) \in \{1 \dots P\}^2, i \neq j, r \in \{1 \dots R\}$.

Then the following constraint is introduced to ensure that the separation times previously used are correctly associated with the appropriate separation times corresponding to the runway used:

$$S_{ij} = \sum_{r=1}^R S_{ij}^r z_{ir} z_{jr} \quad \forall (i, j) \in \{1 \cdots P\}^2, i \neq j. \quad (17)$$

While Eq. (17) is non-linear, it can be linearised to ensure that the formulation of the Aircraft Landing Problem including runway dependent separation times remains linear. In brief this is done by introducing a variable f_{ij}^r and replacing Eq. (17) by the constraints $f_{ij}^r \geq 0$, $f_{ij}^r \geq S_{ij}^r(z_{ir} + z_{jr} - 1)$ and $S_{ij} = \sum_{r=1}^R f_{ij}^r$. With the addition of Eq. (17), the S_{ij}/ij term in Eq. (6) becomes non-linear. In order to maintain Eq. (6) linear, this term can be replaced by S_{ij} , since from Eq. (17), S_{ij} will be zero if aircraft i and j land on different runways.

2.4.3. Runway restrictions

It can happen that certain aircraft cannot make use of certain runways because, for example, a particular runway is too short for the considered aircraft to land on. This is easily dealt with in our formulation by setting $z_{ir} = 0$ if aircraft i cannot use runway r .

Amending the heuristic algorithms presented in this paper to incorporate all of the above extensions is relatively trivial. For reasons of space, we shall only briefly indicate the major changes that are needed to our heuristics to incorporate these extensions.

3. Previous literature

The Aircraft Landing Problem formulation presented in the previous section is largely based on the work presented in [6]. After relaxing binary variables and strengthening the formulation with additional constraints, the problem is solved optimally in that paper with a linear programming based tree search algorithm. Computational results are presented for instances from OR-Library [4] involving up to 50 aircraft and four runways. A detailed review of published work addressing the Aircraft Landing Problem can be found in [6] and so in this paper we mainly consider work additional to that considered there, or work that makes use of a population based heuristic approach.

A heuristic algorithm is presented in [27] based on the segmentation of time. The time horizon is

divided into time segments that determine sub-problems of the Aircraft Landing Problem. Each subproblem is formulated as a mixed-integer zero-one linear problem as in [6] and solved optimally in turn. Computational results are presented for instances from OR-Library [4] and for randomly generated instances involving up to 75 aircraft and four runways.

The work presented in [18] is a comparison of several type of algorithms for solving the Aircraft Landing Problem. Amongst these are a mixed-integer zero-one linear formulation, an integer linear formulation using time discretisation and two heuristic local search approaches. Computational results are presented for the single runway case and for three types of instances (including some from OR-Library [4]) involving up to 112 aircraft.

Theoretical results are presented in [2] about policies to route aircraft to two runways using a queuing system where the arrival process is modelled as a superposition of independent Poisson processes and the separation times are regarded as dependent service times. In [3] the same authors examine the implications of the aircraft queuing process for airport capacity.

In [9] the multiple runway Aircraft Landing Problem with an objective having the close-up property is addressed using four genetic schemes. Three of these schemes use a genetic algorithm approach while the other scheme uses a genetic programming approach. In one of the genetic algorithm schemes the representation adopted relates to the order in which aircraft land and the runway they land on, while in the other two genetic algorithm schemes the representation adopted relates to the order in which aircraft land (the allocated runway being decided using a heuristic procedure). Computational results are presented for one instance involving 12 aircraft and three runways. A recent paper [23] builds upon the work presented in [9] and computational results are presented there for four instances involving up to 20 aircraft and five runways.

In [17] the multiple runway Aircraft Landing Problem is addressed using a heuristic approach and an exact approach. The heuristic approach involves a genetic algorithm to search a perturbation space. The exact approach is implemented as a

branch and bound algorithm. Pre-processing steps involving time window tightening and partial ordering based on problem data or an upper bound are used. Computational results are presented for instances from OR-Library [4] involving up to 50 aircraft and four runways.

A genetic algorithm and a branch and bound algorithm are presented in [1] for the Aircraft Landing Problem. For the genetic algorithm, possible solutions are represented by individuals whose genes are the landing times of each aircraft. Results are presented for the single runway case for instances randomly generated using an air traffic simulator involving up to 20 aircraft.

In [14,15] a genetic algorithm is presented for the Aircraft Landing Problem. In this algorithm, landing times are allocated by specifying a 30 seconds time slot. The representation adopted encodes for this time slot with infeasible solutions being penalised. The approach is limited to at most two runways since runway allocation uses a zero-one representation. Computational results are presented for two data sets involving 28 and 29 aircraft and two runways. The approach is also applied to the dynamic case of the problem.

In [8] a population heuristic is presented for the single runway Aircraft Landing Problem where the objective function involves deviations from target time. The objective function used is a squared function of deviations. Individuals generated by this heuristic are assigned an unfitness value to quantify constraint violation if an individual corresponds to an infeasible solution. Results are presented for a problem instance based on observations during a busy period at London Heathrow airport. In [7] the algorithm developed in [8] is incorporated into approaches for solving the dynamic case of the problem.

4. Population heuristics

4.1. Overview

Population heuristics are based on the principles of selection and mutation, the main concepts of Darwin's theory of evolution. They mirror evolution in performing manipulations on individuals

which represent possible solutions to the considered problem [5]. Each individual is encoded using a set of chromosomes that define the problem's variables. The fitness of an individual is evaluated with respect to the quality of the solution it represents. An initial population of individuals is generated and operators that model genetic selection, mating and other processes are defined and applied to the population individuals. The standard and most widely known population heuristics are genetic algorithms, whose general framework is presented below:

generate an initial population

repeat

select individuals from the population to be parents

create new individuals as combinations of selected parents

optionally mutate the children

select the children to insert into the population

select the individuals to remove from the population

until termination, whereupon report the best solution encountered

The difference between standard genetic algorithms and other population heuristics relates to the strategies used to implement the steps of this framework. Operators involved in genetic algorithms are mainly random procedures that could be applied to any type of problem. Other population heuristics, such as Scatter Search and the Bionic Algorithm used in this paper, involve deterministic procedures that can include problem specific knowledge. As there exists a large number of possible approaches to the steps involved in a population heuristic, a major consideration is to choose appropriate approaches that will achieve good algorithmic performance.

In particular, it is important to achieve a good balance between procedures that intensify the solution search and procedures that diversify this search.

Finally, as by their very nature population heuristics create new solutions as combinations of previous solutions, individuals that violate some problem constraints can be generated. These

individuals map to infeasible solutions in terms of the problem considered and such infeasible solutions must be appropriately dealt with. Standard ways to handle infeasible solutions are described in [5].

4.2. Scatter search

By contrast to genetic algorithms driven by randomisation, Scatter Search relies on deterministic processes influenced by the problem's context and having special properties for exploiting combinatorial optimisation problems, such as the Aircraft Landing Problem. The specific features of Scatter Search are: individuals are not limited to binary representation; selected parents are not restricted to two; parent mating is structured as a linear combination; a local improvement procedure is applied to all individuals.

