A Method Framework for Automatic Airspace Reconfiguration-Monte Carlo Method for Eliminating Irregular Sector Shapes Generated by Region Growth Method

Zhijian Ye 1,*, Fanhe Kong 2, Baocheng Zhang 1, Wei Gao 1 and Jianfeng Mao 3

1 College of Air Traffic Management, Civil Aviation University of China, Tianjin 300300, China
2 ATC Dep, Dalian ATM Station of Civil Aviation of China, Dalian 116033, China
3 School of Science and Engineering, Shenzhen Key Laboratory of IoT Intelligent Systems and Wireless Network Technology, The Chinese University of Hong Kong, Shenzhen 518172, China

* Correspondence: zjye@cauc.edu.cn; Tel.: +86-130-2222-6938

Abstract: With the growth of air traffic demand in busy airspace, there is an urgent need for airspace sectorization to increase air traffic throughput and ease the pressure on controllers. The purpose of this paper is to develop a method framework that can perform airspace sectorization automatically, reasonably, which can be used as an advisory tool for controllers as an automatic system, especially for eliminating irregular sector shapes generated by simulated annealing algorithm (SAA) based on the region growth method. The two graph cutting method, dynamic Monte Carlo method by changing location of flexible vertices (MC-CLFV) and Monte Carlo method by radius changing (MC-RC) were developed to eliminate irregular sector shapes generated by SAA in post-processing. The experimental results show that the proposed method framework of airspace sectorization (AS) can automatically and reasonably generate sector design schemes that meet the design criteria. Our methodology framework and software can provide assistant design and analysis tools for airspace planners to design airspace, improve the reliability and efficiency of airspace design, and reduce the burden of airspace planners. In addition, this lays the foundation for reconstructing airspace with the more intelligent method.

Keywords: airspace reconfiguration; irregular boundary smoothing; dynamic Monte Carlo method by changing location of flexible vertices; Monte Carlo method by radius changing; Voronoi diagram; graph cutting; multi-objective optimization

1. Introduction

In recent years, the air traffic flow of China has been increasing at an average annual rate of 11 percent, and air transport is developing rapidly. With the rapid development of national traffic volume, China’s airports and airspace system are under increasing pressure, which leads to frequent flight delays. In 2010, the fixed assets investment of Chinese airports increased from 44.150 billion Yuan to 56.08 billion Yuan, with a compound annual growth rate of 2.54% [1]. Thus in order to solve the imbalance between demand and capacity, specific measures include strengthening the construction of infrastructure and expanding fleet was taken. However, according to the statistics of CAAC (Civil Aviation Administration of China), the average flight punctuality rate in China maintained at a low level (as shown in Figure 1) in the past few years [2].
workload is considered in most of these DAC models. Gerdes, Temme and Schultz [23] clearly distinguished DAC from DAS. DAS is a flexible method, encompassing the idea of unstructured and (rigid) structured airspace, and differing from DAC, where predefined airspace blocks are combined to form new structures (merging and splitting). DAS approach creates a less familiar setting for controllers but can be used for situations where no basic knowledge concerning the best airspace structure is available. This paper mainly solves the problem of increasing the number of sectors or improving

Figure 1. Average flight punctuality rate and total number of flights.

This indicates that the government needs to not only strengthen the construction of infrastructure, but also give advice on how to manage the airspace effectively. The current Chinese National Airspace architecture is reaching the ability limits to accommodate the increases in traffic demand. As the key limiting factor, today’s sector boundaries are largely determined by historical user profiles, which have evolved slowly over time. Consequently the sector geometry has stayed relatively constant despite the fact that route structures and demand have changed dramatically over the years. There are usually two forms of traffic flow growth in airspace. The first is that the traffic flow of different routes has different growth rates with time. Some sectors have excessive task load, while others have a relatively small task load. In this case, the problem that throughput is limited by sector capacity can be solved by adjusting sector structure and keeping the number of sectors unchanged. The second is that the traffic flow of all routes in the whole airspace has increased significantly. The control workload cannot be effectively reduced by just adjusting sector structure without airspace sectorization (AS) [3,4].

Although AS is the most potential method to increase throughput of airspace as well as to better use scarce resources, there is a serious challenge to realize it. Except for the accessibility of airspace sectorization by air traffic controller, a lot of constraints must be considered in accordance with the demand of air traffic control operation. Since AS needs to consider the constraints of the balance of task load, shape, connectivity and compactness, which are beyond the limits of human ability, a mathematical optimization tool is needed to support this process [5,6].

In terms of problem solving technology, AS is essentially a special partitioning technology. Partitioning technology has been widely used in the analysis and design of various agricultural [7], computer [8,9], communication [10,11], economic [12], sea [13], road [14], intermodal transport [15], power [16] and other partitionable systems and networks. AS and these partitioning problems have similarities in load distribution, but it makes AS different from these partitioning problems that coordination workload varies with relative position relationship between AS boundary and routes.

A lot of AS research has been carried out. Delahaye, et al. [17] initiated the study of the balanced sector of airspace in order to increase air traffic control capacity in high density areas. Genetic algorithms were used to compute a balanced regrouping of air traffic control sectors thus to optimally reduce the number of controller teams [18]. A decision support tool based on fuzzy logic was proposed to determine the number of open sectors during a given time period [19]. Related researches include dynamic airspace configuration (DAC), dynamic airspace sectorization (DAS) and airspace configuration. In DAC, the airspace is adjusted in real-time to accommodate fluctuating demand patterns [5,20–24]. A detailed overview of recent work of DAC is given by Zou, et al. [25]. Transitional workload is considered in most of these DAC models. Gerdes, Temme and Schultz [23] clearly distinguished DAC from DAS. DAS is a flexible method, encompassing the idea of unstructured and (rigid) structured airspace, and differing from DAC, where predefined airspace blocks are combined to form new structures (merging and splitting). DAS approach creates a less familiar setting for controllers but can be used for situations where no basic knowledge concerning the best airspace structure is available. This paper mainly solves the problem of increasing the number of sectors or improving
sectors caused by the rapid growth of China’s traffic volume. Our objective is to adjust the sector structure to increase capacity. Civil Aviation of China stipulates that as long as the sector structure changes, controllers must carry out pre-job training. The problem of transitional workload is another problem of controller’s assignment with minimizing training cost. AS in this paper is closer to DAS, because it is a strategic or tactical AS before the implementation of AS scheme.

Flener and Pearson [26] systematically surveyed the algorithmic aspects of methods for automatic airspace sectorization, for an intended readership of experts on air traffic management. Most of the airspace sectorization models were based on two approaches. The first one is the graph-based model. In this type of model, a graph is constructed whose vertices represent the intersections of the existing trajectories while edges represent segments [14,17,27–29]. Several recent published papers also use this type of model [25,30,31]. The graph-based model could use some mature technology in graph partition, such as the MIN-CUT graph partition used as minimizing the coordination task load. However, a graph partition does not define the sector boundaries, so actual sectors have to be constructed from the resulting vertex sets in a geometric post-processing step. The second one is the region-based model in which the airspace is initially partitioned into some type of regions, which are smaller than the studied sectors, so that the combinatorial problem of partitioning these regions in principle needs no geometric post-processing step [32–40]. Unfortunately, shapes of obtained sectors, which use the region-based model still have to need post-processing. Although the region-based model is essentially a down-top combination approach, in which workload among the sectors was balanced by maintaining the sector shape parameters within acceptable limits, but it made them operationally undesirable that there were some corners or Z-shape among these sectors even after post-processing [41,42].

Ref. [43,44] developed another top-down bi-partition approach to cut airspace into sectors. Xue [20], Gerdes, Temme and Schultz [23], Xue [45], Gerdes, et al. [46] also used top-down approach to redesign airspace sector while the generation of sector boundary was based on Voronoi diagrams. With the Voronoi diagram, the convexity requirement is automatically satisfied. The success of this approach depends heavily on the location of Voronoi generating points. For example, in airspace of 200 km × 200 km, if the alternative Voronoi generating points are distributed every 10 km, there will be 400 points. In the case of dividing the airspace into five sectors, the number of alternative locations at five production points is 400⁵. If this approach is applied to larger airspace and divided into more sectors, the computation time may be unacceptable. When estimating the coordinating task load, Xue [20] pointed out that the boundary-crossing of air traffic control center was neglected in his work. Obviously, this will lead to an unbalanced sector task load. If the sector generated by the algorithm is applied to the actual sector, the actual workload of the sector near the studying airspace boundary will be larger than the result of calculation. In addition, if all sectors are Voronoi diagrams, in some cases, it is difficult to find a suitable seed point for Voronoi cell (VC) generation to balance the workload and meet other constraints.

