Centrum voor Wiskunde en Informatica

REPORT*RAPPORT*

Evolutionary computation in air traffic control planning

C.H.M. van Kemenade, C.F.W Hendriks, H.H. Hesselink, and J.N. Kok

Computer Science/Department of Software Technology

CS-R9550 1995

Report CS-R9550
ISSN 0169-118X

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

Evolutionary Computation in Air Traffic Control Planning

C.H.M. van Kemenade

CWI
P.O. Box 94079, 1090 GB Amsterdam, The Netherlands
kemenade@cwi.nl

C.F.W. Hendriks and H.H. Hesselink

Informatics division
National Aerospace Laboratory NLR
Anthony Fokkerweg 2, 1059 CM Amsterdam, The Netherlands
{hendr,hessel}@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

Air Traffic Control is involved in the real-time planning of aircraft trajectories. This is a heavily constrained optimization problem. We concentrate on free-route planning, in which aircraft are not required to fly over way points. The choice of a proper representation for this real-world problem is non-trivial. We propose a two level representation: one level on which the evolutionary operators work, and a derived level on which we do calculations. Furthermore we show that a specific choice of the fitness function is important for finding good solutions to large problem instances. We use a hybrid approach in the sense that we use knowledge about air traffic control by using a number of heuristics. We have built a prototype of a planning tool, and this resulted in a flexible tool for generating a free-route planning of low cost, for a number of aircraft.

AMS Subject Classification (1991): 68T20
CR Subject Classification (1991): G.1.7, I.2.8, J.m
Keywords & Phrases: evolutionary computation, genetic algorithms, constrained optimization, air traffic control
Note: Paper is to be presented at the Sixth International Conference on Genetic Algorithms, Pittsburgh 1995

1. Introduction

Air Traffic Control (ATC) is concerned with real-time planning of aircraft trajectories. A trajectory describes the position of the aircraft as a function of time. The shape of trajectories is heavily constrained. The most important constraints are the separation standards, as stated by the ICAO (Internation Civil Aviation Organisation). For example, according to these standards the minimal distance between two aircraft should be at least 8 nm[1] ($\approx$ 14800 *meters*) if they fly in the same horizontal plane. If they do not fly in the same horizontal plane, they should be vertically separated. Two aircraft flying below 29000 ft[2] ($\approx$ 8839 *meters*) are said to be vertically separated if the vertical distance between these aircraft is at least 1000 ft ($\approx$ 305 *meters*). There is also a length separation

[1] 1 nautical mile = 1,852 meters
[2] 1 foot = 0.3048 meters

rule, which states: if two aircraft follow the same path, then for every point along the path there should be at least a five minute separation between the times that the aircraft pass a certain point on the path. Aircraft are said to be in conflict if at least one of these separation standards is violated.

Currently ATC planning is mainly a human activity. Though sophisticated tools for monitoring meteorological conditions, locating aircraft and for communication exist, the actual planning is still mainly done manually. Due to the increasing volume of air traffic, new automated tools to assist the controller in making a planning become necessary.

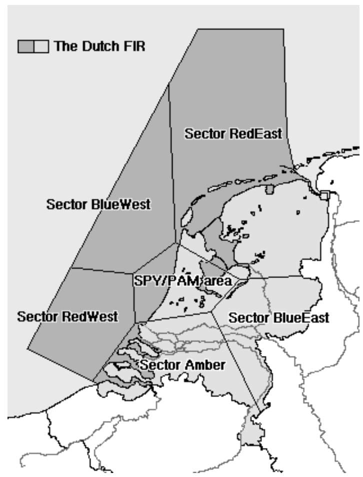

Figure 1: A division of the airspace above the Netherlands in sectors

To be able to handle the situation and to reduce the probability of errors, often an artificial structure is imposed on the airspace. As an example Figure 1 shows the sectors in the airspace above the Netherlands. The airspace is divided in sectors of approximately 200 $\times$ 200 kilometers. In these sectors a limited number of way points is introduced. Typically there is just a number of way points at the boundary and a number of way points in the center of a sector. An aircraft is assumed to fly in an approximately straight line between way points. Figure 2 shows an example of an aircraft trajectory defined by the three filled dots.

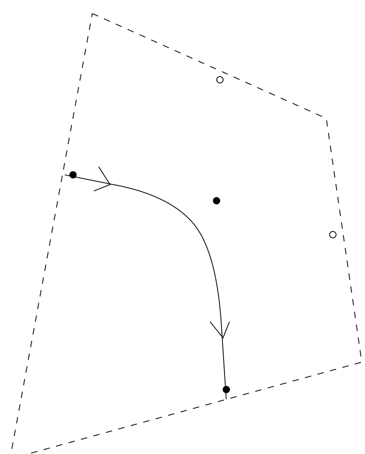

Figure 2: A traditional ATC sector

An alternative is the free-route planning in which aircraft trajectories are not restricted to go via way points. This results in an increase of the number of possible trajectories, and can increase the capacity of a sector. However, planning can become more difficult because there are much more possibilities for the routes of aircraft, i.e. there is a substantial larger degree of freedom.

In this article we consider free-route planning based on evolutionary computation. The free-route planning problem is assumed to be a hard constrained optimization problem. All aircraft trajectories interact by means of the separation standards. Apart from the separation standards there are many other constraints, for example concerning the shape of the trajectory because characteristics of the aircraft determine bounds upon velocity, acceleration, and total distance traveled. There can also be restrictions on the available airspace (for example that aircraft are not allowed to fly in military airspace). It is difficult in free-route planning to balance the importance of constraints and objectives that a good plan should satisfy. One of the central objectives is to find a conflict free planning of minimal cost. This means that the total traveled distance of all aircraft should be minimized under the restriction that no new conflicts are introduced. A certain degree of flexibility in planning tools for free-route planning is desired. For example, a tool should offer a way to handle soft constraints. A characteristic of the free-route problem is its dynamic nature. New aircraft can appear or planned aircraft can deviate from their planned trajectory. Both these situations have to be handled gracefully: the planning has to be adjusted on the fly, and the new planning should not be too far away from the previous planning. It is also desirable that a tool can create alternative plans, among which a human controller can choose. It should be adaptive in the sense that it can cope with additional constraints imposed by a human controller.

Concluding, free route planning is a highly complex problem with numerous constraints.

