Air traffic conflict resolution by genetic algorithms

Frédéric Medioni, Nicolas Durand, Jean-Marc Alliot

Air Traffic Conflict Resolution by Genetic Algorithms

Frédéric Médioni Nicolas Durand Jean-Marc Alliot

ENAC[1] ENSEEIHT[2] ENAC

1. Ecole Nationale de l'Aviation Civile
2. Ecole Nationale Supérieure d'Electronique, d'Electrotechnique, d'Informatique, et d'Hydraulique de Toulouse

Abstract. The resolution of Air Traffic Control (ATC) conflicts is a constrained optimization problem: the goal is to propose, for a certain number, n, of aircraft, which might be in conflict in a near future, trajectories that satisfy the separation constraints between aircraft, and minimizes the delays due to the conflict's resolution. The type of conflict resolution trajectories we use allows to split the problem in two steps: first we choose, and freeze, what we call a *configuration* of the problem, i.e. for each aircraft, the direction in which the aircraft is diverted, and for each pair of aircraft, which of the two aircraft passes first at the crossing point of the two aircraft trajectories. We can then compute the optimal trajectories corresponding to this *configuration*, by solving a simple linear optimization problem. Thus we can use an Genetic Algorithm, along with a linear optimization algorithm, such as the *simplex* algorithm: the elements of the population, on which the GA operates, code *configurations* of the problem, and are evaluated using a linear optimization program.The advantage of this approach is that we get, as well as the *fitness* of an element of the population, the local optima corresponding to the *configuration* coded by this element. The GA actually searches for the global optimum among these local optima. We applied this methods to conflicts in which up to 6 aircraft are involved, and obtained really promising results.

1 Introduction

During the first years of civil aviation, the low performances of aircraft, and the fact that they only flew under good visibility conditions, enabled pilots to ensure their own safety. Then, aircraft began to fly faster, and needed to be able to fly even under bad visibility conditions : there was a need for Air Traffic Control. The constantly increasing number of aircraft flying at the same time in one given

area led to the division of the airspace in several sectors. The goal of Air Traffic Control is to ensure the aircraft *separation*, *i.e.* the fact that the distance between two aircraft is larger than a given value (the *standard separation*), while minimizing delays due to possible alterations of the aircraft routes. A *conflict* is said to occur when two or more aircraft are not *separated*. Ensuring conflicts resolution while minimizing delays is a complex problem, which is still empirically solved by air traffic controllers. But this way of handling ATC conflicts will probably not be efficient much longer, because the number of aircraft flying, and therefore the number of conflicts to solve simultaneously, is increasing. The automation of ATC would probably allow for a gain in capacity. Several studies have been made in that direction :

- the SAINTEX project [AL92] works like an *expert system*, and tries to minimize delays in some cases.
- Karim Zeghal [Zeg93] presents reactive techniques, which are robust to perturbations, but do not lead to optimal trajectories.
- the AERA 3 project [NFC+83], [Nie89b], [Nie89a], only seeks for optimal trajectories in the case of conflicts involving two aircraft.
- ARC 2000 [K+89], [FMT93], uses priority rules to limit the number of aircraft involved in a conflict, and so be able of optimizing their trajectories. There is no search for a global optimum.
- an ENAC and CENA research group has been using Genetic Algorithms since 1992 to solve *en route* ATC conflicts, and to find optimal trajectories for the aircraft (*cf* [AGS93], [DDAS94]).

We will now present another application a Genetic Algorithm, along with a linear optimization algorithm, to the resolution of ATC conflicts.

2 Background

2.1 Hypotheses

The trajectories generated must meet several criteria: they must be compatible with the aircraft performances; they must be simple enough to be easily transmitted to pilots, either by the controllers, or automatically. This means that the number of changes of heading, altitude, or speed, should be limited.

The hypotheses we will make are actually more restrictive: in the following, we will assume that the aircraft trajectories are in an horizontal plane. We will consider only conflicts involving aircraft flying at the same altitude. Furthermore, we assume that aircraft speeds are constant.

2.2 The optimal collision avoidance trajectory

We will first consider conflicts involving only two aircraft, and alter the trajectory of a single aircraft, and we will then extend the results of this study to the resolution of conflicts involving n aircraft, where $n \geq 2$.