A general description of Scatter Search and Path Relinking, its generalised form, is given in [21]. It synthesises the underlying principles to these two approaches and describes many designs to implement Scatter Search and Path Relinking strategies. Each of the articles [19,20,22] emphasises a specific feature of the Scatter Search process and how it can be adapted to different optimisation problems. A general framework for Scatter Search is

generate an initial population called the reference set

improve each individual in the reference set

repeat

select a subset of the reference set

create a new individual as linear combination of the subset

improve the new individual

update the reference set

until termination, whereupon report the best solution encountered

The Scatter Search process has been adapted and applied to many optimisation problems. In relation to this paper, recent Scatter Search applications for scheduling problems, especially flow-shop scheduling and job-shop scheduling, can be found in [26,28].

4.3. The Bionomic Algorithm

The Bionomic Algorithm is less well-known than Scatter Search and was first presented in 1994 [10]. As with Scatter Search, its underlying strategy involves creating new solutions as linear combinations of old solutions and procedures involved in this process are designed to use problem specific knowledge. The specific features of a Bionomic Algorithm are: a maturation step to improve individuals; structured construction of parent sets based on a graph which represents the population structure; parent selection based on fitness and distance between individuals; a generational approach to replace the population. A general framework for the Bionomic Algorithm is

generate an initial population

improve each individual in the initial population

repeat

build a graph that represents the population structure

compute parent sets from this graph

create new individuals as linear combinations for each parent set

improve each new individual

update the population with some of the best new individuals

until termination, whereupon report the best solution encountered

As far as we are aware, the Bionomic Algorithm has not been applied to scheduling problems before. Applications in other areas can be found in [11,12,16,29].

5. Algorithmic details

Elements involved in the implementation of our heuristics for the Aircraft Landing Problem are

- representation of an individual,
- evaluation of an individual,
- generation of the initial population,
- selection of parents,
- generation of children,
- duplication test,

- local improvement of an individual,
- population replacement.

Many of these elements are common to both heuristics either because they correspond to basic features of population heuristics or they are directly related to the Aircraft Landing Problem. The major element that differs relates to the selection of parents. The representation adopted and the evaluation of an individual draw upon previous work presented in [7,8]. The use of Scatter Search and the Bionomic Algorithm and the local improvement procedures have not been presented previously in the literature however.

Previous work in relation to the Aircraft Landing Problem in the literature has considered runways to be sufficiently well spaced so as to enable s_{ij} , the separation required between aircraft landing on different runways, to be zero. Therefore in this paper and for ease of presentation, we shall henceforth also take s_{ij} to be zero. Amending the heuristics presented to deal with the case where s_{ij} is non-zero is easily done however.

5.1. Individual representation

As the key element of population heuristics are individuals, the first step towards an implementation of such a heuristic is to define an individual representation adapted to the considered problem.

An individual represents a possible solution to the problem. For the multiple runway Aircraft Landing Problem, an individual must provide information about the scheduled landing time and the allocated runway for each aircraft waiting to land. Thus the representation we define contains two variables for each aircraft: the proportion of time that elapses before the scheduled landing time and the runway allocated.

Fig. 2 gives a representation of an individual with P aircraft where y_i is the proportion of time as defined in Section 2.2 and r_i is the runway allocated to aircraft i .

5.2. Individual evaluation

The evaluation of an individual is used to address its performance regarding the objective

aircraft	1	2	...	P
proportion	y_1	y_2	...	y_P
runway	r_1	r_2	...	r_P

Fig. 2. Individual representation for P aircraft.

under consideration and to deal with the constraints. Two values are introduced here to capture these two issues: a fitness value and an unfitness value. This use of two distinct values to evaluate a solution in a population heuristic context was first introduced in [13] in conjunction with an appropriate population replacement scheme.

The fitness value is defined as the value of the objective function used, it quantifies the performance of the individual. If the objective function is non-linear, the fitness function is given by Eq. (9), in which case the higher the fitness, the better the solution. If the objective function is linear, the fitness function is given by Eq. (10), in which case the lower the fitness, the better the solution.

The unfitness value measures the violation of the separation time constraints. Time window and runway allocation constraints (see Section 2.2) are automatically satisfied by our representation. Having regard to Eq. (6) and recalling that we are taking s_{ij} to be zero, a convenient measure of constraint violation (or unfitness) is given by

$$\sum_{i=1}^P \sum_{\substack{j=1 \\ j \neq i, x_j \geq x_i, r_i=r_j}}^P \max(0, S_{ij} - (x_j - x_i)). \tag{18}$$

Considering Eq. (18), for two aircraft i and j where j lands after ($x_j \geq x_i$) and on the same runway ($r_i = r_j$) then we need the difference ($x_j - x_i$) in the landing times to be at least S_{ij} . Hence the term $\max(0, S_{ij} - (x_j - x_i))$ will always be zero if separation is satisfied, strictly positive otherwise. The higher unfitness, the more infeasible the solution.

Both fitness and unfitness values are used to compare individuals. An individual with a zero unfitness value is better than any individual with a positive unfitness. Between two individuals with a zero unfitness value (so both are feasible), the best one is the one with the best fitness value corresponding to the objective used.

5.3. Initial population

Some initial individuals must be created to form the initial population of the process.

The majority of the initial individuals are randomly generated. To create a random individual, we randomly generate two values for each aircraft: a proportion value between 0 and 1 and a runway number between 1 and R .

Seeding the initial population with fit individuals that have been heuristically generated is a common practice in population heuristics as it can help the search to find good near-optimal solutions. However too many highly fit initial individuals may provoke an early convergence of the population into areas of the solution space containing only suboptimal solutions.

For the Aircraft Landing Problem, we generate three heuristic individuals, based on the earliest, target or latest times. Aircraft are first ordered with respect to the times being considered. Each aircraft in this sequence is considered in turn in order to be allocated to a runway. If there exists a runway with no aircraft then the aircraft is allocated to this empty runway. Its scheduled landing time is the time used to generate the ordered sequence. Otherwise the aircraft is assigned to the runway that enables it to land as early as possible while satisfying the separation time constraints with all the aircraft already scheduled on this runway. The time proportion values are computed from the scheduled times once all aircraft have been considered. Three individuals can be generated using this heuristic as three sequences can be defined using earliest, target or latest times. Note that, as with randomly generated individuals, individuals generated in this way are not guaranteed to be feasible.

The size of the population is set to 100 individuals, so the initial population contains three heuristic individuals, with the remainder being randomly generated. If the earliest, target or latest times are runway dependent then the above procedure can create $3R$ heuristic individuals (based on ordering with respect to earliest/latest/target for each of the R runways). Hence in this case the initial population contains $3R$ heuristic individuals, with the remainder being randomly generated.

5.4. Parent selection for Scatter Search

Parent selection in our Scatter Search algorithm is a binary tournament selection scheme based on individual fitnesses. In binary tournament selection two individuals are selected at random from the population and the one with the best fitness is kept as a parent. Each execution of this selection scheme provides one individual to play the role of parent. Unlike standard genetic algorithms, the number of parents in Scatter Search can be more than two. For our implementation of Scatter Search, we set the number of parents to three. The binary tournament selection scheme is repeated three times to select three individuals. At the end of the parent selection stage for Scatter Search, we have a single parent set containing three individuals.

5.5. Parent selection for the Bionomic Algorithm

Parent selection for the Bionomic Algorithm is a structured procedure where the underlying principle is to give more opportunities to individuals with a better fitness to be selected as parents and prevent individuals too close to each other from being selected as parents together.