2. Evolutionary Free-Route Planning

One of the reasons for basing our tool on an evolutionary approach is that evolutionary computation is already applied successfully to a variety of difficult constrained optimization problems, such as Job Shop scheduling problems, see for example [Nak91] and [BUMK91]. Furthermore the evolutionary approach has been shown to be a robust optimization method that can handle a large variety of constraints. Other efforts in dealing with ATC planning problems can be found in [AGJS93] and [Ger94]. Both handling with lower air traffic densities than used in the tests presented in this paper.

In a standard evolutionary algorithm all problem-specific knowledge is incorporated in the fitness function. Our approach is a hybrid one, in the sense that part of the knowledge is put in the operators and the representation. We use a hybrid approach for the following reasons:

- some subproblems of free-route planning can be solved easily by means of deterministic algorithms,
- the fitness function becomes simpler as part of the constraints are enforced already by the system, and
- it yields a significant reduction of the size of the search space.

There are a number of constraints, like the characteristics of the aircraft and geographical restrictions, that involve a single aircraft trajectory and that are relatively simple. In order to handle these constraints we introduce an abstract representation of trajectories. First we define a set M of parameterized maneuvers:

$$M = \{straight(\mathrm{d}t), curve(\mathrm{d}\alpha), accelerate(\mathrm{d}v), altitude(\mathrm{d}h)\}$$

Each maneuver $m \in M$ has a parameter with (continuous) range R_m. The maneuver $straight(\mathrm{d}t)$ represents a straight line of duration $\mathrm{d}t$. The actual length of this piece of trajectory is determined by the velocity of the aircraft. (Though not strictly a maneuver, flying in a straight line is also added.) The $curve(\mathrm{d}\alpha)$ represents a change of direction of $\mathrm{d}\alpha$ degrees. The $accelerate(\mathrm{d}v)$ represents a change of velocity. A typical value for the acceleration of aircraft is used to determine the actual time it takes to perform the last two maneuvers. The $altitude(\mathrm{d}h)$ describes a change in altitude. Change of altitude is done at constant vertical velocity.

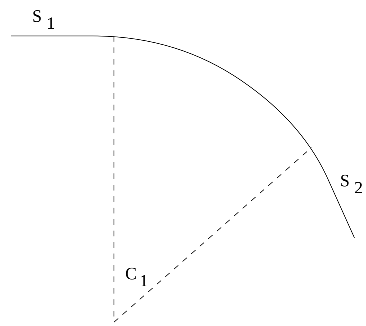

Figure 3: A simple trajectory

A trajectory is represented as a list of these maneuvers together with an appropriate value for the parameter of each maneuver. An example of a simple trajectory is shown in Figure 3. This trajectory is described by the list: $(straight(s1), curve(c1), straight(s2))$. All knowledge regarding the actual shape of the trajectory of an aircraft, performing a given list of maneuvers, is located in a mapping function which computes a list of connected straight lines representing the actual trajectory. Hence we have two representations for trajectories: one of which is a sequence of maneuvres, and a derived representation which is a list of connected lines. A complete planning is then represented by a set of these trajectories. These plannings are taken as individuals in a population. We use the following mutation operator. It acts directly upon the abstract representation and applies the following steps to an individual:

1. determine the time of the first conflict t_{conf},
2. select at random one of the aircraft involved in this conflict and extract its trajectory,
3. select at random the following parameters:
   - $t_{resolve} \in [0, t_{conf}]$,
   - $m \in M$,
   - $x_m \in R_m$, and
   - $t_{straight} \in [0, 300]$.
4. construct a trajectory according to the rule:

$$head(m, 0, t_{resolve}) \cdot m(x_m) \cdot straight(t_{straight}) \cdot m'$$

   (where $head(m, 0, t)$ denotes the part of trajectory m corresponding to interval $[0, t]$, the operator $\cdot$ is a concatenation operator, and m' is a list of maneuvers guiding the aircraft to its destination).

This mutation operator is conflict driven, instead of randomly driven. It modifies trajectories belonging to aircraft involved in the first detected conflict. This approach is chosen for two reasons. First the introduction of redundant maneuvers will result in abundant length of trajectories, and thus a lower quality of the planning. Furthermore the probability of new conflicts being introduced will be enlarged by these redundant maneuvers. The second reason is more sophisticated. Due to the separation standards, trajectories are interacting, and this results in the fitness of a trajectory being correlated to its context, which is determined by the surrounding trajectories. A modification at a certain time influences the future context of the trajectory. The first conflict first driven approach helps to reduce the probability that a mutation makes a previously added maneuver obsolete.

It is difficult to define a good recombination operator using the current representation. Mixing trajectories from different individuals does not make much sense, as trajectories are evolved in the context of surrounding trajectories. The quality of a trajectory is strongly related to this context.

Designing a proper fitness function for the free route planning problem is a challenging problem by itself. Often one defines a fitness function by taking the objective and maps this to a number in a straightforward way. In our case the objective is to find a conflict-free plan with a minimal traveled distance for each aircraft. A straightforward mapping to numbers could result in the following fitness function:

$$f(i) = -\#conflicts - \alpha \sum_{j \in i} \frac{\int_j \mathrm{d}x}{|dest_i - src_i|}.$$

The first term describes the total number of conflicts in the sector. In a conflict-free plan, this number is zero. The second term describes the relative excess distance summed over all aircraft in the sector. The factor α is introduced to balance the relative importance of conflicting objectives.

Although this fitness function does represent the objective, it does not result in a proper driving force for the evolution. The main problem is that conflicts need to be solved in the right order. One has to start with the first conflict (in time), and not with the last one. Intuitively, changing a later conflict is useless if the earlier ones are not solved. Hence to cope with the free-route planning problem a more advanced fitness function needs to be defined which guides the search in the proper direction.

The calculation of the fitness function is done in two stages. First the fitness of trajectory i within an individual is expressed as a value in range $[0, 1]$ using the following formula:

$$f_{traj}(i) = \omega_{conf} \frac{t_{conf}}{t_{plan}} + (1 - \omega_{conf})\, e^{-\omega_{detour}\left(\frac{\int_{traj} \mathrm{d}x}{|dest - src|} - 1\right)}$$

The first term represents the relative time before the first conflict with any other trajectory within the individual arises. The second term results in a penalty when a detour is taken. This penalty is zero in case of a straight trajectory from *src* to *dest* and one in case of an infinitely large detour. The exponential is needed to perform a scaling of this penalty term. A solution where a number of aircraft make a small detour is preferable above a solution where a single aircraft makes a large detour. For fairness and to prevent aircraft form running out of fuel.