Let us consider two aircraft as two points on the Euclidean plane, a_1 and a_2. At time t, their positions are given by the pairs of coordinates $(x_1(t), y_1(t))$ and $(x_2(t), y_2(t))$. At time t_o aircraft a_1 and a_2 are respectively at points O_1 and O_2, and are supposed to reach points D_1 and D_2 at time t_f. Let C be the point on which the trajectories of the two aircraft would meet if there was no deviation. The collision avoidance trajectories must then satisfy the following constraint:

$$\forall t \in [t_o, t_f], \quad (x_1(t) - x_2(t))^2 + (y_1(t) - y_2(t))^2 \geq d^2 \tag{1}$$

where d is the chosen horizontal standard separation.

We now want to find the optimal collision avoidance trajectory for aircraft a_1 (the trajectory of aircraft a_2 remaining unchanged), *i.e.* the trajectory which leads to the smallest increase of length, compared with the trajectory of a_1 with no deviation, while satisfying constraint (1). It has been shown (*cf* [Dur94]) that in this case, the trajectory of a_1, in the reference system of aircraft a_2, is composed of three phases:

1. a line segment on which constraint (1) is not saturated;
2. an arc of circle on which constraint (1) is saturated;
3. a second line segment on which constraint (1) is not saturated.

Both line segments are tangent to the circle of center C and of radius d (see figure 1). This type of trajectory is difficult to follow precisely for a pilot. This is why we will introduce two other collision avoidance modellings, which will lead to longer but easier trajectories: the *turning point* and the *offset* modellings. These two modellings are presented and compared in [Dur94]. We briefly summarize them here.

2.3 The turning point collision avoidance

The principle of the turning point modelling is simple: the two line segments described above are extended until the arc of circle is reduced to a turning point (*cf* figure 1).

It has been shown that under some additional hypotheses (ensuring that the collision avoidance is started soon enough and that the angle between the initial trajectories of the planes is not too small), the shortest turning point trajectory with no conflict is no more than 5% longer than the optimal collision avoidance trajectory (see [Dur94] for details).

2.4 The offset collision avoidance

The offset modelling for collision avoidance (*cf* figure 2) uses collision avoidance trajectories which are parallel to the initial trajectories, but moved aside. The collision avoidance trajectory is then composed of three linear phases:

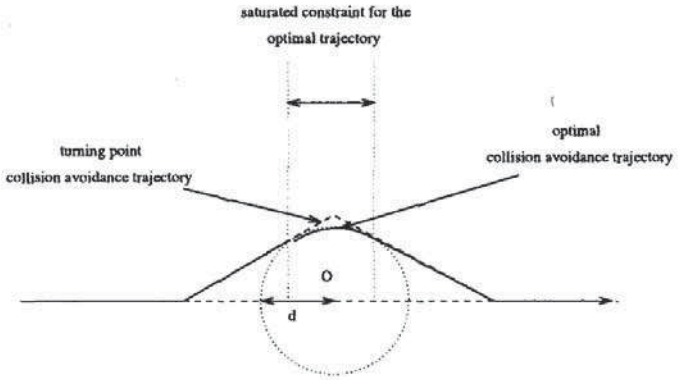

Fig. 1. Comparison optimal trajectory / turning point

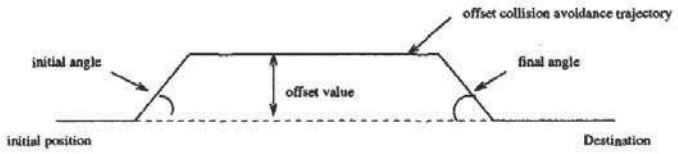

Fig. 2. The offset modelling

1. the *initial phase*: first, the aircraft veers off course with a certain angle (the *initial angle*), until it reaches a certain distance from its original trajectory (this distance is called the *offset value*). The direction to which the aircraft is deviated is called the *offset direction*, the initial angle will always be counted positive.
2. the aircraft then follows a trajectory which is parallel to its initial trajectory.
3. the aircraft finally gets back onto its initial trajectory, with an angle called the *final angle*.

Both the initial angle and the final angle can be fixed. It will be the case in the following. Furthermore, they will be equal, and their common value will be noted β. The duration of the second phase (*i.e.* the time during which the aircraft flies along a line segment which is parallel to its original trajectory) has no influence on the delay due to the collision avoidance. It should however be made as short as possible: while on its deviated trajectory, an aircraft can get in an unpredicted conflict with another one, and the aircraft should anyway be back on its original trajectory when it arrives into a new control sector (about ATC sectors, see [DASF94a], [DASF94b], [DAAS94], [DDAS94]).

