# A tool to design Functional Airspace Blocks

*Charles-Edmond Bichot, Nicolas Durand*
LOG (Laboratoire d'Optimisation Globale)
DSNA — ENAC
7 avenue Édouard Belin, 31055 Toulouse, France
lastname@recherche.enac.fr

## Abstract

This paper focuses on the selection of a technical tool for the establishment of functional airspace blocks in Europe. This paper shows that the creation of functional airspace blocks is a partitioning problem. Some state-of-the-art partitioning libraries and two metaheuristics are applied to this problem. Comparisons have been made between these libraries and metaheuristics. Results show that the Fusion Fission metaheuristic performs better. The purpose of this paper is not to give the best partition of Europe into functional airspace blocks. It only presents a study which compares and suggests different tools for the establishment of functional airspace blocks in Europe.

## Keywords

functional airspace blocks, partitioning, metaheuristics

## 1 Introduction

In the context of the Single European Sky Regulations of March 2004 [549], the European Commission has issued a mandate to the Eurocontrol agency: "Mandate on Support for Establishment of Functional Airspace Blocks (FABs)". The main idea of the Single European Sky (SES) Regulation about the establishment of FABs is to enforce regional cooperation, with the intention of enabling air traffic control to operate efficiently. The SES Regulation calls upon States to reconfigure upper airspace (upper flight level 285) into FABs.

There is no easy definition of what a functional airspace block is. Indeed, the SES Regulation gives some indications by defining what makes a functional airspace block in [549]. It first defines the airspace block as "an airspace of defined dimensions, in space and time, within which air navigation services are provided". Then it defines a functional airspace block as an airspace block which needs to fulfil three overriding attributes which are central to its functionality:

- designed on the basis of operationals requirements;
- more integrated management of the airspace;
- free from the constraints of national borders.

The SES Regulation recalls that the objectives of the single European sky are:

- to enhance current safety standards and overall efficiency for general air traffic in Europe;
- to optimize capacity;
- to minimize delays.

Furthermore, FABs shall respect seven criteria [551]:

1. be supported by a safety case;
2. enable optimum use of airspace, taking into account air traffic flows;
3. be justified by their overall added value, including optimal use of technical and human resources, on the basis of cost-benefit analysis;
4. ensure a fluent and flexible transfer of responsibility for air traffic control between air traffic services unit;
5. ensure compatibility between the configuration of upper and lower airspace;
6. comply with conditions stemming from regional agreements concluded within the ICAO, and
7. respect regional agreements.

Within the framework of the Eurocontrol mandate, a stakeholder consultation has been made, to support the establishment of FABs. The stakeholder consultation indicates a level of unease over the mechanism for assessing each FAB against the seven criteria presented above [Eur05].

The issue of reconfiguring airspace into FABs is addressed in the airspace part of the SES Regulation, and it is clear form the definition that the design of airspace is a key issue in the development of FABs. However, it is at the same time widely recognized that FABs are much more than an airspace design issue. The establishment of FABs, like any regional cooperation in ATM, is a complex process [Hal05].

This article focuses on how to optimize functional airspace blocks design. It presents a powerful tool for searching FABs configurations. This tool is based on a new method, called Fusion Fission, which is described in [BADB04]. This paper is organized as follows: Section 2 describes which problem this article focuses on and tries to resolve. The functional airspace block optimization process is presented more formerly in section 3. Section 4 presents some state-of-the-art partitioning libraries which are adapted to the FAB problem. The Fusion Fission method is shortly described in section 5. Section 6 shows a comparison between the Fusion Fission tools results and other state-of-the-art softwares. After the conclusion, the Fusion Fission FABs result is drawn in the annex.

## 2 Optimizing FABs

The study was done at the LOG[1] laboratory on the Functional Airspace Block Optimized Process[2]. The FABOP project is a study in the "strategic" level of air traffic control, *i.e.*. methods are applied long before any tactical control of aircraft. The FABOP project respects the definition of what a functional airspace block is. Thus national borders are not taken into account, when operational requirements and management of the airspace are taken into account.

The FABOP project aim is safety. To increase safety in Air Traffic Control, a solution consists in decreasing the workload of air traffic controllers. The air traffic controller workload depends on three factors: traffic supervision, conflicts resolution and coordination [All96]. The goal of FABOP project is to minimize the coordination load between controllers. To minimize the coordination, an algorithm of Fusion Fission is used. Thus, by minimizing the coordination load, the safety is increased.