Population structure is captured by an adjacency graph. Each individual of the current population is represented by a node. Nodes have an inclusion frequency value that is set to the rank (ties broken arbitrarily) of the corresponding individual when population individuals are sorted by fitness. Fitter individuals are represented by a node with a higher inclusion value and thus can appear more often in a parent set. For the non-linear objective, individuals are sorted into ascending fitness order (since this objective is a maximisation one, cf Eq. (9)). For the linear objective, individuals are sorted into descending fitness order (since this objective is a minimisation one, cf Eq. (10)). We define

N the population size and also the number of nodes in the adjacency graph,
 F_n the rank of individual n and the inclusion frequency of node n , $n \in \{1 \dots N\}$, $F_n \in \{1 \dots N\}$.
 In our implementation, we use a population of size $N = 100$, as commented in Section 5.3.

A distance measure is introduced to quantify, in terms of solution structure, how close two individuals are to each other. This distance measure is used to determine if an edge must exist between the two corresponding nodes in the adjacency graph. In terms of our implementation of the Bionomic Algorithm for the Aircraft Landing Problem, we define

- d_{ij} the total distance evaluation between individual i and j ,
- d_{ij}^k the distance evaluation corresponding to aircraft k between individual i and j ,
- y_i^k the proportion value associated with aircraft k in individual i ,
- r_i^k the runway number associated with aircraft k in individual i .

We define

$$d_{ij} = \sum_{k=1}^P d_{ij}^k \quad \text{with } d_{ij}^k = \begin{cases} 1 & \text{if } r_i^k \neq r_j^k, \\ |y_i^k - y_j^k| & \text{otherwise,} \end{cases}$$

$$\forall (i, j) \in \{1 \dots N\}^2, \quad i \neq j. \quad (19)$$

Here the contribution of any particular aircraft to the distance between two individuals is one if the aircraft land on different runways in the two individuals and the difference in proportions if the aircraft land on the same runway in the two individuals. Informally the higher the distance value the more different the individuals are.

In addition to these distances, a distance threshold Θ is introduced to assess whether two individuals are distant enough or not. The distance for all pairs of individuals are compared against this threshold. If $d_{ij} < \Theta$, individuals i and j are considered as close to each other and thus an edge must exist between their corresponding nodes in the adjacency graph. In our implementation Θ was first set to $P/10$. If the adjacency graph computed with this value contains more than half the maximum number of edges in a N node undirected graph ($N(N - 1)/2$ edges), Θ is halved and a new graph is computed (this process of halving Θ continuing until the graph has an appropriate number of edges). This process of halving Θ was introduced because preliminary computational re-

sults indicated that the more edges in the adjacency graph the less numerous and various were the parent sets generated (as outlined below) from the adjacency graph. In such a situation producing new (non-duplicate) individuals was difficult.

In the Bionomic Algorithm, the adjacency graph yields parent sets through the computation of a maximal independent set. A maximal independent set is a set of maximal cardinality where none of the selected nodes are linked by an edge. The logic here is that constructing parent sets in this way means that individuals selected to play a role of parent are reasonably spanned over the problem search space, since an edge exists between two individuals i and j if and only if they have structurally similar solutions ($d_{ij} < \Theta$).

To illustrate the concept of a maximally independent set a possible adjacency graph for six nodes is presented in Fig. 3. The maximal independent sets corresponding to this graph are $\{2, 3, 4, 6\}$, $\{2, 3, 5\}$ and $\{1, 3, 4\}$. In the case of parent set selection, it is sufficient to generate only one of the possible maximal independent sets. Note in particular that there is no requirement to generate the maximal independent set that contains the most nodes.

The process of computing a maximal independent set is described below. The adjacency graph involved in this process is a copy of the adjacency graph constructed as described above.

repeat

- select one of the nodes at random
- insert this node into the parent set

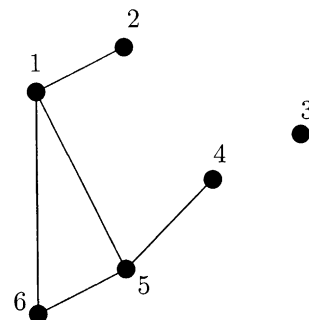


Fig. 3. Example graph to illustrate maximal independent sets.

remove the selected node and all its neighbours from the graph
until the graph is empty

Once this process is executed, a new parent set P_S has been constructed. The nodes involved in P_S have their inclusion frequency value decreased by one (i.e. $F_n = F_n - 1 \quad \forall n \in P_S$). If its inclusion frequency value becomes zero, a node is removed from the main adjacency graph. The inclusion frequency value characterises the number of times an individual will be selected to be in a parent set. New parent sets are computed until the main graph is empty, that is all nodes have been selected as many times as the initial value of their inclusion frequency.

For the Bionomic Algorithm the parent selection stage generates many parent sets (at least N) of various sizes. Sets are computed such that fitter individuals are more often selected than less fit individuals (by means of the inclusion frequency mechanism) and such that individuals in a given parent set are reasonably spanned over the problem search space (by means of the adjacency graph and maximal independent set mechanism). Computationally the complexity associated with generating the adjacency graph is $O(PN^2)$ and the complexity associated with identification of all parent sets from the adjacency graph is (in the worse case) $O(N^3)$. Hence the overall worse case computational complexity here is $O(PN^2 + N^3)$. Since we use a fixed population size $N = 100$ we have a linear relationship between computation time and problem size P .

5.6. Child generation

The child generation procedure for both Scatter Search and the Bionomic Algorithm produces one child from each parent set, and this child may be feasible or infeasible. To create a new individual, a new set of proportion values must be computed and a new set of runway numbers must be determined. For each aircraft, the new proportion value is computed as a weighted linear combination of the corresponding parent proportion values. Random weights are used in the linear combination process in order to introduce diversity to the new

individual. The resulting solution may be infeasible. This does not have serious consequences to the quality of the population as the improvement procedure (see below) is applied to any new individual and, in any event, infeasible solutions can be dealt with using unfitness. Runway allocation is decided in an equally weighted probabilistic manner for each aircraft. Utilising y_j^k , the proportion value associated with aircraft k in individual i , our generalised child generation procedure is

let $p(1), p(2), \dots, p(K)$ be the parents in the parent set
 generate K numbers $w_i, i = 1, \dots, K$, at random between 0 and 1
 set $W = w_1 + \dots + w_K$ and $w_i = w_i/W$ for $i = 1, \dots, K$
 set $y_{\text{child}}^k = \sum_{i=1}^K w_i y_{p(i)}^k \quad \forall k \in \{1 \dots P\}$
repeat for each aircraft in turn
 select one of the K parents at random
 assign its runway number to the runway variable of the current aircraft in the child
until all aircraft have been considered

Scatter Search produces only one child at each generation, as the above procedure is executed once using the set of $K = 3$ parents chosen. By contrast, the Bionomic Algorithm produces a number of parent sets and the above procedure is executed once for each of these parent sets in turn, producing one child for each parent set.

5.7. Duplication test

In order to maintain a good level of diversity in the population new individuals are discarded if they duplicate an individual already present in the population. A new individual is checked for duplication against all the current individuals immediately after its creation.

For the single runway case, the duplication test simply checks whether the *landing sequence* of two individuals are identical, that is if all the aircraft land in exactly the same order. If so both individuals represent the same solution. Note here that our local improvement procedures (see below) are such that if the landing sequences are the same, the solutions after local improvement will

also be the same. Computationally it is clear that checking for duplication before embarking on local improvement is preferable to the contrary.

