[EN-0011] Airspace Design using a Workload Node-based GA

(EIWAC 2010)

+Xavier Ruiz-Hernández, J. M. Rísquez, V.F. Gómez

Air Navigation & Systems Directorate
INECO, S.A.
Madrid, Spain
[xavier.ruiz | jrisquez | fgomez]@ineco.es

Abstract: This optimization algorithm used for airspace design follows the rules of the evolutionary programming and uses a core based on genetic algorithms (in short, GA) inspired in the Mendelian genetics present in nature. These algorithms consist of a search technique based on probability and converging to a sufficient solution by maintaining the elitism when keeping the best element in the population.

The algorithm presented uses workload nodes instead of small airspace regions or traffic flows like other initiatives in this field, and it takes into account parameters like the workload produced by aircraft, the location of the conflicts or the possible/impossible frontier points to consider specific safety issues. Thanks to the difference of using workload nodes the algorithm directly knows where and when the workload is produced.

All theses nodes are encoded into chromosomes forming individuals that will be part of the initial population. The GA will subject the population to specifically dedicated mutations, and the quality of each individual will be evaluated by a fitness function, with the objective of pruning the search space. A new generation will be bred from the best individual in a new iteration of the algorithm. The iterations will be repeated until a sufficient fitness is achieved.

Keywords: Genetic algorithm, GA, Workload nodes, Airspace design, Evolutionary computation, Efficiency, OP-DIS

1. INTRODUCTION

Some studies foresee that world passenger traffic will grow by over 4% per year through 2020, aircraft movements by 2.8% and air freight will increase by 5.4% per annum. The Asia/Pacific region projects the highest rates of growth with an average increase of 6.3% per year as stated in [1]. For this reason, several efficiency, safety and environmental improvement initiatives are now being developed, both at an international scale, with programs like ASPIRE (Asia & Pacific Initiative to Reduce Emissions) [2]; and at a regional/local level, with the creation of routes, sectorizations, functional airspace blocks et cetera. These initiatives share the aim of allowing a dynamical adaptation to the forecasted demand, and of guaranteeing the traffic separation in a safe and efficient way.

From this perspective, the e-TLM (enhanced Traffic Load Monitoring) [3] concept was developed to adapt the sectorization to be used, and its associated capacity, to the forecasted demand in a pre-tactic horizon. Hence, this concept re-sectorizes in real time, but with the unpleasant constraint of only using the pre-designed sectorizations obtainable by recombination of all sectors in airspace. For this reason, we propose in this paper the tool OP-DIS for dynamically creating optimized sectors.

This application is in charge of drawing and proposing new sectorizations, starting from scratch, that become more efficient than those pre-designed by following an empirical and classical way.

The promising approach based on GAs that is explained in this paper is an optimization module to avoid the constraints of using pre-existing sectors. OP-DIS uses an algorithm to design sectors in a given airspace for a specific traffic forecast. The analyzed approach is different from other similar initiatives because, as it is explained, it uses workload nodes instead of airspace regions (as in [4]), network partitioning (as in [5]) or traffic flows (as in [6]).

Other initiatives have been undertaken using other algorithms and techniques like constraint programming, as in [7], or geometric algorithms like BSP or Pie-Cut heuristics, as in [8]. Even though, promising sector-less initiatives were investigated in [9] and experimented in [10], where controllers manage flight safety individually.

By using GA algorithms, appropriate results are obtained in a satisfactory time, even by means of traditional non-dedicated hardware nor software (a common computer may be used for this purpose). In conclusion, this approach to the airspace design using a workload node-based GA improves the efficiency in the creation of dynamic sectorizations.

2. ALGORITHM SELECTION CHRITERIA

2.1 Search methods

Search methods are those that look for solutions that accomplish the statement of a problem. This statement must define all the elements of the problem, including its restrictions and requirements, to find a proper solution. Most of the problems usually have more than one solution, typically finding a maximum or a minimum, reducing the field of study to an optimization problem and its resolution techniques to optimization methods.