When we limit the collision avoidance trajectories to those with an offset, the search for the shortest one is a linear optimization problem, with linear constraints (after some additional simplifications we will describe in the next section). We will then extend these results to the case of a conflict involving n aircraft. The turning point collision avoidance modelling leads to a shorter delay than the offset collision avoidance modelling (*cf* [Dur94]). But the technique we are going to present now uses the fact that the offset modelling leads to linear constraints.

3 A conflict involving n aircraft

Let us consider n aircraft, flying in one ATC sector, from time t_o to time t_f, which may get involved in conflicts. Clearly, the n aircraft are globally separated, if and only if each of them is separated from the others (*i.e.* if and only if constraint (1) is satisfied for each pair of aircraft).

We will now assume that $t_o = 0$. Let us consider two aircraft, a_i and a_j. Let ϕ_{ij} be the angle between their original trajectories (see figure 3), t^i_{ij} (resp. t^j_{ij}) the time at which a_i's original trajectory (resp. a_j's) intersects $a'_j s$ (resp. a_i's), and v_i and v_j the norms of the speed vectors of a_i and a_j (which are assumed to remain constant on $[t_o, t_f]$). Let us consider the orthonormal basis (C_{ij}, E_j, E_i), where C_{ij} is the intersection of the original trajectories of aircraft a_i and a_j, and where the x-axis (E_j), is directed by a_j's speed vector (see figure 3).

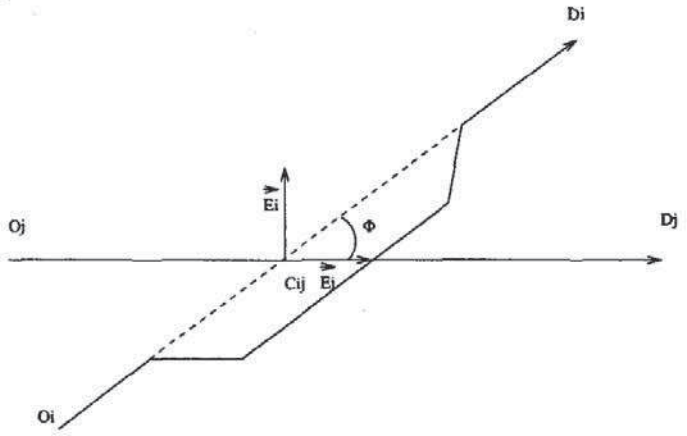

Fig. 3. a conflict involving two aircraft, a single one being deviated.

By writing the coordinates of the aircraft a_i and a_j in this basis, as a function of time $t \in [t_0, t_f]$, we get the following separation constraint, for $t \in [t_o, t_f]$:

$$(v_i - v_j)^2 t^2 + (2v_i v_j \cos(\phi_{ij})(t^i_{ij} + t^j_{ij}) - 2{v_i}^2 t^i_{ij} - 2{v_j}^2 t^j_{ij})\ t \\ +{v_i}^2 {t^i_{ij}}^2 + {v_j}^2 {t^j_{ij}}^2 - 2v_i t^i_{ij} v_j t^j_{ij} \cos(\phi_{ij}) - d^2 \geq 0 \qquad (2)$$

It is a second degree inequation in t. Since it is true for a large enough t, it is true for all if and only if its discriminant is negative, which leads to the following inequation:

$${v_i}^2 {v_j}^2 \sin(\phi_{ij})^2 (t^i_{ij} - t^j_{ij})^2 \geq d^2({v_i}^2 + {v_j}^2 - 2v_i v_j \cos(\phi_{ij})) \qquad (3)$$

Which yields in turn the two following conditions, depending on which of the two aircraft reach C_{ij} (the intersection of the two trajectories) first:

- If aircraft a_i passes behind aircraft a_j, we get:

$$(t^i_{ij} - t^j_{ij}) v_i v_j \sin(\phi_{ij}) \geq d\sqrt{{v_i}^2 + {v_j}^2 - 2v_i v_j \cos(\phi_{ij})} \quad (4)$$

– If aircraft a_j passes behind aircraft a_i, we get:

$$(t^j_{ij} - t^i_{ij}) v_i v_j \sin(\phi_{ij}) \geq d\sqrt{{v_i}^2 + {v_j}^2 - 2v_i v_j \cos(\phi_{ij})} \quad (5)$$

Note: condition (3) is necessary and sufficient for separating two aircraft on $t \in]-\infty, +\infty[$. We wanted a necessary and sufficient condition for their separation for $t \in [t_o, t_f]$. On this time interval, condition (3) is sufficient but not necessary. The constraints we use are too strict, and we may miss optimal admissible solutions. This can be limited by introducing a superior bound for the offset values: if we know that two aircraft will be separated as long as they are not deviated with an offset value larger to this superior bound, we do not take into account the separation condition for these two aircraft (*cf* [Med94]).

