Centrum voor Wiskunde en Informatica

REPORT*RAPPORT*

Evolutionary air traffic flow management for large 3D-problems

C.H.M. van Kemenade, J.M. van den Akker, and J.N. Kok

Evolutionary Air Traffic Flow Management
for Large 3D-problems

C.H.M. van Kemenade

CWI

P.O. Box 94079, 1090 GB Amsterdam, The Netherlands

kemenade@cwi.nl

J.M. van den Akker

Informatics division

National Aerospace Laboratory NLR

Anthony Fokkerweg 2, 1059 CM Amsterdam, The Netherlands

vdakker@nlr.nl

J.N. Kok

Leiden University

Department of Computer Science

P.O. Box 9512, 2300 RA Leiden, The Netherlands

joost@wi.leidenuniv.nl

Abstract

We present an evolutionary tool to solve free-route Air Traffic Flow Management problems within a three-dimensional air space [5]. This is the first evolutionary tool which solves free-route planning problems involving a few hundred aircraft. We observe that the importance of the recombination operator increases as we scale to larger problem instances. The evolutionary algorithm is based on a variant of the elitist recombination algorithm. We show a theoretical analysis of the problem, and present the results of experiments.

1. INTRODUCTION

Previous work on application of Evolutionary Algorithms (EA's) to the Air Traffic Flow Management (ATFM) problem [1, 3, 7] handles only small problems involving few aircraft within a two-dimensional air space. We develop an evolutionary tool that can handle three-dimensional ATFM planning problems involving a few hundred of aircraft.

For smaller 2D-problem instances we obtained good results with a mutation based EA [7]. This system solves problems involving up to 20 aircraft within a square 2D sector of size $200 \times 200\,$km. Using these problem instances it appeared to be impossible to define a good recombination operator. All aircraft are relatively close to one another, and as a result of the small size of the 2D sector the probability that a combination of trajectories from different plans resulted in good offspring is small. When making the transition to the large scale problems we are studying currently, involving up to 800 aircraft within a 3D-sector of size $2000 \times 2000\,$km, there are more aircraft which do not get into

each others proximity. As a result different conflicts can be resolved in parallel, and a recombination operator can bring the solutions together in a single plan. So by scaling the size of the problem instances, the recombination operation changed from a virtually useless operator to an important operator.

There is no reason for abandoning the use of a probabilistic algorithm, as deterministic algorithms can not be guaranteed to give a solution within a reasonable amount of time either on the ATFM problem. This problem, which will be defined formally in section 2, is assumed to be NP-hard. Hence we have to rely on effective (probabilistic) approximation methods. The plans are created before the aircraft departs. Real-time planning is only used when aircraft deviate from the original plan, and should involve only a subset of all aircraft trajectories. On the ATFM problem a probabilistic method can be as good as any deterministic method, when designed carefully. Deterministic methods can also end up in deadlock situations, or encounter problems which can not be solved by that method. Our method has been designed in such a way that it searches a broad class of possible solutions, and it is not biased toward specific solutions in this class. This helps in preventing deadlock situations, and improves global search.

In section 2 we discuss air traffic flow management problems, in section 3 we show a theoretical analysis, in section 4 we introduce the evolutionary algorithm, section 5 gives the experimental set-up and the results, and finally we draw conclusions and discuss further work in section 6.

