Sectorization Optimization of Barcelona FIR Using Genetic Algorithms

Héctor Fernández, Marçal Ferrer, Ben Thomas

[a]*Escola d'Enginyeria de Telecomunicació i Aeroespacial de Castelldefels, Castelldefels, Spain*

Abstract

The main objective of this project is to optimize the sectorization of the Barcelona Flight Information Region (FIR) through the application of a genetic algorithm. Firstly, six complexity parameters, organized into four task areas or objectives affecting Air Traffic Control (ATC) workload have been identified, afterwards implementing the most relevant two, and later 3, of these tasks in a weighted sum objective function provided to our algorithm.

Additionally, this work also examines the effect of modifying algorithm parameters such as the population size and crossover fraction.

Finally, a Pareto front has been generated by varying the weight given to our objectives. Through this process, optimal points in the Pareto front have been obtained, helping us identify the most favorable solutions based on our priorities.

Overall, this project has studied how the use of a genetic algorithm could improve the efficiency of sectorization in high traffic areas, as well as to allow for greater flexibility in the criteria used to determine said sectorization.

Keywords: Sectorization, Genetic Algorithm

1. Introduction

Nowadays, ATC is essential in ensuring safe and efficient air travel. With demand, and therefore traffic continuing to grow, the need to enhance ATC operations through the use of efficient sectorization methodologies is crucial to meet said demand.

This project aims to address this necessity by optimizing airspace sectors, which minimize and balance ATC workload, via the use of a Genetic Algorithm (GA) in our case study, the Barcelona FIR. The data used was obtained from Eurocontrol, considering all flights above FL300 on a day of July of 2008 in a 15 minute interval with high traffic.

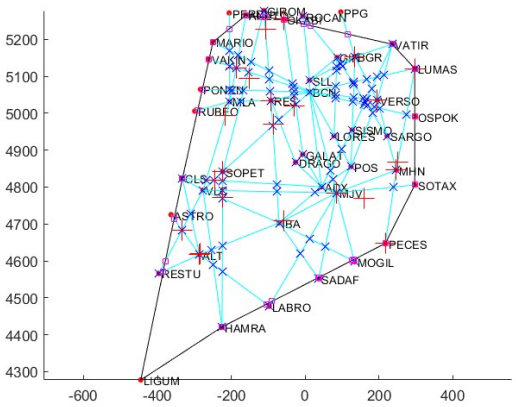

Figure 1: Inital Data

1.1. Constraints

A maximum of 15-17 aircraft per sector have been contemplated in order to maintain safety thresholds, given that our data consists of 87 aircraft inside the FIR, this results in a minimum of 6 sectors needed. However, due to the existence of areas with a very high density of aircraft, 7 sectors have been decided.

1.2. ATC workload

Given that our aim is to minimize ATC workload, we must first define and compute the variables which affect it. In this project, six complexity parameters are considered, which are organized into four ATC tasks:

- **Background tasks**: Airway length and sector area.
- **Transition tasks**: Number of transfer points between sectors and FIRs.
- **Recurring tasks**: Number of aircraft per sector.
- **Conflict tasks**: Number of airway intersections (conflict points).

Having defined the four main tasks that make up the ATC workload, an objective function (to be minimized) for each of them is defined. These consist of computing the Relative Standard Deviation (RTSD) for our complexity parameters. This is done for two reasons: Firstly, minimizing the RTSD will result in balanced values of our complexity parameters between sectors, and secondly, the values will be normalized and able to be compared between them in spite of originally having very different magnitudes.

The number of transfer points between sectors, however, has not been implemented to the objective function through the

RTSD, this is because it's the only value which its total value is not constant, thus minimizing the RTSD of this parameter would not necessarily result in a low amount of transfer points. In this case, we've opted for minimizing the sum of all the transfer points between sectors.

With regard to background and transition tasks, their objective functions are composed of a weighted sum of two parameters. The weight of each parameter has been determined by what we consider to be more time-consuming or contribute more to workload.

Specifically, in the objective function of background tasks, we've considered a 70-30 relation with sector area and length, as we estimate that large areas are could be significantly more difficult to cover in case of adverse meteorological conditions, as opposed to monitoring long airways, which would not require a lot of additional workload.