In the multiple runway case, as in our formulation runways are identical, an individual contains as many landing sequences as there are runways available for landing and consequently checking for duplication is not as straightforward as the single runway case. To illustrate the problem, Fig. 4 presents the landing sequences for two individuals in a two runway, five aircraft, case. The landing sequence on runway 1 for individual 1 is identical to that on runway 2 for individual 2 (and vice versa). It is clear that these two individuals are effectively identical, that is they represent the same solution.

To avoid the situation shown in Fig. 4 occurring, each generated individual has its runway allocation variables r_i (cf Fig. 2) changed in the following manner:

- extract the R landing sequences from the individual
- order these sequences by increasing order of the smallest aircraft number they contain
- assign the first sequence to runway 1, the second sequence to runway 2, ..., etc.
- update the runway allocations in the individual

For the example given in Fig. 4 this procedure would leave individual 1 as currently but would change the runway allocation in individual 2 so that it is identical to individual 1. Once this runway reallocation procedure has been done, duplicate checking can proceed in the same manner as outlined for the single runway case above. Computationally duplicate checking is of complexity $O(PN + RP \log(P))$.

If the problem contains runway dependency (as discussed in Section 2.4) then the above runway reallocation procedure is not used and the runway allocation variables are not changed at all before duplicate checking.

5.8. Individual local improvement

As discussed in Section 4.2 for Scatter Search and Section 4.3 for the Bionomic Algorithm, these heuristics call for a local improvement of each individual that is based on problem specific knowledge. The improvement procedure we define depends on the objective considered and is applied to each landing sequence of the individual considered as fixed. As will become apparent when the details are discussed below, the landing times obtained via this improvement are optimal if the landing sequence considered is feasible.

For the non-linear objective, as we have the close-up property, it is possible to compute optimal landing times with a simple polynomially bounded procedure for each landing sequence of the considered individual. This procedure is applied independently to each landing sequence:

```

let  $q(s)$  be the  $s$ 'th aircraft in the ordered landing
sequence on the runway being considered
set  $s = 1$  and  $x_{q(1)} = E_{q(1)}$ 
repeat
  set  $s = s + 1$ 
  set  $x_{q(s)} = \min[L_{q(s)}, \max[E_{q(s)}, x_{q(t)} + S_{q(t),q(s)}]$ 
     $t = 1, \dots, s - 1]$ 
until all aircraft in the ordered sequence have been
considered
    
```

If this procedure computes landing times that are feasible (satisfy the separation time constraints, Eq. (6)) then they are optimal for the given landing sequence considered. The computational complexity of this procedure is $O(P^2)$.

For the linear objective, the improvement procedure corresponds to solving a simple linear program (LP). This procedure is applied to fixed landing sequences, so (in terms of the mathematical formulation of the problem) all the binary variables are set to known values, based on the aircraft landing in sequence order on their allocated

Landing sequence - individual 1	Landing sequence - individual 2
Runway 1: first 1 then 2 then 4	Runway 1: first 3 then 5
Runway 2: first 3 then 5	Runway 2: first 1 then 2 then 4

Fig. 4. Landing sequences for two different individuals representing the same solution.

runways. The only variables that remain to be computed are the scheduled times for each aircraft. Thus the LP to solve for this local improvement involves making the aircraft land as close as possible to their target time given the landing sequences. More technically this LP involves minimise (10) subject to (1), (4)–(8), (11)–(15) but with all binary variables set; it has $O(P)$ variables and $O(P^2)$ constraints. This is modelled using ILOG Concert Technology [24] and solved using ILOG Cplex software [25].

However, given the landing sequences on each runway, the associated LP problem may be feasible or infeasible. If it is feasible, Cplex computes optimal landing times for the landing sequences based on the original objective (Eq. (10)). If the LP for any particular runway is infeasible, a new LP is modelled based on the same landing sequence but aiming to minimise the infeasibility of the solution. Assuming (for ease of discussion) that the LPs for all R runways are infeasible the new LP would be minimise (18) subject to (1), (4), (5), (7), (8), (11)–(15) but with all binary variables set. This is an LP since although the objective (Eq. (18)) is nonlinear it is easily linearisable. Cplex hence computes optimal landing times for each aircraft with respect to their landing order in the sequences such that these times minimise the separation time constraint violation on each runway. This LP is always feasible.

5.9. Population replacement

After the generation of a new (non-duplicate) child and its improvement, the child is inserted into the population. It is standard within population heuristics to keep the population size constant and hence one member of the population must be selected for removal. In our Scatter Search heuristic the individual that is chosen to be removed from the population is the current worst individual, that is the individual with the higher unfitness value or at same unfitness, the individual with the worst fitness regarding the objective used.

It can happen that the population member discarded is actually better than the new inserted child (i.e. the child is worse than any member of

the population). Whilst this might seem perverse, previous experience has indicated that this helps maintain population diversity and, in any event, the inserted child is likely to depart from the population at the next iteration.

The population replacement scheme for the Bionomic Algorithm as commonly presented in the literature [11,16,29] is a generational approach, that is many children are inserted into the population at the same stage. However our preliminary computational experience indicated that adopting such an approach for the Bionomic Algorithm for the Aircraft Landing Problem would not lead to good quality results. Specifically we observed that inserting many children at each generation led to a quick convergence of the population and produced low quality results.

Consequently for our Bionomic Algorithm we chose to insert a single child into the population, as in Scatter Search. Limited computational experience indicated that this prevents the population converging too quickly and increases solution quality. The child chosen is the best child generated over all the parent sets (the best child being the one with the smallest unfitness or at same unfitness, the one with the best fitness regarding the objective used). The removal of a population member is done in the same manner as in Scatter Search above.

5.10. Heuristic overview

In order to provide a complete overview of our Scatter Search (SS) and Bionomic Algorithm (BA) heuristics for the Aircraft Landing Problem we give below a framework for the steps used:

generate the initial population as in Section 5.3
improve and evaluate each individual in the population as in Sections 5.8 and 5.2

repeat

generate parent sets as in Section 5.4 for SS or as in Section 5.5 for the BA
generate children from the parent sets as in Section 5.6
eliminate any children that are duplicates of population members as in Section 5.7

improve and evaluate each (remaining, non-duplicate) child as in Sections 5.8 and 5.2

add the best child to the population as in Section 5.9

until termination, whereupon report the best solution encountered

Computationally the most time-consuming part of our heuristics relates to the improvement and evaluation of each non-duplicate child. The computational complexity per child of this part is $O(P^2)$ for the non-linear objective and for the linear objective involves the solution of an LP with $O(P)$ variables and $O(P^2)$ constraints. The principal computational difference between the two heuristics implemented is that for Scatter Search we have to generate just a single parent set which leads to just a single child to improve and evaluate at each iteration, whereas in the Bionomic Algorithm we have to generate a number of parent sets (complexity $O(PN^2 + N^3)$) and each of these parent sets in turn gives rise to a single child to be improved and evaluated.

6. Computational results

The Scatter Search and Bionomic Algorithm heuristics presented above were implemented in C++ on a 2GHz Pentium PC with 512MB of memory. Computational results are presented in this section for 13 instances, publicly available from OR-Library [4], involving from 10 to 500 aircraft. Varying the number of runways and the objective adopted for these 13 instances means we consider 101 problems, in total.

Results for each algorithm and each objective are presented in Tables 1–4. Problems of the size shown in Tables 2 and 4 are much larger than those considered previously by the majority of authors in the literature. Each heuristic was replicated (executed) ten times on each problem.