Each air traffic controller supervises a limited space, called an air traffic sector. Controllers have qualifications to work only on a set of sectors, which is a functional airspace block. The FABOP project consists in cutting the European airspace into FABs. Currently blocks almost never cross countries borders. A new organization of blocks only based on flows of aircraft and not on borders is studied. Because "it is well known that controller-controller coordination is easier and more effective inside an air traffic control unit (a block) than between air traffic control units" [Hal05], the FABOP project searches to maximize flows of aircraft inside FABs and further to minimize flows of aircraft between FABs.

Functional airspace blocks shall respect seven criteria (see section 1). Some criteria are technical, some others are political. The FABOP project only tries to solve technical criteria. Because the FABOP project aims at enforcing safety, the first criterion is respected. The FABOP project works on European flight plans. These data are converted into flow of aircraft between sectors. Thus the second criterion is respected too. The FABOP project includes the cost-benefits analysis of [Eur06] for the sizes of FABs, which is the third criterion. Because the Fusion Fission algorithm minimizes the coordination, the fourth criterion is respected. The lower airspace is taken into account by the FABOP project, which respect the fifth criterion. And the last two criteria are clearly political.

The study area presented in this paper is limited to the air traffic above eleven states: Germany, France, United Kingdom, Switzerland, Belgium, Netherlands, Austria, Spain, Denmark, Luxembourg and Italy. This area is called the country core area in [BA05].

A recommendation of the European organization for the safety of air navigation, Eurocontrol, is to create FABs of around 25 sectors [Eur06]. Let $Card_{opt}$ be this number of sectors, the optimal cardinal of the number of sectors per FAB. A second recommendation is to create FABs with a maximum of 50 sectors. Then, because there are 640 sectors in the study area, approximately 26 blocks have to be defined.

## 3 The Air Traffic Control partitioning problem

The SES Regulation idea is to partition the European airspace into FABs. Thus the FABOP project uses partitioning methods to design the FABs.

The airspace composed of FABs, sectors, and flows of aircraft can be viewed as a graph. The graph vertices are air traffic sectors and the edges are flows of aircraft between sectors. The weight of a vertex is the number of aircraft which are going through the corresponding sector in a day. And the weight of an edge is the num-

[1] Laboratoire d'Optimisation Globale, www.recherce.enac.fr

[2] FABOP, www.recherche.enac.fr/opti/FABOP

ber of aircraft which are going from one sector of the edge to another. The weight of a vertex can be different than the sum of the weights of its connected edges, because arrival and departure of aircraft. Arrivals and departures could not be considered as a sector, because its size would be too large. Some aircraft are going out of the states of the study area. These aircraft are also counted in sectors weights. Thus, the FABOP problem is a graph partitioning problem.

Therefore, the creation of FABs into the European airspace is a graph partitioning problem. Classically, the graph partitioning problem consists in dividing the vertices of a graph into several parts such that the number of vertices in each part is roughly the same and the number of edges-cut is minimized. Because FABs can have up to 50 sectors, with an average of 25 sectors, the number of vertices in each part can be very different. The ratio of the maximum number of sectors of a part to that of the average number of sectors is called the imbalance of the partition. Then, the vertex's number imbalance is $Imb_v = \frac{50}{Card_{opt}} = 2$. To conclude, the FABOP problem is a relaxed graph partitioning problem into $k = 26$ parts with vertex's number imbalance $Imb_v = 2$.

The graph partitioning problem can be formerly defined as follows: Given a graph $G = (V, E)$, where $V$ is the set of vertices and $E$ is the set of edges. $\forall v \in V, w(v)$ is the weight of the vertex $v$, $W(V) = \sum_{u \in V} w(v)$ and $\forall e = (u, v) \in E, w(u, v)$ is the weight of the edge $e$. Find a partition $\pi_k$ of $k$ disconnected subsets $V_1, \dots, V_k$ of $V$ such that their union is $V$, $\forall i \in \{1, \dots, k\}$, $Card(V_i) \leq Imb_v * Card_{opt}$, $W(V_i) \leq Imb_w * \frac{W_V}{k}$ ($Card(V_i)$ is the number of vertices of the set $V_i$) and the sum of the weights of the edges crossing borders parts is minimized. The sum of the weight of the edges crossing border parts is returned by the $cut$ function:

$$Cut(\pi) = \sum_{i,j,i \neq j} Cut(V_i, V_j)$$

where,

$$Cut(V_i, V_j) = \sum_{u \in V_i, v \in V_j} w(u, v)$$

Because the FABs constraints consists in minimizing aircraft flows between FABs and maximizing aircraft flows in FABs, the $Ncut$ objective function is better designed than the $Cut$ objective function. Indeed, the $Ncut$ objective function [SM00] minimizes the sum of the weights of the edges between parts while maximizing the sum of the weight of vertices of each part.

$$Ncut(\pi) = \sum_i \frac{Cut(V_i, V - V_i)}{W(V_i)}$$

Because there is no recommendation from Eurocontrol about the weights (*i.e. the number of aircraft* ) of the blocks in [Eur06], this paper assumes that the imbalance of the weight must not be larger than the imbalance of the number of vertices. Thus, the imbalance of the weight, $Imb_w$, must be lower than 2.

To summarize, the creation of FABs in Europe is a graph partitioning problem. The number of FABs is roughly 26 regarding the paper study area, and each FAB must have less than 50 sectors.

# 4 Presentation of some state-of-the-art partitioning libraries

Some state-of-the-art graph partitioning packages have been searched to solve the FABs graph partitioning problem.

## 4.1 The METIS library

The METIS library [KK98] of George Karypis and Vipin Kumar from the department of computer science and engineering at the university of Minnesota, USA, is used.

The METIS library is the most popular graph partitioning library. It is very often used to make some comparisons between graph partitioning problems. This library is based on the Multilevel method. It provides two graph partitioning applications, `pmetis` and `kmetis`. Their objective is to partition a graph into k equal-size parts while minimizing the $Cut$ objective function. `pmetis` uses a recursive bisection, and `kmetis` uses a multi-way algorithm.

This library is not directly designed for the FABs problem, because it minimizes the $Cut$ objective function and not the $Ncut$ objective function and moreover because the imbalance of the partition is not an input of the algorithm.

## 4.2 The GRACLUS library

The GRACLUS library of Inderjit Dhillon, Yuqiang Guan and Brian Kullis [DGK04] from the department of computer sciences at the university of Texas, USA, is based on the METIS library and a kernel k-means algorithm.

The `graclus` application is an adaptation of the METIS library. It can be set to minimize the $Ncut$ objective function. Thus the `graclus` application is well designed to the FABs problem. After some observations, to obtain results as good as possible, the number of local search step is set to 4, while default is 0.

### 4.3 *The JOSTLE library*

The serial JOSTLE library of Chris Walshaw [Wal02] from the university of Greenwich, UK, is another state of the art partitioning package. It is probably, at the time of writing (January 2007), the best graph partitioning software which minimizes the $Cut$ objective function.

The `jostle` software has a customizable balance tolerance. This tolerance range is from 1. to 1.5. Thus, the JOSTLE library is not perfectly designed for the FABs problem, but is better designed than the METIS library. The lowest $Ncut$ value of the graph returned by the `jostle` software has been searched. There is 51 possible partitions into 26 parts corresponding to the different imbalance parameter values. Results show that the imbalance parameter for which the $Ncut$ objective function returns the lowest value is $1.37$.

### 4.4 *The SCOTCH library*

The SCOTCH library of François Pellegrini [Pel06] from the LaBRI (Laboratoire Bordelais de Recherche en Informatique) at the university of Bordeaux, France, gives a complex software, but very powerful.

This is a high customizable library. The partitioning program is based on a dual bisection mapping algorithm, which uses a Multilevel algorithm as a bisection strategy. Partitions can be refined with different methods: Fiduccia-Mattheyses, Gibbs-Poole-Stockmeyer or greedy graph growing methods. The algorithm searches to decrease the $Cut$ objective function, with a customizable imbalance of the weight of the parts.

It occurs that the defaults parameters give good results. The selection operator was not used in the strategy of the dual recursive bisection. The imbalance ratio of the Fiduccia-Mattheyeses method was changed, which is originally set for a perfectly balanced partition.