In transition tasks, we've considered a 50-50 relation between transfer points between sectors and FIRS, as we approximate the time to complete these tasks to be the roughly same.

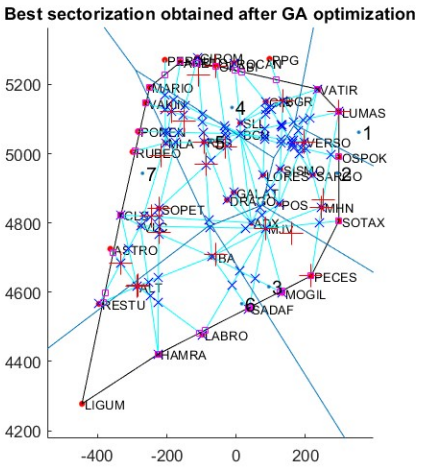

Figure 2: 2 objective sectorization result

2. Objective Function

Once our four partial objective functions have been decided, they have to be implemented to the genetic algorithm in order to obtain a solution. However, these partial objectives are competing with each other, meaning that optimizing one of them, results in worsening the others. Thus, to obtain a viable solution, we need to prioritize some of them.

2.1. 2-task objective function

Firstly, only two partial objectives have been taken into account, resulting in the following objective function:

$$Obj = \alpha Obj_1 + (1 - \alpha) Obj_2 \tag{1}$$

Where α is a parameter between 0 and 1 which determines the weight of each objective in the function. Note that the partial objectives of background and transition tasks were obtained in the same manner.

In this case, we've chosen to prioritize the number of aircraft per sector as well as the number of conflict points as our partial objectives. This is because we've identified conflict resolution as the most time consuming and complex task ATC has to solve, furthermore, the 17 or less aircraft per sector threshold needs to be maintained.

With regard to α, we've assigned it a value of 0.6, therefore having a 60-40 relation in favour of conflict resolution. This has been done because the number of potential conflicts increases exponentially with the number of airway intersections, as a lot of intersection points close together result in the creation of subsequent conflicts when trying to solve the initial ones. The results of applying this values are the following:

- **Background tasks**: 43.53654
- **Transition tasks**: 91.12649
- **Recurring tasks**: 34.09101
- **Conflict tasks**: 41.36035

It can be observed that the two objectives introduced in the objective function are the ones with the lowest values, thus they've been minimized at the expense of the other two.

2.2. 3-task objective function

When using three tasks, we've added the number of transfer points between sectors and FIRs, as we consider this parameters to be more relevant than background tasks, this is because transfering tasks require an active action from ATC, instead of passively being monitored, which is the case for background tasks. In this case, all objectives are given the same weight (1/3). The result of this scenario is the following:

- **Background tasks**: 48.33101
- **Transition tasks**: 72.25462
- **Recurring tasks**: 37.12128
- **Conflict tasks**: 47.65388

When comparing with the previous result, a decrease in the value of transition tasks is observed, as well as a slight increase in all other objectives. This, as previously mentioned, is because these objectives are in competition with one another. The conclusions that can be extracted from this comparison is that trying to optimize more objectives leads to less optimal solutions.

3. Study of Algorithm parameters

In order to study the effect multiple parameters have on the genetic algorithm in an absolutely robust manner, it should be analyzed the effect each one of them has not only on the result

