Dynamic Airpace Configurations Generated by Evolutionary Algorithms

Marina Sergeeva, Daniel Delahaye, Leila Zerrouki, Nick Schede

DYNAMIC AIRSPACE CONFIGURATIONS GENERATED BY EVOLUTIONARY ALGORITHMS

Marina Sergeeva, Daniel Delahaye ENAC Toulouse, France

Leïla Zerrouki, Nick Schede EUROCONTROL Brétigny-sur-Orge, France

Abstract

This paper focuses on the process of generating a sequence of sector configurations composed of two airspace component types Sharable Airspace Modules (SAMs) and Sectors Building Blocks (SBBs). An algorithm has been developed that manages the main features of the dynamic sectors configuration (including sector design criteria). In order to make it run efficiently a pre-processing step will be presented to create a graph modelling of the inputs. Based on this initial graph, a mathematical model is defined which can be summarized by a multi-periods geometric graph partitioning problem. State, space, objective function and constraints will be also presented. Due to the induced complexity, a stochastic optimization algorithm based on artificial evolution is then proposed. A two layer chromosome is used for such a genetic algorithm for which recombination operators are proposed. Evaluation of the algorithm will be presented with a comparison to existing tools and operational approach.

Introduction

This paper presents an ongoing research work performed in the scope of SESAR program to assess the concept of Dynamic Airspace Configuration (DAC) in highly complex free routing areas [1].

The first section of the paper presents an overview of past work related to dynamic sectorisation and dynamic airspace configurations. It helps to distinguish new elements of the approach proposed in this paper.

In order to set-up the operational context and identify the operational objectives and constraints, the second section provides a high level description of the DAC concept in the framework of SESAR program.

The technical solution presented in the third and fourth sections consists in a graph modelling of the inputs and a mathematical model which can be summarized by a multi-periods geometric graph partitioning problem. Due to the induced complexity, a stochastic optimization algorithm based on artificial evolution is then proposed.

The resulting airspace configurations are assessed in section V, by comparing the solution to currently used tools and operational methods. Section VI concludes the paper and gives some perspectives of future work within SESAR program and its potential applications.

Previous Related works

A number of researchers, mainly from Europe and USA have achieved certain results in the development of the solution of the DAC problem [2]. Currently, airspace is partitioned into functional blocks or pre-defined sectors that may be combined into bigger sectors when the traffic is low. Existing support tools enable the computation for the flow management planning phase of the sector opening schemes, which minimize sector overloads based on pre-defined sector configurations. The ICO tool developed at EUROCONTOL is one of them; cf. [3] and [4]. More in a research perspective, a methodology for dynamic sectorisation for highly complex Area has been developed [5]. In preparatory work of SESAR program, “Dynamically Shaped Sectors” early project proposed a heuristic for automated sector design which combines best practice of sectors design and clustering techniques [6].

In some of the existing works the DAC problem was solved using approaches that are based on the initial division of the airspace into small 2D or 3D areas [7]–[9], such as elementary sectors for example. Other researches are concentrated on redesigning the airspace boundaries. Stochastic local search, Mathematical programming [7], [10], Computational geometry [4], Evolutionary algorithms [9], [11]–[13],

Global optimization and several other techniques have been used in prior research on airspace optimization. Several different complexity metrics have been developed [14], [15], and applied [16] in those works in order to evaluate resulting sector configurations.

In [9], the DAC problem is solved by using an improved genetic algorithm (iGA). On the first step of the algorithm a graph model is constructed that represents the airspace static structure. Then, the algorithm solves a graph-partitioning problem. It aims to balance the sector workload under the constraint of ensuring safety. The iGA includes multiple populations and hybrid coding. The algorithm aims to find balanced airspace configurations with an optimal number of sectors. New criteria were included in the algorithm: sector balancing, coordination workload and sector average flight time. The algorithm includes two steps. In the first step, the number of sectors is determined and airspace partition is accomplished. In the second step, sector boundary optimization is implemented. Both of the two steps are done using a hybrid encoding multi objective GA on the basis of the weighted graph model. The method has been proven and is being applied to the airspace of North China.

Authors of [10] have proposed to use relevant air traffic complexity metrics instead of flight counts and sector capacities. In this work possible combinations of elementary sectors are explored, instead of the small subset of pre-defined configurations currently being used, so as to offer the maximum capacity to the incoming traffic. Artificial neural networks are used to predict the sector status from the relevant complexity metrics. In this work the trained network [17] and the most relevant metrics are used to predict when the controlled sector will be normally loaded, overloaded, or under-loaded, and when an airspace configuration should be recombined. The decision to reconfigure sectors is driven by the prediction made by the neural network. A classical tree search algorithm is used to build all the valid sector configurations from an initial set of elementary sectors. The tree search algorithm explores all possible airspace configurations, among which only one is chosen using defined evaluation criterion. The computed configurations were compared to the actual configurations archived by ATC centers.

The next two methods are more freeform and have not used any elements of the current functional airspace block design.

In [18] authors have described a method for partitioning airspace into smaller regions based on a peak traffic-counts metric. The method was used for partitioning the airspace over the United States into 466 sectors, once for each hour of the 24-hour day. The algorithm consists of two phases. In the setup phase, a weighted graph is created based on traffic flow patterns. Topological structure of the airspace (pathways of a flow pattern) can be described as a graph. Nodes of the graph are represented as an averaged position coordinates of flights that enter or leave a given sector through a boundary and links between them represent the flow. The flow is computed as a number of aircraft along a link of the graph, and is assigned as its weight. The weight of the node is computed as a number of aircraft located in the area close to it. The final phase consists of three steps that are repeated in a cycle. During the first step one spectral bisection method splits a given graph into two sub-graphs such that the node-link connectivity is preserved in each sub-graph. On the next step, graphs weight peaks are computed. The sub-graph with the highest weight is selected for further partitioning. The process is repeated until the termination criterion is met.

Authors of [19] have proposed an airspace modeling technique based on the air route structure and a spectral clustering based dynamic airspace configuration algorithm to partition the graph model of the airspace into subgraphs. The performance of the algorithm was validated with real air traffic data. In this method, first, a weighted graph model using the airspace information and air traffic data is built. Then, a novel airspace sectorisation algorithm is applied. The algorithm is based on a graph search algorithm. Each designed sector contains one subgraph, and has a smooth polygonal boundary. Sector boundaries created by this algorithm should satisfy the minimum distance constraint, which means that the main flow paths and intersections should be at a minimum distance away from the sector boundaries. The authors have as well proposed various evaluation metrics of resulting sectorisation. The simulation results have shown that the new sectorisation computed by this algorithm outperforms the currently used sectorisation.

Most of the previously developed approaches do not take into account the stability of the generated configurations over time and some important airspace design aspects. As a matter of fact, one must be able to generate airspace configurations that have minimum changes from one time period to another with reduced airspace complexity, and more balanced workload among airspace sectors.

