Airspace sectoring by evolutionary computation

Daniel Delahaye, Marc Schoenauer, Jean-Marc Alliot

▶ **To cite this version:**

Daniel Delahaye, Marc Schoenauer, Jean-Marc Alliot. Airspace sectoring by evolutionary computation. IEEE 1998, International Conference on Evolutionary Computation, May 1998, Anchorage, United States. pp 218 - 223, 10.1109/ICEC.1998.699504 . hal-00937715

HAL Id: hal-00937715

https://enac.hal.science/hal-00937715v1

Submitted on 29 Apr 2014

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Airspace Sectoring by Evolutionary Computation

D. Delahaye (LOG) * M. Schoenauer (CMAPX) † J.M. Alliot (CENA) ‡

Abstract

This paper addresses the classical graph partitioning problem applied to the air network :

> One considers an air transportation network with aircraft inducing a control workload. This network has to be partitioned into K balanced sectors for which the cutting flow is minimized.

1 Introduction

Aircraft have to follow routes in the sky in a way to increase the navigation accuracy, a route being described by a list of beacons the plane have to follow to reach his destination. Because there is a small number of beacons on the ground, they often symbolize crossings of airways and generate risks of collision between aircraft. Therefore, pilots must be helped by an air traffic controller who has a global view of the current traffic distribution in the airspace and can give orders to the pilots to avoid collisions. As there are many aircraft simultaneously present in the sky, a single controller is not able to manage all this traffic, and airspace is then partitioned into several sectors, each of them being assigned to a controller. Nowadays, this sectoring is done by some airspace experts (in an empirical way) who apply rules they have learned from experience. This way of working is relevant because it takes into account several practical aspects but is limited to the local zone it treats. This process can be improved by an automatic approach in order to give an "optimal" sectoring for the whole airspace which could be refined by experts.

Before specifying what is a good sectoring, control workload on a sector has to be defined. According to the controllers themselve, workload can be divided into three parts which correspond respectively to the conflict workload, the coordination workload and the trajectories monitoring workload of the aircraft present in the sector :

- the conflict workload gathers the different actions of the controller to solve conflicts ;
- the coordination workload is the information exchanges between a controller and the controllers in charge of the bordering sector or between a controller and the pilots when an aircraft crosses a sector boundary.
- the monitoring aims at checking the different trajectories of the aircraft present in the sector.

Then a good sectoring must have balanced sectors according to the control workload and must reduce the global coordination workload which depends of the traffic flow cut by the

*Laboratoire d'Optimisation Globale, CENA/ENAC, 5 Avenue Edouard Belin, 31055 Toulouse CEDEX (delahaye@recherche.enac.fr)

†Centre de Mathématiques Appliquées de l'Ecole Polytechnique (marc@cmapx.polytechnique.fr)

‡Centre d'Etudes de la Navigation Arienne (alliot@dgac.fr)

sector boundaries. Furthermore any sectoring must take some constraints into account coming from the Air Traffic Control system :

- a pilot must not encounter the same controller during his flight to prevent superfluous coordinations; this means that an aircraft crossing a sector will encounter two and only two sector boundaries. Then that synthesized sectors have to be convex according to the routes (route convexity constraints);
- a sector boundary has to be at least at a given distance from each route crossing (safety constraint). When a controller has to solve a conflict, he needs a minimum amount of time to develop a solution. Each controller managing individually his sector, if a sector boundary is too close to a crossing point, he is not able to solve any conflicts because he has not enough time between the coordination step (with the previous sector where the aircraft comes from) and the time the aircraft reaches the crossing point. The minimum delay is fixed at seven minutes and can be converted into a distance once the aircraft speed is known.
- an aircraft has to stay at least a given amount of time (a few minutes) in each sector it crosses to give enough time to the controller to manage the flight in his sector (min stay time constraint). This constraint can be expressed by a minimum distance between two boundaries cutting the same route link.