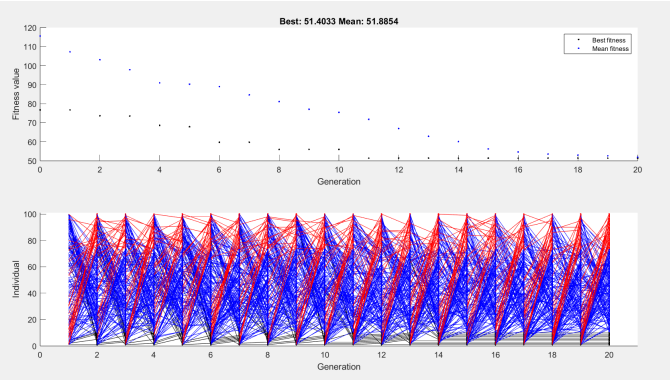

Figure 3: Population size = 100

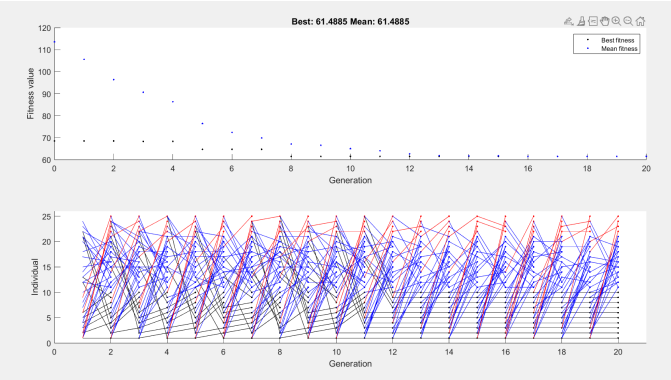

Figure 4: Population size = 25

but on every single other parameter, taking into account all situations at once. However, due to the lack of computational resources it will be considered the effect each variation provokes in the results individually.

The variations to be studied not only include the numerical results the algorithm returns once the program is run, but the total run time also, regarding it's heavy importance in the real word. It would, of course, be optimum to have large population sizes and number of generations, but not realistic if the results are obtained in a non acceptable time-frame. The user has to adapt to their resources and their needs in order to obtain their best code, there is not one greatest solution due to the trade-offs that are produced when attempting to optimize specific fields.

3.1. Population size

As stated beforehand, it would be optimum to have the largest population size possible due to the results being the most optimum in that case. However, run time increases importantly and is the limiting factor. Another factor to consider is how the mean population fitness level compares to that of the best individual. In the case of a higher population size, it should converge less for the same amount of maximum generations, as the "pull" the algorithm has to bring the each individual towards the best is distributed between more individuals. For lower population sizes, using the same argument, it should be observed a higher convergence at equal conditions (see Figures 3 and 4)

To visualize the latter, the population size has been doubled in the first case, and halved in the second. The results are as expected, with (in general) much lower standard deviations and better fitness values in the first case but less convergence than in the second.

3.2. Maximum generations

The effect the maximum generations has is similar to that of the population size. The general solution will be more optimum for higher values of the parameter, but the running time will be increased also. They differ in the fact that average fitness level will be more convergent if the algorithm is given more "time" (in the form of generations). It is worth noting that the genetic algorithm cannot continue optimizing for eternity, therefore there comes a point where the user has to decide if the possible extra optimization provided by the extra generations is worth the compilation time.

To quantify the differences, the code has been compiled for 30 and 10 generations, the differences are as expected.

3.3. Crossover function

The crossover parameter specifies what part of the population is composed by children that have arisen from parents, not been subjected to random mutation. Let it be considered first the extremes:

- When the parameter goes to 0, all children except elite arise from random mutation, leading to a faster run speed. Consequently, convergence is very fast but the fitness value is not optimum.
- If the parameter goes to 1, all children arise from parents and not from random mutation. The run time for this method is higher that for it's counterpart, but the optimization is better for the same amount of maximum generations as it does not rely as much on pure randomness. The mean fitness value, however, converges less towards the best.

The optimum value may lie somewhere in-between the extremes, as it should be considered that the maximum number of generations can be altered in order to decrease the run time, and increase optimization while CF is increased.

3.4. Elite Count

The elite count parameter sets the amount of individuals that are passed on to the next generation without modification. Let it be first considered both extremes:

- If the count is very close to the population size, the run time is very short (the computer processes only a small part of the population size) and the optimization lacks much sense. Convergence would be high, as all elites are very similar.

- If the count is close to 0, the run time is very long (the computer has to process practically all the population size) and convergence is also similar, due to the fact that the elite influence is not large, the solution can be used but there is room for optimization.

All of the other cases lie in the middle ground between both the latter ones. It is most convenient to define a number of elite that accounts for a small (but not irrelevant) percentage of the population size wherein the solution will be more optimum even though there is a little less convergence. Run time also has to be a factor to consider, noting that is decreases when the EC goes to the population size.

4. Multi-Objective GA Optimization (Pareto Study)

In this section, the objective is to achieve the pareto graph of the topic using two methods. Pareto points are those that have the minimum value of the objectives shown before.