The solutions to these problems will be those optimizing a function that assesses their correctness, called fitness function. A famous example of the optimization problems is the Travelling Salesman Problem (TSP) [11] where given a list of cities and their distances, a traveler visits each city exactly once and following the shortest route.

This combinatorial optimization problem may have several solutions (permutations) when linking all the city-pairs accomplishing the conditions of the problem. However, one of these solutions, the shortest one, will achieve the best score in a fitness function used to calculate the total number of kilometers.

For a small amount of cities (or, in general, parameters) this problem may be solved by a combinatorial way, evaluating all the solutions. However, when a high number of cities are included in the problem, the number of combinations responds to (n-1)!/2, and hence, the assessment of all the combinations becomes nearly impossible. In this type of problems, called NP-hard (Non-Deterministic Polynomial-time hard), the size of the search space does not grow linearly with the number of parameters, and the current computers may not be able to solve them in a reasonable time. This leads to focus on the search of non-optimal but acceptable solutions in order to solve the problem at least in a satisfactory manner.

2.2 Problem modeling

The resolution of this type of problems is treated as restrictions problems, where these restrictions may be part of two categories: hard restrictions, when once unaccomplished offer an invalid solution and soft restrictions that might be unaccomplished, if their consequences are assessed with the fitness function.

A multimodal problem is that problem that may have several different solutions, which usually are a global maximum/minimum and several local ones. Several solutions may produce the same results once assessed by a fitness function, because of the parameters included in the function. All these solutions are included in the Pareto front, including all the solutions in which it is impossible to obtain a better solution or improve the result from a specific parameter without worsening other parameters.

2.3 Resolution techniques

Several resolution techniques may be used to solve this type of problems, depending on the completeness and the locality of the solution, as follows:

Exhaustive search: useful when the search space is limited and with a reduced size, because all the solutions are listed and then assessed. The best option according to the fitness function is selected. The main problem of this type of search is that it requires a high amount of computing memory, becoming a useless method in most of the problems.

Local search: this type of technique uses a solution generator to start from an initial solution (or baseline) that will be assessed by the fitness function and that will be iteratively modified until the expected solution is found. The solution generator will produce solutions following proximity, randomization and refining criteria to ease the obtaining of solutions covering all the search space. The particularities of the modifications will produce nearby solutions to the first one until an inflection point. Usually, the solutions reached are local (instead of global) maximum or minimums.

Methods with partial solutions: these methods build partial solutions, variable after variable, and a complete solution is not obtained until all the variables have been assigned. The variables are allocated trying to maximize or minimize parameters from the fitness function. This type of methods is based on decision trees where branches are consecutively added. To optimize its resolution time, it is usually established an upper bound (when minimizing), and the algorithm follows this branch of the tree until a partial solution exceeds this limit, when the entire branch will be discarded.

Global methods: this type of methods tends to escape from local minimums/maximums, and efficiently covers all the search space. They are mainly based on randomization and use a fitness function to discard solutions. Most of the evolutionary algorithms are included in this category.

3. GENETIC ALGORITHMS

3.1 What is "Genetics"?

Genetics (from ancient Greek γενετικός *genetikos*, "genitive" and that from γένεσις *genesis*, "origin") is the science studying genes, the inheritance and the variation of the organisms in the nature. The term *genetics* was proposed to describe this field by the British scientist William Bateson in a private letter to Adam Sedgewick in 1905, in [12]. Previously, Gregor Mendel in 1865 had categorized the external characteristics of the pea plants and called them *characters*. In his experiments, he had analyzed the variation of the different types of peas.

Inheritance and variation are the baseline of the Genetics. In Prehistory, human beings used their intuition about inheritance mechanisms to domesticate and improve animals and plants. In modern research, Genetics provides important tools to investigate about specific functions of the genes, as the interaction between genes. In organisms, the genetic information resides in chromosomes, where it is stored in the sequence of molecules of Deoxyribonucleic acid (DNA).

In Genetics jargon, the verb *encode* typically means that a gene contains the instructions to synthesize a specific protein. Genetics determine most of the (but not the whole) appearance of the organisms, having the difference in environment and other random factors also impact on it.