### 4.5 *Simulated Annealing*

Metaheuristics are useful tools to resolve problems for which specific algorithms have not been designed, or are not powerful. One of the earliest but always very powerful metaheuristic, is the Simulated Annealing. The algorithm of Simulated Annealing of this paper is based on the article of David Johnson, Cecilia Aragon, Lyle McGeoch and Catherine Schevon [JAMS89]. The authors present an application of Simulated Annealing on the unweighted bisection graph partitioning problem. Thus, to resolve the FABs partitioning problem, some details were changed in this algorithm.

Because Multilevel algorithms used in the other packages are extremely fast, less than one second of computation, the computation time of the Simulated Annealing algorithm has to be limited. About 2 minutes of computations are allowed to the algorithm. It is not a to long time, but allows relatively good results.

## 5 The Fusion Fission method

Because lots of the state-of-the-art partitioning packages where not well designed for the FABs partitioning problem, a new method was proposed by the LOG (laboratoire d'optimisation globale) laboratory at the DGAC, DSNA/DTI/SDER and ENAC, France. A graph partitioning method which can have large imbalance, and which can be used with the $Ncut$ objective function has been created.

The Fusion Fission method is a metaheuristic, and as many metaheuristics, it comes from the real life. The Fusion Fission method comes from the nuclear process. The nuclear process can be viewed as a matter reorganization in an optimization process which tends to create atoms of great internal cohesion. In the nature, the atom with the greatest internal cohesion is the iron atom $Fe$, with 56 nucleons in a range of 2 to 235 for other atoms. We can imagine that a reorganization of the nucleons of the atoms naturally tends to create iron atoms, if the number of nucleons and the sort of nucleons allows it.

The graph partitioning problem has some similarities with the nuclear process. The objective of the graph partitioning problem is to find a low energy organization of the parts of a graph. Thus, it is easy to compare atoms with parts and nucleons with vertices. Then, like in the nature, the process to find an organization of low energy consists in merging or splitting atoms. So, parts of the partition are successively merged or split by the Fusion Fission process.

The Fusion Fission method consists in repeating a perturbation process of a partition till a stop condition is reached. The perturbation process starts with a partition of the graph and successively applies fusion or fission to the partition.

A nuclear reaction can only be created in a very high temperature plasma. Thus the Fusion Fission method includes a “temperature” which is used to channel the process. The temperature is used to stop the perturbation process after a definite number of pass.

In the nature, when a fusion or a fission is done, some nucleons are ejected by the process. These nucleons join another atoms, or make other fissions if they have a very high energy. The number of ejected nucleons is know with a certain probability, and the “rules” of ejected nucleons for the reactions can be written. Then some “rules” are defined to eject randomly in a fusion or in a fission some vertices of its part. These rules are

| Algorithms | FF | SA | scotch | graclus | jostle | pmetis | kmetis |
|---|---|---|---|---|---|---|---|
| $Ncut$ | 5.85 | 6.04 | 6.11 | 6.25 | 6.29 | 6.88 | 7.40 |
| Vertex imbalance | 1.80 | 2.30 | 1.80 | 1.81 | 1.82 | 1.88 | 1.75 |
| Weight imbalance | 1.87 | 1.92 | 1.72 | 1.75 | 1.37 | 1.13 | 1.03 |
| Computation time | 121s | 122s | $\ll$1s | $\ll$1s | $\ll$1s | $\ll$1s | $\ll$1s |
| Distance from FF | - | 3.2% | 4.3% | 6.4% | 7.0% | 15.0% | 20.9% |

Table 1: Average of the results of 100 permutations of the air traffic flow graph

automatically adjusted during the execution of the algorithm by a self-learning function. A rule is just a probability function to eject a certain amount of vertices of the atom. Vertices which have a proportionally low connection[3] with the other vertices of the same part are ejected first.

The process of fusion consists just in merging two parts. The new part is the contraction of the two former parts minus the random number of vertices "ejected" by the rule. Then, if it is possible, these vertices are merged with a different part with which they are the most connected.

The process of fission is much more complex. To split a part into two new parts, a bisection method must be used. There is a great number of bisection methods. An agglomerative method based on the percolation process was chosen. But before splitting the part, like in the fusion process, some vertices can be "ejected" of the part. Then these vertices can be merged with a partition with which they have big connection, or they can split other parts, in a chained fission reaction.