2. Air Traffic Flow Management Problem

Air Traffic Flow Management (ATFM) is involved in planning the movements of aircraft. A complete plan describes the trajectories of all involved aircraft. A trajectory defines the exact position of an aircraft as a function of time, so it corresponds to a path with additional temporal information. Two trajectories are conflicting when at a certain time the separation between these trajectories is too small. The minimal required separation between trajectories is 16 nautical miles in the horizontal plane (1 nautical mile = 1,852 meters), or 2000 ft vertical separation (1 foot = 0.3048 meters). Usually a plan is created for a sector of limited size. A sector contains a number of layers, called flight-levels. The altitudes of different flight-levels are chosen in such a way that aircraft flying in different flight-levels are never in conflict. An ATFM plan assigns a single trajectory to each aircraft. These trajectories should be chosen in such a way that there are no conflicts between aircraft. Furthermore the length of all trajectories and the number of maneuvers should be kept low. There is also a fairness requirement that states that the additional distances and maneuvers should be divided among all the aircraft (i.e. it should not be the case that only a few aircraft do all the maneuvering and do all the extra flying). The primary target of a planning is to let all aircraft move from their entry to their exit location in such a way that none of the trajectories are conflicting. As a secondary target, one can minimize the number of maneuvers, the additional distance, and try to satisfy the fairness requirement. Maneuvers are usually uncomfortable for the passengers, and too much additional distance may lead to a delayed arrival of the aircraft. Currently ATFM is based on a restricted model of the air space, the so called network model. This model assumes a fixed network of corridors within the air space, each containing a number of flight-levels. An aircraft is assumed to fly through a corridor from beacon to beacon. Intersections of corridors are always marked by beacons. Only near those beacons, an aircraft is allowed to switch to a different corridor. So this model introduces a kind of three dimensional highway network. The main difference compared to a highway network is that flight control within this ATFM network is more strict. Aircraft are assumed to follow the trajectory which is agreed upon in advance, and pilots have to ask for clearance when they want to deviate from the planned trajectory. The network model restricts the number of possible trajectories. As a result the network model does not use the full capacity of the air space. Due to the increasing amount of air traffic, the air space above Europe is almost saturated. Increased accuracy of navigation equipment and the availability of better computers allows for less restricted air space models. One such model

is the free-route model which allows arbitrary shaped trajectories, has a larger degree of flexibility, but it results in a completely different planning problem. In this paper we propose an evolutionary algorithm to create such a planning, and we study how this algorithm scales with respect to the size of the problem instances.

3. THEORETICAL ANALYSIS OF THE ATFM PROBLEM

We assume that a planning has to be created for a square sector of size $s \times s$, containing l separate flight-levels. The source and the destination of the aircraft are 2D-locations, chosen at random within the sector, using a uniform distribution. The flight-level of the aircraft at the entry- and exit-location can be chosen freely. The entry and exit locations of flights in a planning do not have to correspond to actual locations of airports. When aircraft are departing or approaching an airport their trajectory is managed by the controllers at the corresponding airport. Such a local Air Traffic Control center at an airport usually manages lower regions of the airspace that do not overlap with the high altitude sector we are considering here. We may hence assume that the aircraft enter the airspace at some distance from the airport. We assume that all the aircraft have the same velocity v. The flights are to be planned within the interval $[0, t_{hor}]$. The time of entry is selected at random within this interval. This time of entry is accepted if the aircraft can reach its exit location within the interval $[0, t_{hor}]$, when flying in a straight line.

In order to predict the number of conflicts we can use physical models describing the number of collisions between a set of gas molecules in a box per unit of time [2]. When translating such a model to our case we get the following formula,

$$E[\#Conf] = \alpha \frac{t_{hor} \, d_{sep} \, v}{s^2 l} n^2.$$

Here v is the average velocity. The constant α is introduced to account for the non-uniform distribution of aircraft over the sector. The density of aircraft will be highest near the center of the sector, and lowest near its borders. Within the gas model this constant is one as the gas molecules are distributed uniformly over the complete volume. The complexity of a problem instance scales quadratically with the number of aircraft involved. This is tested as follows. We have varied the number of aircraft from 100 to 800. For each number of aircraft 25 random problem instances are created. For each problem instance 1000 random plans, containing only straight-line trajectories, were generated. Figure 1 shows the average number of conflicts detected as a function of the number of aircraft, and a fitted curve showing a quadratic scaling in the number of aircraft. Observe the close match between the fitted curve and the simulation results. Figure 1 also contains the average number of conflicts of the best plan $E[\#Conflicts \; best]$ which is estimated by taking an average over 25 independent problem instances (for each instance the #conflicts in the best plan out of 1000 random plans is used). When comparing the result for a horizontal separation d_{sep} of 16 nm to those for a separation of 8 nm we observe the linear scaling in this parameter. This is in accordance with the gas model.