3.2 Genetic algorithmia

Evolutionary computation is a global method and uses concepts inspired on biology, as population, mutation, inheritance and survival of the fittest to generate iterative improved solutions for a problem. These methods are divided into evolutionary algorithms (e.g. genetic algorithms) and collective intelligence (e.g. ants algorithms, as in [13])

A genetic algorithm is a search method (heuristic) based on probability. Under a very weak condition (the algorithm must maintain the elitism, keeping unmodified the best element in the population), the algorithm tends to converge in probability to the optimum result. When the number of iterations increases, the probability to reach the optimum value tends to 1.

3.3 Main characteristics of genetic algorithms

The main general characteristics that all genetic algorithms should accomplish follow mainly four steps: initialization, selection, reproduction and termination (see Figure 1).

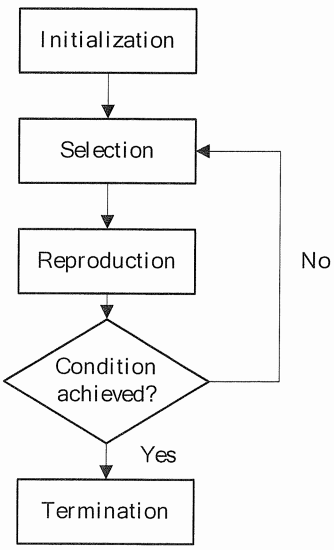

Figure 1 Genetic algorithm workflow

3.3.1 Initialization

An initial population is generated. This population is formed by strings called chromosomes or the genotype. Each of these chromosomes encodes a potential solution in a proper data structure (called individuals or phenotype). Typically, this solution is based on binary codification (genes or bits), but other meta-data structures may be used (see Figure 2). This initial solution, with the intention of accelerating the execution of the algorithm, must accomplish all the hard restrictions. Moreover, in spite of being a random solution, it is often seeded by a nearby solution or with a high fitness one.

3.3.2 Selection

All the chromosomes are evaluated by a fitness function that includes as parameters all the soft restrictions that should be taken into account for the expected solution. Many selection methodologies can be used like the stochastic universal sampling or the truncation selection (where only a specific proportion is selected in each iteration). Regardless of the used methodology, the individuals with the highest score (elitism) will be selected and will be used to breed the next generation.

3.3.3 Reproduction

The next generation is created using the elitist individuals from the previous iteration. Two basic genetic operators of generation are crossover and mutation. Crossover (or recombination) consists in the creation of new individuals bred from their parents. In this method, the parents are divided by one or multiple crossover points and recombined between them to create a pair of children.

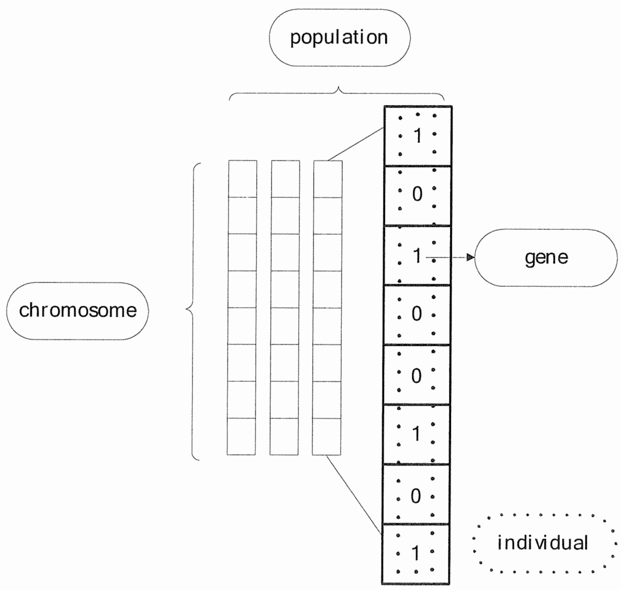

Figure 2 Main genetics components

Mutation is used to maintain diversity and to expand the search space, by modifying one or more bits in the genetic sequence in an individual. This arbitrary modification may be completely random or semi-random, to force the individual to a specific region of the search space or to achieve a certain value in the fitness evaluation function.