The notation used in these tables is

Z_{opt} : value of the optimal solution,

Z_{best} : value of the best-known solution (if the optimal solution is not known),

T : total execution time in seconds for the ten replications,

G_{best} : percentage gap associated with the best solution found over the ten replications,

G_{FCFS} : percentage gap associated with the first-come first-served solution.

Computation times associated with producing the first-come first-served solution are not given in detail in Tables 1–4, because they are comparatively small. Over all 101 test problems, the largest computation time associated with any first-come first-served solution was only 1.4 seconds.

For problems for which the optimal solution is not known the best-known solution value Z_{best} given is the best solution value we have observed during the course of our computational work. The percentage gap G_{best} associated with the best solution found over the ten replications is calculated with reference to Z_{opt} (or Z_{best} if Z_{opt} is not known) as $100|Z_{\text{opt}} - \text{best solution value}|/|Z_{\text{opt}}|$. One complication is that for some problems Z_{opt} (or Z_{best}) is zero in which case the appropriate percentage gap is taken to be zero if and only if the best solution found is also zero, otherwise it is undefined.

Each instance has been executed with up to five runways, or less if fewer runways were sufficient to land all aircraft optimally. For the non-linear objective this would mean that all aircraft could land at their earliest landing time. For the linear objective this would mean that all aircraft could land at their target time. So the range of values used for R can be different in certain cases depending upon the objective used.

6.1. Non-linear objective

Tables 1 and 2 present results for the non-linear objective for a set of instances involving from 10 to 50 aircraft and from 100 to 500 aircraft respectively. For comparison purposes, the termination criterion adopted is the same for both heuristics and is set, based on preliminary tests, to 50,000 children generated (per replication).

With respect to the termination criteria adopted we found that, over the 52 problems presented, obtaining the final solution (G_{best}) shown required

Table 1
Results for non-linear objective—small instances

P	R	Z_{best}	SS		BA		FCFS G_{FCFS}
			T	G_{best}	T	G_{best}	
10	1	4849	6	0	57	0	3.13
	2	5924	6	0	64	0	3.17
	3	6185	8	0	65	0	0.71
	4	6237	9	0	66	0	0
15	1	18,337	8	0	56	0	5.11
	2	19,948	8	0	47	0	0.39
	3	20,078	10	0	50	0	0
20	1	35,632	9	0.28	70	0.28	6.35
	2	38,524	10	0	45	0	0.37
	3	38,664	12	0	47	0	0
20	1	20,001	9	0.95	70	0.59	5.76
	2	22,888	9	0	47	0	1.08
	3	23,659	11	0	50	0	0.17
	4	23,955	12	0	49	0	0
	5	24,140	13	0	51	0	0
20	1	19,381	9	6.11	71	1.03	20.43
	2	26,021	11	0	47	0.06	0.78
	3	26,495	11	0	50	0.02	0.02
	4	26,699	12	0	52	0	0.01
	5	26,732	13	0	49	0	0
30	1	-2,847,013	15	0	68	0	0
	2	-8943	15	0	58	1.06	108.48
	3	0	14	0	58	0	0
44	1	-23,266	24	0	89	0	0
	2	644,749	21	0	91	0	3.00
	3	646,432	21	0	87	0	0
50	1	728,837	23	19.33	122	4.65	Infeasible
	2	797,116	22	0	99	0	0.02
	3	799,417	24	0	105	0	0
Average			13	0.9	65	0.3	5.7

over 40,000 children (per replication) for 11 problems with Scatter Search and 28 problems with the Bionomic Algorithm. It required less than 10,000 children for 40 problems with Scatter Search and 14 problems with the Bionomic Algorithm.

Computational tests with these problems with regard to seeding the initial population with heuristic individuals (see Section 5.3) and the quality of solution produced showed that no seeding produces worse results for all problems with Scatter Search; and worse results for 9 problems, but better results for 2 problems, with the Bionomic Algorithm. So the initial population used is seeded with

three heuristic individuals. For all 52 problems shown in those tables, the final solution provided by our Scatter Search and Bionomic Algorithm heuristics was always feasible and (due to this seeding) dominates the first-come first-served solution. The first-come first-served solution was infeasible for one problem.

Problems presented in Tables 1 and 2 have not yet been solved optimally. Hence, for these tables, G_{best} is calculated against the best-known solution value. Result quality is globally very high for these two tables. The overall gap for Scatter Search and the Bionomic Algorithm is 0.9% and 0.3% respec-

Table 2
Results for non-linear objective—large instances

P	R	Z_{best}	SS		BA		FCFS G_{FCFS}
			T	G_{best}	T	G_{best}	
100	1	10,926,459	66	14.57	177	11.34	17.26
	2	13,690,290	53	0.57	188	0.43	1.65
	3	14,037,508	50	0.01	186	0.09	0.49
	4	14,090,232	48	0	186	0	0
150	1	14,713,752	106	14.80	223	12.38	20.44
	2	19,829,588	116	0.11	261	0.08	0.64
	3	20,126,522	90	0.02	282	0	0.14
	4	20,136,384	84	0	283	0	0
200	1	21,827,542	157	15.19	323	13.29	16.10
	2	26,786,444	154	0	397	0.07	1.39
	3	27,409,004	163	0.01	362	0.05	0.25
	4	27,484,328	137	0.01	493	0.12	0.12
	5	27,506,007	120	0	411	0	0
250	1	27,619,476	202	15.22	416	13.32	15.81
	2	33,676,680	181	0.18	467	0.15	0.80
	3	34,330,656	229	0.03	442	0.05	0.27
	4	34,401,946	201	0	461	0	0.01
	5	34,405,915	179	0	517	0	0
500	1	45,677,264	498	17.52	1268	15.94	18.35
	2	61,557,052	688	0.01	1170	0.10	0.26
	3	63,641,468	725	0.03	1212	0.02	0.04
	4	63,886,852	674	0	1249	0	0
	5	63,891,976	687	0	1455	0	0
		Average	244	3.4	540	2.9	4.1

tively for small instances and 3.4% and 2.9% respectively for large instances. For 30 of the 52 problems, the Bionomic Algorithm produces a solution equal to that produced by Scatter Search, for 13 problems it produces a better solution and for 9 problems Scatter Search produces a better solution than the Bionomic Algorithm. Overall, the Bionomic Algorithm out-performs Scatter Search with respect to quality of solution when the non-linear objective is used.

For every problem, Scatter Search terminates faster than the Bionomic Algorithm. This difference lies in the parent selection scheme adopted. In Scatter Search, a single parent set is constructed and a single child is generated and improved at each iteration. By contrast, in the Bionomic Algorithm, many parent sets are constructed from the current population and so at each iteration many children are generated and must be improved.

The largest execution time in [Tables 1 and 2](#) is approximately 1450 seconds (Bionomic Algorithm for 500 aircraft and five runways). This corresponds to 0.3 seconds of computation per aircraft per replication which is very reasonable given the size of the problem.

6.2. Linear objective

[Tables 3 and 4](#) present results for the linear objective for a set of instances involving from 10 to 50 aircraft and from 100 to 500 aircraft respectively. These instances are the same as those considered in [Tables 1 and 2](#). For comparison purposes, the termination criterion adopted is the same for both heuristics and is set, based on preliminary tests, to 10,000 children generated.