Delahaye, et al. [47] have proposed the use of floating-point chromosomes, and have developed special operators to maintain the integrity of the chromosome for use in a floating point representation, contending that the convergence rate is faster in the floating-point case. When the studying space is discretized by n VCs (Voronoi cells) and the number of sectors is k, the number of possible groupings will be $\delta_n^k$ as following.

$$\delta_n^k = \frac{1}{n!} \sum_{i=0}^{n-1} (-1)^i \frac{n!}{i!(n-i)!} (n-i)^k.$$  \hspace{1cm} (1)

They pointed out that this kind of combination is a non-deterministic polynomial hard problem and that stochastic optimization is a good candidate to address it. Pawlak, et al. [48] also used this type of floating-point chromosomes to group cells by the finite-element method. Sergeeva, et al. [49] developed a more complex floating-point chromosome for 3D reconfiguration, which includes two parts, the first part represents coordinates of sector centers and the second part contains the associated vertical
extension of each sector center. However, these chromosomes are hard to understand especially how they work in the genetic algorithm.

The reconfiguration problem is well-known for the NP-hard problem [47,50,51]. Simulated annealing (SA) is a well-studied local search and meta-heuristic used to address discrete and, to a lesser extent, continuous optimization problems. It is a probabilistic technique for approximating the global optimum of a given function. Specifically, it is a meta-heuristic to approximate global optimization in a large search space. It is often used when the search space is discrete. The key feature of simulated annealing is that it provides a mechanism to escape local optima by allowing hill-climbing moves (i.e., moves, which worsen the objective function value) in hopes of finding a global optimum [52].

Johnson, et al. [53], [54] is the pioneer who uses SA to graph coloring and number partitioning. Rahimian, et al. [55] proposed a fully distributed algorithm called JA-BE-JA, which uses local search and simulated annealing techniques for two types of graph partitioning: Edge-cut partitioning and vertex-cut partitioning. These partitioning algorithms have some similar objective, such as minimizing cut cost and balancing weights of vertexes. However, they still are different from AS, which require balancing the total workload (monitoring and coordinating task loads), minimizing the coordination task load, as well as thinking about dwell time, convex shape, connectivity and compactness.

In summary, the bottom-up method is prone to appear C-shaped, zigzag, trapezoidal and other irregular shapes, while the top-down method is prone to appear as a narrow triangle, which is not easy to ensure convexity. In addition, the method of a random Voronoi diagram is like looking for a needle in a haystack to get an optimal solution. It is doubtful that the multi-layer random Voronoi graph without boundary reprocessing not only guarantees the workload balance of the graph, but also guarantees the convexity of the sector. Therefore, we must design an innovative solution to the AS problem.

The aim of this paper is to establish a framework of solving methods including a series of multi-objective (balance the task load, minimize the task load imbalance, minimize the total coordinating task load, minimize the cost of short dwell time and minimize the cost of reentering the same sector) solving methods that can perform AS automatically at the strategically planning stage, while meeting the constraints of connectivity and compactness in optimal process, as well to smooth the boundary for an accessible shape of the sector. In order to improve the reliability of the design, we synthesized the region growth method and the graph cutting method. Firstly, we use the region growth method based on the Voronoi diagram to generate the initial optimal solution. Then, two graph cutting algorithms are used to eliminate irregular sector boundary caused by region growing algorithm.

This paper makes the following specific contributions:

1. Both of two graph cutting algorithms proposed by us can eliminate the irregular sector boundary caused by the region growing algorithm as well as other criteria can also be guaranteed.
2. The MC-CLFV algorithm based on flexible vertices generated deducted from the simulated annealing algorithm (SAA) result greatly simplifies the complexity of graphic cutting algorithm.
3. The MC-RC algorithm shows some advantages to reduce narrow blocks while the MC-CLFV algorithm is superior to the MC-RC algorithm for increasing traffic throughput.
4. We also found that appropriately allowing some non-convex boundaries is a measure to reduce the total workload, and blindly pursuing convex boundaries will increase the total workload.
5. The solution framework proposed in this paper improves the reliability of obtaining the optimal design scheme in airspace, and will reduce the design load and completion time of designers.

These studies are of great significance to improve air traffic throughput and maintain the task load of controllers at a reasonable level. This paper consists of five sections. Beginning with the introduction, Section 2 describes problem formulation. Section 3 elaborates the details of the solution method, which including SAA and the two post-processing method. Experimental results for a realistic scenario are
reported in Section 4. Section 5 presents discussions. Conclusions and opportunities for further work are presented in Section 6.

2. Problem Formulation

2.1. Boundary and Route Network

In order to clearly show the problem, we use Taiyuan Airspace (TYN, Taiyuan Airport is defined as TYN by the International Air Transport Association with three-character code) as the background in the following definitions of airspace structure. The boundary points are expressed as $B$. $B$ is a $n \times 2$ matrix, and is listed anti-clockwise. The boundary line is the red dotted line in Figure 2. Each row of matrix $B$ represents a geographic coordinate of a point.

![Figure 2. Route network and boundary in TYN airspace.](image)

The route network consists of all segments (blue line in Figure 2) in airspace. Table 1 is an example of the route network shown in Figure 2. Due to the paper length limit, we listed only a part of the data. Despite having the same name, the segment is distinguished by the route course, such as B208 (N), B208 (M) and B208 (S). Each segment includes the coordinates of the starting and ending points and the daily traffic volume.
Table 1. Examples of the route network and daily traffic volume.

| Route | Starting Point Coordinates ($\varphi_1, \lambda_1$) | End-Point Coordinates ($\varphi_2, \lambda_2$) | Starting Point | End Point | Daily Traffic Volume $f$ (Flights/Day) |
|-------|--------------------------------------------------|---------------------------------------------|----------------|-----------|--------------------------------------|
| B208(N) | $39^\circ29.5'$ 112 12.1 | $37^\circ45.0'$ 112°37.1' | TODAM | TYN | 146 |
| B215(E) | $37^\circ45.0'$ 112 37.1 | $38^\circ17.3'$ 114°05.9' | TYN | ISGOD | 90 |
| G212(W) | $37^\circ45.0'$ 112 37.1 | $36^\circ28.4'$ 110°30.6' | TYN | OKVUM | 17 |

2.2. Voronoi Diagrams Generating Method

VCs are adopted to discretize the reconfiguration airspace in this paper. Given some generating points (seeds of the Voronoi diagram), the Voronoi diagram divides airspace into a group of convex polygons with no overlaps. In theory, the higher the number of generating points, the greater the solution space. Although the small granularity can result in more balanced sectors, it will take more time to calculate. The distribution of generating points in space also affects the solution. The following three methods were used to generate seeds of the Voronoi diagrams.

2.2.1. Seeds Were Generated Randomly in the Studying Airspace

Within the square airspace surrounding the airspace boundary, random point was generated. If this point falls within the airspace boundary, a random seed is successfully generated. Repeat this step over and over again to generate $n$ generating points. Then the Voronoi diagrams can be generated. Figure 3 is an example of a Voronoi diagram generated by 90 randomly generating points.

![Random generating seeds](image)

**Figure 3.** Random generating seeds.

2.2.2. Seeds Were Taken Every $D_{\text{min}}$ Kilometers along Routes

Generating points for constructing VCs include starting and end points of every route, as well as some intermediate points along routes separated by at least $D_{\text{min}}$ kilometers. If the length of $r$ segment is $D_r$, then the number of intermediate points equals the following:

$$m_r = \frac{D_r}{D_{\text{min}}} - 1,$$

(2)
where \( \cdot \) is the operator of rounding down. Less \( D_{\min} \) means more VCs with small granularity when discretizing the airspace. Figure 4 is a Voronoi diagram generated by taking a point every 30 km along the route (pink coloring) as generating points.