DAC operational concept in SESAR

In comparison with a sectorisation based on a fixed route network, the SESAR program introduces the User Preferred Routing/free routing concept to enable the airspace users to freely plan and fly the 4-D trajectory that best copes with their business or mission needs. Contrary to a fixed-route network a FRA environment will produce a much larger number of different trajectories for which the dynamic nature and flexibility of the DAC process will cater most efficiently.

Increasing adaptability of Sectors Configurations

The DAC concept in SESAR proposes increasing levels of adaptability of the sectors configuration to match with the capability of each Air Traffic Control Center regarding types of free routes implementation, ATC licensing, and airspace delegation agreements.

Four levels of configurations adaptability have been identified:

DAC Level-1 is the lowest level of airspace adaptability, which provides an improvement to the current operational process at minimum cost. Instead of using predefined sectors configuration, Level-1 uses dynamically computed configurations by combining existing elementary sectors provided by an automated optimization process. In level-1 any sector combination which forms controllable airspace blocks is eligible and may be used during the day of operation.

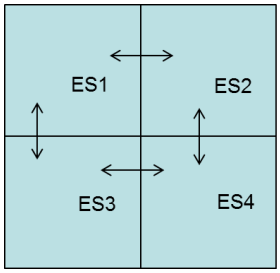


Figure 1. DAC level-1 Elementary Sectors components (top/lateral view)

DAC level-2 increases the adaptability of the airspace to the traffic pattern by delineating from the nominal sectors, a limited number of airspace components that can be recombined laterally and/or vertically within an automated and optimized sector configuration process to form controllable airspace blocks. Two airspace structures are specified in level-2: the Sector Building Blocks (SBB), are permanently busy areas with a high traffic load delineated by recurring traffic patterns, while Sharable Airspace Modules (SAM) are less busy with a temporary high traffic load that can be used in the sector configuration computation process to equally balance the traffic load among the sectors. SBBs in level-2 can be sufficiently large to be workable and controlled separately. SAMs cannot be used separately in a configuration. The number of the elementary sectors resulting from the combination of the airspace components is known and limited. However, the number of possible configurations will be of a greater number than in level-1, which allows a better adaptation to changes of the traffic pattern.

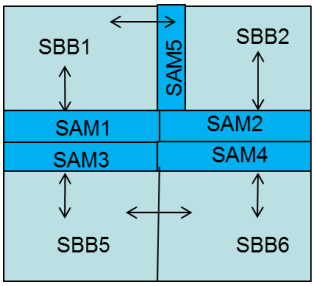


Figure 2. DAC level-2 SBBs and SAMs components (lateral view)

DAC level-3 combines a larger number of SBBs and SAMs. The SBBs are defined by a less structured traffic pattern which makes them smaller and their workability as a separate entity not ensured.

However, the SBB component is preserved as a reference for controllers training and licensing purposes.

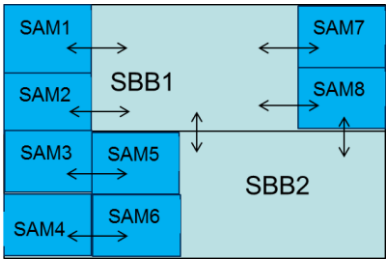


Figure 3. DAC level-3 SBBs and SAMs components (top view)

DAC Level-4 combines only SAMs which are not related to any particular traffic pattern. It is the most challenging level since there will be no more predefined airspace component known and full generic ATC validation is a prerequisite for the implementation of this level.

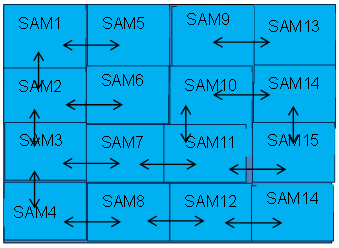


Figure 4. DAC level-4 SAMs components (top/lateral view)

DAC processes overview

Each DAC level is associated with an airspace design approach which produces the airspace components that will be used in the airspace configuration process. Elementary sectors, SBBs and SAMs are the result of a sector design process dedicated to level 1, 2 and 3, while DAC level 4 is itself a sector design process performed dynamically.

Airspace design for level-1 is similar to the current process which produces elementary sectors; however supporting automated tools are proposed at the aim of increasing the efficiency of sector design process [20]. Sector design for level-2 and 3 which delineates SBBs and SAMs is a subject of future research.

Because of the huge number of possible sectors configurations enabled by more modularity of the sector design phase, the definition of sector configurations and opening schemes will require the use of automated support tools. Given the organizational framework of the concerned airspace, the challenge is to organize, plan and manage airspace configurations to meet User Preferred Routing, in a Free Route environment with enough flexibility to respond to changes in traffic demand, to unexpected events including weather, and to any update in airspace reservation within the Advanced Flexible Use of Airspace concept (AFUA) in the optimum way, while maintaining the safety targets.

The Optimum Configuration process will be able to build Configurations of sectors based on the grouping/de-grouping of elementary sectors and reallocation of Sharable Airspace Modules. The process will at least provide either a configuration with the minimum number of ATC working positions without imbalance or, a configuration with the minimum of imbalance considering the number of ATC working positions available.

It is obvious that workload assessment is a key requirement for generating workable sector configurations and more so in a Free Route User preferred trajectory environment where there might be very little historical data available and due to its very nature no possibility of managing trajectories by using published routings.

The workload assessment in FRA must be achieved through evolution/development of tools. These tools must support load assessment in generic airspace defined and managed through SBBs and SAMs. Workload assessment in a free route environment is only possible based on a trajectory that is well described and shared by the NOP.

Figure 5 provides an overview of the sector design and sector configuration processes described above within the overall context of airspace organization and management in SESAR.

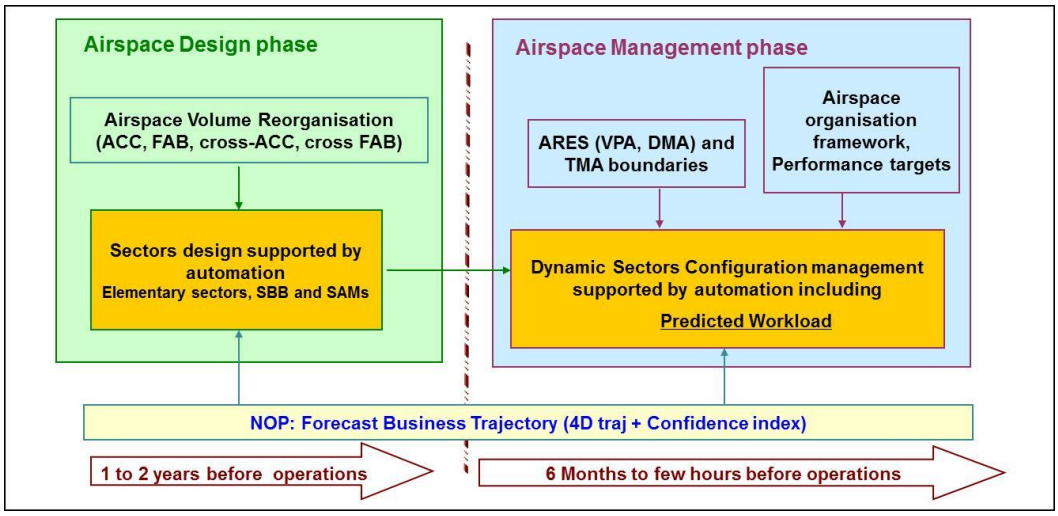


Figure 5. DAC processes overview

DAC models and algorithms

Airspace modelling

The first stage in the optimization process consists of modeling the real problem using a mathematical abstraction which should be as faithful as possible. The modeling stage then consists of characterizing the state space and the objective space.