The Fusion Fission algorithm is more precisely presented in [BADB04].

# 6 Comparisons with state-of-the-art partitioning packages

One day of European traffic can be converted to a graph as it is described in section 3. Such a graph can be built for different days of data. But in this paper only results which are obtain for the Friday, $17^{th}$ June 2005 were presented. The graph is stored using the CHACO graph input file format describes in [HL95]. This format has been chosen for compatibility reasons. The METIS, JOSTLE, GRACLUS and SCOTCH libraries both used this graph input file format.

The aim of this section is to find the graph partitioning package which finds the best European FABs partition. During the graph partitioning process, a method can return a particularly good or bad partition, depending on the input configuration of the graph. To avoid this kind of problem, benchmark were made with one hundred permutation of the initial graph, and then, average and range of the results were returned. A permutation only consists in renaming each vertex of the graph. Indeed,the structure of the graph is unchanged.

All results presented in this paper are found with an Intel Pentium 4, 3GHz processor with 1Go RAM computer, which uses a GNU/Linux Debian operating system. Every specific graph partitioning package partitions the graph in less than one second, but metaheuristics are greatly slower with an average of 121 seconds for Fusion Fission and 122 seconds for Simulated Annealing. The computational times of the metaheuristics was limited. They can find better solutions with more time. We have found that a roughly 2 minutes computation time is enough but not too long regarding results found.

The computational results are shown in table 1. Algorithms were sorted by their $Ncut$ values. First, it can be noticed that metaheuristics have better results than the other algorithms. But the difference of computation time between metaheuristics and the other algorithms can explain these differences. The Fusion Fission algorithm has the best $Ncut$ average. It is just followed by the Simulated Annealing algorithm with an $Ncut$ average which is $3.2\%$ worth than the Fusion Fission. Then come the Multilevel methods. The scotch algorithm is the first of them. Its $Ncut$ average is $4.3\%$ larger than for the Fusion Fission algorithm. Then comes the graclus algorithm. The three state of the art graph partitioning packages which are not design to solve this relaxed $k$-way graph partitioning problem come at the end. The jostle algorithm has a better $Ncut$ average than the METIS package, because it takes into account algorithm has an imbalance parameter.

The vertex and weight imbalance of the partitions found for each algorithm can be shown in figure 1. The horizontal line represents the threshold of $Imb_v \leq 2$ or $Imb_w \leq 2$. The Simulated Annealing vertex average is really above this threshold, and it seems that it has too much weight imbalance above this threshold too. As it was expected, the jostle, pmetis and kmetis algorithms have very low weight imbalance compared to the thresh-

[3] The connection between a vertex and its part is the sum of the weight of the edges connecting this vertex to vertices of the same part

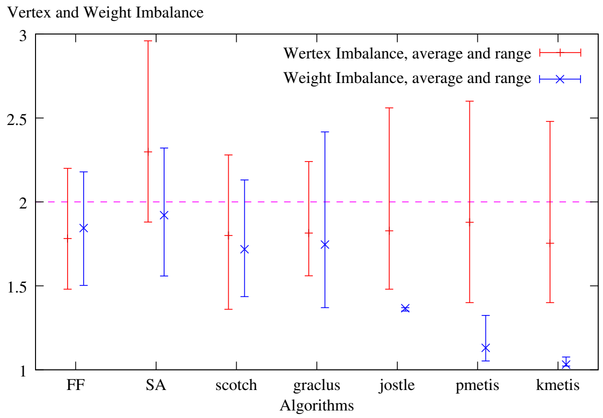

Figure 1: Algorithms average and range results of the vertex and weight imbalance of 100 permutations of the air traffic flow graph

old. As it is explained in subsection 4, this is due to their very strict weight partitioning constraints. However, both of their vertex imbalance have a too big range, with an upper range of approximately $2.5 \gg 2$. Partitions found by Fusion Fission and graclus have good vertex imbalance, both on average and range. Thus, the Fusion Fission algorithm most respects the FABs partitioning problem constraints.