![Figure 4. Seeds generated by taking a point every 30 km along the route.](image1)

2.2.3. Hexagonal Seed

The generating points are evenly and staggeringly distributed in the studying airspace. The resulting Voronoi diagram is a regular hexagon. Figure 5 is an example of a hexagonal Voronoi diagram generated in the studying airspace.

![Figure 5. Hexagonal seed.](image2)
2.3. Reconstruct Voronoi Cells According to the Airspace Boundary for Coloring Correctly

After discretizing the airspace with Voronoi cells (VCs), in order to visualize which sector these Voronoi cells belong to, the Voronoi diagram needs to be colored. Voronoi cell coloring in the airspace boundary can be easily colored by calling the fill or pitch function of MATLAB, but it is not easy to color Voronoi cells near the airspace boundary. Ordinary Voronoi diagrams have the following basic properties [56].

- All the points on the common edges have equal distances to their neighbor generating points.
- The generating point of VC is the closest generating point to any point within this VC.
- The line between two generating points is perpendicular to the common edge of the two Voronoi polygons.

These characteristics of VCs will be utilized to calculate the intersection point between the edge of the boundary and edge of the VC.

Reconstructing VCs and coloring all cells for visualization are very important for reconfiguration. We noticed that color could not be filled in some open VCs according to the index of which sector these cells belonged to. In the process of coloring, just the inner part of the VC crossing by airspace boundary is preferred to retain, so we developed a method to reshape these VCs in the next part for coloring all cells correctly.

The procedure for the solution of the intersection point between the edge of the boundary and the edge of the VC is defined as follows:

Firstly, all endpoints of the boundary line were sorted anticlockwise. The equation of each boundary line is derived from Table 1. The equation of a boundary line can be expressed as Equation (3). $k_1$ is the slope of this line and $b_1$ is the intercept of a straight line on $y$ axis. According to the characteristic that all the points on the common edges have equal distances to their neighbor generating points, the common edges must be perpendicular to the line between two generating points, which generate VC that passed through by boundary segment. As a result, we can get the function of common edges as Equation (4). $k_2$ is the slope of this line and $b_2$ is the intercept of a straight line on $y$ axis.

The intersection point can be obtained from two of these linear equations.

$$ y = k_1x + b_1. \tag{3} $$
$$ y = k_2x + b_2. \tag{4} $$

Thereafter, only points within the airspace boundary of VC which is intersected by the boundary segment and intersection point are retained as the new vertices set of these VC. After anti-clockwise sorting of these new vertices sets of these VC, every VC can be correctly colored by the program. Figure 6 is an example that the Voronoi cell diagram is reconstructed and colored according to the airspace boundary. With this reconstruction and coloring, it is not only easy to distinguish which sector VCs belong to, but it is also easy to solve the intersection of the route and boundary VCs.
2.4. Task Load Measurement and Objective Function

The research focusing on the controller workload has been carried out by a desire to balance the task load, understand occupational stress, eliminate operational errors, enhance safety and improve throughput. Many workload calculation methods have been proposed in previous studies based on dependent variables: Physical activity [57], physiological indicators [58–60], simulation models of the controller’s tasks [46,47,61–64] and subjective ratings [65,66]. Gianazza [67] have reviewed literature about workload prediction and compared several machine learning methods on the problem of learning workload prediction models from historical data. Surprisingly, there is no globally accepted definition for the controller workload—“Controller workload is a confusing term and with a multitude of definitions, its measurement is not uniform” [68]. From the perspective of technical feasibility, the task load calculation method based on the statistical controller’s tasks in the field or simulated tests is easier than others.

Each sector is typically positioned with one or two controllers. A radar controller is responsible for radio communications with the aircraft, monitoring the radar screen to maintain safe separation and communicating with other controllers. When two controllers work a sector, the second is an associate controller, known as a data-controller. The data-controller typically receives the flight-plan information and helps plan and organize the flow of traffic within the sector. During exceptionally busy periods, a third controller may be assigned to the team, although three-member teams are not typically planned for. The task load is the sum of the time spent by controllers performing tasks [69].

Task load calculation model designed by Oktal and Yaman [64] considered three kinds of task load, the monitoring task load, the conflict task load and the coordination task load. The monitoring and the conflict task loads occur inside the sector while the coordination task load between the sector and its adjacent sector. Due to the fact that the air traffic controllers in China usually build an aerial overpass bridge at the intersection of the route to ensure that flights that may be conflicting have been transferred to the specified flight levels before they arrive at the intersection, it indicates that the monitoring task load includes the conflict task load. For this reason, our task load calculation model only considers monitoring and coordinating task loads. Formulas of our task load calculation model is similar to these designed by [47,49], while the tiny difference lies in that we consider the duration of the task, not just the number of flights in our task load model. We suppose that the monitoring

![Voronoi cells are reconstructed and colored according to the airspace boundary.](image-url)
task load of one aircraft is proportional to flight time. This ratio is the average time per unit of time that the aircraft needs to be monitored by the controller (including conflict resolution by telephone). The coordinated task load of an aircraft is equal to the average time for the controller to complete the takeover or handover when this aircraft cross sector boundaries.

2.4.1. Task Load Measurement

Given that the airspace studied has \( n \) VCs, each \( VC_i, i \in \{1, \ldots, n\} \), must be assigned to some sector \( s_j, j \in \{1, \ldots, k\} \) where \( k \) is the number of desired sectors to be opened. If \( VC_i \) is assigned to \( s_j \) sector, then \( x^j_i = 1 \), else 0. The relationship between task load of \( j \)th sector and task load of \( VC_i \) is expressed as follow:

\[
wl_j = \sum_{i=1}^{n} x^j_i \cdot wl_i, \forall j \in \{1, \ldots, k\}, \forall i \in \{1, \ldots, n\}. \tag{5}
\]

\(wl_i\) includes two kinds of task loads, namely monitoring and coordinating task loads, as mentioned before.

\[
wl_i = w_{i}^{\text{mot}} + w_{i}^{\text{cod}}. \tag{6}
\]

Monitoring the task load measurement:

Assume that there are \( m \) routes passing through \( VC_i \). Then monitoring task load was calculated as follows:

\[
w_{i}^{\text{mot}} = \alpha^{\text{mot}} \cdot \sum_{r=1}^{m} f_r \cdot \left( \frac{l'_r}{v_r} \right). \tag{7}
\]

\(l'_r\) is the length of route \( r \) in \( VC_i \), km.
\(v_r\) is the average speed of flights on route \( r \), km/h.
\(f_r\) is the hourly traffic volume passing through route \( r \), number/h.
\(\alpha^{\text{mot}}\) is the statistical monitoring time per hour per flight by field test, seconds/h.

The intersection of all routes with the VC they pass through is computed and stored in a structured array before the task load is computed, as shown in Figure 7.

The coordinating task load measurement:

The coordinating task load is calculated as follows:

If \( \forall r \in \{1, \ldots, m\} \) passing through \( VC_i \) and both of the boundaries of \( VC_i \) were the sector boundary, then:

\[
w_{i}^{\text{cod}} = \alpha^{\text{cod}} \cdot \sum_{r=1}^{m} \left( \gamma^r_i \cdot f_r \right). \tag{8}
\]

\(f_r\) is the hourly traffic volume passing through route \( r \), number/hour.
\(\alpha^{\text{cod}}\) is the statistical coordinating time per hour per flight by field test, seconds/hour.