3.3.4 Termination

The iterative process of selection-evaluation-reproduction is repetitively produced until a termination condition is reached. Typical conditions are limit of execution time, specific number of algorithm iterations, reach of a threshold satisfying certain criteria, etc.

4. OP-DIS MODELLING

The algorithm used for the development of OP-DIS is based on genetic algorithms, and it basically shares the main characteristics indicated in the previous point: initialization, selection, reproduction and termination. Furthermore, some initial (and essential) steps are needed to achieve the objective of the algorithm.

4.1 Preliminary phase

Prior to executing the GA, some steps must be carried out in order to prepare the algorithm. First of all, a traffic forecast is needed. The objective of this GA is to produce a traffic-dependent sectorization, and hence this will be the main input for the algorithm. The format used by the GA is a traffic obtained with the traffic simulator RAMS (however, other flight events formats could be easily implemented). In this case, specifically the file *flightevents.out* is used, containing the log of all the theoretical events that would occur during the flights. A filter of the file is done in order to discard the unnecessary data like negligible events or trivial information. For each event produced it is also needed the type of event, its coordinates (x, y), flight level, time and indicative. The events required are shown in Table 1. This selection has been done keeping only the most important actions, in order to simplify the model reducing the number of points and events, without losing reliability on the results.

All these flight events are obtained from the flight events file, except the event "crossing points" that relates to those coordinates where several traffics converge but at different times, thus not producing a "conflict". These points are mathematically calculated by means of the crossing equations of their expected trajectories.

Table 1 Events considered in the GA algorithm

Event	Abbreviation
Sector Pierce	#SP
Sector Exit	#SX
Start of cruise	#CR
End of cruise	#EOCR
Conflict	#CF
Crossing point	None

The importance of all these event points lies in the workload produced by them at the coordinates where they occur; coordinates that are the spots at where the Air Traffic Controller must pay special attention or even communicate with the aircraft. This workload will be a parameter of the fitness function, and one of its main factors. The user may introduce a fixed workload value for each category of event.

The only events not producing workload are the crossing points (since they are normal points of the regular trajectories), but they will be used for a better defining of the trajectories and to highlight the most crowded regions of the airspace.

For an optimized use of the algorithm, two simplifications are done: 1) the irregular trajectories like those coming from military flights, calibration flights, circular flights, etc. are removed; and 2) coordinates are rounded off in order to group those events produced at very close positions, effectively reducing the number of points. After these two operations, a numbered list of all the points (called outstanding points) is obtained and ready to be introduced and encoded into the algorithm.

Finally, the airspace to be optimized is obtained. All inner boundaries of sectors are removed and only the outer boundary is preserved. The algorithm will create new inner boundaries defining a given number of sectors within the outer boundary. Consequently, internal events due to an inner sector change (i.e. internal sector pierces and exits) must be removed, since these events are due to the inner boundaries and not to the natural trajectory of the aircraft.

4.2 Initialization phase

At this step, the genetic algorithm must be initialized. The population is prepared to encode the outstanding points detected in the previous phase. To achieve this, each chromosome will be initialized with an individual, i.e. a candidate solution representing a complete sectorization. For the purpose of this representation, consider a fixed number n of sectors and the number m of detected outstanding points in the previous phase. An individual will then be an array of exactly $n \cdot m$ genes.

A gene in an individual will encode the membership of a given outstanding point to a particular sector in the following way: given a sector s_i (with i between 1 and n) and an outstanding point p_j (with j between 1 and m), the gene $g_{(i-1)m+j}$ will have a value of 1 if p_j belongs to the sector s_i, or 0 if p_j does not belong to the sector s_i. Note that $(i-1)\cdot m+j$ varies between 1 and $n \cdot m$ when considering all possible sectors and outstanding points.