The fitness of a individual j is taken equal to be a weighted sum over the fitness of all trajectories within the individual according to the formula:

$$f(j) = \sum_{i \in j} \omega_i \cdot \mathrm{f}_{traj}(i)$$

under the restrictions $\omega_i \geq 0$ and $\sum \omega_i = 1$. A proper choice of the weights ω_i is important in order to get proper convergence. Three possible choices for these weights are considered:

1. Set $\omega_i = 1$ for the least fit trajectory and for all other trajectories j with $j \neq i$ set $\omega_j = 0$. This corresponds to observing the worst performing trajectory only,
2. Set $\omega_i = \frac{1}{\#aircraft}$ for all trajectories i. This corresponds to taking the average over all trajectories.

3. Use a fitness based weighting, where bad performing trajectories have a relatively large influence on the fitness of a planning:

$$\omega_i = \frac{W_i}{\sum_j W_j}$$

$$\text{where } W_i = e^{-\alpha \cdot f_{traj}(i)} \text{ and } \alpha \geq 0.$$

The third method can be seen as a generalisation of the other two methods. By setting $\alpha = 0$ all weights ω_i are made equal, which corresponds to the second method. As α gets larger, more emphasis is put upon the bad performing trajectories. In the limit $\alpha \to \infty$, this method will correspond to the first method, where the fitness of a plan is completely determined by its worst performing trajectory.

This two-phase computation of the fitness of a plan has an important additional advantage. Calculating the fitness value for a trajectory is one of the most time-consuming parts of the complete algorithm. As most trajectories appear in more than one individual a significant amount of time is saved by applying an incremental calculation scheme, which uses stored partial results.

The fitness function puts much emphasis on nearby conflicts. As the first conflict is moved in time the fitness function gets less sensitive to the actual time a conflict arises. When applying a fitness proportional selection scheme this will result in less selective pressure when the first conflict moves ahead in time and the fitness increases. In order to avoid this kind of scaling problem a ranking scheme is used [Bak85, GD91]. In a ranking scheme the individuals are sorted and ranked on fitness. The probability of being selected for reproduction is coupled to the rank. By adjusting the probability of survival the selective pressure can be set. Ranking is often combined with a steady-state algorithm [Sys91, Whi89]. Such a steady-state algorithm replaces just a small part of the population during each iteration. Hence the rank will be recalculated often and a more aggressive search is obtained.

As the search space is very large and it is assumed to contain many good solutions, an aggressive search method seems to be appropriate. The algorithm is terminated if it does not enhance its best solution for more than τ iterations. In our experiments we used $\tau = 100$.

3. Experiments

All experiments involve the creation of a planning for a two-dimensional square area of 200×200 kilometers. (This limitation is purely for convenience and does not correspond to a conceptual limitation of the model.)

Three different fitness functions are compared on a set of 24 random problems. The three fitness functions we compare are the fitness based weighting with values $\alpha = 0, 6, \infty$. For each test problem 40 independent runs were performed for each fitness function, resulting in a total of 960 runs for each point in the graph.

A single test problem is generated by selecting for each aircraft an entrance time, entrance position and exit position at the boundary of the sector, all at random. The entrance and exit positions should be on different sides of the sector. A test problem constructed according to these rules is accepted if it does not have any initial conflicts. A conflict is called an initial conflict if both aircraft involved in the conflict have just entered the sector. A typical result of a free-route planning involving 20 aircraft is shown in figure 4.

Figure 5 shows the rate of success of the method as a function of the number of aircraft in the sector. A single run is successful if a planning is constructed such that:

- the planning is free of conflicts, and
- all aircraft leave the sector at their exit position.

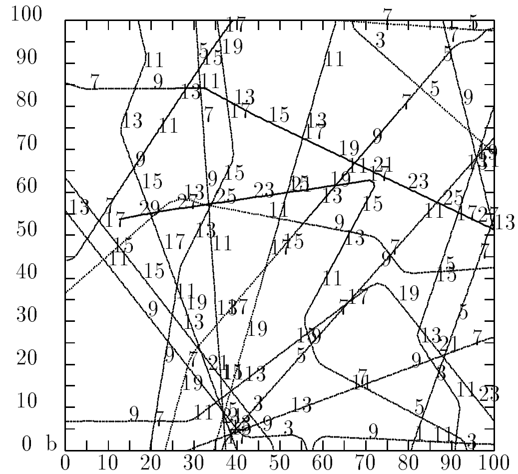

Figure 4: A typical example of a free-route planning. The number inside the sector are an indication of time.

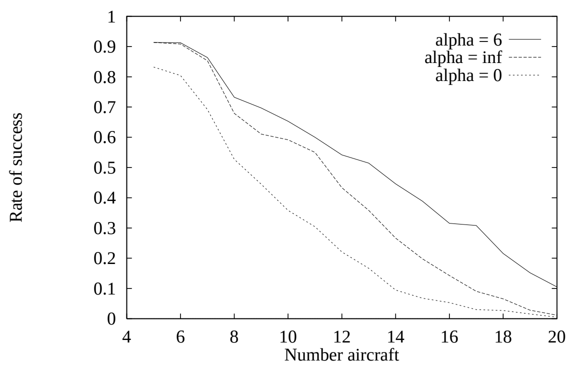

Figure 5: Rate of success

The second item is checked in order to prevent solutions where part of a trajectory is outside the sector.

For each successful run the average deviation from the shortest path is calculated, where the length of the shortest possible path is assumed to be the Euclidean distance between the entrance and the exit point. So a deviation of 1.1 corresponds to 10% extra distance traveled relative to the shortest path. Figure 6 shows the average deviation over all successful runs as a function of the number of aircraft in the sector.

Another important measure is the deviation of the worst performing aircraft in a plan. If this deviation is large compared to the average deviation a plan is said to be unfair. If a plan is not fair, this might result in aircraft deviating from their estimated time of arrival, or even worse, aircraft running out of fuel. Figure 7 shows the deviation of the worst performing aircraft averaged over all successful runs.