Then the problem we have to solve can be summarized by the following :

> one consider an air transportation network with flows on it inducing a workload spread over the airspace. This workload has to be partitioned into K balanced sector in order to minimize coordination and which satisfies the route convexity constraint, the safety constraint and the min stay time constraint.

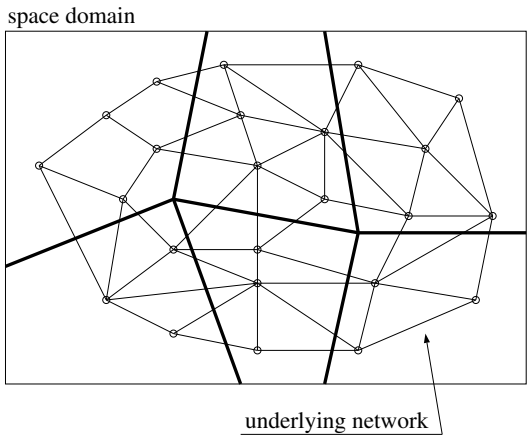

Figure 1: Voronoi diagram in a two dimensional space

A first try has been developed in [1] where the three dimensional airspace was partitioned into polyhedral sectors. This approach was based on Genetic Algorithm for which the chromosome was describe by K points in the airspace (class centers). The positions of these K class centers were randomly distributed in the airspace for which all the domain points were aggregated to their nearest class center. Then, each chromosome, synthesized a random three dimensional Voronoi diagram (see figure 1)

Good results have been given by this technique but it remains some weak points from the operational point of view :

- polyhedral sectors (convex sector in a 3 dimensional space) are not adapted for the air traffic because aircraft do not stay on a stable three dimensional network (different climb rates) ;

- operational sectors must have vertical planar boundaries to help controllers for coordination of aircraft in climb or in descent ;
- Voronoi diagrams are well adapted for sectors design in 2 or 3 dimensions with space convexity which is stronger than route convexity inducing a strong restriction of the state space.

A new approach is now presented which try to circumvent the previous drawbacks. Instead of sectorizing the physical airspace this new technique partitions an underlying network connecting the domain points where workload has been registered.

2 Problem Modeling

Before creating sectors, the original network has to be built by applying the following steps :

- A loaded day of traffic is chosen.
- All the crossings between all the trajectories are registered
- The associated workload is computed for this network.
- An initial network is created where all the nodes represent a crossing (this network is supposed to be connected, this mean that each nodes of the network can be reached from all the other nodes by an undirected path).
- All the links greater than 14 minutes are removed and stocked in a new set of link.
- The remaining connected components (and their associated workloads) are gathered into a new single node (a connected component is a connected subgraph).
- A new contracted network is created from the two new sets of nodes and links.

This new network will be partitioned into sectors.

The classical graph partitioning problem is usually defined the following way :

> Let $G = (V, E, w)$ be an undirected connected graph, where $V = v_1, v_2, ..., v_n$ is the set of nodes, $E \in V \times V$ is the set of edges and $w : E \rightarrow N$ defines the weight of the edges. The Graph Partitioning Problem is to divide the graph into K connected components $P_1 ... P_k$, such that the sum of the weights of edges between the component is minimal, and the weight of the components are nearly equal.

The problem we have to solve is more complex because when an edge joining two components is cut, a new weight appears on it which is shared by the two associated components. This new weight depends on the cutting flow on the edge and is summarized by the coordination.

This means that the global weights of the network before and after partitioning are not the same (the later being heavier). So, the balance for the a priori partitioning (before cutting the edges) is not the same as the balance for the a posteriori partitioning (after cutting the edge).

Constraint satisfaction

- Route convexity constraint Each sector being synthesized by a connected component of a network build from the routes actually used (supposed to be the min distance route in the network) this constraint is naturally satisfied.
- Safety constraint and min stay time constraint Those constraints are partially satisfied

by construction of the network where all the small links have been removed. For the longer links (link in the new network), the possible cuts are forced to be in a central zone of the edges, 7 minutes flight distant from both extremities.