As a consequence of this, we can define two types of sets of genes: the sets of genes providing information about a given sector; and the set of genes providing information about a given outstanding point. In the first case, and considering the sector s_i, we can say that the set S_i defined in (1)

$$S_i = \{g_{(i-1)m+j} \mid 1 \le j \le m\} =$$

$$= \{g_{(i-1)m+1}, g_{(i-1)m+2}, g_{(i-1)m+3}, \ldots, g_{(i-1)m+m}\} \quad (1)$$

contains the m genes that relates each outstanding point with sector s_i, where j varies between 1 and m. In the second case, and considering the outstanding point p_j, we can say that the set P_j defined in (2)

$$P_j = \{g_{(i-1)m+j} \mid 1 \le i \le n\} =$$

$$= \{g_{(1-1)m+j}, g_{(2-1)m+j}, g_{(3-1)m+j}, \ldots, g_{(n-1)m+j}\} \quad (2)$$

contains the n genes that relates each sector with the outstanding point p_j, where i varies between 1 and n.

Obviously, one outstanding point cannot be assigned to two different sectors in one individual, thus one and only one of the genes in each P_j can have a value of 1. This conforms a hard constraint in the system, and has the effect of avoiding non-valid individuals.

As an illustration consider Figure 3, where a practical example is shown. In such example, the airspace is to be divided in 3 different sectors (hence, $n = 3$). In this airspace, all the aircraft in it generate 4 outstanding points (hence, $m = 4$). Therefore, the number of genes in each individual will be $3 \cdot 4$, thus each individual will contain 12 genes from which we can obtain the sets $S_1 = \{g_{(1-1)4+1} = g_1, g_{(1-1)4+2} = g_2, g_{(1-1)4+3} = g_3, g_{(1-1)4+4} = g_4\}$, $S_2 = \{g_5, g_6, g_7, g_8\}$ and $S_3 = \{g_9, g_{10}, g_{11}, g_{12}\}$ (shown in different colors in the figure), and the sets $P_1 = \{g_{(1-1)4+1} = g_1, g_{(2-1)4+1} = g_5, g_{(3-1)4+1} = g_9\}$, $P_2 = \{g_2, g_6, g_{10}\}$, $P_3=\{g_3, g_7, g_{11}\}$ and $P_4 = \{g_4, g_8, g_{12}\}$ meaning for instance that genes 1, 5 and 9 are related to the same outstanding point and therefore one and only one of them can have a value of 1, and so on. Once all genes have been set to a valid value, the individual is ready to be replicated or mutated as will be explained in the following sections.

The number of chromosomes that will form the population will be equal to the number of mutations (in our case, 38) and in a first step, all chromosomes will be filled with the same individual to start with a harmonic population where different mutations are to be applied.

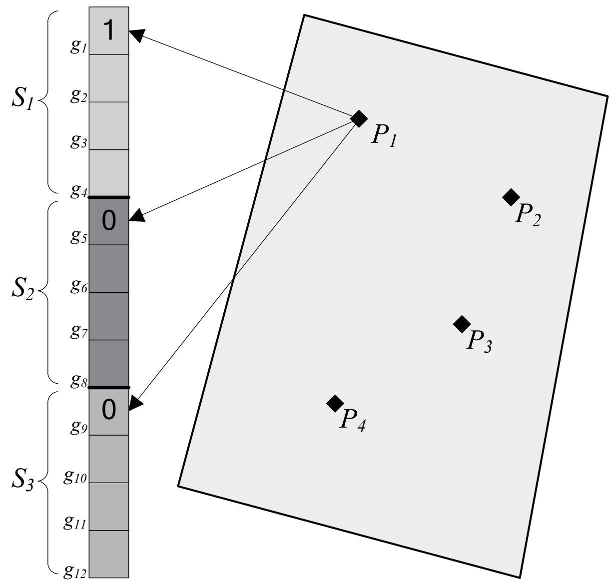

Figure 3 Point P_1 assignment to sector S_1

As mentioned in the previous sections, chromosomes may be initialized with individuals following a certain solution or using an individual encoding a sectorization close to a good solution, i.e. not having a high fitness score. For this reason, the initialization phase will not randomly fill the genes in the chromosome but will create a *block* solution dividing all the notable points in coordinate bands (e.g. vertical bands or latitude bands) or encoding a sectorization actually used or with a known fitness score.