The deviations of aircraft a_i and a_j change the relative positions of their two trajectories. It changes the intersection time. Let us consider the consequences of the offset collision avoidance on the separation constraints for aircraft a_i and a_j, where $1 \leq i < j \leq n$.

The deviation of the two aircraft has two consequences on intersection times $t^i_{i,j}$ and $t^j_{i,j}$. These consequences depend on the offset angle, named β, on the offset direction, and on the offset value. If aircraft a_i is deviated, with its offset value being d_i, we get:

– whatever the offset direction may be, the deviation of the aircraft causes a delay, and $t^i_{i,j}$ increases of $\frac{d_i \tan(\frac{\beta}{2})}{v_i}$.
– When a_i's offset direction is such that a_i's new trajectory is outside angle ϕ $\frac{d_i \cot(\phi_{i,j})}{v_i}$ is added to $t^i_{i,j}$. If a_i's new trajectory is inside this angle, this value is subtracted from $t^i_{i,j}$.
– In the same way, when a_i's offset direction is such that a_i's new trajectory is outside angle ϕ, $t^j_{i,j}$ increases of $\frac{d_i}{v_j \sin(\phi_{i,j})}$.

Thus, by modifying t^i_{ij} and t^j_{ij} in conditions (4) and (5), we get a linear inequation involving d_i and d_j. Its coefficients depend on the offset direction for each aircraft, and for each pair of aircraft, on which one passes behind the other.

For instance, if both aircraft are deviated outside, as shown on figure 4, and if a_i passe behind a_j, we get:

$$\begin{aligned}(t^i_{ij} + \frac{d_i \tan(\frac{\beta}{2})}{v_i} + \frac{d_i \cot(\phi_{ij})}{v_i} + \frac{d_j}{v_i \sin(\phi_{ij})} - t^j_{ij} - \frac{d_j \tan(\frac{\beta}{2})}{v_j} \\ - \frac{d_j \cot(\phi_{ij})}{v_j} - \frac{d_i}{v_j \sin(\phi_{ij})}) v_i v_j \sin(\phi_{ij}) - d\sqrt{{v_i}^2 + {v_j}^2 - 2v_i v_j \cos\phi_{ij}} \geq 0 \quad (6)\end{aligned}$$

This condition, as we mentioned before, is a sufficient condition for the separation of the aircraft, only during the second phase of the offset collision avoidance trajectory (when the aircraft flies along a line segment, parallel to its original trajectory). In the following, we will only consider these constraints. The

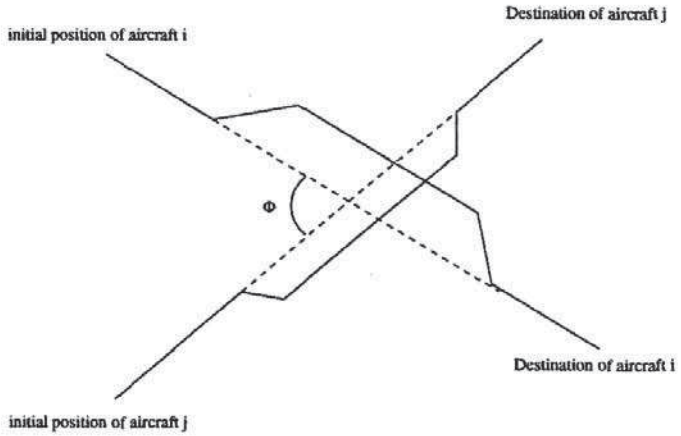

Fig. 4. A conflict involving two aircraft, both of them deviated to the outside

initial and final phases of the offset collision avoidance (when the aircraft leaves its original trajectory to get onto a parallel line segment, and when it gets back onto its original trajectory) lead to non linear constraints, which we will ignore. We will assume that aircraft are deviated soon enough, and will stay on their collision avoidance trajectories long enough, to allow us to ignore these constraints.