If \( \forall r \in \{1, \ldots, m\} \) passing through \( VC_i \) and two of the boundaries of \( VC_i \) was the sector boundary, then \( \gamma^r_i = 2 \). This rarely happens unless a single VORONOI diagram is large. If \( \forall r \in \{1, \ldots, m\} \) passing through \( VC_i \) and just one of the boundaries of \( VC_i \) was the sector boundary, then \( \gamma^r_i = 1 \). If \( \forall r \in \) passing through \( VC_i \) and none of the boundary of \( VC_i \) was the sector boundary, then \( \gamma^r_i = 0 \).
2.4.2. Objective Function

Four criteria were included in our objective function for the evaluation of a solution. The equation of objective function is:

$$F = (\alpha_1 \cdot \text{cost}_{imb} + \alpha_2 \cdot \text{cost}_{tot}^{\text{cod}} + \alpha_3 \cdot \text{cost}_{tot}^{\text{SDT}} + \alpha_4 \cdot \text{cost}_{tot}^{\text{rin}}).$$

(9)

$\alpha_1$, $\alpha_2$, $\alpha_3$ and $\alpha_4$ represent the weight of different objective components.

Minimize the task load imbalance:

The first criterion minimizes the level of the task load imbalance among all controlled sectors in configuration. The task load imbalance of all controlled sectors is then computed with the following equation:

$$\text{cost}_{imb} = \sum_{j=1}^{k} |wl_j - \mu|$$

(10)

where:

- $k$: Number of desired sectors to be opened;
- $j$: $j$th sector;
\( \mu \): Average task load, \( \mu = \frac{\sum_{j=1}^{k} w_l}{k} \);

\( w_l \): Total task load in \( j \)th sector.

Minimize the total coordinating task load:

Coordinating task load is the key factor of the total task load. In the process of airspace sectorization, it is almost impossible to decrease the monitoring task load. A good airspace configuration plan must be a plan with the minimum coordination task load. Minimizing coordination task load can not only reduce the total task load, but also keep the sector consistent with the main traffic flow. The total coordinating task load, \( w_{\text{tot}}^{\text{cod}} \), is equal to the sum of task load of all VCs as following equation.

\[
 w_{\text{tot}}^{\text{cod}} = \sum_{i=1}^{n} w_i^{\text{cod}}, \quad i \in \{1, \ldots, n\}. \tag{11}
\]

Minimize the cost of a short dwell time (SDT):

The sector capacity is calculated based on MAP (Monitor Alert Parameter) from the Federal Aviation Administration (FAA [70], which is roughly 5/3 of the average sector flight time (in minutes). This means that sector capacity increases with dwell time (average sector flight time). Since the airspace studied was an en-route airspace, we arbitrarily set the minimum of dwell time as 240 s. The cost of the short dwell time was measured as follows:

\[
 \text{cost}_{\text{SDT}}^{\text{tot}} = \sum_{j=1}^{k} \text{cost}_{\text{SDT}}^{j}. \tag{12}
\]

For any flight \( a \in \{1, \ldots, m\} \) in sector \( j \),

\[
 \text{cost}_{\text{SDT}}^{j} = \sum_{a=1}^{m} \left[ t_{\text{min}}^{\text{dw}} - t_{\text{dw}}(a, j) \right] \cdot \varepsilon_{\text{dw}}. \tag{13}
\]

\[
 t_{\text{dw}}(a, j) = t_{\text{out}}(a, j) - t_{\text{in}}(a, j). \tag{14}
\]

\( t_{\text{dw}}(a, j) \) is the \( a \)th flight’s dwell time \( t_{\text{dw}}(a, j) \) in the \( j \)th sector and \( t_{\text{min}}^{\text{dw}} \) was set as 240 s.

\( \varepsilon_{\text{dw}} \) is the penalty coefficient of the short dwell time,

\[
 \varepsilon_{\text{dw}} = \begin{cases} 
 \sqrt{\exp\left[ t_{\text{min}}^{\text{dw}} - t_{\text{dw}}(a, j) \right]}, & \text{if } \left[ t_{\text{min}}^{\text{dw}} - t_{\text{dw}}(a, j) \right] > 0 \\
 0, & \text{if } \left[ t_{\text{min}}^{\text{dw}} - t_{\text{dw}}(a, j) \right] \leq 0
\end{cases}. \tag{15}
\]

Minimize the cost of reentering the same sector:

We hoped that every trajectory entered the same sector less than or equal to one time. To some extent, it means this cost will cause the optimization search process to discard those non-convex sectors and disconnected sectors.

\[
 \text{cost}_{\text{in}}^{\text{tot}} = \sum_{r=1}^{m} \text{cost}_{\text{in}}^{r}. \tag{16}
\]

\[
 \text{cost}_{\text{in}}^{r} = \sum_{j=1}^{k} \beta \cdot (t_{\text{in}, j} - 1). \tag{17}
\]

If \( r \) route does not pass through the \( j \) sector, then \( t_{\text{in}, j} = 0 \); if \( r \) route passes through the \( j \)th sector more than once, the cost of \( r \) route reentering in the \( j \) sector will increase with reenter times. \( \beta \) is the reenter penalty factor. We arbitrarily set it as 100 s.

2.5. Constraints

Three constraints, task load, connectivity and compactness were considered.
2.5.1. Task Load Constraint

In theory, the highest value for the R-side task load for a 15-min period is 900 s (15 min times 60 s/min), assuming (unrealistically) that a controller can effectively use all 900 s of available time and that a second controller is handling the D-side task load. When the task load rollup exceeds a certain threshold (around 600 s), it is assumed that two controllers are working the traffic [69]. The maximum acceptable task load in each sector is set as 3420 s for a 1-h period. Assuming that there is only one sector in the studying airspace at first, the task load increases gradually with time. Assuming that there is only one sector in the research airspace at first, the task load will gradually increase with time. When the task load increases to its maximum, the partition of the airspace is triggered. In order to balance the workload, the best partitioning result is that the workload of each sector is half of the maximum workload. Over time, when the task load of both sectors reaches the maximum task load, the best division is that the task load of each sector is 2/3 of the maximum task load. It can be inferred that when airspace is divided into \( k \) sectors, the optimal average task load can be expressed as the following:

\[
\mu = \frac{k-1}{k} w_{l_{\text{max}}} \tag{18}
\]

When the airspace keeps \( k \) sectors unchanged, and the traffic flow of different sectors is not consistent with the growth of time, the sector structure also needs to be adjusted. Before changing the number of sectors from \( k \) to \( k+1 \), the variation range of the average task load is as follows:

\[
\frac{k-1}{k} w_{l_{\text{max}}} \leq \mu \leq w_{l_{\text{max}}} \tag{19}
\]

Equations (18) and (19) are based on the trigger mechanism of sector division, which is when the traffic flow of all sectors increases to \( w_{l_{\text{max}}} \). The real environment may be that the traffic flow growth in each sector is not balanced, some sectors may not grow, and some sectors have reached the maximum. It also needs to be adjusted at this time. The more sectors there are, the more likely the growth imbalance will be. Then the trigger mechanism will fail. In the case of an unbalanced growth of traffic flow in each sector, it is necessary to adjust the sector structure as long as a single sector reaches its maximum. In this case, the task load of the busiest sector can usually be reduced by adjusting the structure of airspace without increasing the number of departments. Assuming this happens, there are currently \( k \) sectors, the total workload is \( w_{l_{\text{tot}}} \) and the average workload is \( w_{l_{\text{tot}}}/k \). Then, the following equation is used to constrain the task load of each sector.

\[
\beta_1 \cdot w_{l_{\text{tot}}}/k \leq w_{l_i} \leq \beta_2 w_{l_{\text{max}}} \tag{20}
\]

\( \beta_1 \) is the minimum deviation from the average of total task load. The smaller the value of \( \beta_1 \) is, the more feasible solutions are. \( \beta_2 \) is the design redundancy parameter. If the design cycle is longer, the \( \beta_2 \) should be smaller, leaving enough room for traffic flow growth.