4.3 Selection phase

The OP-DIS algorithm uses a set of specific mutations that are applied to the chromosomes to try to maintain their elitism. The applied mutations force the change of the individuals, generating new combinations that in most of the cases will be worst than their parents. However, a small amount will obtain a better fitness result that will improve the population. In our algorithm only the best individual will be cloned to the rest of the population (i.e. the parent is the individual with the best fitness and the children will be 37 clones of this one). The rest of individuals in each iteration will be discarded.

The mutations applied to the chromosomes will be different for each individual and are focused to improve the weaknesses that they may have. Mutations consist of basic variations on the assignment of outstanding points from a sector to another sector, modifying the bit in the related genes. Most of the mutations are only a few times (or even never) producing the best result in the fitness function for most of the scenarios. However, in particular situations they are useful to obtain an optimized result, and hence their utility is very scenario dependant. An explanation of the 38 mutations can be found in Table 2.

Table 2 Mutations explanation

Mutation	Addressed to...	Mutation	Addressed to...
1	Remove re-entries, modifying the following point in the flow, in the flight direction.	20	Randomly change the assigned sector of the most western point.
2	Remove re-entries, modifying the previous point in the flow, in the flight direction.	21	Match all the points of a flight that crosses two sectors to only one sector, in flight direction.
3	Remove re-entries, modifying the following point in the flow, opposite to the flight direction.	22	Match all the points of a flight that crosses two sectors to only one sector, opposite to flight direction.
4	Remove re-entries, modifying the previous point in the flow, opposite to the flight direction.	23	Match all the points of a flight that crosses two sectors to only one sector, opposite to flight direction, for three flights.
5	Remove airspace discontinuities, modifying the following point in altitude direction.	24	Convert to one sector all the points in a column (with same x and y coordinates but different altitude)
6	Remove airspace discontinuities, modifying the previous point in altitude direction.	25	Randomly change the assigned sector of the three most northern points.
7	Remove airspace discontinuities, modifying the following point opposite to altitude direction.	26	Randomly change the assigned sector of the three most southern points.
8	Remove airspace discontinuities, modifying the previous point opposite to altitude direction.	27	Randomly change the assigned sector of the three most western points.
9	Modify all the points of the first sector in a minority (with less than 15% in points of a sector in a flight level), for a selected flight level, opposite to altitude direction.	28	Randomly change the assigned sector of the three most eastern points.
10	Modify all the points of the first sector in a minority (with less than 15% in points of a sector in a flight level), for a selected flight level, in altitude direction.	29	Randomly change the sector of a point of the sector with more traffic.
11	Modify all the points of the first sector in a majority (with more than 15% of points in a sector in a flight level) for a selected flight level, opposite to altitude direction.	30	Randomly change the sector of several points of the sector with more traffic (the number of points to be modified is randomly selected).
12	Modify all the points of the first sector in a majority (with more than 15% in points of a sector in a flight level), for a selected flight level, in altitude direction.	31	Modify the assigned sector of a point that generates a restriction.
13	Modify all the points of all sectors in a minority (with less than 15% in points of a sector in a flight level), for all flight levels.	32	Modify the assigned sector of a complementary point that generates a restriction.
14	Modify a point out of two consecutive ones, belonging to different sectors, in the flight direction.	33	Remove a re-entry, modifying the following point to it, in flight direction.
15	Modify a point out of two consecutive ones, belonging to different sectors, opposite to the flight direction.	34	Remove a re-entry, modifying the previous point to it, opposite to flight direction.
16	Reverse the sectors of two consecutive flight levels.	35	Remove a re-entry, modifying the following point to it, in flight direction.
17	Randomly change the assigned sector of the most northern point.	36	Remove a re-entry, modifying the previous point to it, opposite to flight direction.
18	Randomly change the assigned sector of the most southern point.	37	Convert to one sector all the points in a flight level.
19	Randomly change the assigned sector of the most eastern point.	38	Randomly select a point and randomly change its assigned sector.