4. EVOLUTIONARY ALGORITHM

The elements of the population are plans. A plan is a set of trajectories, one for each flight to be scheduled. The plans in the initial population involve only straight-line trajectories from the entry to the exit location at a fixed flight-level. If we consider a problem with three flight-levels, then this yields already 3^n possible flight plans (n is the number of flights). It seems to be a reasonable choice to consider only trajectories with a small number of maneuvers. We do this by starting from an initial population containing straight-line trajectories only, and use the recombination operator to search for non-conflicting combinations of trajectories. Trajectories containing many additional maneuvers are not likely to be good due to the fairness requirement: a solution with only straight-line trajectories is an optimal solution, according to our fitness measure. As evolution proceeds, more complex trajectories arise due to the application of a mutation operator.
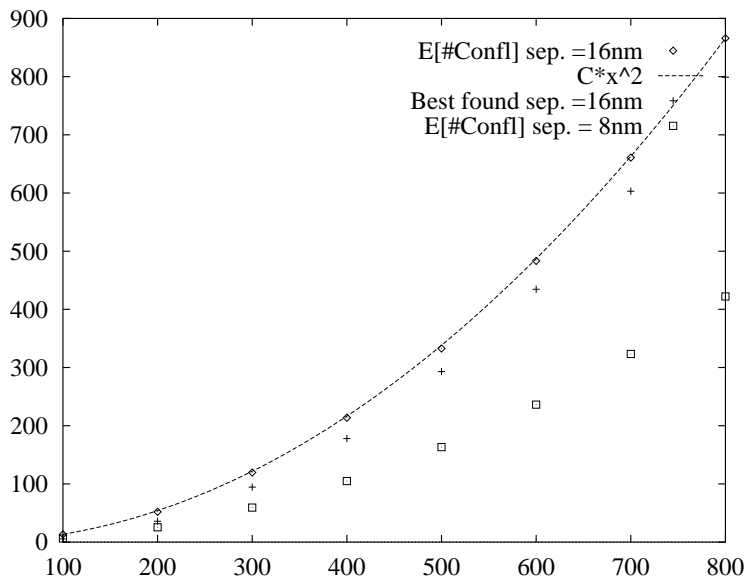
Figure 1: $E[\#conf]$ as a function of n for a horizontal separation of 8 nm and 16 nm.

The fitness is defined as follows:

$$f(plan) = -C(plan) - \frac{M(plan) + R(plan)}{d}$$

Here $C(plan)$ is the total number of conflicts within the plan, $M(plan)$ is the number of avoidance maneuvers, and $R(plan)$ is the sum of all the distances traveled by the aircraft divided by the summation over the lengths of the straight-line trajectories. The constant d is chosen such that $(M(plan) + R(plan))/d$ never becomes larger than one. In this way a fitness $f(plan) > -1$ corresponds to a conflict free planning.

The Elitist recombination algorithm [6] is used as a basis for our algorithm. This algorithm does a random pairing of all elements of the population, without using fitness information. Each pair of parents creates two offspring by means of recombination. The best two individuals among parents and offspring are propagated to the next generation. This competition between parents and offspring prevents rapid duplication of relatively fit individuals, and as a result decreases the probability of premature convergence. Within our implementation each offspring competes with only one of its parents.   This scheme has a lower selective pressure than the standard elitist recombination scheme. This competition with one parent is also used in the deterministic crowding scheme [4]. But deterministic crowding lets offspring compete with the most similar parent. The Elitist recombination algorithm is chosen as it does not deteriorate the average fitness when inferior offspring is produced due to the (population) elitism, and it prevents too rapid convergence of the population (the offspring always has to compete against its parents).

The main operator in our algorithm is the recombination operator. If two conflicts involve two disjoint sets of aircraft, then often it is possible to resolve these conflicts independently. A good recombination operator should be able to merge separate resolutions. The recombination operator creates one offspring using two parents. It starts with an empty plan for the offspring. Then it iteratively selects an aircraft and takes the corresponding trajectory from one of its parents. This

```
while ¬ ready do
    /* apply Elitist recombination for N_gen generations */
    for i ← 1 to N_gen do
        elitist recombine Population
    /* introduce new trajectories by means of mutation */
    for i ← 1 to (F_mut × #conflicts best) do
        pl ← random plan from Population
        fl ← random conflicting trajectory from pl
        k ← 0
        repeat
            k ← k + 1
            if k < N_try then
                ntr ← mutate(fl)
            else
                ntr ← random straight(fl)
        until nconflicts(pl\fl, ntr) < nconflicts(pl\fl, fl)  ∨  k ≥ N_try
        if nconflicts(pl\fl, ntr) < nconflicts(pl\fl, fl) then
            pl ← ntr ∪ pl\fl
```

Figure 2: Pseudo-code of main loop of Evolutionary ATFM

parent is selected as follows. One of the parents is selected at random, and it is checked whether the corresponding trajectory would introduce new conflicts within the child. If it does not introduce conflicts this parent is selected, otherwise the other parent is selected. The trajectory of the selected parent is added to the child. So aircraft are selected in a random order, and for each flight the primary parent is selected at random. This selection procedure results in a preference for non-conflicting trajectories. This bias helps the recombination operator to scale-up well with the size of the problem instances. When a (purely random) uniform recombination operator is applied it likely results in offspring performing worse than the parents, when a few hundred of aircraft are involved.