2.5.2. Connectivity Constraint

The connectivity constraint ensures that the sectors are not fragmented [28]. In other words, a sector must be a contiguous portion of airspace. Connectivity of a sector was assured by checking whether every VC in this sector has at least one common edge with another VC or not. If each VC in a certain sector has no common edge with another VC, then this sector with this VC is not connective.
Definition: Undirected graph \( D = (VC, E) \), VC set \( VC = \{VC_1, VC_2, ..., VC_n\} \) belongs to sector \( s_i \), if two Voronoi cell have a common edge between each other, then the two cells are adjacent. Define the adjacency matrix \( M = (M_{ij})_{n \times n} \) as follows:

\[
M_{ij} = \begin{cases} 
0, & \text{if } VC_i \text{ to } VC_j \text{ is not adjacent} \\
1, & \text{if } VC_i \text{ to } VC_j \text{ is adjacent}
\end{cases}
\]

Similarly, we define the reachable matrix \( R = (R_{ij})_{n \times n} \) of graph \( D \) as follows:

\[
R_{ij} = \begin{cases} 
0, & \text{if } VC_i \text{ to } VC_j \text{ is not reachable} \\
1, & \text{if } VC_i \text{ to } VC_j \text{ is reachable}
\end{cases}
\]

\( R \) can be calculated by the adjacency matrix \( M \). If the sum of every column of \( R \) equals \( n \), then all VCs in sector \( s_i \) are connected, otherwise, the conclusion is contrary.

Two methods are used to guarantee connectivity. One is to reject the new solution directly, keep the original solution and search on the basis of it. The other is to repair the disconnected sector into connected sector through the following methods. According to the adjacency matrix of a sector, the numbers of connected areas were calculated in this sector. If there are more than two connected areas in any sector, only the area with the largest number of VCs are reserved, and the other isolated area is allocated to some sector with the smallest task load and this sector must be adjacent to these isolated areas. When the simulated annealing temperature is higher, connectivity of sectors will be repaired with a larger probability. On the contrary, when the temperature is low, connectivity of sectors will be repaired with a small probability. While these disconnected solutions were rejected with a larger probability.

2.5.3. Compactness Constraint

In the process of using the neighborhood search method to generate new solutions, there are a few VCs very far from their assigned sector. This is called non-compactness. Compactness of a sector is defined as follows.

\[
\delta_j = \sum_{i=1}^{n} \frac{D_{ij}^j z_{ij}}{n} \forall i \in \{1, \ldots, n\}, \forall j \in \{1, \ldots, k\}.
\]

\( D_{ij}^j \) is the distance between generating point of \( VC_i \) and the center of \( s_j \). If \( VC_i \) located in sector \( s_j \), \( z_{ij} = 1 \), otherwise, \( z_{ij} = 0 \). \( \delta_j \) is the average distance of all VC to the center of \( s_j \). The average distance is set as the threshold to assure the compactness of sectors.

Firstly, all VCs that distance between \( \mu_j \) and themselves were larger than the threshold value were found. Then, the numbers of connected regions were found out in the undirected graph composing of these cells. Each of them was allocated unconditionally to the nearest sector.

2.5.4. Number Constraint of Flexible Vertices

The vertices of a sector boundary can be classified according to their sharing by multiple sectors. All vertices of a sector boundary are divided into two categories: Flexible vertices and fixed vertices. The classification of vertices is illustrated in Figure 8.
The blue-coated sector has a vertex shared (two-zone shared vertex) with the red-coated sector, as well as with the green-coated sector. There is a vertex shared by the blue-coated sector, the red-coated sector and the green-coated sector. Both of the two-zone shared vertex and three-zone shared vertex are called flexible vertices. The other vertices marked with a red triangle in the blue-coated sector in Figure 8 were fixed. We found that when the number of flexible vertices (NFV) in a sector was less than three, the sector shape generated by SAA would inevitably appear C shaped. Therefore, we had to force the NFV to be greater than three. This constraint was not used for the SAA search process, but only for the sector evaluation and post-processing. If the number of flexible points in the sector generated by SAA was less than three, the SAA process was repeated.

Figure 8. Flexible and fixed vertices.

3. Solution Method

There are three common design questions related to meta-heuristics: The representation of solutions handled by the algorithm, new solution generation method and the definition of the objective function that will guide the search. The definition of the objective function has been discussed in the previous section. In this section, we will introduce the representations of solutions and solution method of airspace reconstruction problem. The framework of the solution method is shown in Figure 9.

3.1. Solutions Expressions

The representation was based on the proposed reconfiguration problem modeling. When calculating the task load, the representation of solution was similar to the assignment problem. If $V_C_i, \forall i \in \{1, \ldots, n\}$, is assigned to sector $s_j, j \in \{1, \ldots, k\}$, then $z_{i}^{j} = 1$, otherwise $z_{i}^{j} = 0$. Here, an example is given to show the representation of this solution. Suppose there are 10 cells allocated to three sectors.
Figure 9. Framework of the solution method.
The first type of representation of solution $z = \begin{cases} z_1^1, z_1^2, \ldots, z_1^n \\ z_2^1, z_2^2, \ldots, z_2^n \\ \vdots \\ z_m^1, z_m^2, \ldots, z_m^n \end{cases}$ was shown in Table 2. If VCs (1, 2, 3) are adjacent, as well (4, 5, 6) and (7, 8, 9, 10), then the solution show in Table 2 is a feasible one. This type of representation of the solution was used to calculate the task load.

Table 2. First type of representation of solution $z$.

| VC | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|----|---|---|---|---|---|---|---|---|---|----|
| Sector 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Sector 2 | 0 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| Sector 3 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |

The second type of representation of solution $x = \{x_1, x_2, \ldots, x_n\}$ was shown in Table 3. This type of representation of the solution was used to renew the current solution.

Table 3. Second type of representation of solution $x$.

| VC | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|----|---|---|---|---|---|---|---|---|---|----|
| Sector | 1 | 1 | 1 | 2 | 2 | 3 | 3 | 3 | 3 | 3 |

Two kinds of these representations can be transformed into each other.

3.2. K-Means Clustering to Generate the Initial Feasible Solution

Given a set of observations ($p_i$ is coordinate of generation point of VC$_i$, $i = 1, 2, \ldots, n$), $k$-means clustering [73] aims to partition the $n$ observations into $k$ ($\leq n$) sets $s = \{s_1, s_2, \ldots, s_k\}$ so as to minimize the within-cluster sum of squares (i.e., variance). Formally, the objective is to find:

$$\arg\min_S \sum_{j=1}^{k} \sum_{p_i \in s_j} ||p_i - \mu_j||^2,$$

where $\mu_j$ is the centroid of points in $s_j$.

After $k$-means clustering, the sector of VC$_i$ will become clear. The second type of representation of the solution was used to represent $k$-means clustering results. For example, $x = \{1, 1, 1, 2, 2, 3, 3\}$, means VC$_1$, VC$_2$ and VC$_3$ belong to sector 1, VC$_4$ and VC$_5$ belong to sector 2, and VC$_6$ and VC$_7$ belong to sector 3.

3.3. Neighborhood Search Strategy

3.3.1. Variable Neighborhood Search

To prevent local optimization, the following two variable neighborhood search methods were used in the search process.

Algorithm 1, RC_PNB search algorithm:

Firstly, we needed to calculate the workload of all sectors and find out the sectors with the minimum task load. Assuming that the sector with the minimum task load was the sector $s_j$, and VC$_{i1}$, VC$_{i2}$, VC$_{i3}$, VC$_{i4}$ (all Voronoi cells with crimson color of dark blue sector with the smallest task load as showed in Figure 10) adjacent to at least one of the VC in the sector $s_j$, we defined set
{VC_{i1}, VC_{i2}, VC_{i3} \ldots VC_{iA}} as PNB_{sj}, propagable neighborhood of sj. These VC of PNB_{sj} might be distributed in one sector or multiple sectors.

**Figure 10.** An example of the propagable neighborhood (crimson) of the sector (dark blue) with the smallest task load.

A variable neighborhood search algorithm was used according to a certain probability as following.

**Algorithm 1 RC_PNB Search Algorithm**

\[
x_0
\]

PNB_{sj} = PBN\_find\_fun(x,M,\text{sn}_{tl\_min});

A = \text{length}(PNB_{sj});

B = \text{rand}(A);

If rand > 0.5

\[
A_1 = A(1:B,:);
\]

else

\[
A_1 = A(B:end,:);
\]

end

\[
x(A_1) = \text{sn}_{tl\_min}
\]

Connectivity constraint check and assurance algorithm, \(x\rightarrow x_1\)

Compactness constraint assurance, \(x_1 \rightarrow x_2\)

If \(F(x_2) < F(x_0)\)

\[
x_0 = x_2
\]

else

\[
x_0 = x_0
\]

end
Algorithm 2, PN_VC search algorithm:

All VCs adjacent to other sectors in the sector $s_j$ with the lowest task load are called expandable boundaries points (EBP). A neighborhood search algorithm of expandable boundary points (EBP_NB search algorithm) in minimum task load sector $s_j$ was developed as following.