The state space represents the set of parameters of the system upon which we may act in order to optimize one (or more) objective(s). Examination of the properties of the state space then assists us in the choice of a suitable optimization method. In most industrial optimization problems, variables of the state space must remain within a sub-domain defined by a set of constraints. One very important point that is characterizing the state space is its dimension. Generally, the higher the dimension n of the state space X is, the harder it is to find the optimum.

The objective space represents the set of criteria which we wish to optimize. Within the context of industrial optimization problems, we seek to determine global optima and to avoid being trapped on local optima.

First, we need to model the airspace. Let's consider a given airspace which consists of a known set of 3D airspace blocks (Figure 6).

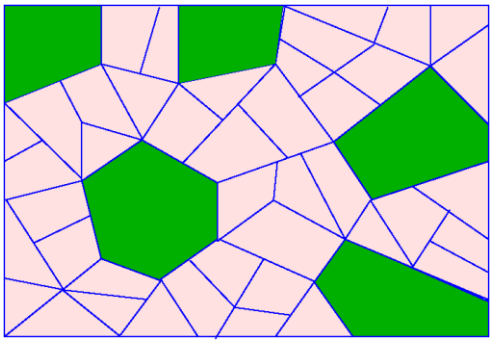


Figure 6. Initial airspace blocks (2D projection). The green blocks are SBBs and the pink ones are SAMs

The airspace area is split into several altitude layers. The shape of those blocks can vary depending on the altitude level. A set of aircraft trajectories that are crossing this area is known as well. For the purpose of DAC, traffic data is taken just for one day. There are two type of blocks used in our model: sharable (SAMs) and non-sharable (SBBs). Each controlled sector is built of at least one SBB. Building of the controlled sector starts from choosing a central block, which can be chosen only among non-sharable blocks. Even if the number of controlled sectors is different in two successive configurations, centers of controlled sectors will be chosen among the same group of non-sharable blocks. This partly ensures continuity between successive airspace configurations.

State space modelling

Our model is based on a weighted graph model of the airspace (Figure 7). Let a graph $G = (N, L)$ represent the airspace, where N is the set of nodes and L is the set of links.

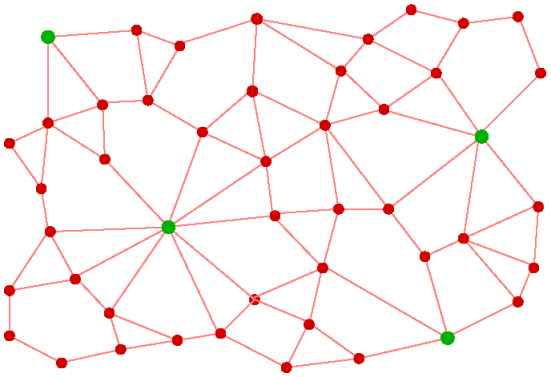


Figure 7. Initial graph. Green nodes represent non-sharable blocks (SBBs)

In this graph each node represents a sharable or non-sharable block and each link represents the relation *is neighbor with* between two blocks, it means that when two blocks share a common vertical or horizontal border (Figure 7), a link is built between them. Weight of the node represents a workload and weight of the link represents the flow between two sectors.

As for the evaluation metric, several of them have been included in the model. In order to estimate the complexity of the traffic in each block a convergence metric [17] is used and for the evaluation of the traffic load - an occupancy count. Occupancy count is computed as a number of aircraft inside the block at each minute. Link's weight of the graph is computed as a number of aircraft crossing the border between two airspace blocks connected by this link. The workload of airspace blocks is computed for each chosen time period, as well as transfer traffic between each pair of neighboring blocks (Figure 8). In order to be able to compute the total number of overloads in configuration, for each minute of chosen day we are computing the number of aircraft in each block.

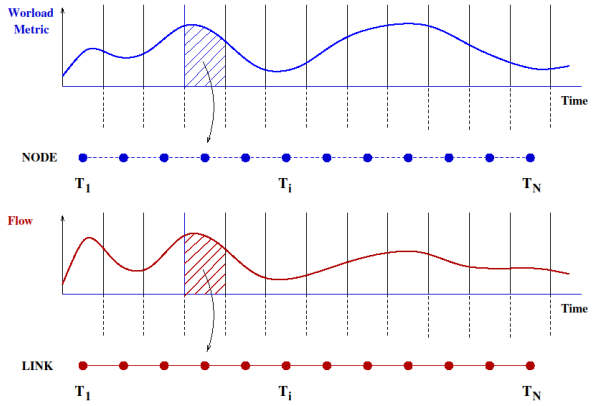


Figure 8. Graph time extension

In order to be able to compute re-entry events inside created controlled sectors, we register the list of airspace blocks crossed by each trajectory with the associated time horizon (Figure 9).

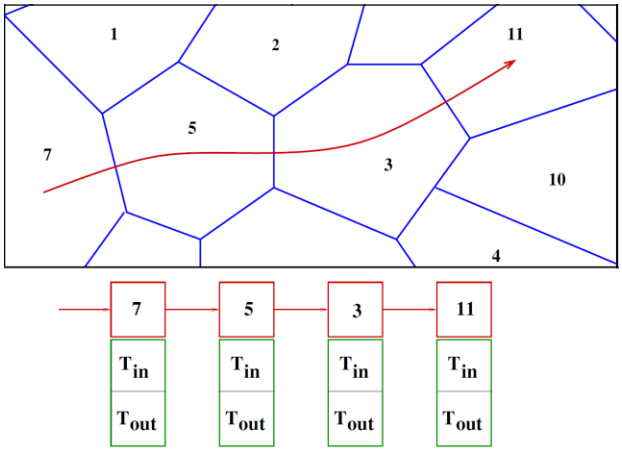


Figure 9. List of airspace blocks associated to a given trajectory

A graph partitioning problem

Based on the airspace model described above, the DAC problem consists in finding an optimal partitioning of the graph into several sub-graphs for each time period. The regular graph partition problem is defined on data represented in the form of a graph $G = (N, L)$, where N is the set of nodes and L the set of links, such that it is possible to partition G into smaller components with specific properties. Having a connectivity constraint on elementary sectors belonging to the same sector, such graph partitioning is a geometric graph partitioning. In this case, nodes belonging to the same component have to be connected. The Figure 10 gives an example of

partition of a graph into three connected components. As it can be seen on the figure, for each pair of nodes belonging to the same component, there is a path connected them for which all the elements of such path (links and nodes) belong to this component. If we note N_k the set of nodes belonging to the component k, the connectivity constraint can be modelled by: $\forall$ pair of node $(i, j) \in N_k$ $\exists$ a path $P = \{l_1, l_2, \ldots l_d\}$ connecting node i and node j for which each link l_i has its origin and destination nodes in N_k.