With respect to the termination criteria adopted we found that, over the 49 problems presented,

Table 3
Results for linear objective—small instances

P	R	Z_{opt}	SS		BA		FCFS G_{FCFS}
			T	G_{best}	T	G_{best}	
10	1	700	4	0	60	0	0
	2	90	24	0	45	0	0
	3	0	39	0	34	0	0
15	1	1480	6	0	90	0	1.35
	2	210	45	0	49	0	9.52
	3	0	46	0	43	0	0
20	1	820	8	0	99	0	68.29
	2	60	48	0	58	0	116.67
	3	0	62	0	63	0	n/d
20	1	2520	8	0	95	0	0
	2	640	52	0	55	0	0
	3	130	46	0	57	0	0
	4	0	56	0	52	0	0
20	1	3100	9	0	100	0	74.84
	2	650	50	0	61	3.08	16.92
	3	170	54	0	43	0	5.88
	4	0	56	0	68	0	n/d
30	1	24,442	158	0	274	0	0
	2	554	70	0	101	3.61	59.21
	3	0	54	0	87	0	0
44	1	1550	195	0	79	0	0
	2	0	118	0	124	0	0
50	1	1950	42	52.05	287	36.15	Infeasible
	2	135	121	0	196	0	425.93
	3	0	139	0	181	0	n/d
Average			60	2.1	96	1.7	37.1

n/d means not defined.

obtaining the final solution (G_{best}) shown required over 5000 children (per replication) for 19 problems with Scatter Search and 37 problems with the Bionomic Algorithm. It required less than 1000 children for 20 problems with Scatter Search and 8 problems with the Bionomic Algorithm.

Computational tests with these problems with regard to seeding the initial population with heuristic individuals (see Section 5.3) and the quality of solution produced showed that no seeding produces worse results for 21 problems, but better results for 5 problems, with Scatter Search; and worse results for 29 problems, but a better result for 1 problem, with the Bionomic Algorithm. So the initial population used is seeded with three

heuristic individuals. For all 49 problems shown in those tables, the final solution provided by our Scatter Search and Bionomic Algorithm heuristics was always feasible and (due to this seeding) dominates the first-come first-served solution. The first-come first-served solution was infeasible for one problem, the same as with the non-linear objective.

Times given in Tables 3 and 4 are larger than those in Tables 1 and 2 for the non-linear objective. The primary reason for this lies in the child improvement scheme adopted. For the linear objective this scheme involves solving LPs (with $O(P)$ variables and $O(P^2)$ constraints), whereas for the non-linear objective this scheme only involves a polynomially bounded procedure of com-

Table 4
Results for linear objective—large instances

P	R	Z_{best}	SS		BA		FCFS G_{FCFS}
			T	G_{best}	T	G_{best}	
100	1	5611.70	119	30.06	554	14.51	30.27
	2	452.92	342	5.67	487	54.73	172.60
	3	75.75	390	0	466	87.46	342.81
	4	0	336	0	439	n/d	n/d
150	1	12329.31	227	44.96	925	33.90	63.50
	2	1288.73	608	7.87	845	25.95	103.47
	3	220.79	668	8.88	803	195.88	552.16
	4	34.22	647	16.74	788	292.40	3473.49
	5	0	607	0	762	n/d	n/d
200	1	12418.32	256	17.95	1417	16.67	20.94
	2	1540.84	959	9.19	1287	38.54	129.80
	3	280.82	1021	21.59	1203	290.09	764.25
	4	54.53	993	2.77	1168	474.47	3947.88
	5	0	956	0	1158	n/d	n/d
250	1	16209.78	381	22.15	2011	23.58	24.28
	2	1961.39	1266	18.80	1835	50.18	137.42
	3	290.04	1454	17.48	1710	198.01	903.97
	4	3.49	1445	271.63	1688	13216.91	70752.44
	5	0	1386	0	1662	n/d	n/d
500	1	44832.38	1237	3.24	5852	1.03	5.10
	2	5501.96	3836	3.72	5379	37.47	56.91
	3	1108.51	4560	1.98	5158	182.69	462.60
	4	188.46	4413	22.98	4977	1186.81	2027.94
	5	7.35	4421	0	4887	22308.44	52628.71
		Average	1355	22.0	1978	1936.5	6830.0

n/d means not defined.

plexity $O(P^2)$. Except for 5 of the smaller problem instances, Scatter Search is, as previously, faster than the Bionomic Algorithm. The largest execution time in Tables 3 and 4 is approximately 5850 seconds (Bionomic Algorithm for 500 aircraft and one runway). This corresponds to 1.2 seconds of computation per aircraft per replication which is reasonable given the size of the problem.

Problems presented in Table 3 were solved optimally in [6]. Hence, for this table, G_{best} is calculated against the optimal solution value. The overall gap for Scatter Search is 2.1% with all but one of the 25 instances being solved optimally. The Bionomic Algorithm does not perform as well on these instances, but still performs reasonably with an overall gap of 1.7%. Problems presented in Table 4 have not yet been solved optimally and G_{best} for this table is hence calculated against

the best-known solution value. For 22 of the 49 problems shown in Tables 3 and 4, the Bionomic Algorithm produces a solution equal to that produced by Scatter Search, for 5 problems it produces a better solution and for 22 problems Scatter Search produces a better solution than the Bionomic Algorithm. So Scatter Search outperforms the Bionomic Algorithm with respect to quality of solution when the linear objective is used.

It is clear from Table 4 though that as the number of runways increases the gap often tends to very high values, with this trend being much more drastic for the Bionomic Algorithm than Scatter Search. The results presented in this table for the Bionomic Algorithm with the linear objective stand in marked contrast to the results presented in Table 2 for the same algorithm and the same

large problems but with the non-linear objective. The first-come first-served solution performs very poorly, with high gap values, even for the smaller problems.

6.3. Averaging

The large gap values seen in Tables 3 and 4 are (in part) a consequence of the low Z_{opt} and Z_{best} values. As the reader will probably appreciate, it is standard practice within the literature to evaluate heuristic algorithm performance by calculating percentage deviation from optimal (or best-known) solutions, exactly as we have done in those tables. What is unusual about the problems presented in Tables 3 and 4 though is that we have a wide variation in solution values and in particular, have some very low values.

The issue can be simply illustrated by comparing performance for Scatter Search for two cases of the same problem in Table 4. For the problem with $P = 500$ and the case where $R = 1$, the best gap obtained by Scatter Search is 3.24%. Recalling the linear objective aims to minimise the overall cost, this gap indicates that this solution costs 3.24% more than the best-known solution whose value is 44832.38. Hence the *excess cost* incurred in this case is $3.24(44832.38)/100 = 1453$. For the same problem but with the case where $R = 4$, the best gap obtained by Scatter Search is 22.98%. By comparison with the previous solution, it appears at first sight to be much worse, by a factor of $22.98/3.24 = 7$ in fact. But for this case the value of the best-known solution, 188.46, is smaller and the excess cost incurred is $22.98(188.46)/100 = 43$ only. Comparing excess costs, the solution cost for $R = 4$ is $1453/43 = 34$ times smaller than the solution cost for $R = 1$. In other words, in excess cost terms, the solution for $R = 4$ is over one order of magnitude *smaller* than the solution for $R = 1$. Whereas, in percentage gap terms, the solution for $R = 4$ is approximately one order of magnitude *larger* than the solution for $R = 1$. Thus there is a difference of over two orders of magnitude when we compare solutions depending upon whether we use excess cost or percentage gap.