As in Figure 11a, one VC (pink color) in EBP was randomly selected, all VCs around this VC but not belonging to $s_j$ (red color) were called expandable boundary points of this VC (EBP_VC). A neighborhood search algorithm of propagable neighborhood of VC (PN_VC search algorithm) in minimum task load sector $s_j$ was developed as following.

### Algorithm 2 PN_VC Search Algorithm

```plaintext
x = x_0
Random select a VC in EBP_VC
Check out PN_VC of this VC
Allocate PN_VC to $s_j$ and update x
Connectivity constraint check and assurance algorithm, $x \rightarrow x_1$
Compactness constraint assurance, $x_1 \rightarrow x_2$
If $F(x_2) < F(x_0)$
   $x_0 = x_2$
else
   $x_0 = x_0$
```

(a) Expansible boundary points

Figure 11. Cont.
3.3.2. Connectivity Constraint Check and Assurance

After calculating the reachable matrix $R$ of the sector, if the sum of each column of $R$ is less than $n$, all connected components of the sector should be detected. If a sector is divided into several blocks by using the neighborhood search method, only the largest blocks in the sector are retained, other smaller blocks are unconditionally allocated to the sector connected to these components with the smallest distance.

3.3.3. Compactness Constraint Assurance

In order to ensure the compactness of the sector, the average distance from the generating points of VCs to the center point of all VCs was set as a threshold to assure compactness of sectors. If some VCs were far from their assigned sector more than a threshold in Equation (21), they would be allocated to the adjacent sector with the smallest distance; otherwise, they would be kept in the original sector.

3.4. Simulated Annealing Algorithm

Simulated annealing starts from a state $x_0$ and continues until temperature $T$ cools down to $T_{\text{min}}$ at a cooling rate $\sigma$. In the process, the call `neighbor`($x$) should generate a randomly chosen neighbor of

![Figure 11. An example of expandable boundary points (a) and expanded boundaries (b) in PN_VC search.](image-url)
a given state \(x\); to increase the search space, repeat call \(\text{neighbor}(x)\) \(L\) times at a specific temperature. The call \(\text{random}(0, 1)\) should pick and return a value in the range \([0, 1]\), uniformly at random. The algorithm determines whether the new solution \(x_{\text{new}}\) is better or worse than the current solution \(x\). If the new solution is better than the current solution, it becomes the next solution. If the new solution is worse than the current solution, the algorithm can still make it the next solution. The algorithm accepts a worse solution based on an acceptance function \(1/(1 + e^{\delta T})\). Since both \(\delta = E(x_{\text{new}}) - E(x)\) and \(T\) are positive, the probability of acceptance is between 0 and 1. Lower temperature leads to smaller acceptance probability. At the same time, larger \(\delta\) leads to smaller acceptance probability. The best solution and value of objective function were recorded at every specific temperature. After the simulated annealing, the corresponding solution of the minimum value of all objective function at different temperatures was extracted as the optimal solution. The following pseudo code presents the details of the simulated annealing algorithm:

### Algorithm 3 Simulated Annealing Algorithm

1. Let \(x = x_0\).
2. \(n = \text{length}(x_0)\)
3. \(T = T_0\)
4. \(M = 100, \% m\) is the iteration times at different temperatures
5. \(T_{\text{min}} = 1e^{-3}\)
6. \(\text{Partition} = []\); 
7. While \(T > T_{\text{min}}\)
8. \(\text{Temp} = \text{zeros}(m,n + 1)\);
9. for \(k = 1:m\)
10. If \(\text{rand}(1) > 0.6\) 
11. \(\text{RC_PNB search algorithm, } x_{\text{new}} \leftarrow \text{neighbor}(x)\)
12. else 
13. \(\text{PN_VC search algorithm, } x_{\text{new}} \leftarrow \text{neighbor}(x)\)
14. end
15. If \(F(x_{\text{new}}) < F(x_0)\) 
16. \(x \leftarrow x_{\text{new}}\)
17. Elseif \(1 + e^{\frac{F(x_{\text{new}}) - F(x_0)}{T}} \geq \text{random}(0, 1)\) 
18. \(x \leftarrow x_{\text{new}}\)
19. End
20. \(\text{temp}(k,:) = [x, F(x)];\)
21. end
22. \([F_{\text{min}}, \text{index}] = \text{min(temp(:,end))}; \% \text{Find the optimal partition at current temperature.}\)
23. \(\text{Partition} = \text{[partition; temp(index,end)]};\)
24. \(T = T \times \sigma; \% \sigma\) is cooling rate
25. End
26. \([F_{\text{opt}}, \text{index1}] = \text{min(Partition(:,end))};\)
27. \(x_{\text{opt}} = \text{Partition(index1,1:end-1); } \% \text{the final state is } x_{\text{opt}}, \text{optimal value of objective function is } F_{\text{opt}}\)

### 3.5. Post-Processing

After solving the model by SAA, the boundaries of each sector are obtained. Even considering the compactness of the objective function, sector boundaries generated by SAA, like all bottom-up methods, still appear as zigzag, ladder or C-shaped. Drew [41] used the Douglas-Peucker (DP) algorithm to smooth shared boundaries of the sectors generated by mixed integer programming, there are still some obvious problem, such as boundaries with a large zigzag shape. Li, Wang, Hwang and Hwang [29] also developed a bisection method based on the shortest path to smooth boundaries of the sectors. How these methods ensure the balance of task load among sectors is questionable, because they adjust boundaries based on distance parameters rather than task load. However, the most critical criterion
for sector division is task load balancing, because only task load balancing can give each sector the opportunity (buffer) to increase traffic throughput.

Generally, a convex hull can be constructed for each pair sectors generated by SAA (or any other methods). Then, there must be a line that can bisect this convex hull into two sectors with the balance task work. We found that when all these lines are drawn, these lines cannot clearly divide the studied airspace into the required number and shape of sectors. Therefore, this method was abandoned.

To illustrate the principle of our algorithm, we first observed the boundary characteristics of blue-coated sectors from Figure 8. Taking the blue-coated sector shown in Figure 8 as an example, we called the boundary of the studied airspace as the outer boundary, and the irregular boundary between the blue-coated sector and the adjacent (red or pink) sector as the shared boundary. Our algorithm was to smooth these irregular shared boundaries, and try to ensure convexity and task load balance of all sectors, as well as connectivity and compressibility. Next, all fixed and flexible vertices (described in Section 2.5.4) in each sector are found out by repeating this process. If all boundary vertices are connected by a straight line, the sector that coincides with the original sector can be obtained. However, in this way, the task load of the sector will be different from that of the original sector, resulting in unbalanced task load of the sector. Therefore, our idea was to reposition these flexible vertices and rebuild sector boundaries to balance workload sectors.

In order to guarantee the near-convexity of sector shape in the process of rebuilding sector boundaries, we added a near-convexity evaluation function ($\text{cost}_{\text{cvx}}^{\text{tot}}$):

$$\text{cost}_{\text{cvx}}^{\text{tot}} = \sum_{i=1}^{k} C_i^{\text{cvx}} \cdot$$ (23)

$$C_i^{\text{cvx}} = \frac{SA_i^{\text{cvx}} - SA_i}{SA_i^{\text{cvx}}} \cdot$$ (24)

$SA_i^{\text{cvx}}$ is the area of the convex hull of the vertex of the $i$th sector, in square meters.

$SA_i$ is the area of the $i$th sector, in square meters.

$C_i^{\text{cvx}}$ is the concave coefficient of shape for the $i$th sector. The smaller $C_i^{\text{cvx}}$ is, the less concave the polygon is and the closer it is to the convex shape.