We have so far implicitly assume that the original trajectories of aircraft were secant. If the trajectories of two aircraft are parallel (the aircraft may face each other, or one aircraft may fly fast behind a slower one), similar calculi lead to similar linear conditions. What is now important to know is not whether a_j passes behind a_j, or not, but whether a_j passes to the left of a_j, or to the right. For the sake of simplicity, we will assume in the remainder of this article that the trajectories of the planes are secant.

4 A linear, but strongly combinatorial problem

In the previous section, we established that, after some simplifications, the conditions for the separation of each pair of aircraft, lead to a linear constraint. So we have $\frac{n(n-1)}{2}$ linear constraints. We add to these constraints the following ones: $d_i \geq 0$ for $1 \geq i \geq n$, which leads to a total of $\frac{n(n+1)}{2}$ constraints, for a conflict involving n aircraft (or $\frac{n(n+2)}{2}$ constraints if we impose superior bounds for the offset values d_i).

We now want to minimize the global delay, *i.e.* the sum of the delays of aircraft, due to the conflict resolution, under these constraints. The aircraft are delayed only during the initial and final phases of the collision avoidance trajectories. Their delay is a function of the initial angle β. For an aircraft a_i, with offset value d_i this delay is:

$$2\frac{d_i \tan(\frac{\beta}{2})}{v_i}$$

The sum of the delays is then, for the n aircraft:

$$S(d_1, .., d_n) = 2 \sum_{i=1}^{n} \frac{d_i \tan(\frac{\beta}{2})}{v_i}$$

This function is linear in d_i for $1 \leq i \leq n$.

When the directions of the offsets are fixed for all the aircraft, as well as which aircraft passes behind the other for each pair of aircraft, the offset values leading to the minimal global delay are the solutions of a linear optimization problem, with $\frac{n(n+1)}{2}$ constraints. This problem can be solved, by an algorithm such as the *simplex* algorithm, for instance. But the best we will get can only be a local optimum. We might even get no solution at all.

The linear constraints' coefficients depend on the offset direction for each aircraft and on which aircraft passes behind the other for each pair of aircraft. So these data have to be fixed, before the linear optimization program is run. If n aircraft are involved in the conflict to solve, there are $2^{\frac{n(n+1)}{2}}$ possible combinations for these data. An exhaustive search for a global optimum among the local ones would imply to solve as many linear optimization problems. When n grows, this number quickly becomes too large: 32 768 for $n = 5$, 2 097 152 for $n = 6$.

Two questions naturally arise:

1. Isn't there a fast way to establish that a whole class of such data combinations lead to a linear problem with no solution, and to ignore all the combinations belonging to such a class ?
2. Isn't it possible to group different combinations into one single connected component onto which could be run an optimization program, even with non linear constraints ?

Positively answer to one or both questions would have allowed to reduce the number of combinations to take into account, either by ignoring some of them, or by grouping several ones. Unfortunately, even in the very simple case of a conflict involving only two aircraft, answers to these question are very complex (they mainly depend on the value of the initial angles β, compared to the angle between the trajectories of the aircraft). It gets even more complex for three aircraft, and doesn't lead to a significant combinatorial simplification anyway (*cf* [Med94]).

5 The use of Genetic Algorithms

These combinatorial difficulties lead us to try on this problem a Genetic Algorithm, along with a linear optimization algorithm, such as the "simplex" algorithm.

5.1 The algorithms

To solve the linear problems presented above, we use a program called *lp_solve*, elaborated by the *Design Automation Section, Eindhoven University of Technology.* It uses a *simplex* algorithm, such as the one described in [OH68].

The Genetic Algorithm we use is similar to the ones that are described in [Gol89] and [Mic92]: The elements of the initial population are randomly generated. Then each element is evaluated, by the computation of its *fitness* (the more the element is adapted to the problem to solve the higher, its fitness is). Then, mutation and crossover are randomly applied to the population elements, with probabilities P_c and P_m. At this point, a new population is created, and this process is repeated. At each iteration, a new population is obtained. An iteration is called a *generation.*

To avoid premature convergence toward a local optimum we use the *sharing* technique, described in [GY].

5.2 The encoding

The chromosome of an element of population codes a situation from which the linear optimization will be started: for each aircraft, the offset direction, and for each pair of aircraft, which one passes behind the other. This is coded by an sequence of $\frac{n(n+1)}{2}$ bits, treated as an integer (or as several integers if n is large): the n first bits code the offset directions for the n aircraft (1 for a deviation to the left, 0 for a deviation to the right), and the $\frac{n(n-1)}{2}$ bits, for each of the $\frac{n(n-1)}{2}$ pair of aircraft (a_i, a_j), with $i < j$, which aircraft passes behind the other (1 if a_i passes behind a_j, 0 otherwise).