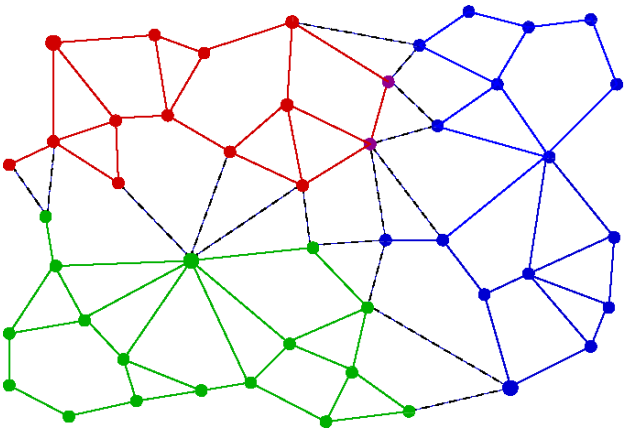


Figure 10. Example of a partition of a graph into three connected components

For a given time period, the state space will be modelled the following way: $X = \{N_1, N_2, \ldots, N_K\}$ where N_i represents the set of nodes belonging to the component i. Based on those sets of nodes, it is easy to determine the links shared by two components (black dash line of the Figure 10). Having a problem with several time periods, such set of nodes are computed for each time period:

$$\begin{cases} t_1 & K_1 & X_1 = \{N_{11}, N_{12}, \ldots, N_{1K_1}\} \\ t_2 & K_2 & X_2 = \{N_{21}, N_{22}, \ldots, N_{2K_1}\} \\ & \ldots & \ldots \quad \ldots \\ t_N & K_N & X_N = \{N_{N1}, N_{N2}, \ldots, N_{NK_1}\} \end{cases}$$

where K_i represents the number of component at time period t_i.

Based on the state space definition, we must model the associated objective function.

Objective function

The objective function as well as the constraints included in our model is designed according to requirements developed in co-operation with operational experts from EUROCONTROL. Five criteria have been included in our objective function for evaluation of a solution.

The first criterion measures the level of the workload imbalance among all controlled sectors in configuration for each time period t_i $(i = 1..N)$. The workload of the controlled sector is computed as the sum of workload of airspace blocks that it is composed of. The workload imbalance of all components (controlled sectors) for the time period t_i is then computed with the following formula:

$$u_i = \sum_{k=1}^{K_i} \frac{\|Wc_{ik} - C\|}{C}$$

where K_i is the number of components in the configuration for period t_i, Wc_{ik} is a total weight of all nodes (blocks workload) included in the component k (a workload of the sector) for period i, C is the sector capacity (a value of the workload that we are trying to obtain for each controlled sector).

Then, the total imbalance associated to the whole planning is given by:

$$U = \sum_{i=1}^{T} u_i$$

where T is a number of periods.

The second criterion measures the total number of overloads in each controlled sector of the configuration. For each one minute of the period we are computing the number of aircraft in the controlled sector, and if this number exceeds a given value during several minutes consequently, we register an overload. Then we summarize the number of overloads of all controlled sectors for each time period:

$$v_i = \sum_{k=1}^{K_i} vcs_i$$

where v_i is the total number of overloads for period t_i and vcs_i is the number of overloads in one controlled sector.

Then, the total number of overloads associated to the whole planning is given by:

$$V = \sum_{i=1}^{T} v_i$$

The third criterion measures the transfer traffic between neighboring components. When two neighboring blocks belong to different controlled sectors, the traffic flow between them is cut by the sector border, this increases the coordination workload. The total flow cut for the time period t_i is given by:

$$f_{c_t} = \sum_{\substack{(i,j)\in L \\ i\in N_{tm}\ j\in N_{tn} \\ m\neq n}} f_{ij}^t + f_{ji}^t$$

where $f_{ij}^t + f_{ji}^t$ - the flow cut between nodes *i* and *j*, in both directions, for the time period t.

The total flow cut for a given planning is then computed with the following formula:

$$F_c = \sum_{t=1}^{T} f_{c_t}$$

The fourth criterion counts the number of re-entry events. Using the list of nodes (crossed airspace blocks) for each trajectory and the set of nodes of each component, it is possible to compute the number of re-entry events for each time period, which will be noted N_r.

The last criterion measures the number of short transits inside each sector (N_s). It is relatively easy to compute efficiently the number of aircraft that are staying too short time inside one sector using trajectories that are represented by its associated list of blocks with entering and exit times.

All those criteria are then aggregated into a single objective function:

$$y = \alpha_1 U + \alpha_2 F_c + \alpha_3 V + \alpha_4 N_r + \alpha_5 N_s$$

Application of EAs to the sector design Problem

Evaluation Algorithms

Typically, graph partitioning problem falls under the category of NP-hard problem and stochastic optimization is a good candidate to address it. For this type of problem there could not be only one solution, there may be several optimal (or near optimal) solutions and we should be able to find most of them, as long as they have to be later evaluated and refined by airspace experts. This last point makes us reject classical simulated re-annealing optimization which updates only one state variable. On the other hand, Evolutionary Algorithms (EAs) maintain and improve a population of numerous state variables according to their fitness and will be able to find several solutions. Thus, EAs appear to be relevant to solve the sector design problem.

Coding the chromosome

In the previous section we have proposed a way of modeling the airspace configuration as a set of connected components for each time period. The coding used for this problem consists in representing connected components by sub-sets of nodes.

EAs use techniques inspired by evolutionary biology such as inheritance, mutation, natural selection, and recombination (or crossover) to find an approximate solution of the optimization problem.

An individual (a solution of the problem), is represented by a list of parameters, called chromosome. The chromosome used in our algorithm is presented in Figure 11.