In order to overcome the issue of large variation of solution values, we propose to average results,

for each problem, over all values of R . Each problem has been solved for varying R values, but given a particular problem, the underlying set of aircraft, time windows, separation times and objective function coefficients are the same. In the light of this we believe that, in our particular context, an averaging process is legitimate.

We define, for each problem,

Z_R the value of the best-known or optimal solution when R runways are used,
 G_R the best percentage gap as previously defined when R runways are used,
 R_{max} the largest number of runways considered.

Then the new average value we use to gain (percentage) insight into the excess cost for each problem is $100(\sum_{R=1}^{R_{\text{max}}}(G_R/100)|Z_R|)/R_{\text{max}})/(\sum_{R=1}^{R_{\text{max}}}|Z_R|/R_{\text{max}})[= (\sum_{R=1}^{R_{\text{max}}}G_R|Z_R|)/(\sum_{R=1}^{R_{\text{max}}}|Z_R|)]$. The numerator is the average excess cost over all values of R considered for the problem and the denominator is the average solution value. This averaging approach provides better insight into the performance of both Scatter Search and the Bionomic Algorithm for the large problems with the linear objective than the simple averaging adopted previously.

Table 5 shows the gaps computed using this averaging approach for both heuristics and both objectives. Although solution values with the non-linear objective do not display the same variation as those of the linear objective, results for the non-linear objective are also computed using this averaging approach for comparison purposes. Values in Table 5 still show that Scatter Search and the Bionomic Algorithm perform better with the non-linear objective than with the linear objective, as observed previously.

6.4. Basic local search heuristic

In order to provide insight into the computational effectiveness of our Scatter Search and Bionomic Algorithm heuristics we implemented a basic local search heuristic for the single runway Aircraft Landing Problem. This heuristic examines repositioning single aircraft in the landing sequence, and swapping the positions of pairs of

Table 5
Averaged percentage gaps

P	Non-linear objective			Linear objective		
	SS	BA	FCFS	SS	BA	FCFS
10	0	0	1.65	0	0	0
15	0	0	1.74	0	0	2.37
20	0.09	0.09	2.13	0	0	71.59
20	0.17	0.10	1.26	0	0	0
20	0.94	0.18	3.33	0	0.51	62.25
30	0	0	0.34	0	0.08	1.31
44	0	0	1.53	0	0	0
50	6.06	1.46	0.01	48.68	33.81	27.58
Average	0.9	0.2	1.5	6.1	4.3	20.6
100	3.17	2.48	4.13	27.89	18.38	44.62
150	2.95	2.46	4.23	40.87	36.38	83.40
200	2.53	2.26	3.04	17.02	26.15	62.26
250	2.60	2.28	2.88	21.77	31.64	63.48
500	2.69	2.46	2.87	3.34	16.31	35.31
Average	2.8	2.4	3.4	22.2	25.8	57.8

aircraft in the landing sequence, in order to try and improve the solution. This repositioning of an aircraft and swapping of pairs of aircraft, is repeated until no improvement in the landing sequence can be found. Here we define an improvement as a less infeasible solution if the current sequence is infeasible or a better quality solution if the current sequence is feasible. Each landing sequence examined is improved using the local improvement scheme discussed in Section 5.8.

Table 6 presents the time consumption and percentage solution gap of this basic heuristic for both objectives across the 13 instances. These results were produced by applying our heuristic three times—each time using a different initial landing sequence found by ordering aircraft with respect to earliest, latest and target times.

Results provided by this heuristic are of high quality, but at the expense of computation time. Time varies from 1 second for 10 aircraft to more than 78 hours for 200 aircraft with the linear objective. Problems of 250 and 500 aircraft have not been executed with the linear objective because of this high time consumption. With the non-linear objective, time varies from effectively zero for 10 aircraft to more than 44 hours for 500 aircraft. Comparing the times given in Table 6 with those

Table 6
Results with the basic heuristic

P	Linear objective		Non-linear objective	
	T	G _{best}	T	G _{best}
10	1	0	0	0
15	2	0	0	0
20	6	0	0	0
20	7	0	0	0
20	9	0	0	0
30	7	0	0	0
44	20	0	0	0
50	965	10.51	4	0
100	5482	0	29	0
150	43,077	0	276	0
200	281,332	0	1113	0
250	–	–	2923	0
500	–	–	160,592	0

given in Tables 1–4, the time required by this heuristic exceeds that required by our heuristics when $P \geq 50$ for the linear objective and when $P \geq 150$ for the non-linear objective.

6.5. Runway dependent problems

None of the problems considered above involves any runway dependence, such as considered in Section 2.4. Problems relating to runway dependent

Table 7
Results for runway dependent problems

Problem	P	R	Z_{best}	SS		BA		[9] or [23]
				T	G_{best}	T	G_{best}	G_{best}
[9]	12	3	11.25	9	0	44	0	0
[23] problem 1	12	3	3.50	8	7.14	49	7.14	0
[23] problem 2	15	3	12.25	10	0	47	0	2.04
[23] problem 3	20	5	8.75	12	0	49	11.43	180.00
[23] problem 4	12	3	4.25	7	29.41	47	0	164.71
Average				9	7.3	47	3.7	69.4

For [23] problem 3 a detailed solution that is supposed to have an associated objective function value of 12.0 is given in [23], but the solution presented actually has a different objective function value.

times have been given previously in the literature in [9,23]. These papers contain five problems, all have runway dependent earliest times and one has runway restrictions. For these problems the target time T_i for each aircraft i is runway independent and set to $\min[E_i^r | r = 1 \cdots R]$. So the target time for each aircraft is the minimum earliest time over all runways. The objective adopted in [9,23] is non-linear and defined by minimise $\sum_{i=1}^P (x_i - T_i)^2$. Because of the definition of the target time, this objective reduces mathematically to the same function as our non-linear objective (Eq. (9)).

Note that for these problems the target time for some aircraft may be less than the earliest time on some runways. Although this may seem counter-intuitive the fact that we have a non-linear objective of the form given by Eq. (9) means that our heuristics can be applied directly without modification. Had we had a linear objective (Eq. (10)) then our heuristics could still have been applied, but would have needed minor modification with respect to the linear program solved in the improvement procedure (Section 5.8).

Table 7 presents the results for our Scatter Search and Bionomic Algorithm heuristics for the problems given in [9,23]. The termination criterion adopted for both algorithms for these problems is when 50,000 children have been generated. The Z_{best} solution value given corresponds to the best solution encountered, either in [9,23] or in our computational work. It is clear from that table that, on average, both our heuristics produce better quality solutions than the algorithms presented in [9,23]. As we would expect

from the results given in Tables 1 and 2, the Bionomic Algorithm out-performs Scatter Search on these problems. Computationally the times for Scatter Search and the Bionomic Algorithm are small. No computation times were presented in [9], and in [23] detailed computation times were not given, only a statement that the solutions derived were found in less than 5 minutes on a desktop PC. By contrast our largest computation time is less than one minute.

7. Conclusions

In this paper we considered the multiple runway Aircraft Landing Problem. Two different objective functions were considered, a non-linear objective function which has the close-up property and a linear objective function. We presented two population heuristic algorithms, Scatter Search and the Bionomic Algorithm, which have not been applied to the problem previously in the literature. Computational results were presented for a large number of problem instances involving up to 500 aircraft and five runways. These instances are much larger than those that have been considered by the majority of workers in the literature.