All the chromosomes built this way do not correspond to solutions of the collision avoidance problem. They code for linear problems, some of which are unfeasible (*i.e.* have no solution), as we've already seen it. But we have no way of knowing *a priori* which element of the population codes an unfeasible linear problems. We will see in the remainder of this article how these elements will be treated.

The mutation operator is classical: first, an element is chosen in the population, with probability P_m ; then, one of the $\frac{n(n+1)}{2}$ bits of its chromosome is randomly chosen, and modified.

The crossover operator is the classical uniform crossover operator. Two element are chosen in the population, with probability P_c, to be the "parents". The parents' bits are randomly distributed between the two "children". To accelerate the algorithm's convergence, we use a technique described by Samir Mahfoud and David Goldberg in [MG92], which is inspired by the Simulated Annealing algorithm. Each child is compared to its best parent. If the child's fitness is higher, it takes its parent's place in the population. If the child's fitness is lower, it takes its poarent's place with probability of law $e^{-\frac{\Delta}{T}}$, where Δ is the difference between the parent's fitness and the child's one, T decreases along the generations.

5.3 Implementation

Each chromosome codes for each aircraft, the offset direction, and for each pair of aircraft, which one passes behind the other. We will say that a chromosome codes for a *configuration* of the collision avoidance problem. As we have already stated, for a fixed configuration, the optimal collision avoidance trajectories are determined by the offset values, which are solutions of a linear optimization problem. We can thus compute, for each element corresponding to a feasible linear problem, the n offset values leading to the shortest delay, with a linear optimization program, and then compute this delay.

It is natural to consider that the shorter this delay is, the better the corresponding configuration will be, and to take for the fitness of an element a decreasing function of the delay given by the linear program applied with the configuration coded by the element's chromosome. However, some configurations might lead to unfeasible linear problems, *i.e.* problems for which all constraints cannot be satisfied at the same time. These configurations might be numerous, and if the fitness of all corresponding elements is set to zero, the genetic algorithm would lose much of its efficiency.

We want to evaluate these elements, to be able to compare one to the other. A linear optimization problem is unfeasible when all of its constraints cannot be satisfied simultaneously. We considered that a chromosome coding an unfeasible problem is better than another one, when more of its constraints can be satisfied simultaneously. So we evaluated these elements the following way: constraints of the linear problem are removed one by one and the linear optimization program is applied after each constraint's removal until the problem is feasible. The constraints to be removed are chosen randomly, to avoid the side effects of possible symmetries. The chromosome's fitness is then given by a decreasing function of the number of constraints that had to be removed to make the problem feasible.

The two different function used for the evaluation of the population's elements have to be adjusted in order to ensure that any elements corresponding to an unfeasible linear problem has a lower fitness than the ones which correspond to feasible problems.

We actually used a slightly different technique to evaluate the elements corresponding to unfeasible problems, which is better adapted to the collision avoidance problem and more efficient. Instead of removing constraints one by one, we remove them "aircraft by aircraft": all the constraints in the coefficient of which a given offset value appears are removed at a time. The results presented in the next section were obtained using this technique.

6 Results

It is difficult to evaluate the results we obtained, as it is difficult to obtain, with another optimization technique a solution known to be optimal we could compare with our best solution. For a conflict involving up to six aircraft however, it is still possible to treat all the possible configurations with a linear optimization

program, and to determine which one leads to the global optimum. For a six aircraft conflict, this takes a little more than three hours (there are 2 097 152 different configurations).

We consider the following situation: all the aircraft fly at the same speed (400 knots), at time t_o they are regularly distributed on a semi circle centered in C, of radius 100 nautical miles, and their initial trajectories gather in C (see figure 5). Note that the symmetry of this situation, useful for a man to understand it and make sure there actually is a conflict, do not change the way the machine handles it.