The mutation operator is the only operator that introduces new trajectories within our algorithm. A new trajectory is created by making a copy of one of the existing trajectories and adding some additional maneuvers to it. Two different mutation operators have been defined:

level-mutation: modifies an existing trajectory by adding a random change of flight-level at a random time.

detour-mutation: introduces a detour by changing its heading by $\pm 45^o$, flying for a random duration in this new direction, and then changes its heading towards the exit-location again.

Note that the detour mutation can cancel previously inserted maneuvers. Although these mutation operators are relatively easy to implement one can not guarantee that a single application of these operators will produce a good trajectory. New trajectories that do not resolve any conflicts will rapidly be filtered out of the population by the EA (such trajectories only introduce a penalty and no benefit for the fitness). If the probability that good trajectories are produced becomes small, premature convergence becomes likely. In order to prevent this we have taken the mutation out of the selection-production cycle. We run the EA for a few generations and then we apply mutation to the plans present within the population. A mutation is only accepted if the produced trajectory increases fitness when replacing the original trajectory. Using this scheme we get a more constant production of well-performing trajectories.

Figure 2 summarizes the main loop of the evolutionary ATFM in pseudocode. The ready-predicate becomes true when a conflict-free solution is obtained, or when the maximal number of function evaluations is reached, and nconflicts(pl, tr) counts the number of between trajectory tr and the trajectories with plan pl.

Our evolutionary algorithm is designed in such a way that we get a good balance between exploration and exploitation. Good exploration is obtained because two independent applications of the recombination operator to the same pair of parents are still likely to result in different offspring and from the bias towards better trajectories within the mutation operator. Good exploitation comes from the bias towards preserving better combinations in the recombination operator, and from Elitist recombination algorithm with low selective pressure.

Our experience is that incorporation of problem specific knowledge is usually necessary when solving large problems. In this case we have introduced knowledge in several ways. We start with a population containing only straight-line trajectories. Within acceptable plans the trajectories usually do not contain many maneuvers, so this is a reasonable starting point. A single application of the mutation operator can only introduce a small number of additional maneuvers. Furthermore knowledge is incorporated in the recombination operator as it makes a biased decision when it has to decide which parent will provide the trajectory for a certain aircraft. Using this knowledge the probability that a well-performing offspring is created is increased, especially for problems involving many aircraft. Designing biased operators is not a trivial task. One has to introduce a bias to enhance the probability that an operator produces reasonable offspring, but introducing too much bias might easily lead to premature convergence. An important guideline we used is that independent operator application should likely result in different offspring, even when using the same pair of parents.

An aspect of our algorithm is that it does not rely too much on local search methods. Instead it uses a kind of random generation of trajectories by means of the mutation operator. The primary goal of the EA is to find those trajectories that perform well and combine these trajectories to a complete planning. Such a probabilistic approach does not use an explicit local search method. As a result it does not easily get trapped in a kind of deadlock situation where no enhancements are possible.

5. Experiments and Results

During the experiments we varied the number of aircraft to be planned, and as a result the expected number of conflicts in the sector. The parameters shown in Table 1 were used during all the experiments. A single application of the current mutation operators introduces one avoidance maneuver. Such an avoidance maneuver contains several elementary maneuvers. For example an application of the detour-mutation introduces two additional maneuvers, the first is a change of heading by $\pm 45^o$, and the second is a curve to move the heading of the aircraft towards the exit location.