$\text{cost}_{\text{cvx}}^{\text{tot}}$ is the concave coefficient of shape for all sectors. The smaller $\text{cost}_{\text{cvx}}^{\text{tot}}$ is, the better the overall convexity of all sectors is. In this way, this performance parameter can be added to Equation (9) to form the following new evaluation function, which can be used to evaluate the new sector generated by the random change of the position of these flexible vertices.

$$F = \alpha_1 \cdot \text{cost}_{\text{imb}} + \alpha_2 \cdot \text{w}^{\text{tot}} + \alpha_3 \cdot \text{cost}_{\text{SDT}}^{\text{tot}} + \alpha_4 \cdot \text{cost}_{\text{tin}}^{\text{tot}} + \alpha_5 \cdot \text{cost}_{\text{cvx}}^{\text{tot}} \cdot$$ (25)

To prevent a concave shape, we added a constraint that the angle between all adjacent lines (counterclockwise) connected to the three-share point should not be greater than 180 degrees. In the MC-CLFV algorithm, this kind of constraint is called concave shape checking.

In order to prevent sector boundaries from getting too close to some important points (such as large traffic flow routes, intersections, airports, etc.), the minimum distance constraint between departmental boundaries and important points is added [29]. In the MC-CLFV algorithm, this kind of constraint is called minimum distance checking.

3.5.1. MC-CLFV Algorithm

As mentioned earlier, the key to smooth the boundary is to locate these flexible vertices. Therefore, the problem of sector boundary smoothing becomes the problem of locating these flexible vertices. Monte Carlo techniques can be used to ensure high quality, robust designs [74,75]. In order to improve the search efficiency, we developed a dynamic Monte Carlo method by changing the location of flexible vertices (MC-CLFV).
Given all flexible vertices \((VF_1, VF_2, \ldots, VF_n)\) belong to set \(VF\), and all fixed vertices \((VD_1, \ldots, VD_n)\) belong to set \(VD\). \(B\) is boundary of airspace. \(L\) is the air routes passing by airspace. Then the shape of airspace, \(G\), is a function of the above vectors, which can be defined as follows.

\[
G = g(VF, VD, B, L)
\] (26)

\(f\) is the vector of flow on air routes \(L\). Then the objective function (25) under the specific sector shape is the function of traffic flow and shape, which can be expressed as the following.

\[
F = F(G, f)
\] (27)

Before describing the algorithm, we give the following definitions.
\(\Gamma\) is the standard deviation of task load for all sectors.
\(\Gamma_0\) is the initial the standard deviation.
\(F_0\) is the value of evaluation Equation (27).
\(r_0\) is base search radius, which is equal to half of the average distance between all flexible vertices.

In search process of the MC-CLFV algorithm (Algorithm 4), a random location point around each flexible vertex within the radius is used to replace the location of these flexible vertexes in \(VF\). Then \(G\) and \(F\) change with locations of these flexible vertices.

The search process of the Algorithm 4 is as following.

---

**Algorithm 4 MC-CLFV Algorithm**

Check out all flexible vertices \(VF_1, VF_2, \ldots, VF_n\) and all fixed vertices \(VD_1, \ldots, VD_n\) in \(s_i\), \(i = 1, \ldots, k\), sorted them counter-clockwise

Setting base search radius, \(r_0\), and its decrease rate in every iteration, \(\sigma^1 = 0.96\)

Calculate the value of the evaluation function, \(F_0\), Calculate initial the standard deviation \(\Gamma_0\)

While \(\Gamma_0 > \tau\) (\(\tau\) is maximum acceptable standard deviation)

\[
r = r_0 \times \sigma^1
\]

For \(i = 1: \text{length}(VF)\)

\[
\theta = 2\pi \times \text{rand}()
\]

\[
x(VF_i) = x(VF_i) + r \times \text{rand}() \times \cos \theta;
\]

\[
y(VF_i) = y(VF_i) + r \times \text{rand() } \times \sin \theta;
\]

end

\(VD\) and renewed \(VF\), constitute a new shape of sector.

If concave shape checking or minimum distance checking is not true

Continue;

end

\(G = g(VF, VD, B, L)\)

Calculate \(F = F(G, f)\)

\(\Delta F = F_0 - F_0\)

\[
\Gamma = \sqrt{\frac{1}{k} \sum_{j=1}^{k} (F_j - \bar{F})^2}
\]

If \(\Delta F \leq 0\) and \(\Gamma \leq \Gamma_0\) and \(\beta_1 \times \text{mean}(F_j, j = 1 \ldots k) \leq \text{min}(F_j, j = 1 \ldots k)\) and \(\max(F_j, j = 1 \ldots k) \leq \beta_2 \times \text{w}_{\text{w}}\text{max}\)

\[
F_{\text{opt}} = F_0
\]

\[
\Gamma_0 = \Gamma
\]

\(G_{\text{opt}} = G\)

end

end
3.5.2. MC-RC Algorithm

When some sector produced by SAA is surrounded by other sectors (hub-sector), the MC-CLFV algorithm cannot partition airspace properly. Especially, when traffic flow is crossed in the center of the airspace, or when an important airport is located in the airspace center. Basu, Mitchell and Sabhnani [43] also developed a ‘Pie Cutting’ method for more flexibility during sectorization. However, they did not specify what airspace the pie-cut method would apply to. Traffic flows and routes in different airspace have different characteristics, so the shape or partition method of the sector should be different. The concept of generic sectors proposed by the Federal Aviation Administration (FAA) supports this view. Generic sectors are defined as segments of airspace in air route traffic control centers (ARTCCs) that controllers could manage without significant specialized training or experience, beyond what they would normally acquire to become certified [76,77]. The following Monte Carlo method by radius changing (MC-RC) was developed for this type of situation.

The sector whose geometric center is closest to the geometric center of the airspace is chosen as the hub-sector. A circle, which its center is the geometric center of hub-sector and radius is the average distance between center and points consisting of the convex hull of the hub-sector, is constructed. By adjusting the radius, the task load of the circular region can be equal to that of the original sector.

There is an overlapping region between two convex hulls of adjacent sectors. A line, which passes through the center of the hub-sector and the center of the overlapping region, is called the spoke. The spoke will divide the overlapping areas into two adjacent sectors. In theory, near the overlapping area of the convex hull, there must be a spoke that can divide task load of two sectors equally. Such a spoke can be found by keeping the starting point of the spoke (center of the hub-sector) unchanged and only changing the bearing of spoke from center. Figure 12 shows the initializing hub and spoke.

Figure 12. Initializing hub and spoke.
The search process of the MC-RC algorithm (Algorithm 5) is as following.

Configuration $G$ is determined by dynamic variables $r_h, \theta = (\theta_1, \theta_2, \ldots, \theta_{k-1})$ and static variables $B, L$. $k$ is the number of sectors. Graph can be expressed as $G = g(r_h, \theta, B, L)$. $r_h$ is the radius of the hub sector. $\theta$ is the vector consisting of a three bearing of the spoke. $B$ is the boundary of airspace. $L$ are air routes passing by airspace. Then the value of the objective function in Equation (9) can be expressed as $F = F(G, f)$. $f$ is the vector of flow on air routes $L$. Given that the value of the objective function of every sector is $F_j, j = 1, 2, \ldots, k$, under configuration $G$ and traffic flow $f$. Then $F_{\text{ave}}$ is the average of value of task load of all sectors. Given that $F_h$ is the value of the task load of the hub sector, then the task load difference between $F_h$ and $F_{\text{ave}}$ can be expressed as the following.

$$\delta_h = F_{\text{ave}} - F_h.$$  \hspace{1cm} (28)

If $\delta_h > 0$, $r_h$ will increase by $\frac{\delta_h}{F_{\text{ave}}}$ to make $F_h$ equal to $F_{\text{ave}}$, and vice versa. Therefore, $r^{i}_h$ at ith iteration can be expressed as the following.

$$r^{i}_h = r^{(i-1)}_h \left(1 + \frac{\delta_h}{F_{\text{ave}}} \right).$$ \hspace{1cm} (29)

$\Gamma$ is used to represent the standard deviation of the objective function. The pseudo-codes for solving $r^i_h$ and $\theta^i$ by the Algorithm 5 is shown as the following.