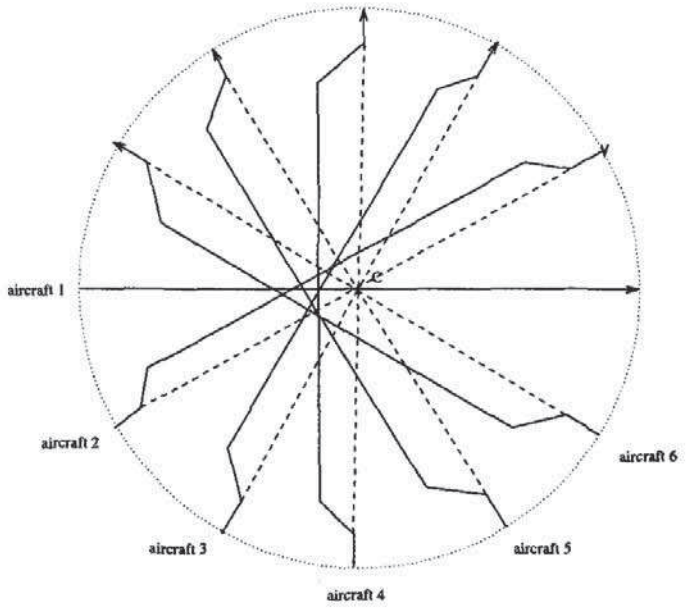

Fig. 5. *Optimal collision avoidance system for a six aircraft conflict.*

There are two equivalent optimal solutions: either the five first aircraft are deviated to the right, the sixth one goes straight, and for $i < j$, a_i passes behind a_j, or the first aircraft goes straight, the five others are deviated to the left, and for $i < j$, a_j passes behind a_i. We represented this latter solution on figure 5.

Our algorithm found these two optimal solutions. We can compare, for a conflict involving 6 aircraft the time needed to compute the delays corresponding to all the configurations and the time required by our algorithm to reach the optimum.

We ran our program 50 times on the situation presented above. We used populations of 150 elements, and the number of generation was fixed to 98. 39 of the 50 runs lead to one of the two optimal solutions. The distribution of the number of generation before one of the two optimal solution was found, over the 50 runs, is shown on table 1. For instance we found an optimal solution in less than 20 generations 16 times.

The number of generations required to reach an optimal solution is not very meaningful. We present in table 3 the mean value and the standard deviation,

Number of runs	Number of generations needed
16	0 - 20
7	20 -40
7	40 - 60
7	60 - 80
2	80 - 98
11	no optimal solution found in less than 98 generations

Table 1. distribution of the number of generation before an optimal solution is found

Number of runs	Number of calls to the linear optimization program needed
2	0 - 10 000
18	10 000 - 20 000
8	20 000 -30 000
7	30 000 - 40 000
3	40 000 - 50 000
1	50 000 - 55 000
11	no optimal solution found in less than 98 generations

Table 2. Distribution of the number of calls to the linear optimization program needed before an optimal solution is found

over the 39 successful runs, of the CPU time, the number of elements evaluations (Nb of evaluations), and the number of calls to the linear optimization program (Nb of calls), required to reach an optimal solution. Table 2 shows the distribution of this number of calls to the linear optimization program. An exhaustive search would require 2 097 152 calls to this program.

We obtained these results on a HP 720 station.

There are about 250 evaluations at each generation (each of the 150 elements is evaluated, and some other evaluations are needed by the crossover operator).

For each of the first generations, up to as many as 1500 calls to the linear optimization program may be required, whereas this number later decreases to less than 300. Indeed, the evaluation of an element corresponding to an infeasible linear optimization problem requires several calls to the linear optimization program (one after each removal of a group of constraints, the constraints being removed "aircraft by aircraft"), until the problem is feasible. Generation by generation, the elements of population get globally better, the number of elements

	Mean value	Standard deviation
Time (s)	99.0	51.8
Nb of evaluation	9696	6707
Nb of calls	23193	11818

Table 3. Number of calls to the linear optimization program

that correspond to unfeasible problems decreases, and the mean value of the number of constraints to remove also decreases.

With the numerical values we have used, the two optimal solutions led to a 9,6 minute global delay. Each time no optimal solution was found (11 runs out of 50), an under-optimal solution was found, leading to a 10,9 minute global delay.

7 Conclusion

Such a technique is useful to solve only highly combinatorial problems. For a conflict involving three aircraft, only 64 calls to the linear optimization program are required for an exhaustive search, and only 1 024 are required for a conflict involving four aircraft. In these cases an exhaustive search will clearly be faster than the technique presented here.

This technique allows to join the advantages of linear optimization (a quick and efficient search for a local optimum) to those of genetic algorithms, which help finding global optima, and often propose several different solutions to one problem, which is of great interest if we want to help controllers by proposing possible collision avoidance trajectories among which they will be able to choose, rather than completely automate air traffic control.