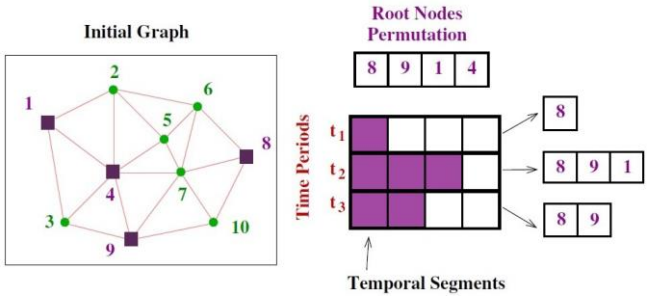


Figure 11. Chromosome structure

The chromosome consists of two parts. First part includes a table with permutation of root nodes (their indexes) and a table that contains temporal segments. For each time period this second table will contain the list of root nodes, chosen from nodes that represent non-sharable blocks, and that are used in construction of the controlled sectors. Root nodes are used as starting nodes for graph partitioning (see Figure 12). Second part contains connected components, coded as lists of nodes belonging to same component. The method ensures that nodes of the same sub-set are connected by at least one path.

For each time period several nodes are selected from the list of non-sharable nodes as shown in Figure 11. Those nodes will be considered as root nodes. The minimum number of root nodes that can be selected equals to 1 and the maximum equals to the number of all non-sharable blocks i.e., to the number of maximum controlled sectors in the configuration. Here, nodes 1, 4, 8 and 9 are non-sharable nodes and potential centers of controlled sectors. Three time periods are considered and three temporal time segments are generated first randomly. At the time period 1 there is one sub-set created with root node 8, at the second period - 3, with root nodes 8, 9 and 1, and etc. (see Figure 11).

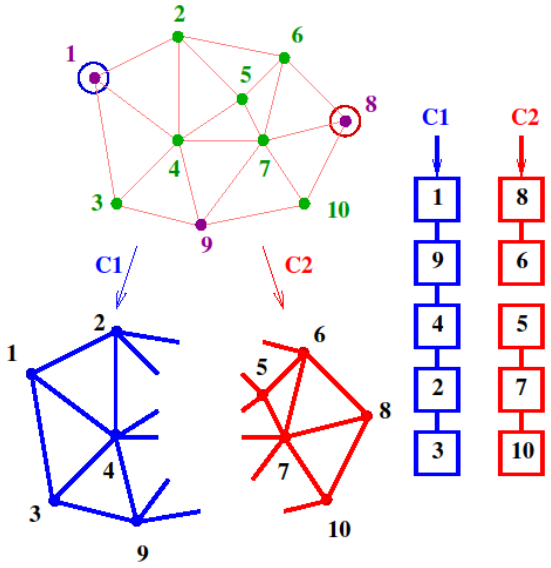


Figure 12. Example of the coding graph partitioning for one time period

The initial permutation of root nodes ensures a random mapping between temporal segments and root nodes (this avoids the same root node to be associated with the first temporal segment). The way of coding chromosomes with temporal segments ensures the stability in time of controlled sectors shapes. As a matter of fact, for successive time periods, similar list of root nodes will be used as a sector centers, ensuring those volume of airspace being controlled by the same controller.

Chromosome Initialization

In order to create an initial population of solutions, we should be able to generate chromosomes randomly. First, the root node order is selected by the mean of the initial permutation. Then, temporal segments are randomly built. For each time period several root nodes are chosen from the permutation table. The number of chosen root nodes per time period is first generated randomly and is equal to the number of controlled sectors in the configuration. The first root node in the permutation table participates in the partitioning process for each time period. Finally, based on those chosen root nodes a greedy heuristic is used to build the initial partitioning which is summarized in Figure 13.

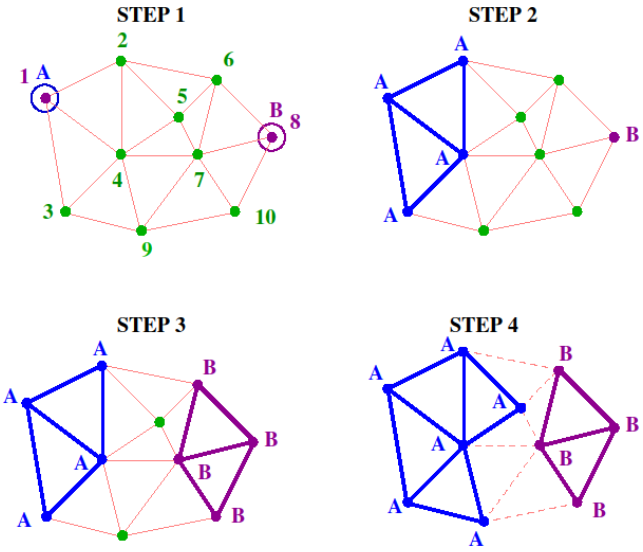


Figure 13. Greedy heuristic is used to create initial partitions

In this example, there are two root nodes, marked with labels *A* and *B* (step 1). On a first step, all nodes connected with one of the root nodes are marked respectively, with the label *A* or *B* (step 2 and

3). Then, the algorithm makes several iterations, running each time through the permutation list of all other nodes, until every node receives its label. At each iteration, the algorithm checks all neighbors of a non-labeled node and if at least one of its neighbors is labeled, the node receives the same label (step 4). If several neighbors of the node have different labels, then the label that represents the lightest component (with the smallest workload) is chosen. If the node does not receive a label, it is left without it for the next iteration.

Recombination operators

After creating the first population, each solution is evaluated, and a value of fitness is returned by a fitness function. This initial population undergoes a selection process that identifies best solutions. Then, three following recombination operators are applied to individuals: *nothing*, *crossover*, or *mutation*. The associated probability of application are respectively $(1-p_c-p_m)$, p_c and p_m.

Crossover between two parent solutions results into two new child chromosomes, which are added to the next population. During the application of crossover operator, chromosomes of parents are mixed or exchange between each other. The purpose of mutation in EAs is to allow the algorithm to avoid local minimum by preventing the population of chromosomes from becoming too similar to each other, which could cause slowing or even stopping evolution. These processes ultimately result in the next population of chromosomes *POP(k+1)* that is different from the initial generation. This generational process is repeated in cycle N_G times, where N_G is a required number of generations.

Several recombination operators have been developed in our algorithm.

First type of operators has been designed to modify the first part of the chromosome. In order to change initial root nodes, a mutation operator has been developed. It consists in changing the order of root nodes by randomly exchanging two nodes in the permutation table. The mutation operator for temporal segments changes the number of segments by adding or removing one segment i.e. adding or removing one controlled sector into configuration. One slicing crossover operator has been developed for temporal segments as well. Two parents with time intervals are used to create two children by randomly selecting one gene position and by exchanging all genes from such position.

The second type of operators for the second part of the chromosome consists of two mutation operators. Before applying any mutation operator a time period is selected either randomly, either according to the associated graph partitioning performances, meaning that configuration with a worst objective function will have more chances to be selected. Then one of the graph partitioning mutation operators is applied.