From the three compared fitness functions the function with $\alpha = 0$ method is consistently the worst. This method has the lowest rate of success, as can be seen from figure 5. Furthermore the relative

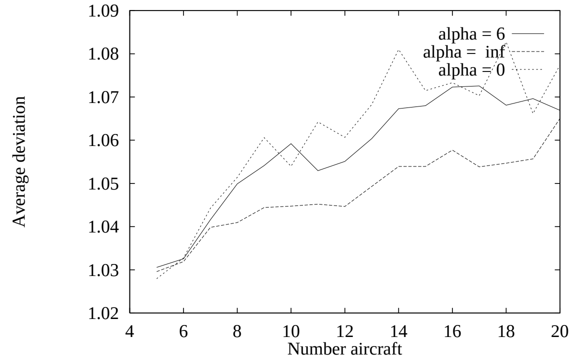

Figure 6: Average deviation of aircraft from shortest path

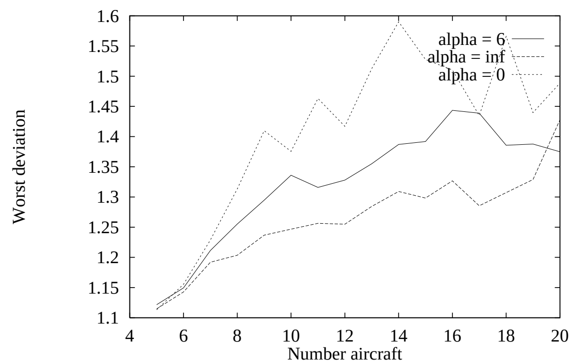

Figure 7: Deviation of worst performing aircraft

deviations from the shortest path are the largest when using $\alpha = 0$, as can be seen from figure 6 and 7. This low performance for $\alpha = 0$ is a result of the low selective pressure enforced by this method. A closer look at the plans produced by this method reveals the reason of this bad performance. Often a single aircraft is sent in the wrong direction, thereby reducing the size of the problem. The remaining conflicts are solved relatively fast, but then it seems to be impossible to guide the last aircraft back to its exit position without making drastic changes to many other trajectories, so the problem can not be solved completely any more. The problem is that mutations which are good in the short term are often bad in the long run. This effect becomes more dominant as the number of aircraft in the sector increases. This corresponds to the relative fast drop of the rate of success as the number of aircraft increases.

Taking $\alpha = \infty$ is much better. The rate of success of this method is higher than that of $\alpha = 0$, as can be seen from figure 5. Unfortunately the rate of success drops fast as the number of aircraft increases. When $\alpha = \infty$, then the value of the fitness function is almost completely determined by the time of the first conflict. The other conflicts within the sector do not have any influence on the fitness value. This lack of differentiation between different plans having the same first conflict is assumed to be the reason for the fast drop of rate of success when increasing the number of aircraft for $\alpha = \infty$.

Further evidence for this is the low number of iterations used when $\alpha = \infty$. Fast convergence often corresponds to bad exploration of the search space.

Taking $\alpha = 6$ has the overall highest rate of success. This method is somewhere in between the previous two methods. It puts a relative high emphasis on the worst trajectory just like for $\alpha = \infty$. The advantage of taking $\alpha = 6$ is that it does differentiate between different plans that have the same first conflict, because all trajectories influence the fitness value.

When looking at the quality of the solutions in case of success, preference should be given to $\alpha = \infty$. This method results in the lowest deviations from the shortest path, as can be seen from figures 6 and 7. When taking rate of success and quality into consideration it seems best to use $\alpha = \infty$ if the number of aircrafts is below 14 and use $\alpha = 6$ otherwise.

4. Conclusions and Further Work

In the Evolutionary Computation community a lot of research is done on artificial test problems, constructed to show certain specific effects. Real world problems are often more diverse than the most advanced test suite can capture. For example, in our case it was non-trivial to map our problem to a representation, which could be handled by an evolutionary algorithm. Furthermore it was important to make a clear distinction between the actual objective and the fitness function. The final result of the planner has to be tested against the objectives to measure its quality. The fitness function on the other hand has to be streamlined to guide the search in the proper direction.

Due to the lack of a sufficient number of real-world test cases and other methods to handle this planning problem, it is difficult to determine the relative strength of the evolutionary computation method. Taking the difficulty of this kind of planning problems into account, the results are promising. A solution of good quality can be obtained for problems involving up to 20 aircraft, within a reasonable amount of time.

When allowing up to 14 aircraft in the sector results of even better quality can be obtained, by using a different value of α.

Further research is needed to find a good recombination operator. We also like to show the robustness of the method with respect to changes in the planning. This robustness is important, as the system has to be able to cope with additional constraints enforced by the human controller.

Concluding, the evolutionary approach has resulted in a successful planner for free-route planning. New constraints can be incorporated easily by modifying the fitness function. Due to this kind of flexibility the model is a valuable tool for the free-routing problem.

5. Acknowledgements

We like to thank dr. Henk Blom from the National Aerospace Laboratory NLR for his helpful comments.

References

[AGJS93] J. Alliot, H. Gruber, G. Joly, and M. Schoenauer. Genetic algorithms for solving air traffic control conflicts. In 9^{th} *conference on AI Applications*, 1993.

[Bak85] J.E. Baker. Adaptive selection methods for genetic algorithms. In *First International Conference on Genetic Algorithms*, 1985.

[BUMK91] S. Bagchi, S. Uckun, Y. Miyabe, and K. Kawamura. Exploring problem-specific recombination operators for the job shop scheduling. In *Fourth International Conference on Genetic Algorithms*, 1991.

[GD91] D.E. Goldberg and K. Deb. A comparative analysis of selection schemes used in genetic algorithms. In *Foundations of Genetic Algorithms - 1*, 1991.

[Ger94] I.S. Gerdes. Application of genetic algorithms to the problem of free-routing for aircraft. In *First IEEE conference on Evolutionary Computation*, 1994.

[Nak91] R. Nakano. Conventional genetic algorithms for job shop problems. In *Fourth International Conference on Genetic Algorithms*, 1991.

[Sys91] G. Syswerda. A study of reproduction in generational and steady–state genetic algorithms. In G. Rawlins, editor, *Foundations of Genetic Algorithms - 1.* Morgan Kaufmann, 1991.

[Whi89] D. Whitley. The genitor algorithm and selective pressure. In *Third International Conference on Genetic Algorithms*, pages 116–121, 1989.