The Fusion Fission algorithm not only found partitions for a fixed number of parts, but for a range of values around this number. Then, when the Fusion Fission algorithm resolves the FABs partitioning problem, it finds partitions with a partition cardinal range: $\{19, \ldots 30\}$. And, for more than $90\%$ of the graph permutations, a partition cardinal range: $\{18, \ldots 31\}$. To compare all of the partitions found by the Fusion Fission algorithm with other partitions, the two better Multilevel algorithms, scotch and graclus, computed the one hundred permutations of the air traffic flow graph for a number of parts btween 18 to 31. Results of the computation of the Fusion Fission algorithm with a number of part, 26 and for the 13 computations of the scotch algorithm and for the 13 computations of the graclus algorithm are displayed figure 2. In this figure, each algorithm results are presented as a ratio of the $Ncut$ average to that of the $Ncut$ average of graclus. The partitions found by the Fusion Fission algorithm are better than those of scotch for a number of parts btween 22 to 28. And the partitions found by the Fusion Fission algorithm outperforms the partitions found by graclus for a number of parts between 21 to 29.

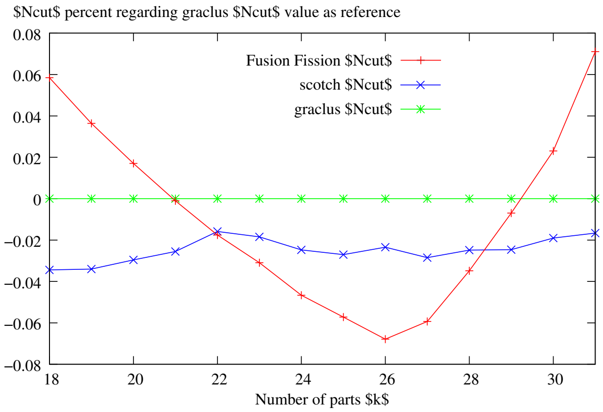

Figure 2: Comparisons between Fusion Fission results for $k = 26$, with GRACLUS and SCOTCH results for $k \in \{18, \ldots, 31\}$. Each algorithm results are presented as ratio of the $Ncut$ average to that of the $Ncut$ average of graclus.

# Conclusion

This paper focuses on technicals tools for the establishment of functional airspace blocks in Europe. This paper shows that the creation of functional airspace blocks is a partitioning problem. The state-of-the-art partitioning libraries METIS, GRACLUS, JOSTLE and SCOTCH have been used to solve this problem. Because these libraries where not perfectly designed to solve this problem, due to imbalance constraints, two metaheuristics have been used. The first is a simulated annealing method applied to graph partitioning. The second is a Fusion Fission method applied to graph partitioning. Comparisons have been made between these methods. The result is that the Fusion Fission performs better than the other methods.

The purpose of this paper is not to give the best partition of the Europe into functional airspace blocks. It only presents a study which compares and suggests different tools for the establishment of functional airspace blocks in Europe. Figures that can be shown in the annex never attempts to be the perfect solution for the establishment of functional airspace blocks in Europe. They just show what may be the single European sky with functional airspace blocks. Because some of the seven criteria presented in the introduction are not respected or not fully respected, tools presented in this paper must be enhanced to respect all of this criteria.

| Partitions | Number of part | $Ncut$ | $w_{imb}$ | $v_{imb}$ |
|---|---|---|---|---|
| The 6/17/2005 partition | 55 | 25.10 | 4.64 | 3.33 |
| The best FF partition | 26 | 4.87 | 1.69 | 1.64 |