Discrete graph partitioning is a classical NP_HARD problem [2] and then no polynomial algorithm has been identified to solve it. So, the K graph partitioning is usually investigated by iteratively applying a bi-partitioning heuristic on the successive created subcomponents[3, 4], which is definitively a suboptimal approach. Even, for the bi-partitioning problem, the most powerful heuristic developed by Goemans and Williamson [5] ensure to be at least at 80 percents from the optimum.

So, according to the size of our network (about 1000 nodes), classical combinatorial optimization is not relevant and stochastic optimization seems to be more suitable.

3 Evolutionary Algorithm

3.1 Introduction

Evolutionary algorithms (EAs) are heuristic computer search techniques whose mechanics are based upon the principles of natural selection found in the biological world [6, 7, 8, 9, 10, 11]. They have be used to obtain solutions to a diverse set of known NP-hard problems including task scheduling, graph-theoretical problems, VLSI layout, automatic control, numerical integration, etc .... Empirical evidence strongly suggests that EAs can outperform other optimization techniques such as simulated annealing [12].

Before an EA can be run, a suitable coding for the problem must be devised. A fitness function is also required, which assigns a figure of merit to each coded solution. During the run, parents must be selected and recombined to generate offspring. These aspects are described below.

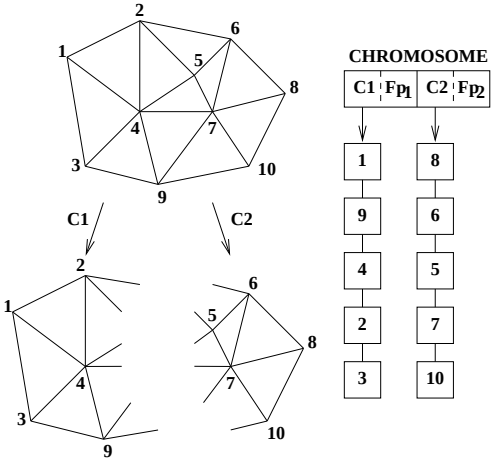

Figure 2: Coding of the chromosome

3.2 Data Coding

This step consists of converting each point of the state domain into a chromosome used by the Evolutionary Algorithm. It is assumed that a potential solution may be represented as a set of parameters. These parameters (known as genes) are joined together to form a string of values (often referred to as chromosome). In genetic terms, the set of parameters represented by a particular chromosome is referred to as the genotype. The genotype contains the information required to construct an organism which is referred to as the phenotype. In our problem, the state variables (which contain all the information needed to define the connected components) consist of the partition of the set of nodes into subsets, each node belonging to one and only one subset (with no empty subset). Furthermore, a power factor is associated to each subset to define the limit between two different components (see figure 2).

To create an initial population of individuals (random graph partitioning), the following

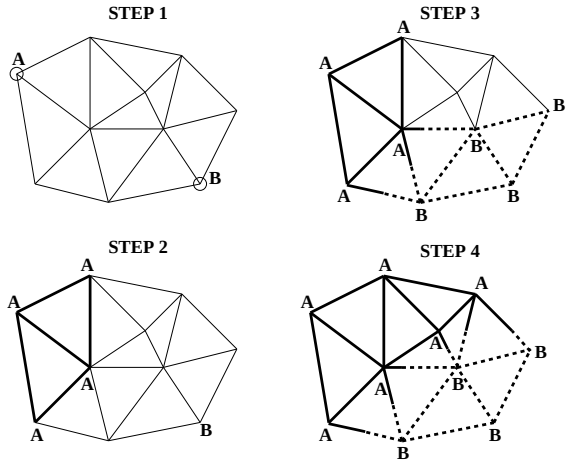

Figure 3: Random graph partitioning

steps are applied (see figure 3) :

1. K different nodes are randomly selected from the network (K represent the number of connected components (2 in the example)). Those nodes are the initial K connected components and are then labeled with different symbols (A and B here) ;

2. the neighbors of a component i ($i = 1..K$) are checked (a node is said to be a neighbor of a connected component if there is a link between this node and a node belonging to the connected component). If this node is free, then it is associated to the component i else this node is already associated to another component and the link joining the current component to the neighbor node is randomly cut into two segments.