Centrum voor Wiskunde en Informatica
REPORT*RAPPORT*

Evolutionary computation in air traffic control planning

C.H.M. van Kemenade, C.F.W Hendriks, H.H. Hesselink, and
J.N. Kok

Computer Science/Department of Software Technology

CS-R9550 1995

Report CS-R9550
ISSN 0169-118X

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

Evolutionary Computation in Air Traffic Control Planning

C.H.M. van Kemenade

CWI
P.O. Box 94079, 1090 GB Amsterdam, The Netherlands
kemenade@cwi.nl

C.F.W. Hendriks and H.H. Hesselink

Informatics division
National Aerospace Laboratory NLR
Anthony Fokkerweg 2, 1059 CM Amsterdam, The Netherlands
{hendr,hessel}@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

Air Traffic Control is involved in the real-time planning of aircraft trajectories. This is a heavily constrained optimization problem. We concentrate on free-route planning, in which aircraft are not required to fly over way points. The choice of a proper representation for this real-world problem is non-trivial. We propose a two level representation: one level on which the evolutionary operators work, and a derived level on which we do calculations. Furthermore we show that a specific choice of the fitness function is important for finding good solutions to large problem instances. We use a hybrid approach in the sense that we use knowledge about air traffic control by using a number of heuristics. We have built a prototype of a planning tool, and this resulted in a flexible tool for generating a free-route planning of low cost, for a number of aircraft.

AMS Subject Classification (1991): 68T20
CR Subject Classification (1991): G.1.7, I.2.8, J.m
Keywords & Phrases: evolutionary computation, genetic algorithms, constrained optimization, air traffic control
Note: Paper is to be presented at the Sixth International Conference on Genetic Algorithms, Pittsburgh 1995

1. Introduction

Air Traffic Control (ATC) is concerned with real-time planning of aircraft trajectories. A trajectory describes the position of the aircraft as a function of time. The shape of trajectories is heavily constrained. The most important constraints are the separation standards, as stated by the ICAO (Internation Civil Aviation Organisation). For example, according to these standards the minimal distance between two aircraft should be at least 8 nm[1] ($\approx$ 14800 *meters*) if they fly in the same horizontal plane. If they do not fly in the same horizontal plane, they should be vertically separated. Two aircraft flying below 29000 ft[2] ($\approx$ 8839 *meters*) are said to be vertically separated if the vertical distance between these aircraft is at least 1000 ft ($\approx$ 305 *meters*). There is also a length separation

[1] 1 nautical mile = 1,852 meters
[2] 1 foot = 0.3048 meters

rule, which states: if two aircraft follow the same path, then for every point along the path there should be at least a five minute separation between the times that the aircraft pass a certain point on the path. Aircraft are said to be in conflict if at least one of these separation standards is violated.

Currently ATC planning is mainly a human activity. Though sophisticated tools for monitoring meteorological conditions, locating aircraft and for communication exist, the actual planning is still mainly done manually. Due to the increasing volume of air traffic, new automated tools to assist the controller in making a planning become necessary.

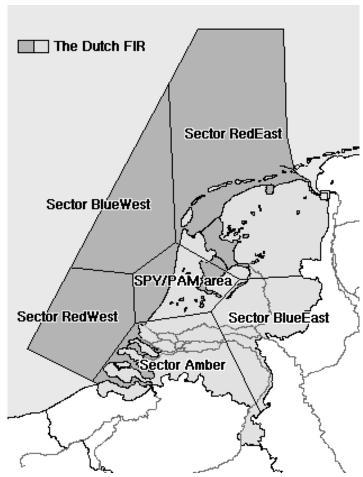

Figure 1: A division of the airspace above the Netherlands in sectors

To be able to handle the situation and to reduce the probability of errors, often an artificial structure is imposed on the airspace. As an example Figure 1 shows the sectors in the airspace above the Netherlands. The airspace is divided in sectors of approximately 200 $\times$ 200 kilometers. In these sectors a limited number of way points is introduced. Typically there is just a number of way points at the boundary and a number of way points in the center of a sector. An aircraft is assumed to fly in an approximately straight line between way points. Figure 2 shows an example of an aircraft trajectory defined by the three filled dots.

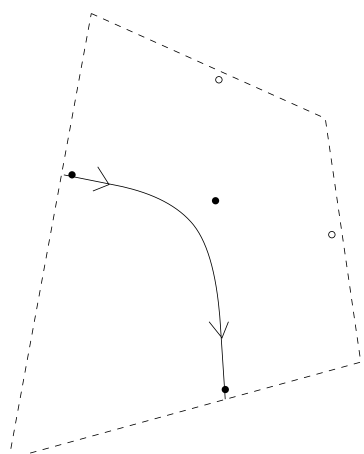

Figure 2: A traditional ATC sector

An alternative is the free-route planning in which aircraft trajectories are not restricted to go via way points. This results in an increase of the number of possible trajectories, and can increase the capacity of a sector. However, planning can become more difficult because there are much more possibilities for the routes of aircraft, i.e. there is a substantial larger degree of freedom.

In this article we consider free-route planning based on evolutionary computation. The free-route planning problem is assumed to be a hard constrained optimization problem. All aircraft trajectories interact by means of the separation standards. Apart from the separation standards there are many other constraints, for example concerning the shape of the trajectory because characteristics of the aircraft determine bounds upon velocity, acceleration, and total distance traveled. There can also be restrictions on the available airspace (for example that aircraft are not allowed to fly in military airspace). It is difficult in free-route planning to balance the importance of constraints and objectives that a good plan should satisfy. One of the central objectives is to find a conflict free planning of minimal cost. This means that the total traveled distance of all aircraft should be minimized under the restriction that no new conflicts are introduced. A certain degree of flexibility in planning tools for free-route planning is desired. For example, a tool should offer a way to handle soft constraints. A characteristic of the free-route problem is its dynamic nature. New aircraft can appear or planned aircraft can deviate from their planned trajectory. Both these situations have to be handled gracefully: the planning has to be adjusted on the fly, and the new planning should not be too far away from the previous planning. It is also desirable that a tool can create alternative plans, among which a human controller can choose. It should be adaptive in the sense that it can cope with additional constraints imposed by a human controller.

Concluding, free route planning is a highly complex problem with numerous constraints.