First operator modifies permutation table of all nodes (sharable and non-sharable). In order to change a resulting graph partitioning without changing root nodes order we are simply changing the order of all other nodes. This ensures that the graph partitioning process will not start from the same nodes, and so resulting components will probably consist of different nodes.

Then, the second operator begins by selecting the “worst” component. After that, it starts to search the suitable neighbouring component. In case if the chosen component is not loaded enough, then it searches for the neighbour with the highest workload (the same for the flow cut). On the other hand, if the chosen component is overloaded, then it searches for the neighbour with the lowest workload. This step is carried out statistically (introducing a bias into a random selection). Then we obtain one link which connects these two components. One of the nodes belonging to this link is then moved from one component to the other one (depending if the component is overloaded or not), while verifying that the component which is losing a node remains connected.

Having described our algorithm, the next section presents several examples of results.

Simulation and results

This section presents the results of dynamic sector configurations algorithm applied to Maastricht Amsterdam FIR and free routes traffic.

Scenarios description

Three scenarios have been used to test the algorithm. Each scenario is based on free route trajectories simulated by Lido[1] and Sabre[2] systems, which provide a sample of 2332 full free route trajectories for the 14th of July 2014 crossing Maastricht/Amsterdam Airspace (EDYYDUTA).

Figure 15 representing free route trajectories, compared to the trajectories planned in today's Routes Network in Figure 14, shows a noticeable dispersion of the trajectories and a loss of traffic pattern.

Figure 14. Planned trajectories in fixed-route network

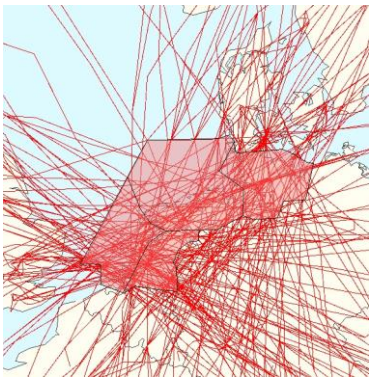


Figure 15. Free route simulated trajectories

The baseline scenario to which the algorithm will be compared characterizes current operational methods and tools. It uses a limited number of predefined sectors configurations, based on known traffic pattern and current organizational framework, including ATC licensing constraints. The tool used to compute the optimum sectors configuration sequence over a traffic day (the opening scheme) is ICO, Improved Configuration Optimizer. ICO algorithm is implemented within NEST simulator (cf. [3] and [4]), the Network Strategic Tool developed by EUROCONTROL.

Two solution scenarios have been tested. The first is a DAC level-1 case, which uses 6 initial elementary sectors as components inputs to the algorithm, cf. figure 16.

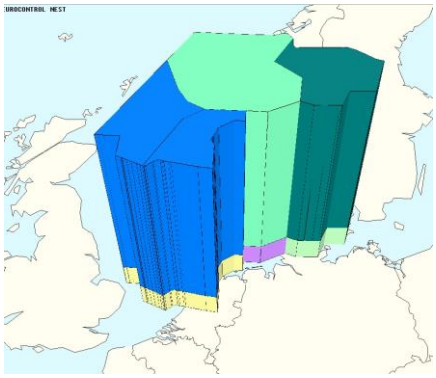


Figure 16. DAC level-1: 6 sectors scenario

The second scenario is a DAC level-2 case. It uses 6 SAMs components obtained by cutting vertically the 6 original sectors at altitude levels 335 and 355 in order to increase flexibility of new sectors configurations built by the algorithm, cf. figures 17 and 18.

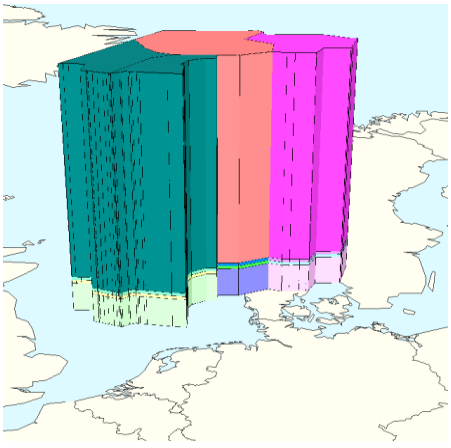


Figure 17. DAC level-2: 6 sectors and 6 SBBs

[1] Lido: Lufthansa Systems flights planning tool

[2] Sabre: Sabre Airlines Solutions, flight planning tool

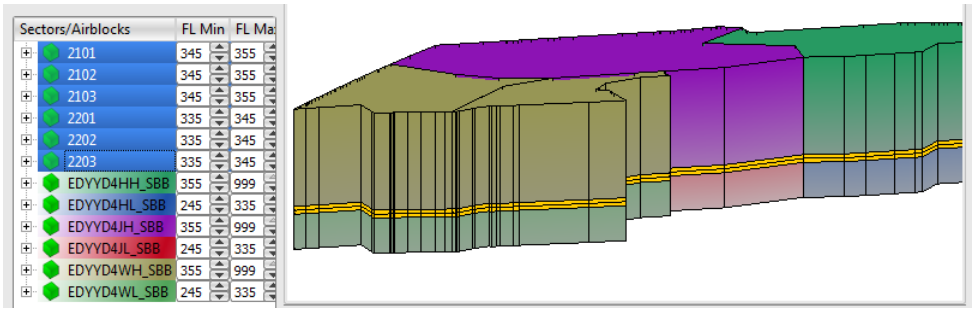


Figure 18. DAC level-2 scenario airspace components vertical limits

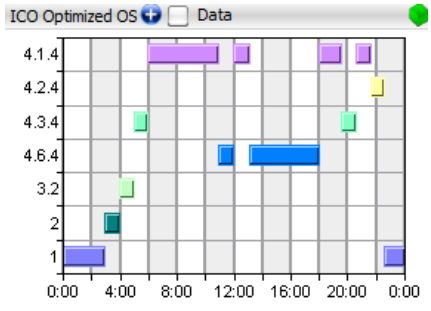
The sectors traffic load is monitored through occupancy count metric. It is assumed that the capacity used for all the sectors for the baseline as well as solution scenarios is a constant value. It corresponds to a maximum number of flights per minute over a sustained period of time (e.g.: maximum of 8 flights per minute over a sustained period of 12 minutes). It is also assumed that the backbone (the number of available controllers) is a constant value of 8 controllers all over the day. Capacity and backbone values are used for technical testing purpose and do not reflect real operational values.

Some Quantitative Results

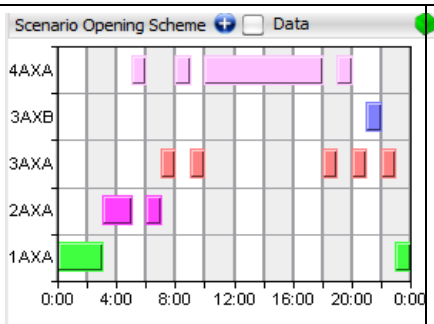
The algorithm has been executed for both solutions scenario and ICO for the baseline scenario. Figure 19 displays the Opening Schemes computed for each scenario and arranged in time tables.