The work we have presented here is a theoretical approach of the resolution of ATC conflicts. We have made some strong hypotheses (plane trajectories and conflicts, constant speed, separation conditions for all t and not only for $t \in [t_o, t_f]$, etc). If a practical use of this technique is wanted, these hypotheses' strength will have to be reduced (*cf* [Med94]).

These results may probably be made better, by adjusting the coefficient that are used by the algorithm (for instance to evaluate the elements). They are nevertheless already really promising.

References

[AGS93] Jean-Marc Alliot, Hervé Gruber, and Marc Schoenauer. Using genetic algorithms for solving ATC conflicts. In *Proceedings of the Ninth IEEE Conference on Artificial Intelligence Application.* IEEE, 1993.

[AL92] Luc Angerand and Hervé LeJeannic. Bilan du projet SAINTEX. Technical report, CENA, 1992. CENA/R92009.

[DAAS94] Nicolas Durand, Nicolas Alech, Jean-Marc Alliot, and Marc Schoenauer. Genetic algorithms for optimal air traffic conflict resolution. In *Submitted to the Second Singapore Conference on Intelligent Systems.* SPICIS, 1994.

[DASF94a] Daniel Delahaye, Jean-Marc Alliot, Marc Schoenauer, and Jean-Loup Farges. Genetic algorithms for partitioning airspace. In *Proceedings of the Tenth Conference on Artificial Intelligence Application.* CAIA, 1994.

[DASF94b] Daniel Delahaye, Jean-Marc Alliot, Marc Schoenauer, and Jean-Loup Farges. Genetic algorithms for air traffic assignment. In *Proceedings of the EuropeanConference on Artificial Intelligence.* ECAI, 1994.

[DDAS94] Daniel Delahaye, Nicolas Durand, Jean-Marc Alliot, and Marc Schoenauer. Genetic algorithms for air traffic control system. *soumis à IEEE Expert '94*, 1994.

[Dur94] Nicolas Durand. Modélisation des trajectoires d'évitement pour la résolution de conflits en route. Technical report, Centre d'Etudes de la Navigation Aérienne, Février 1994.

[FMT93] Xavier Fron, Bernard Maudry, and Jean-Claude Tumelin. Arc 2000 : Automatic radar control. Technical report, Eurocontrol, 1993.

[Gol89] David Goldberg. *Genetic Algorithms.* Addison Wesley, 1989. ISBN: 0-201-15767-5.

[GY] Noel Germay and Xiaodong Yin. A fast genetic algorithm with sharing scheme using cluster analysis methods in multimodal function optimization. Technical report, Université Catholique de Louvain, Laboratoire d'Electronique et d'Instrumentation.

[K+89] Fred Krella et al. Arc 2000 scenario (version 4.3). Technical report, Eurocontrol, April 1989.

[Med94] Frédéric Medioni. Algorithmes génétiques et programmation linéaire appliqués à la résolution de conflits aériens. Mémoire de dea, Ecole Polytechnique, Ecole Nationale de l'Aviation Civile, Juillet 1994.

[MG92] Samir W. Mahfoud and David E. Goldberg. Parallel recombinative simulated annealing: a genetic algorithm. IlliGAL Report 92002, University of Illinois at Urbana-Champaign, 104 South Mathews Avenue Urbana IL 61801, April 1992.

[Mic92] Zbigniew Michalewiicz. *Genetic algorithms + data structures = evolution programs.* Springer-Verlag, 1992. ISBN : 0-387-55387.

[NFC+83] W.P. Niedringhaus, I. Frolow, J.C. Corbin, A.H. Gisch, N.J. Taber, and F.H. Leiber. Automated En Route Air Traffic Control Algorithmic Specifications: Flight Plan Conflict Probe. Technical report, FAA, 1983. DOT/FAA/ES-83/6.

[Nie89a] W.P. Niedringhaus. Automated planning function for AERA3: Manoeuver Option Manager. Technical report, FAA, 1989. DOT/FAA/DS-89/21.

[Nie89b] W.P. Niedringhaus. A mathematical formulation for planning automated aircraft separation for AERA3. Technical report, FAA, 1989. DOT/FAA/DS-89/20.

[OH68] W. Orchard-Hays. *Advanced Linear Programming Computing Techniques.* McGraw-Hill, 1968.

[Zeg93] Karim Zeghal. Techniques réactives pour l'évitement. Technical report, ONERA, June 1993.