For the non-linear objective, our results indicated that our Bionomic Algorithm out-performed our Scatter Search algorithm. For the linear objective, the reverse was true with our Scatter Search algorithm out-performing our Bionomic Algorithm. As commented previously, for the Aircraft Landing Problem it is, in practical terms, very

useful to have the ability to change the objective adopted to reflect prevailing conditions at an airport. Our results indicate that, of the two algorithms we have considered, computational performance can reverse with adoption of a different objective. This indicates that care needs to be taken to ensure an appropriate algorithm is used with the objective adopted.

Regarding computation time, over all 101 problems considered, the largest computation time observed was associated with applying the Bionomic Algorithm (linear objective) to a 500 aircraft, one runway, problem, but this corresponded to only 1.2 seconds of computation per aircraft per replication.

With regard to the basic heuristic, it is clear from the results presented that, as problem size increases, this heuristic is not a computationally effective procedure, although results obtained are of high quality. With regard to runway dependent problems, our results indicate that these can be dealt with effectively, with our algorithms out-performing previous algorithms presented in the literature.

Acknowledgement

The financial support of EPSRC under grant GR/R55160/01 is acknowledged.

References

- [1] J. Abela, D. Abramson, M. Krishnamoorthy, A. De Silva, G. Mills, Computing optimal schedules for landing aircraft, in: Proceedings of the 12th National ASOR Conference, 1993, pp. 71–90.
- [2] N. Bauerle, O. Engelhardt-Funke, M. Kolonko, Routing of airplanes to two runways: Monotonicity of optimal controls, Working paper available from the first author at Inst. for Mathematical Stochastics, University of Hannover, Welfengarten 1, D-30167 Hannover, Germany, 2002. Currently available from: <<http://www.math.tu-clausthal.de/Arbeitsgruppen/Stochastische-Optimierung/VeroeffKLetzt/VeroeffKLetzt.html>>.
- [3] N. Bauerle, O. Engelhardt-Funke, M. Kolonko. On the waiting time of arriving aircrafts and the capacity of airports with two runways, Working paper available from the first author at Inst. for Mathematical Stochastics, University of Hannover, Welfengarten 1, D-30167 Hannover, Germany, 2003. Currently available from: <<http://www.math.tu-clausthal.de/Arbeitsgruppen/Stochastische-Optimierung/VeroeffKLetzt/VeroeffKLetzt.html>>.
- [4] J.E. Beasley, OR-library: Distributing test problems by electronic mail, *Journal of the Operational Research Society* 41 (1990) 1069–1072, Available from: <<http://mscmga.ms.ic.ac.uk/info.html>>.
- [5] J.E. Beasley, Population heuristics, in: P.M. Pardalos, M.G.C. Resende (Eds.), *Handbook of Applied Optimization*, Oxford University Press, 2002, pp. 138–157.
- [6] J.E. Beasley, M. Krishnamoorthy, Y.M. Sharaiha, D. Abramson, Scheduling aircraft landings—the static case, *Transportation Science* 34 (2000) 180–197.
- [7] J.E. Beasley, M. Krishnamoorthy, Y.M. Sharaiha, D. Abramson, Displacement problem and dynamically scheduling aircraft landings, *Journal of the Operational Research Society* 55 (2004) 54–64.
- [8] J.E. Beasley, J. Sonander, P. Havelock, Scheduling aircraft landings at London Heathrow using a population heuristic, *Journal of the Operational Research Society* 52 (2001) 483–493.
- [9] V.H.L. Cheng, L.S. Crawford, P.K. Menon, Air traffic control using genetic search techniques, Paper presented at the 1999 IEEE International Conference on Control Applications, August 22–27, Hawaii, USA, 1999. Currently available from: <<http://www.optisyn.com/Papers.html>>.
- [10] N. Christofides, The bionomic algorithm, Paper presented at the AIRO'94 Conference, Savona, Italy, 1994.
- [11] N. Christofides, Optimal design of telecommunication networks, Paper presented at the CO96 Conference, Imperial College, London, 1996.
- [12] S. Christofides, A. Christofides, N. Christofides, The design of corporate tax structures, *Mathematical Programming* 98 (2003) 493–510.
- [13] P.C. Chu, J.E. Beasley, Constraint handling in genetic algorithms: The set partitioning problem, *Journal of Heuristics* 4 (1998) 323–357.
- [14] V. Ciesielski, P. Scerri, An anytime algorithm for scheduling of aircraft landing times using genetic algorithms, *Australian Journal of Intelligent Information Processing Systems* 4 (1997) 206–213.
- [15] V. Ciesielski, P. Scerri, Real time genetic scheduling of aircraft landing times, in: D. Fogel (Ed.), *Proceedings of the 1998 IEEE International Conference on Evolutionary Computation (ICEC98)*, IEEE, New York, USA, 1998, pp. 360–364.
- [16] E. Elia, G. Salkin, N. Christofides, The bionomic algorithm for the tax structuring problem, Working paper, The Management School, Imperial College, London, 1997.
- [17] A.T. Ernst, M. Krishnamoorthy, R.H. Storer, Heuristic and exact algorithms for scheduling aircraft landings, *Networks* 34 (1999) 229–241.
- [18] T. Fahle, R. Feldmann, S. Gotz, S. Grothklags, B. Monien, The aircraft sequencing problem, in: *Computer Science in Perspective: Essays Dedicated to Thomas*

- Ottmann, Lecture Notes In Computer Science, vol. 2598, 2003, pp. 152–166.
- [19] F. Glover, Genetic algorithms and scatter search: Unsuspected potentials, *Statistics and Computing* 4 (1994) 131–140.
- [20] F. Glover, Scatter search and star-paths: Beyond the genetic metaphor, *OR Spektrum* 17 (1995) 125–137.
- [21] F. Glover, A template for scatter search and path relinking, in: *Artificial Evolution*, Lecture Notes in Computer Science, vol. 1363, 1998, pp. 3–51.
- [22] F. Glover, M. Laguna, R. Marti, Fundamentals of scatter search and path relinking, *Control and Cybernetics* 29 (2000) 653–684.
- [23] J.V. Hansen, Genetic search methods in air traffic control, *Computers and Operations Research* 31 (2004) 445–459.
- [24] ILOG, ILOG Concert technology 1.2, October 2001, ILOG, Inc., 1080 Linda Vista Ave., Mountain View, CA 94043, USA.
- [25] ILOG, ILOG Cplex 7.5, September 2001, ILOG, Inc., 1080 Linda Vista Ave., Mountain View, CA 94043, USA.
- [26] A.S. Jain, S. Meeran, A multi-level hybrid framework applied to the general flow-shop scheduling problem, *Computers and Operations Research* 29 (2002) 1873–1901.
- [27] G. Jung, M. Laguna, Time segmenting heuristic for an aircraft landing problem, Working paper, Leeds School of Business, University of Colorado, Boulder, CO, March 2003. Currently available from: <http://leeds.colorado.edu/Faculty/Laguna/articles/tsh.html>.
- [28] M. Laguna, P. Lino, A. Perez, S. Quintanilla, V. Vails, Minimizing weighted tardiness of jobs with stochastic interruptions in parallel machines, *European Journal of Operational Research* 127 (2000) 444–457.
- [29] V. Maniezzo, A. Mingozzi, R. Baldacci, A bionomic approach to the capacitated p -median problem, *Journal of Heuristics* 4 (1998) 263–280.
- [30] J. Mullins, Trails of destruction, *New Scientist* 2056 (1996) 28–31.