The values in Y-axis are the configuration name, where the first digit informs about the number of sectors used in the configuration. It shows that the algorithm builds configurations composed of a number of sectors which increases and decreases with the traffic profile of the day. The maximum number of sectors in a configuration is 4, which is in line with the limit imposed by the number of available controllers (8). The same is for ICO algorithm in the baseline scenario. It also shows that the number of configurations and configuration changes increase significantly for DAC level-2 scenario. For example, it is proposed 5 different configurations composed of 4 sectors and 3 configurations of 3 sectors and 19 configuration changes over the day.

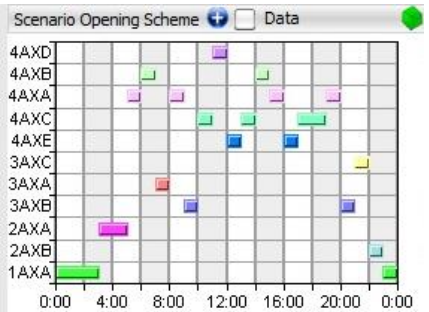
The performance of the opening schemes proposed for the baseline and solutions scenarios are displayed in the next three figures. Figure 20 provides global results regarding the overload and the number of controllers hours generated by the opening scheme of each scenario.



Baseline Opening Scheme: ICO using pre-defined configurations



DAC level-1 scenario: solution algorithm using elementary sectors



DAC level-2 scenario: solution algorithm using 6 SBBs and 6 SAMs

Figure 19. Opening Scheme for baseline and solution scenarios

Figure 21 and 22 presents results about the quality of the computed configurations; i.e. criteria of workload balancing, number of re-entries and short-crossing flights that are taken into account by the algorithm for the selection of the most acceptable configurations in the operational view point.

Figure 20 below shows that the proposed algorithm used with DAC level-2 scenario outperforms the baseline as for overload and number of controllers working hours. DAC level-2 decreases significantly the overload of about 15% compared to the baseline. While DAC level-1 scenario provides better results regarding the number of controllers hours. Nevertheless, the operational acceptance of the proposed configurations is not ensured. Further analysis of each new configuration and also the changes in the configuration sequence must be properly analyzed. An initial expert judgment is provided in the next section.

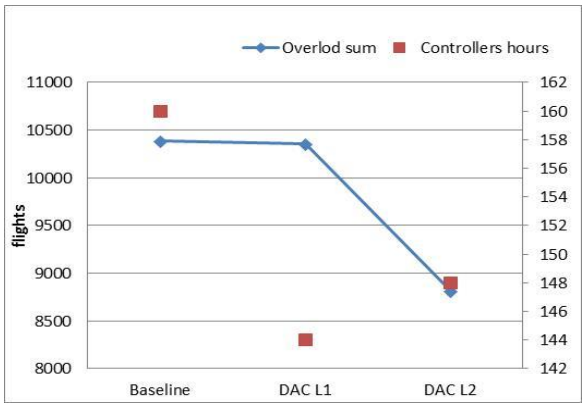


Figure 20. Overload and controllers hours

Figure 21 shows a significant improvement of the quality of the configurations provided by the solution scenarios in terms of workload balancing compared to the baseline. The mean difference of the sectors load within a configuration of the solution scenario is about 20% while it reaches 33% for the baseline configurations. Those results demonstrate that the proposed configurations, which freely combines elementary sectors (DAC level-1) and 6 SBBs and 6 SAMs enable to propose sectors configuration with more balanced sectors load.

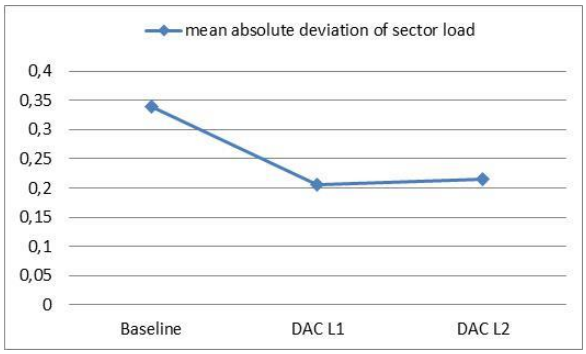


Figure 21. Sector load balancing criteria

Figure 22 presents the average number of re-entering flights and short-crossing flights per configuration for the baseline and solutions scenarios. It clearly shows that the algorithm optimizes both criteria. Similarly to overload criteria, DAC level-2 is the scenario which also best optimizes re-entries and short-crossing criteria due to the increased number of airspace component combinations which allows the algorithm to find a greater number of solutions fitting the traffic structure.

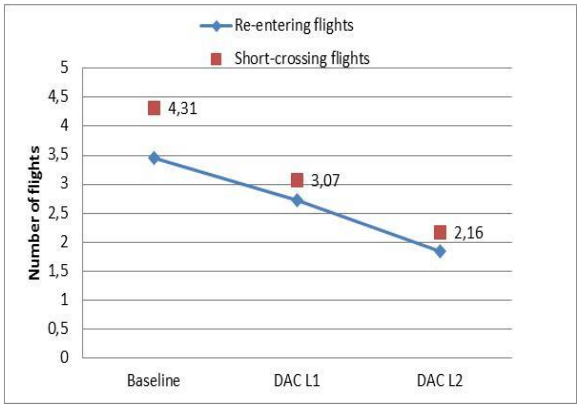


Figure 22. Re-entering and short-crossing flights

Initial analysis of the new configurations

The results presented previously are promising; they demonstrate that the algorithm performs in adequacy with the specified operational objectives and constraints. Nevertheless, further operational evaluation is required to assess the workability of the proposed configurations and their sequence (opening scheme). In the scope of this paper, an initial judgmental assessment is provided for three new configurations proposed in scenario DAC level-2.

Figure 23 presents sectors configuration 4AXE. This configuration is built by grouping all the sectors of the lower level to form an extended lower sector displayed in light-green. This configuration is not operationally used because the current system visualization imposes limit to the size of the sector to be controlled. The maximum length of a leg in a combined sector should be 250NM. This technical constraint is not taken into account by the algorithm.

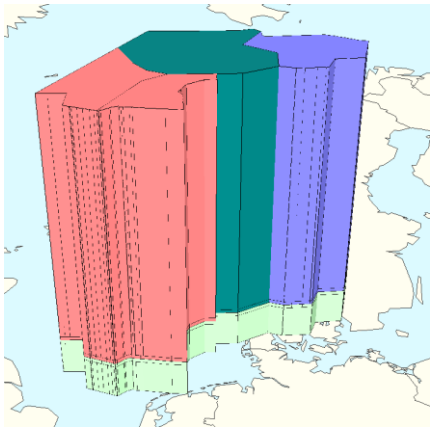


Figure 23. DAC Level-2 Conf 4AXE

Figure 24 presents one often used configuration, "4AXC", in DAC-level 2 scenario. The lower sector displayed in light-blue color, is composed of two grouped sectors (SBBs) which currently require two different ATC licenses, which explains why it is not used in today's configurations. However, the algorithm considering a unique license for all the sectors proves that it can offer better performance regarding sectors load balancing and overload.