2. Evolutionary Free-Route Planning

One of the reasons for basing our tool on an evolutionary approach is that evolutionary computation is already applied successfully to a variety of difficult constrained optimization problems, such as Job Shop scheduling problems, see for example [Nak91] and [BUMK91]. Furthermore the evolutionary approach has been shown to be a robust optimization method that can handle a large variety of constraints. Other efforts in dealing with ATC planning problems can be found in [AGJS93] and [Ger94]. Both handling with lower air traffic densities than used in the tests presented in this paper.

In a standard evolutionary algorithm all problem-specific knowledge is incorporated in the fitness function. Our approach is a hybrid one, in the sense that part of the knowledge is put in the operators and the representation. We use a hybrid approach for the following reasons:

- some subproblems of free-route planning can be solved easily by means of deterministic algorithms,
- the fitness function becomes simpler as part of the constraints are enforced already by the system, and
- it yields a significant reduction of the size of the search space.

There are a number of constraints, like the characteristics of the aircraft and geographical restrictions, that involve a single aircraft trajectory and that are relatively simple. In order to handle these constraints we introduce an abstract representation of trajectories. First we define a set M of parameterized maneuvers:

$$M = \{straight(\mathrm{d}t), curve(\mathrm{d}\alpha), accelerate(\mathrm{d}v), altitude(\mathrm{d}h)\}$$

Each maneuver $m \in M$ has a parameter with (continuous) range R_m. The maneuver $straight(\mathrm{d}t)$ represents a straight line of duration $\mathrm{d}t$. The actual length of this piece of trajectory is determined by the velocity of the aircraft. (Though not strictly a maneuver, flying in a straight line is also added.) The $curve(\mathrm{d}\alpha)$ represents a change of direction of $\mathrm{d}\alpha$ degrees. The $accelerate(\mathrm{d}v)$ represents a change of velocity. A typical value for the acceleration of aircraft is used to determine the actual time it takes to perform the last two maneuvers. The $altitude(\mathrm{d}h)$ describes a change in altitude. Change of altitude is done at constant vertical velocity.

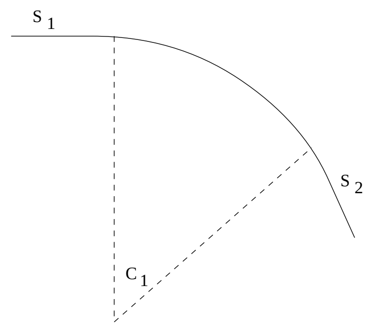

Figure 3: A simple trajectory

A trajectory is represented as a list of these maneuvers together with an appropriate value for the parameter of each maneuver. An example of a simple trajectory is shown in Figure 3. This trajectory is described by the list: $(straight(s1), curve(c1), straight(s2))$. All knowledge regarding the actual shape of the trajectory of an aircraft, performing a given list of maneuvers, is located in a mapping function which computes a list of connected straight lines representing the actual trajectory. Hence we have two representations for trajectories: one of which is a sequence of maneuvres, and a derived representation which is a list of connected lines. A complete planning is then represented by a set of these trajectories. These plannings are taken as individuals in a population. We use the following mutation operator. It acts directly upon the abstract representation and applies the following steps to an individual:

1. determine the time of the first conflict t_{conf},
2. select at random one of the aircraft involved in this conflict and extract its trajectory,
3. select at random the following parameters:
   - $t_{resolve} \in [0, t_{conf}]$,
   - $m \in M$,
   - $x_m \in R_m$, and
   - $t_{straight} \in [0, 300]$.
4. construct a trajectory according to the rule:

$$head(m, 0, t_{resolve}) \cdot m(x_m) \cdot straight(t_{straight}) \cdot m'$$

   (where $head(m, 0, t)$ denotes the part of trajectory m corresponding to interval $[0, t]$, the operator $\cdot$ is a concatenation operator, and m' is a list of maneuvers guiding the aircraft to its destination).

This mutation operator is conflict driven, instead of randomly driven. It modifies trajectories belonging to aircraft involved in the first detected conflict. This approach is chosen for two reasons. First the introduction of redundant maneuvers will result in abundant length of trajectories, and thus a lower quality of the planning. Furthermore the probability of new conflicts being introduced will be enlarged by these redundant maneuvers. The second reason is more sophisticated. Due to the separation standards, trajectories are interacting, and this results in the fitness of a trajectory being correlated to its context, which is determined by the surrounding trajectories. A modification at a certain time influences the future context of the trajectory. The first conflict first driven approach helps to reduce the probability that a mutation makes a previously added maneuver obsolete.

It is difficult to define a good recombination operator using the current representation. Mixing trajectories from different individuals does not make much sense, as trajectories are evolved in the context of surrounding trajectories. The quality of a trajectory is strongly related to this context.

Designing a proper fitness function for the free route planning problem is a challenging problem by itself. Often one defines a fitness function by taking the objective and maps this to a number in a straightforward way. In our case the objective is to find a conflict-free plan with a minimal traveled distance for each aircraft. A straightforward mapping to numbers could result in the following fitness function:

$$f(i) = -\#conflicts - \alpha \sum_{j \in i} \frac{\int_j \mathrm{d}x}{|dest_i - src_i|}.$$

The first term describes the total number of conflicts in the sector. In a conflict-free plan, this number is zero. The second term describes the relative excess distance summed over all aircraft in the sector. The factor α is introduced to balance the relative importance of conflicting objectives.

Although this fitness function does represent the objective, it does not result in a proper driving force for the evolution. The main problem is that conflicts need to be solved in the right order. One has to start with the first conflict (in time), and not with the last one. Intuitively, changing a later conflict is useless if the earlier ones are not solved. Hence to cope with the free-route planning problem a more advanced fitness function needs to be defined which guides the search in the proper direction.

The calculation of the fitness function is done in two stages. First the fitness of trajectory i within an individual is expressed as a value in range $[0,1]$ using the following formula:

$$f_{traj}(i) = \omega_{conf} \frac{t_{conf}}{t_{plan}} + (1 - \omega_{conf})\, e^{-\omega_{detour}\left(\frac{\int_{traj} \mathrm{d}x}{|dest - src|} - 1\right)}$$

The first term represents the relative time before the first conflict with any other trajectory within the individual arises. The second term results in a penalty when a detour is taken. This penalty is zero in case of a straight trajectory from *src* to *dest* and one in case of an infinitely large detour. The exponential is needed to perform a scaling of this penalty term. A solution where a number of aircraft make a small detour is preferable above a solution where a single aircraft makes a large detour. For fairness and to prevent aircraft form running out of fuel.