The first set of results is obtained using straight-line trajectories only. Mutation is only used to (re)introduce straight-line trajectories that got lost. It does not introduce new trajectories. Even for this (restricted) subset of possible trajectories, l for each flight, the search space already is large. The number of possible plans is proportional to l^n. The other experiments use level-mutation, detour-mutation and mixed mutation. The mixed mutation operator first tries to find a better trajectory by applying detour-mutation, if this fails the level-mutation is applied once. The detour-mutation is applied first because it has better convergence properties. During the experiments we varied the number of aircraft. For each problem instance 25 random plans were created. All graphs show the negated fitness. A run is terminated when its negated fitness drops below one, this corresponds to a conflict-free planning. We recall that the fitness measure has been chosen in such a way that $\lfloor -f(plan) \rfloor$ corresponds to the number of conflicts within the plan. Only detour mutation is used, unless stated otherwise.

Figure 3 (left) shows the overall worst solution over all 25 independent problem instances. Different

Parameter		Value	
plannings horizon	t_{hor}	4	hours
number of aircraft	n	100-600	aircraft
velocity	v	900	km/hour
separation	d_{sep}	16	nautical miles
size of sector	s	2000	km
#levels	l	3	
population size		16	
#function evaluations		2500	maximum
#independent runs		25	
#avoidance man. per ac.		4	
#generation inner loop	N_{gen}	2	
rel. #mutations in outer loop	F_{mut}	0.5	
#tries for a mutation	N_{try}	10	

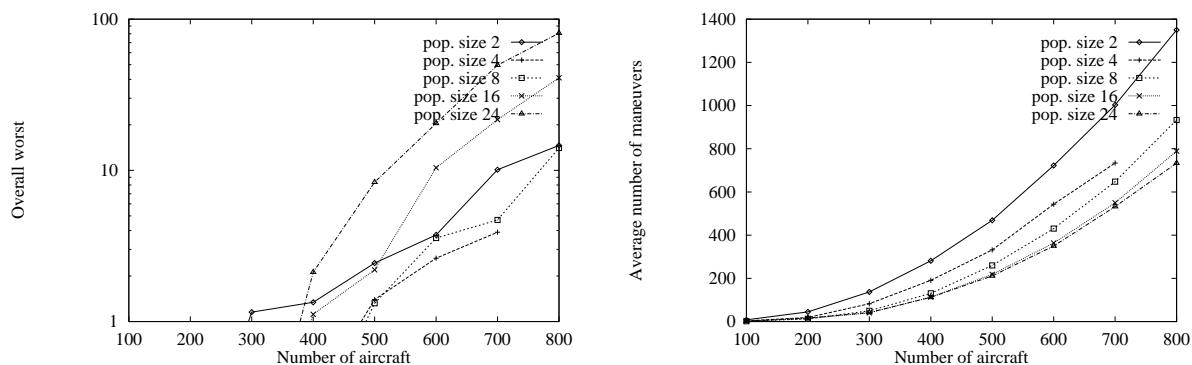Table 1: Parameters used during the experiments

Figure 3: overall worst solution over 25 independent problems (left) and number of avoidance maneuvers (right) for different population sizes

curves correspond to different population sizes. This graph suggests that the worst case behavior gets better when the population size gets smaller. The only exception is in the experiment using the extremely small population size 2, which gets trapped easily. The good performance of small populations is mainly due to the upper limit on the number of function evaluations. The larger the population size, the higher the probability that the algorithm is terminated because it hits this upper limit. Figure 3 (right) shows the average number of avoidance maneuvers per plan as a function of the number of aircraft. We see that the number of introduced maneuvers becomes smaller and hence better solutions are obtained when we increase the population size. When we measure quality in terms of the additional distance to be flown then EA's using larger populations also create solutions of higher quality. Our experiments show that EA's using small populations are more likely to get trapped in a configuration where it seems impossible to make further progress. We conclude that larger populations result in a more reliable convergence process and better solutions.