3. step 2 is repeated till all the nodes of the network are labeled.

This process enable the creation of an initial population of random graph partition into connected components which must be now evaluated with the fitness function.

3.3 Fitness Function

Given a particular chromosome, the fitness function returns a single numerical "fitness", or "figure of merit", which is supposed to be proportional to the "utility" or "ability" of the individual which that chromosome represents. In our application, the two objective of sectors balancing (f_b) and minimization of coordination (f_c) are gathered into a single fitness in the following way :

$$f(chrom) = \alpha f_b(chrom) + (1 - \alpha) f_c(chrom);$$

with

$$f_b(chrom) = \sum_{k=1}^{k=K} \frac{|w(k) - \frac{W}{K}|}{\frac{W}{K}}$$

$$f_c(chrom) = \frac{1}{W} \sum_{k=1}^{k=K} w_c(k)$$

and

$$W = \sum_{k=1}^{k=K} w(k)$$

- $\alpha \in [0, 1]$;
- $w(k)$ control workload in the sector k;
- $w_c(k)$ coordination workload in the sector k.

3.4 Reproduction

During the reproductive phase of the EA, individuals are selected from the population and recombined, producing offspring which will comprise the next generation. Parents are selected randomly from the population using a scheme which favors the more fit individuals ("Stochastic Remainder Without Replacement Selection"). Good individuals will probably be selected several times in a generation, poor ones may not be at all.

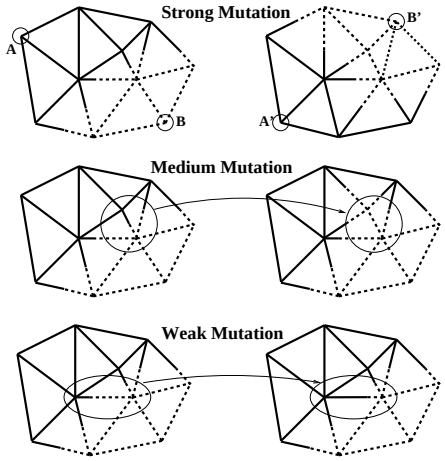

Figure 4: Mutation operators

3.5 Recombination Operators

When a population has been created and the best individuals selected, the associated diversity decreased. To improve this diversity and to have a chance to explore new regions of the state domain, different recombination operators are randomly applied. These operator are stochastic, and modified more or less the structure of the chromosome. In our application crossover and mutation operators have been developed but only mutation has been applied because offsprings generated by crossover needed repairing in order to respect constraints. Repairing operator was very penalizing and then has been abandoned.

Mutation operators can be classified into three categories (see figure 4) :

1. Strong mutation. This operator modified all the connected components by randomly choosing K new initial nodes and propagating the new components as for the initialization process;

2. Medium mutation. This mutation consists in statistically selecting the most unbalanced component in the chromosome and to (statistically) identify the neighboring component which will better correct the weight of the current component by exchanging one node. After applying this operator one must check that the component which loses one node is still connected. As a matter of fact, this operator can break the connectivity as it is shown in the figure 5

   Then the mutation will be accepted only if the new associated component is still connected. To check this property, a connectedness algorithm is applied to the suspected component. The basic step of this algorithm is the fusion of adjacent vertices. We starts with some vertex in the graph and fuse all vertices that are adjacent to it. Then we take the fused vertex and again fuse with it all those vertices that are adjacent to it now. This process of fusion is repeat until no more vertices can be fused. This indicates that a connected component has been "fused" to a single vertex. If this exhausts every vertex in the initial graph, the graph is connected.