The fitness of a individual j is taken equal to be a weighted sum over the fitness of all trajectories within the individual according to the formula:

$$f(j) = \sum_{i \in j} \omega_i \cdot \mathrm{f}_{traj}(i)$$

under the restrictions $\omega_i \geq 0$ and $\sum \omega_i = 1$. A proper choice of the weights ω_i is important in order to get proper convergence. Three possible choices for these weights are considered:

1. Set $\omega_i = 1$ for the least fit trajectory and for all other trajectories j with $j \neq i$ set $\omega_j = 0$. This corresponds to observing the worst performing trajectory only,
2. Set $\omega_i = \frac{1}{\#aircraft}$ for all trajectories i. This corresponds to taking the average over all trajectories.

3. Use a fitness based weighting, where bad performing trajectories have a relatively large influence on the fitness of a planning:

$$\omega_i = \frac{W_i}{\sum_j W_j}$$

$$\text{where } W_i = e^{-\alpha \cdot f_{traj}(i)} \text{ and } \alpha \geq 0.$$

The third method can be seen as a generalisation of the other two methods. By setting $\alpha = 0$ all weights ω_i are made equal, which corresponds to the second method. As α gets larger, more emphasis is put upon the bad performing trajectories. In the limit $\alpha \to \infty$, this method will correspond to the first method, where the fitness of a plan is completely determined by its worst performing trajectory.

This two-phase computation of the fitness of a plan has an important additional advantage. Calculating the fitness value for a trajectory is one of the most time-consuming parts of the complete algorithm. As most trajectories appear in more than one individual a significant amount of time is saved by applying an incremental calculation scheme, which uses stored partial results.

The fitness function puts much emphasis on nearby conflicts. As the first conflict is moved in time the fitness function gets less sensitive to the actual time a conflict arises. When applying a fitness proportional selection scheme this will result in less selective pressure when the first conflict moves ahead in time and the fitness increases. In order to avoid this kind of scaling problem a ranking scheme is used [Bak85, GD91]. In a ranking scheme the individuals are sorted and ranked on fitness. The probability of being selected for reproduction is coupled to the rank. By adjusting the probability of survival the selective pressure can be set. Ranking is often combined with a steady-state algorithm [Sys91, Whi89]. Such a steady-state algorithm replaces just a small part of the population during each iteration. Hence the rank will be recalculated often and a more aggressive search is obtained.

As the search space is very large and it is assumed to contain many good solutions, an aggressive search method seems to be appropriate. The algorithm is terminated if it does not enhance its best solution for more than τ iterations. In our experiments we used $\tau = 100$.

3. Experiments

All experiments involve the creation of a planning for a two-dimensional square area of 200×200 kilometers. (This limitation is purely for convenience and does not correspond to a conceptual limitation of the model.)

Three different fitness functions are compared on a set of 24 random problems. The three fitness functions we compare are the fitness based weighting with values $\alpha = 0, 6, \infty$. For each test problem 40 independent runs were performed for each fitness function, resulting in a total of 960 runs for each point in the graph.

A single test problem is generated by selecting for each aircraft an entrance time, entrance position and exit position at the boundary of the sector, all at random. The entrance and exit positions should be on different sides of the sector. A test problem constructed according to these rules is accepted if it does not have any initial conflicts. A conflict is called an initial conflict if both aircraft involved in the conflict have just entered the sector. A typical result of a free-route planning involving 20 aircraft is shown in figure 4.

Figure 5 shows the rate of success of the method as a function of the number of aircraft in the sector. A single run is successful if a planning is constructed such that:

- the planning is free of conflicts, and
- all aircraft leave the sector at their exit position.

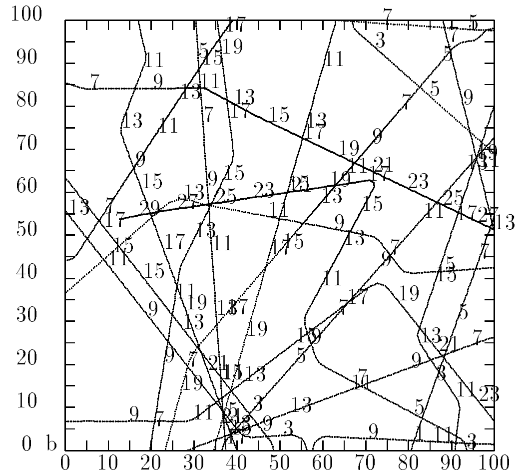

Figure 4: A typical example of a free-route planning. The number inside the sector are an indication of time.

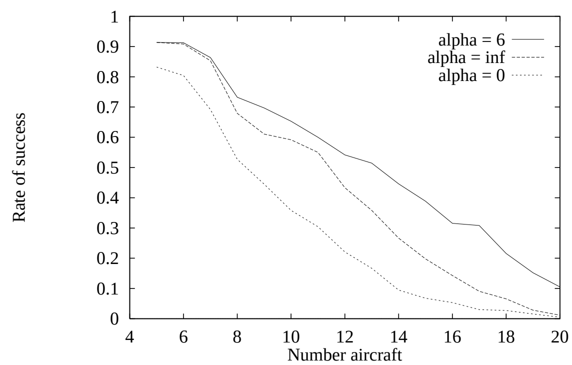

Figure 5: Rate of success

The second item is checked in order to prevent solutions where part of a trajectory is outside the sector.

For each successful run the average deviation from the shortest path is calculated, where the length of the shortest possible path is assumed to be the Euclidean distance between the entrance and the exit point. So a deviation of 1.1 corresponds to 10% extra distance traveled relative to the shortest path. Figure 6 shows the average deviation over all successful runs as a function of the number of aircraft in the sector.

Another important measure is the deviation of the worst performing aircraft in a plan. If this deviation is large compared to the average deviation a plan is said to be unfair. If a plan is not fair, this might result in aircraft deviating from their estimated time of arrival, or even worse, aircraft running out of fuel. Figure 7 shows the deviation of the worst performing aircraft averaged over all successful runs.