For the evaluation of the new sectorizations created (one per each mutation), the main characteristics are evaluated by the fitness function. The fitness function (*FF*) follows the structure shown in (3).

$$\begin{aligned} FF = Fx_1 \cdot Fa + Fx_2 \cdot Fd + Fx_3 \cdot Fr + \\ + Fx_4 \cdot Fwl + Fx_5 \cdot Fconst \end{aligned} \quad (3)$$

where Fx_n corresponds to weighting coefficients introduced by the user depending on their priorities when creating the new sectorization, and Fvalue to the factors taken into account in each sectorization represented in each individual. These factors are explained in Table 3.

For a better understanding of these factors, in this implementation, the proper explanation of these features is the following:

Workload: each of the required events has assigned a specific workload (usually calculated by an external workload monitoring software). For simplification purposes, each event has assigned the same value of workload, regardless of other events produced in the same moment (non-linear workload is discarded).

Table 3 Factors taken into account in fitness function

Factor	Description
Fa	Number of airplanes in the sector with more traffic
Fd	Number of discontinuities in altitude
Fr	Numbers of re-entries in sectors
Fwl	Estimated workload in sectors
Fconst	Number of couples of points breaking constraints

Discontinuity: given certain outstanding points with the same coordinates (x and y) but in different altitudes, a discontinuity is that circumstance when a sector is assigned to two different points in altitude and a second sector is assigned to an intermediate point (in altitude).

Re-entry: given the flow of points that an aircraft is producing, a re-entry is that circumstance when two points belong to the same sector and another intermediate point belongs to a second sector.

Constraint: couples of points where a frontier change is impossible. A restriction is broken when these two points belong to two different sectors. The intolerable frontier points are, in this case,

- Couples of points, one belonging to the external frontier, and separated less than a number of nautical miles, where this number is to be defined by the user.
- Couples of points, one being a #CF, and separated less than a number of nautical miles, where this number is to be defined by the user.
- Couples of points, one being a #SP or a #SX, and separated less than a number of nautical miles, where this number is to be defined by the user.
- Couples of points that the user considers that may not produce a frontier break.

4.4 Reproduction phase

Once all the individuals are evaluated by the fitness function in each iteration, the one having the minor fitness function result is selected (the more penalties, the higher result in the fitness function). When the best individual is selected, it is cloned to the other 37 chromosomes, having at the end of the iteration the same individual replicated in all chromosomes in the population. The phases of selection and reproduction are iteratively repeated until a termination conditions is reached.

The fact of maintaining the best scored individual in each iteration guarantees that the elitism is maintained during the whole process. Usually, in similar algorithms, the technique employed to breed the new generation, when few parameters are taken into account in the fitness function, is to maintain a number of individuals from the last iteration and not only the best one. However, in this research development only the lowest-fitness individual is kept to accelerate the preservation of the elitism, and this fact is feasible since the fitness function contains several parameters and the search space is vastly explored due to the balance produced between them.

In two consecutive iterations, the best individual may be completely different from the previous one, because as the fitness function evaluates the individual (and hence the sectorization) as an overall, one of the parameters of the new individual may have substantially decreased (or increased) in favor of another parameter. This equilibrium eases the exploration of the great majority of the search space.

4.5 Termination phase

The algorithm may terminate following the criteria of the generic genetic algorithms, for instance a given number of iterations, a fixed execution time, etc. In this particular case, the algorithm stops when a threshold in the fitness function is reached. The solution obtained with the algorithm is the assignment of each outstanding point to a sector. A cloud of points linked to this sector is finally producing each of the sectors (see practical case in Figure 4).