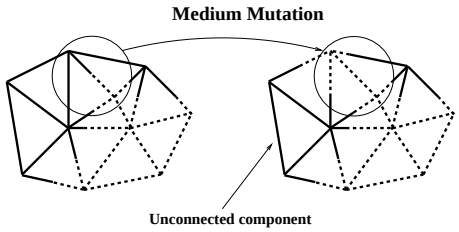

Figure 5: Structure of the test network

3. Weak mutation. This operator works the same way as the previous one but only the respective power factor are modify on the two selected components (no repairing is needed).

When a Medium or a Weak mutation has been decided, the stochastic balancing process induced by there application is applied several

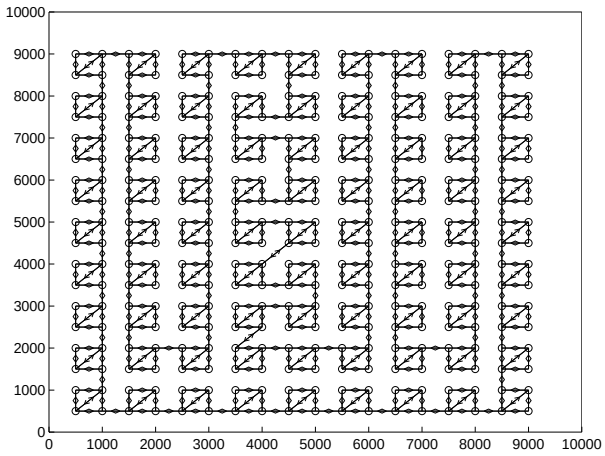

Figure 6: Symmetrical test network

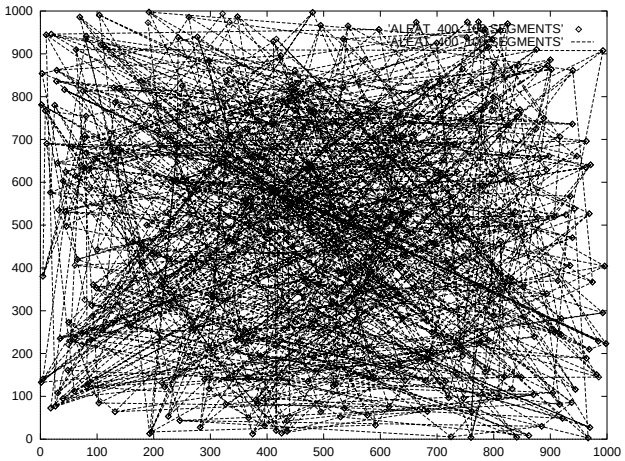

Figure 7: Random test network

times in order to speed up the convergence on the first part of the criterium (balancing).

4 Application to test networks

This method has been successively applied to different kind of networks with several hundred nodes. To investigate the performance of the algorithm, a serie of networks with exact solutions have been used. In all cases, the expected exact solution have been found but sometime further exact solution have been discover by the sharing mechanisms. Figures 6, 7, represent two test networks with exact solutions (324 nodes and 400 nodes respectively).

In the first one, an exact solution with 81 components can be be identified (this symmetrical solution is trivial for a human being because of our brain ability to investigate symmetries); in the second one an exact solution with 100 components has been hidden in this random network. From the computer, both networks represent the same difficulty but for a human being the first one is much more easier. The associated EA parameters were the followings :