Figure 4 (left) shows the average negated fitness of the best solution as a function of the number of function evaluations for a population size 16. The small bulbs in most curves are due to the fact that fitness is calculated over the non-terminated runs, so if a run terminates successfully, the shown
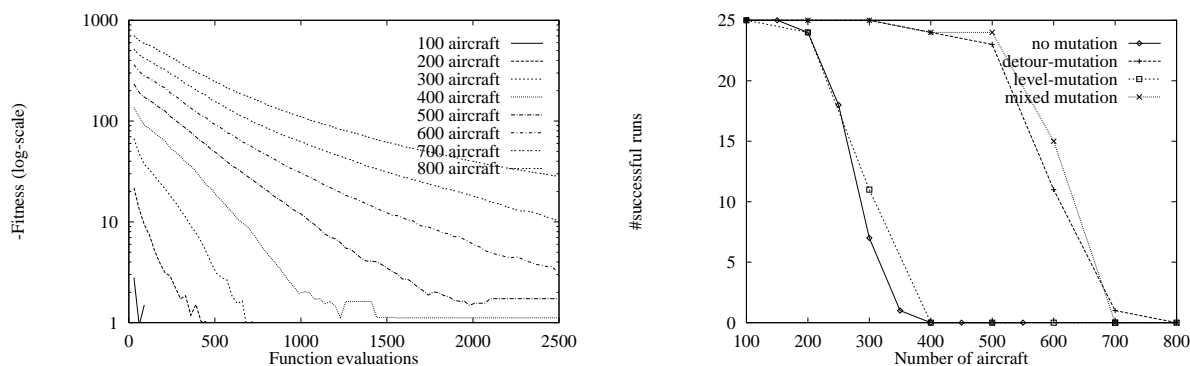
Figure 4: Convergence curves for different problem sizes (left) and number of successful runs for different types of mutation (right)

average fitness can increase. We see that the speed of convergence is dependent upon the problem size. The failure of the algorithm to solve our largest problem instances is probably due to the limit on the number of function evaluations. On the 800 aircraft problem instances the method still obtains a 30-fold reduction in the number of conflicts on average.

Figure 4 (right) shows the number of successful runs out of the 25 independent runs, for the different types of mutation operators. When allowing no mutation, the problem rapidly becomes unsolvable as the number of aircraft increases. Without mutation only solutions involving straight-line trajectories are considered. When mutation is allowed the probability of finding a conflict-free solution is much larger, and the scaling properties are better. The method using level-mutation performs slightly better than the method without specialized mutation operators. This is probably due to the fact that a level-change takes quite some time, and that during a level-change an aircraft claims space within a number of flight-levels. A single level change can also easily introduce dependencies between aircraft.

6. CONCLUSIONS
The free-route planning problem has a search space that grows exponentially when the number of aircraft increases. To be able to handle large scale problems of this type it is required to incorporate knowledge regarding the problem domain. We have done so by means of a non-uniform seeding of the initial population and by designing problem specific evolutionary operators. Introduction of such operators has to be done carefully in order to prevent that certain good solutions are ignored and to prevent premature convergence. The choice of the evolutionary algorithm and the way mutations are introduced also have a strong influence on the balance between exploration and exploitation. A proper balance is necessary to obtain good solutions using only a limited amount of computation. The designed algorithm performs well and requires only small populations.

Given the fact that the airspace above Europe contains at least 9 separate flight levels, the our current tool can route approximately $(3 \times 4 \times 500 =)$ 6000 aircraft within a time span of 12 hours, using a horizontal separation of 16 nautical mile.

Further research will be devoted to real-time replanning. Due to weather conditions, or delays at airports it is likely that some aircraft do not fly according to their planning. Under such circumstances it is important to have rapid replanning tools, which use the current planning as their starting point.

REFERENCES
1. J. Alliot, H. Gruber, G. Joly, and M. Schoenauer. Genetic algorithms for solving air traffic control conflicts. In *Ninth conference on Artificial Intelligence for Applications*, pages 338–344. IEEE Computer Society press, 1993.

2. Department of Civil Engeneering, Princeton University. *Proceedings of the Conference on Safety Issues in Air Traffic Systems Planning and Design*, 1983.

3. I.S. Gerdes. Application of genetic algorithms to the problem of free-routing for aircraft. In *First IEEE conference on Evolutionary Computation*, pages 536–541, 1994.

4. S.W. Mahfoud. Crowding and preselection revisited. In *Parallel Problem Solving from Nature II*, pages 27–36. Springer, 1992.

5. Arnold Field Obe. *International Air Traffic Control; Management of the World's Airspace*. Pergamon Press, Oxford, 1985.

6. D. Thierens and D.E. Goldberg. Elitist recombination: an integrated selection recombination GA. In *First IEEE conference on Evolutionary Computation*. IEEE Press, 1994.

7. C.H.M. van Kemenade, C.F.W. Hendriks, H.H. Hesselink, and J.N. Kok. Evolutionary computation in air traffic control planning. In S. Forrest, editor, *Sixth International Conference on Genetic Algorithms*, pages 611–616. Morgan Kaufmann, 1995.