Table 2: The European airspace partition and the best Fusion Fission partition

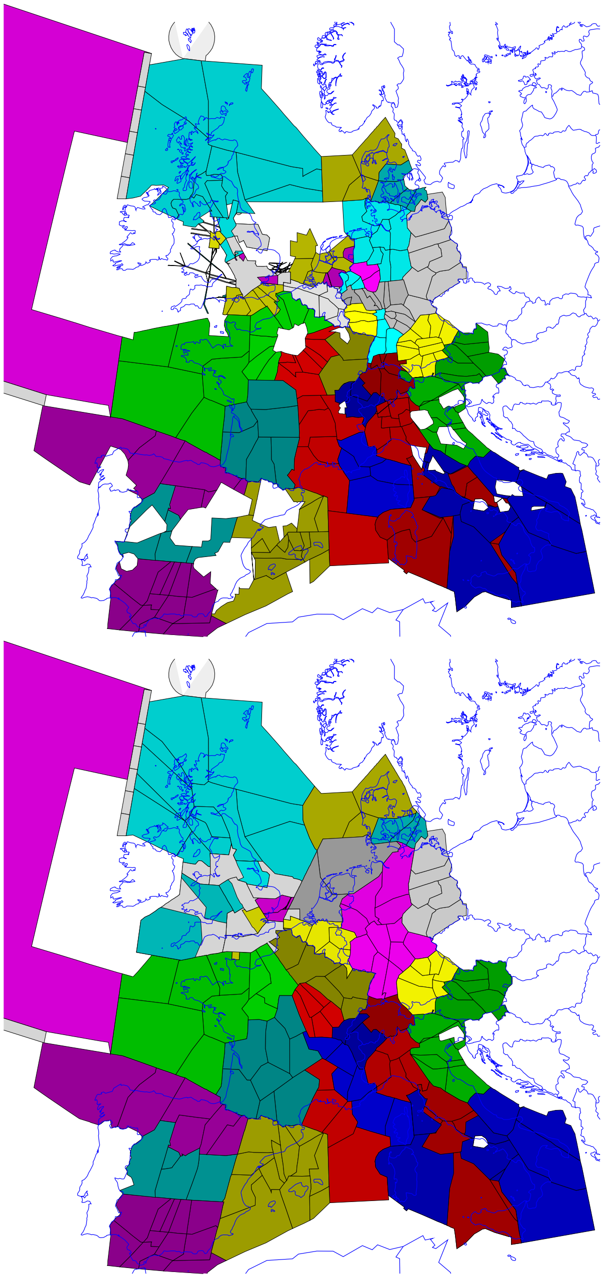

Figure 3: The real European airspace partitioning at flight levels 180 and 280

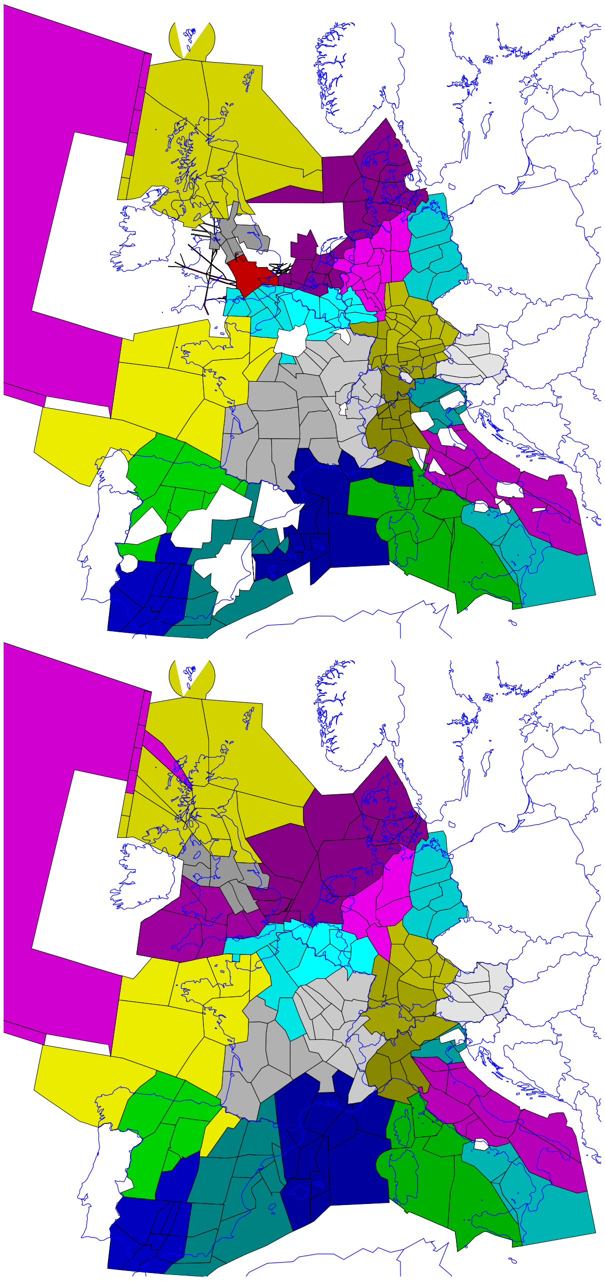

Figure 4: The best European airspace partition found with the Fusion Fission algorithm at flight levels 180 and 280