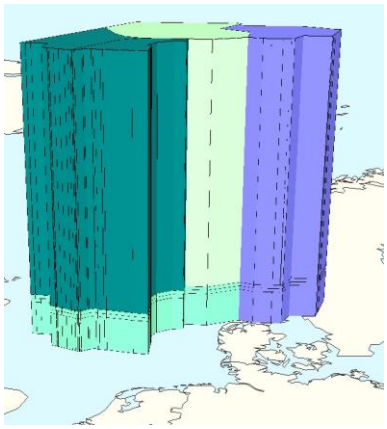


Figure 24. DAC Level-2 Conf 4AXC

In addition, due to the use of SAMs components, can be added to this combination of SBBs, different combinations of SAMs, which leads to new sectors configurations, with a slight modification, as it is presented in configuration '4AXB' displayed in figure 25. The difference between configurations 4AXC and 4AXB is one SAM component that is allocated to the upper sector displayed in pink color in figure 25.

Configurations 4AXC and 4AXB are used consecutively in the opening scheme proposed by the algorithm to better balance the workload, and there is no foreseen operational opposition to the use of either 4AXC and 4AXB configurations or sequence, providing that a unique ATC licensing is applied to their components.

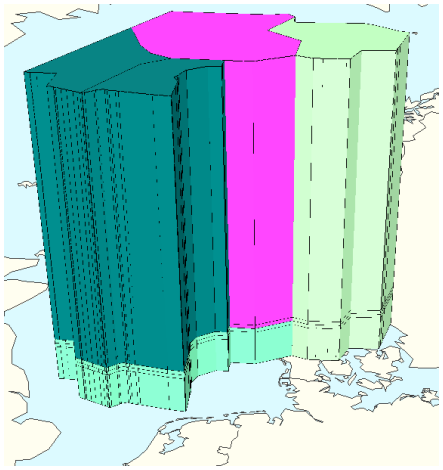


Figure 25. DAC Level-2 Conf 4AXB

Conclusion and next steps

The modelling approach and algorithm solution presented in this paper have been tested and compared to existing technical solution. The provided results demonstrate that the new solution fits with the requirements of the DAC concept and outperforms the baseline solution regarding the minimization of the overload, sector load balancing, as well as re-entering and short-crossing flights, at least for DAC level-1 and level-2 cases. Initial tests have been performed for scenarios DAC level-3. The provided results show that the algorithm is not adapted to that case, since an increasing number of SBBs and SAMs, requires geometrical constraints to control the shape of the sectors, in terms of convexity, balconies, etc. Further investigations will be achieved in order to

define a solution for DAC level-3 sectors configuration.

References

[1] SESAR WP7.5.4 project, Dynamic Airspace Configuration, OSED step2/V2.

[2] Flener P., J. Pearson, Automatic airspace sectorisation: A survey, CoRR, vol. abs/1311.0653, 2013. http://arxiv.org/abs/1311.0653

[3] Vehlac C. S. Manchon, Improved Configuration Optimizer, EEC note No 10/05, 2005.

[4] NEST: https://www.eurocontrol.int/services/nest-modelling-tool

[5] Ehrmanntraut R., McMillan S., Airspace design process for dynamic sectorisation, 26th DASC, 2007, Dallas, USA

[6] Zerrouki L. M. Dalichampt, K. Purves, Dynamically Shaped Sectors, SJU early project validation report, 2010.

[7] Drew M., "A method of optimally combining sectors," in Proceedings of the 9th AIAA Aviation Technology, Integration and Operations (ATIO) Forum. American Institute of Aeronautics and Astronautics, 2009.

[8] Sabhnani G., A. Yousefi, and J. Mitchell, "Flow conforming operational airspace sector design," in Proceedings of the 10th AIAA Aviation Technology, Integration and Operations (ATIO) Forum. American Institute of Aeronautics and Astronautics, 2010.

[9] Yangzhou C., B. Hong, Z. Defu, and S. Zhuoxi, "Dynamic airspace sectorization via improved genetic algorithm," Journal of Modern Transportation, vol. 21, pp. 117–124, 2013.

[10] Gianazza D., "Forecasting workload and airspace configuration with neural networks and tree search methods," Artificial Intelligence, vol. 174(7-8), p. 530549, 2010.

[11] Delahaye D., J. Alliot, M. Schoenauer (CMAPX, and J. Farges, "Genetic algorithms for partitioning airspace," in Proceedings of the 10th IEEE Conference on Artificial Intelligence Application. IEEE, 1994.

[12] Delahaye D., J. Alliot, M. Schoenauer, and J. Farges, "Genetic algorithms for automatic regrouping of air traffic control sectors," in Proceeding of the 4th Annual Conference on Evolutionary Programming. MIT Press, 1995.

[13] Delahaye D., M. Schoenauer, and J.-M. Alliot, "Airspace sectoring by evolutionary computation," in Proceedings of the IEEE International Congress on Evolutionary Computation. IEEE, 1998.

[14] Masalonis A., M. Callaham, and C. Wanke, "Dynamic density and complexity metrics for realtime traffic flow management," in Proceedings of 5th USA/Europe Air Traffic Management Seminar, 2003.

[15] Kopardekar P., A. Schwartz, S. Magyarits, and J. Rhode, "Airspace complexity measurement: An air traffic control simulation analysis," in Proceedings of the 7th USA/Europe ATM 2007 RD Seminar, 2007.

[16] Klein A., M. D. Rodgers, and K. Leiden, "Simplified dynamic density: A metric for dynamic airspace configuration and nextgen analysis," in Proceedings of the Digital Avionics Systems Conference, 2009.

[17] Gianazza D. and K. Guittet, "Selection and evaluation of air traffic complexity metrics," in Proceedings of the 25th Digital Avionics Systems Conference, 2006.

[18] Martinez S., G. Chatterji, D. Sun, and A. M. Bayen, "A weighted graph approach for dynamic airspace configuration," in Proceedings of the AIAA Conference on Guidance, Navigation, and Control (GNC). American Institute of Aeronautics and Astronautics, 2007.

[19] Li J., T. Wang, M. Savai, and I. Hwang, "A spectral clustering based algorithm for dynamic airspace configurations," in Proceedings of the 9th AIAA Aviation Technology, Integration, and Operations Conference (ATIO), 2009.

[20] Sergeeva M., D. Delahaye, "3D Airspace Sector Design by Genetic Algorithm", MT-ITS, Budapest, 2015