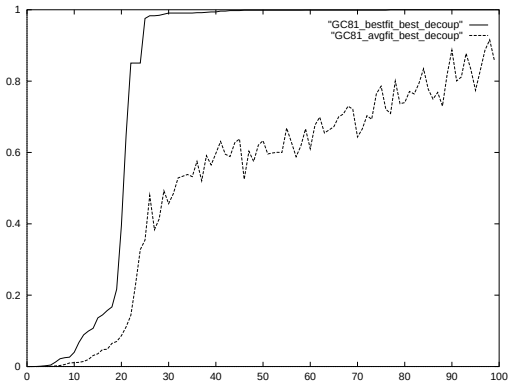

Figure 8: Results

- population size : 100
- number of generations : 100
- probability of mutation : 0.7
- sharing (an adaptative clusterised sharing has been used) : yes
- ellitism : yes

The associated fitness evolution (best and average fitness on population for each generation) are given on figure 8 for the symmetrical network and on figure 9 for the random network.

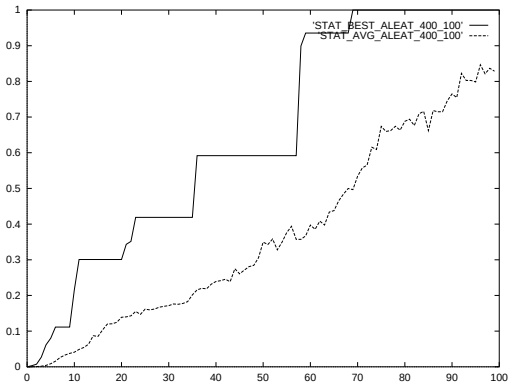

Figure 9: Results

In both cases, the fitness reaches "1.0" which is the optimum according to the fitness calculation. The large evolution steps are due to the random balancing process during the medium and weak mutations. The execution time was about of 10 minutes for both network on a Pentium (133 Mhz).

5 Conclusion

This new approach has given very good results and seems to be very well adapted to the air network partitioning problem. It respects the major operational constraints (the synthesized sectors met the route convexity constraint (not the space convexity), the safety constraint and the min stay time constraint) and can generate three dimensional sectors with planar vertical boundaries after applying an algorithm which synthesis spatial envelops from connected components (this aspect has not been develop for concision). Furthermore, it circumvents the weak points of the first approach[1] by working on the network itself instead of the geographical airspace. The method used to create the initial network depends only of the traffic itself and can investigate organized traffic or Free Route[1] traffic.

Having now a good method to sectorize airspace, the workload definition has to be refined in a way to be more realistic.

References

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

[2] M.R Garey and D.S Johnson, *Computers and Intractability. A Guide to the Theory of NP_Completeness*, Freeman, W.H and Company, 1979.

[3] Chung-Kuan Cheng, "The optimal partitioning of networks," *Networks*, vol. 22, pp. 297–315, 1992.

[4] B Hendrickson and R Leland, "An improved spectral graph partitioning algorithm for algorithm for mapping pararrel computations," *SIAM. J; Sci. Comput*, vol. 16, pp. 452–469, 1995.

[5] M.X Goemans and D.P Williamson, "Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming," in *Proceedings of the 26th Symposium on the Theory of Computing*, 1994.

[6] D.E Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison Wesley, 1989, ISBN: 0-201-15767-5.

[1] In Free Flight environment, aircraft follow direct routes between origins and destinations without staying on the air network

[7] Back Thomas, *Evolutionary Algorithms in Theory and Practice*, Oxford University Press, 1996.

[8] L.J Fogel, A.J Owens, and M.J Walsh, *Artificial Intelligence Through Simulated Evolution*, Wiley and sons. NY, 1966.

[9] D.B Fogel, *Evolutionary Computation. Toward a new Philosophy of Machine Intelligenc e*, IEEE press, 1994.

[10] Z Michalewicz, *Genetic Algorithms+Data Structures=Evoultion Programs*, Pringer-Verlag, 1992.

[11] H.P Schwefel, *Evolution and Optimum Seeking*, Wiley, New York, 1995.

[12] E Aarts and J Korst, *Simulated annealing and Boltzmann machines*, Wiley and sons, 1989, ISBN: 0-471-92146-7.