# Annex: one of the results found

The European airspace area presented in this paper is actually partitioned into 55 blocks. Table 2 presents a comparison between the European airspace partition of June, $17^{th}$ 2005 and the best partition found by the Fusion Fission algorithm for 26 parts. Because the numbers of parts of the two partitions are very different, the $Ncut$ comparison is not relevant. The imbalance of the vertices number of the current European airspace partition into centers of control is really above the Eurocontrol recommendation. The European airspace partition of June, $17^{th}$ 2005 at flight level 180 and 280 can be shown figure 3. The best Fusion Fission partition found is presented by figure 4, it takes four hours of computation.

The European partition into FABs presented by figure 4 is just a solution, it is not the perfect solution and do not attempt to be so. Moreover the criterion used does not take into account all operational requirements and this partition would be operationaly unfeasible. A lot of work remains to be done to apply this algorithm to real world problems.

# References

[549] Regulation No 549/2004 of the European Parliament and of the Council of 10 March 2004 laying down the framework for the creation of the single European Sky, Official Journal of the European Union.

[551] Regulation No 551/2004 of the European Parliament and of the Council of 10 March 2004 on the organisation and use of the airspace in the single European sky, Official Journal of the European Union.

[All96] Jean-Marc Alliot. Techniques d'optimisation stochastique appliquées aux problèmes du contrôle aérien. Mémoire d'habilitation à diriger les recherches (HDR), Mai 1996.

[BA05] Charles-Edmond Bichot and Jean-Marc Alliot. A theoretical approach to defining the european core area. Technical report, LOG - ENAC/CENA, 2005.

[BADB04] Charles-Edmond Bichot, Jean-Marc Alliot, Nicolas Durand, and Pascal Brisset. Optimisation par fusion et fission. application au problème du découpage aérien européen. *Journal Européen des Systèmes Automatisés*, 38(9-10):1141–1173, 2004.

[DGK04] Inderjit Dhillon, Yuqiang Guan, and Brian Kullis. Kernel k-means, spectral clustering, and normalized cuts. In *ACM International Conference on Knowledge Discovery and Data Mining*, 2004.

[Eur05] Eurocontrol final report on european commission's mandate to support the establishment of functional airspace blocks, May 2005.

[Eur06] Eurocontrol. The impact of fragmentation in european atm/cns. Technical report, Eurocontrol, 2006.

[Hal05] Anders Hallgren. Restructuring european airspace: functional airspace blocks. *Skyway*, pages 20–22, autumn 2005.

[HL95] Bruce Hendrickson and Robert Leland. *The Chaco User's Guide*. Sandia National Laboratories, 2.0 edition, 1995.

[JAMS89] David S. Johnson, Cecilia R. Aragon, Lyle A. McGeoch, and Catherine Schevon. Optimization by simulated annealing: an experimental evaluation; part i, graph partitioning. *Operations Research Society of America*, 37(6):865–892, 1989.

[KK98] George Karypis and Vipin Kumar. *METIS : A Software Package for Partitioning Unstructured Graphs, Partitioning Meshes, and Computing Fill-Reducing Orderings of Sparse Matrices*. University of Minnesota, 4.0 edition, september 20 1998.

[Pel06] François Pellegrini. *Schotch and LibScotch*. ENSEIRB - LaBRI, Université de Bordeaux I, 4.0 edition, january 31 2006.

[SM00] Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

[Wal02] Chris Walshaw. *The serial JOSTLE library user guide*. University of Greenwich, 3.0 edition, july 8 2002.

# Author biographies

Charles-Edmond Bichot graduated from the École Nationale de l'Aviation Civile (ENAC) in 2004. He is a Ph.D. student in computer Science at the LOG laboratory of the DSNA/DTI/SDER and of the ENAC.

Nicolas Durand graduated from the École Polytechnique de Paris in 1990 and from the École Nationale de l'Aviation Civile (ENAC) in 1992. He has been a design engineer at the Centre d'Études de la Navigation Aérienne (now DSNA/DTI/SDER) since 1992 and holds a Ph.D. in computer Science (1996). He is currently head of the he Planning, Optimization, and Modelling team of the DSNA/DTI R&D domain.