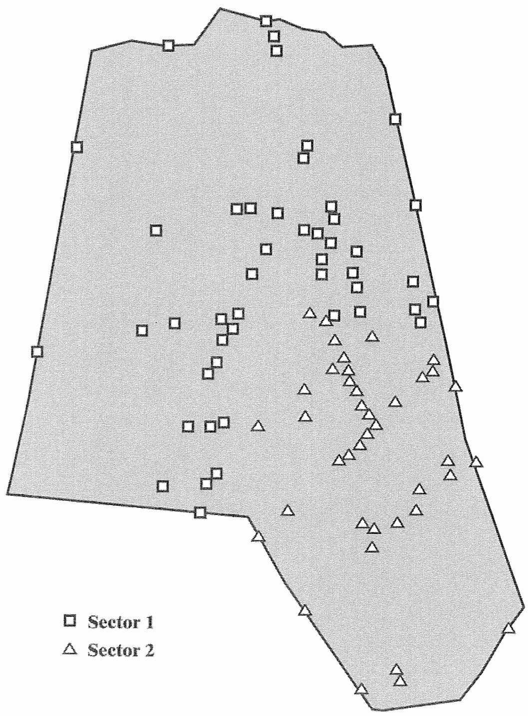

Figure 4 Cloud of points assigned to two different sectors

5. FUTURE EVOLUTION

Future evolutions are foreseen over this research prototype. First of all, as it has been mentioned, the algorithm works with outstanding points and hence, once a set of points is assigned to a specific sector, they provide a solution similar to a cloud of points. However, the solution expected may be a closed boundary for the sector that the points are representing and as a first approach for this issue, it has been used in the tests an external algorithm to produce convex figures from sets of points. These convex polygons are then extended up to the external boundaries of the region and the overlapping areas are removed and only assigned to one of the sectors. However, a better solution may be found, but it has not been deeply analyzed since it was out of the scope of the project.

Additionally, further analysis on the focused mutations may be developed to detect other kinds of alterations in the individuals addressing other parameters that may provide better solutions. Otherwise, the effectiveness of including random mutations may be assessed.

Finally, a more philosophic and pragmatic modification should be made to effectively implement this algorithm in active and real control rooms and it's the fact that the actual current ATCos operation include a training period that depending on the particular unit may be up to three months. The objective of this algorithm is that given an expected forecast of traffic, the new sectorization may be used in a very short time (e.g. the sectorization may be created only in 30 minutes in advance). These different timescales force this algorithm to be useful only in the medium/short term of airspace management phases, leaving out its main real-time functionality.

6. REFERENCES

[1] "Global Traffic Forecast 2006-2025. Executive Summary". Airports Council International. Edition 2007. http://www.airports.org/aci/aci/file/Economics/ACI%20Executive%20Summary.pdf

[2] "ASPIRE, Asia & Pacific Initiative to Reduce Emissions. Strategic Plan". Version 3. 3rd March 2010. http://www.aspire-green.com/mediapub/docs/strategic_plan_2010.pdf

[3] M. M. Dorado. "Future Concepts (MSP, DyR) and pseudo –TFM simulation". AP9 – TIM Atlantic City, November 2005. http://www.tc.faa.gov/acb300/ap5_workshops/documents/Pres11_AENA_AP9TIM_2005.pdf

[4] F.J. Esteban. "Developing optimization algorithms within the airspace scope. 4th IC-SCCE. July 2010.

[5] D. Delahaye, J. M. Alliot, M. Schoenauer and J. L. Farges "Genetic algorithms for partitioning air space". July 1993.

[6] G. R. Sabhnani, A. Yousefi and J. S. B. Mitchell. "Flow conforming operational airspace sector design".

[7] H. trandac, P. Baptiste and V. Duong. "Airspace sectorization by constraint programming".

[8] A. Basu, J. S. B. Mitchell and G. Sabhnani. "Geometric algorithms for optimal airspace design and air traffic controller workload balancing".

[9] V. Duong, G. Gawinowski, J.-P. Nicolaon and D. Smith. "Sector-less air traffic management". 4th USA/Europe ATM R&D seminar (USA). December 2001.

[10] B. Korn, T. Pütz et al. "Is sectorization really required for efficient en-route air traffic control?"

[11] C. Nilsson. "Heuristics for the traveling salesman problem". http://www.ida.liu.se/~TDDB19/reports_2003/htsp.pdf

[12] W. Bateson, F. R. S., naturalist: his essays & addresses, together with a short account of his life. Ed. Beatrice Bateson, Cambridge University Press. 1928.

[13] M. Dorigo and T. Stützle. "Ant Colony Optimization". Massachusetts Institute of Technology. 2004.

7. COPYRIGHT