4.1. Method 1

In this method the pareto graph is obtained by changing the variable α in order to vary the weight of each of the objectives. In this case, only objectives 3 and 4 will be taken into account. The pareto graph is as follows.

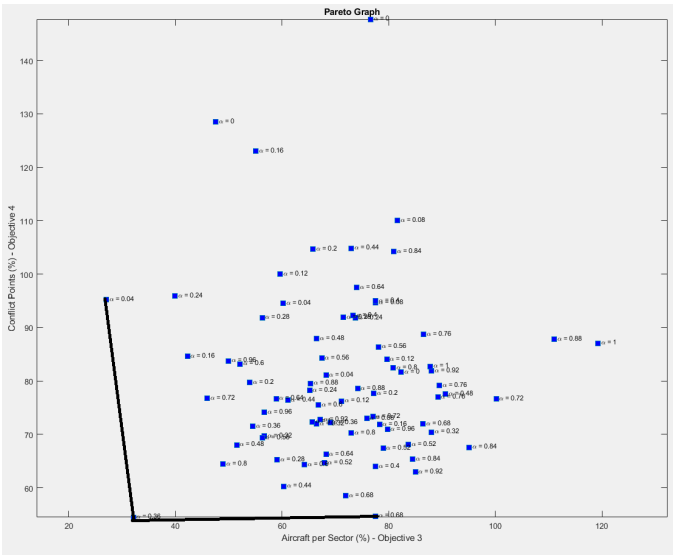

Figure 5: 2D Pareto Graph - Method 1

At every point's side there is the value of α that generated it. In this case, 25 different values of α were taken into account from 0 to 1, and also 3 runs of the genetic algorithm were made for every value of α. This leads to a total of 75 runs of the GA. In the figure 5 it can also be seen the pareto frontier drawn in a black line.

Note that the values that minimize the most both objectives are the values of $\alpha = 0.04$, $\alpha = 0.36$ and $\alpha = 0.68$. The value of $\alpha = 0$ is not considered because it does not take into account the objective 4. These values of α take values from 0.04 to 0.68, thus the range of values is large. Moreover, the value that minimizes the most both objectives is the value of $\alpha = 0.36$. This value gives more weight to objective 3 (see 1, where obj1 and obj2 are objective 4 and objective 3, respectively). For this value of α, the percentage of the relative standard deviation (%) of the aircraft per sector is around 30 % while the RSTD of conflict points is around 50%. Furthermore, the value that minimizes objective 3 the most is the value of $\alpha = 0.04$, but it does not minimize the objective 4. The values of $\alpha = 0.36$ and $\alpha = 0.68$ minimize the most objective 4 but the value of 0.68 does not minimize the objective 3.

The main drawback of this method is probably the computation time. The graph took around 40 minutes to finish up (parameters: MaxGen = 8 and PF = 20, if these values are smaller, the time taken to complete the plot will also be smaller but the solutions will not be that optimal).

The optimal number of sector is 7 because the number of aircraft per sector is well shared between all sectors, impeding to surpass the maximum number of planes per sector (17). Recall that the number of aircraft for the simulation (from 7.55 to 8.10) is 117 and, when dividing it per 17 (max number of aircraft per sector), the number is 6.88, so the number of sector should be 7 or more. In this case, the optimal number is 7 according to aircraft per sector and conflict points.

4.2. Method 2

This methodology consists in using the *gamultiobj* function in MATLAB. This function is a multiple objective genetic algorithm.

This method will be used in two separate ways. The first one is to consider only two objectives (in this case, 3 and 4) plotting a 2D graph. The other way is adding objective 2 to get a 3D pareto graph. The plot shown below (Figure 6) shows the 2D graph involving objectives 3 and 4:

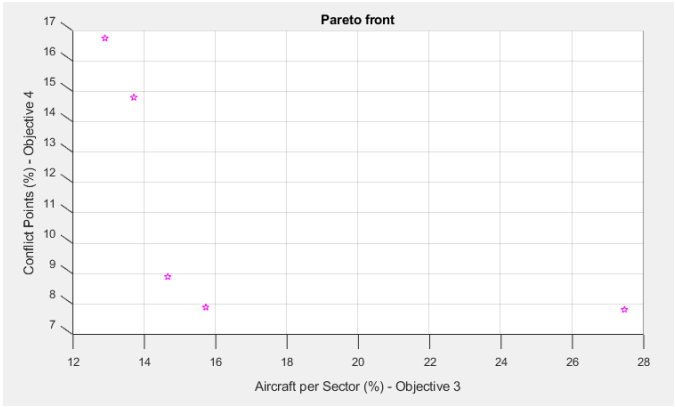

Figure 6: 2D Pareto Graph - Method 2

And, finally, when adding the third objective (objective 2 in the case of this project), the 3D graph that results is as can be seen in Figure 7:

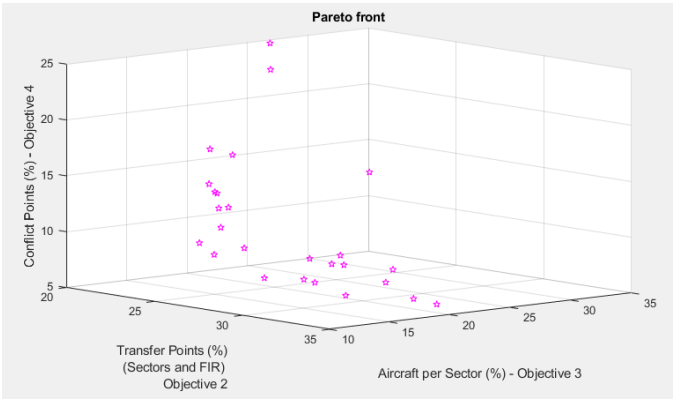

Figure 7: 3D Pareto Graph - Method 2

Note that, for both of the graphs the time taken to complete them was also considerable (for the plots show, the value of the max generations was 20).

Again, the optimal number of sectors is 7.

Furthermore, if one decides to vary the value of some of the GA parameters such as maximum generations, population size or pareto fraction, the following happens:

If the maximum number of generations are changed, the time taken to complete the plot is proportional to the change. For example, if the max generations are smaller, the time necessary will also be smaller. The number of points plotted will also be smaller as the number of generations decrease.

On the flip side, if one decreases the population size (note that the PS has to be always greater than Elite Count parameter), the time taken to finish the graph decreases considerably (the time taken on each generation becomes smaller) as well as the points plotted.

Eventually, if one decreases the pareto fraction, the time taken of the generation does not change at all but the results are quite different. The solutions found are not as optimal as the solutions obtained with a greater value of PF. The points plotted at the graph also decrease.

5. Summary and conclusions

Overall, this project aims to improve the Barcelona FIR sectors using a process called Genetic Algorithm that is based on natural selection. It focuses on the workload of air traffic control and finds a balance between competing jobs. The study also checks the algorithm settings to see the trade-offs between quickness and quality of optimization. During all this study, the optimal number of sector found was 7, based on the maximum number of aircraft per sector (17 considered in this case) and the total number of aircraft in the simulation (117 in the simulation from 7.55 to 8.10).

The study also uses Pareto analysis to find the best solutions, showing the importance of balance in workload. This analysis is performed taking into account two different methodologies as shown above. When comparing the two methods, the *gamultiobj* function (multiple GA) demonstrated a better efficiency than the method 1, which was based on varying the weight of the objectives (while varying the α parameter). This last method took several minutes to complete and aimed only to optimize two objectives while the multi-objective genetic algorithm permits to focus on multiple objectives simultaneously.

The project gives new insights into using genetic algorithms for effective airspace management, especially in busy airspace, showing potential for better air traffic control operations and meeting future flight needs. In short, this work can be a a useful tool for future projects aimed at improving airspace management with advanced optimization techniques. Furthermore, this topic is expected to become more crucial over time, primarily in the busy and congested European airspace. Here, there are certain areas where reducing sectors is not doable, thus alternative solutions should be explored. One of these solutions may be genetic algorithm optimization

References

[1] MATLAB. Matlab genetic algorithm. https://es.mathworks.com/help/gads/ga.html.

[2] MATLAB. Matlab multiple objective genetic algorithm. https://es.mathworks.com/help/gads/gamultiobj-plot-vectorize.html.

[OpenAi] OpenAi. Chat gpt. https://chat.openai.com/.