From the three compared fitness functions the function with $\alpha = 0$ method is consistently the worst. This method has the lowest rate of success, as can be seen from figure 5. Furthermore the relative

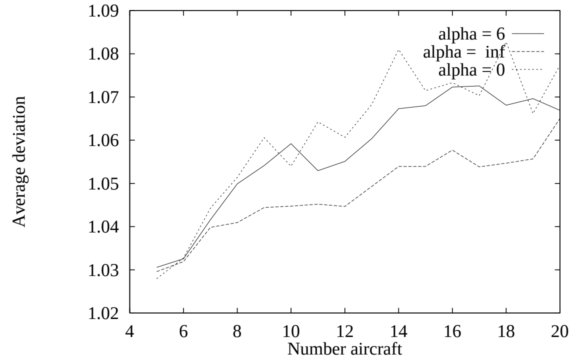

Figure 6: Average deviation of aircraft from shortest path

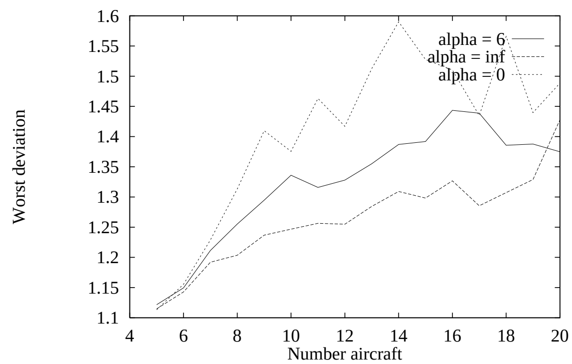

Figure 7: Deviation of worst performing aircraft

deviations from the shortest path are the largest when using $\alpha = 0$, as can be seen from figure 6 and 7. This low performance for $\alpha = 0$ is a result of the low selective pressure enforced by this method. A closer look at the plans produced by this method reveals the reason of this bad performance. Often a single aircraft is sent in the wrong direction, thereby reducing the size of the problem. The remaining conflicts are solved relatively fast, but then it seems to be impossible to guide the last aircraft back to its exit position without making drastic changes to many other trajectories, so the problem can not be solved completely any more. The problem is that mutations which are good in the short term are often bad in the long run. This effect becomes more dominant as the number of aircraft in the sector increases. This corresponds to the relative fast drop of the rate of success as the number of aircraft increases.

Taking $\alpha = \infty$ is much better. The rate of success of this method is higher than that of $\alpha = 0$, as can be seen from figure 5. Unfortunately the rate of success drops fast as the number of aircraft increases. When $\alpha = \infty$, then the value of the fitness function is almost completely determined by the time of the first conflict. The other conflicts within the sector do not have any influence on the fitness value. This lack of differentiation between different plans having the same first conflict is assumed to be the reason for the fast drop of rate of success when increasing the number of aircraft for $\alpha = \infty$.

Further evidence for this is the low number of iterations used when $\alpha = \infty$. Fast convergence often corresponds to bad exploration of the search space.

Taking $\alpha = 6$ has the overall highest rate of success. This method is somewhere in between the previous two methods. It puts a relative high emphasis on the worst trajectory just like for $\alpha = \infty$. The advantage of taking $\alpha = 6$ is that it does differentiate between different plans that have the same first conflict, because all trajectories influence the fitness value.

When looking at the quality of the solutions in case of success, preference should be given to $\alpha = \infty$. This method results in the lowest deviations from the shortest path, as can be seen from figures 6 and 7. When taking rate of success and quality into consideration it seems best to use $\alpha = \infty$ if the number of aircrafts is below 14 and use $\alpha = 6$ otherwise.

4. Conclusions and Further Work

In the Evolutionary Computation community a lot of research is done on artificial test problems, constructed to show certain specific effects. Real world problems are often more diverse than the most advanced test suite can capture. For example, in our case it was non-trivial to map our problem to a representation, which could be handled by an evolutionary algorithm. Furthermore it was important to make a clear distinction between the actual objective and the fitness function. The final result of the planner has to be tested against the objectives to measure its quality. The fitness function on the other hand has to be streamlined to guide the search in the proper direction.

Due to the lack of a sufficient number of real-world test cases and other methods to handle this planning problem, it is difficult to determine the relative strength of the evolutionary computation method. Taking the difficulty of this kind of planning problems into account, the results are promising. A solution of good quality can be obtained for problems involving up to 20 aircraft, within a reasonable amount of time.

When allowing up to 14 aircraft in the sector results of even better quality can be obtained, by using a different value of α.

Further research is needed to find a good recombination operator. We also like to show the robustness of the method with respect to changes in the planning. This robustness is important, as the system has to be able to cope with additional constraints enforced by the human controller.

Concluding, the evolutionary approach has resulted in a successful planner for free-route planning. New constraints can be incorporated easily by modifying the fitness function. Due to this kind of flexibility the model is a valuable tool for the free-routing problem.

5. Acknowledgements

We like to thank dr. Henk Blom from the National Aerospace Laboratory NLR for his helpful comments.

References

[AGJS93] J. Alliot, H. Gruber, G. Joly, and M. Schoenauer. Genetic algorithms for solving air traffic control conflicts. In 9^{th} *conference on AI Applications*, 1993.

[Bak85] J.E. Baker. Adaptive selection methods for genetic algorithms. In *First International Conference on Genetic Algorithms*, 1985.

[BUMK91] S. Bagchi, S. Uckun, Y. Miyabe, and K. Kawamura. Exploring problem-specific recombination operators for the job shop scheduling. In *Fourth International Conference on Genetic Algorithms*, 1991.

[GD91] D.E. Goldberg and K. Deb. A comparative analysis of selection schemes used in genetic algorithms. In *Foundations of Genetic Algorithms - 1*, 1991.

[Ger94] I.S. Gerdes. Application of genetic algorithms to the problem of free-routing for aircraft. In *First IEEE conference on Evolutionary Computation*, 1994.

[Nak91] R. Nakano. Conventional genetic algorithms for job shop problems. In *Fourth International Conference on Genetic Algorithms*, 1991.

[Sys91] G. Syswerda. A study of reproduction in generational and steady–state genetic algorithms. In G. Rawlins, editor, *Foundations of Genetic Algorithms - 1*. Morgan Kaufmann, 1991.

[Whi89] D. Whitley. The genitor algorithm and selective pressure. In *Third International Conference on Genetic Algorithms*, pages 116–121, 1989.