\begin{algorithm}[H]
\caption{MC-RC Algorithm}
\begin{algorithmic}
\State Initializing $r^0_h, \theta^0, \delta_0, i = 0$
\State Calculating configuration parameters, $G^0 = g(r^0_h, \theta^0, B, L)$
\State Calculating of value of objective function, $F^0 = F(G, f)$
\While {$\delta_h \geq 50$ or $\Gamma \geq \tau$}
\State $r^{i}_h = r^{(i-1)}_h \left(1 + \frac{\delta_h}{F_{\text{ave}}} \right)$
\State Generate three random integers between 0 and 360 and assign them to $\theta^i$, $n$ mean at $i$th iteration.
\State $G^i = g(r^i_h, \theta^i, B, L)$.
\State If minimum distance checking is not true
\State \hspace{1cm} Continue;
\State \endWhile
\State $F^i = F(G^i, f)$
\State $\delta_h = F_{\text{ave}} - F^h$
\State $\Gamma = \sqrt{\frac{1}{i} \sum_{j=1}^{k} (F_j - F)^2}$
\State $i = i + 1$;
\State If $F^i < F^0$ and $\delta_h < 50$ and $\Gamma < \tau$ and $\beta_1 \times \text{mean}(F_j, j = 1 \ldots k) \leq \text{min}(F_j, j = 1 \ldots k)$ and $\text{max}(F_j, j = 1 \ldots k) \leq \beta_2 \times \text{w}_{\text{max}}$
\State $F^0 = F^i$
\State $G^{opt} = G^i$
\State End
\end{algorithmic}
\end{algorithm}

Figure 13 is a case of hub-spoke partition by optimizing the radius of the hub sector and bearing of the spoke using the MC-RC algorithm.
4. Results

We extracted one week’s data from Shanxi regional airspace. Since there were few flights between 20:00 p.m. and 7:00 a.m. in Shanxi regional airspace, we only counted the traffic flow data of each segment from 7:00 a.m. to 20:00 a.m. and took the average value to calculate the task load during the reconfiguration process. The parameters used in different algorithms are shown in Table 4.

Table 4. Parameters used in the algorithm.

| Parameters                              | Value       | SAA | MC-CLFV | MC-RC |
|-----------------------------------------|-------------|-----|---------|-------|
| Task load imbalance                      | $\alpha_1 = 4000$ | √   | √       | √     |
| Total coordinating task load            | $\alpha_2 = 200$ | √   | √       | √     |
| Cost of a short dwell time              | $\alpha_3 = 5$ | √   | √       | √     |
| Cost of reentering the same sector      | $\alpha_4 = 1500$ | √   | √       | √     |
| Initial temperature                     | $T_0 = 1e+3$ | √   | √       | √     |
| Termination temperature                 | $T_{min} = 1e-3$ | √   |         |       |
| Cooling rate                            | $\sigma = 0.8$ | √   |         |       |
| Radius decreasing rate in every iteration | $\sigma^1 = 0.96$ | √   |         |       |
| Monitoring task load per 10 min         | $\alpha_{mot} = 22/600 (s/flight)$ | √   | √       | √     |
| Coordination task load required for each handover | $\alpha_{cod} = 9 (s/flight)$ | √   | √       | √     |
| Permissible minimum of dwell time       | $t_{min}^{dw} = 240 (s)$ | √   | √       | √     |
| The minimum deviation from the average of the total task load | $\beta_1 = 0.5$ | √   | √       | √     |
| The design redundancy parameter         | $\beta_2 = 0.95$ | √   | √       | √     |
| The maximum acceptable standard deviation | $\tau = 100$ | √   | √       | √     |
| The maximum acceptable task load in each sector | $w_{max} = 3420 (s)$ | √   | √       | √     |
4.1. SAA Test Result

Based on the three Voronoi diagrams, we used the SAA described in Section 3 to execute the automatic sector optimization calculation for the studying airspace. The results comparing are shown in Table 5.

**Table 5.** Comparing simulated annealing algorithm (SAA) results with three kinds of seeds generating method.

| Seeds Generating Method | TL in Sector1 (s) | TL in Sector2 (s) | TL in Sector3 (s) | TL in Sector4 (s) | Std | TTL in Airspace (s) | Figure |
|-------------------------|-------------------|-------------------|-------------------|-------------------|-----|--------------------|--------|
| Based on randomly       | 3316              | 2409              | 2443              | 2023              | 546.4 | 10,191             | Figure 14 |
| Based on every 50 km    | 2618              | 2459              | 2688              | 2890              | 178.8 | 10,655             | Figure 15 |
| Based on hexagonal      | 2383              | 3144              | 2493              | 2265              | 393.0 | 10,285             | Figure 16 |
| Current configuration   | 3018              | 2503              | 4025              | 3031              | 636.6 | 12,577             | Figure 17 |

Note: TL is the abbreviation of task load. TTL is the abbreviation of total task load. The (s) is the abbreviation of seconds. Std is the abbreviation of standard deviation.

![Figure 14. Reconfiguration results based on random seeds with SAA.](image-url)
Figure 15. Reconfiguration results based on seeds every 50 km with SAA.
Figure 16. Reconfiguration results based on hexagonal seeds with SAA.

Figure 17. Current configuration and task load.
4.2. Post-Processing Result

The first purpose of post-processing was to smooth the boundary while keeping the task load balance and other constrains. The second purpose was to compare the results of post-processing with MC-CLFV and MC-RC. The results of post-processing graph are shown in Figures 18–20.

Figure 18. Post-processing results of Figure 14 with MC-CLFV.
Figure 19. Post-processing results of Figure 15 with MC-CLFV.
4.3. Computational Efficiency Evaluation

The annealing parameters of the SAA algorithm are the initial temperature $T_0$ and termination temperature is $T_{\text{min}}$, and cooling rate of $T_0$ in every iteration is $\sigma$. According to the basic idea of the

---

Table 6. Comparison of results between the MC-CLFV algorithm and MC-RC algorithm.

| Algorithm     | Original Configuration | After Post-Processing | TL of S1 (s) | TL of S2 (s) | TL of S3 (s) | TL of S4 (s) | Std | TTL in Airspace (s) |
|---------------|------------------------|-----------------------|--------------|--------------|--------------|--------------|-----|-------------------|
| MC-CLFV       | Figure 14              | Figure 18             | 3061.6       | 3197.5       | 3134.2       | 3280.9       | 93.2| 12,674            |
| MC-CLFV       | Figure 15              | Figure 19             | 3254.5       | 3076.6       | 3116.2       | 3128.0       | 77.0| 12,575            |
| MC-RC         | Figure 16              | Figure 20             | 3154         | 3143         | 3217         | 2937         | 121 | 12,451            |
| Current config| Figure 17              |                       | 3018         | 2503         | 4025         | 3231         | 632.6| 12,777            |

Note: The abbreviations in this table have the same meaning as in Table 5.

4.3. Computational Efficiency Evaluation

The annealing parameters of the SAA algorithm are the initial temperature $T_0$ and termination temperature is $T_{\text{min}}$, and cooling rate of $T_0$ in every iteration is $\sigma$. According to the basic idea of the
simulated annealing algorithm, the cycle times of outer loop could be $\frac{T_{\min}}{T_0}$, the cycle times of inner loop is set as $m$. Therefore, the calculation time of SAA could be expressed as:

$$T_{SAA} = \log_{\frac{T_0}{T_0}} m T_{inner}$$

where $T_{inner}$ is the time when the inner loop is executed once. The $T_{inner}$ has a relationship with the calculation time of the monitor workload ($L_R L_S$), calculation time of the coordination workload ($L_R L_S L_{C1} L_{C2}$) and calculation time of the structural adjustment of sectors ($L_{C1}^2$). The $L_R$ is the time the program traverses all routes once. The $L_S$ is the time the program traverses all sectors. The $L_{C1}$ is the time the program traverses all Voronoi cells. The $L_{C2}$ is the time the program traverses all boundaries of Voronoi cells. The $T_{inner}$ could be expressed as:

$$T_{inner} = L_R L_S + L_R L_{C1} L_{C2} L_S + L_{C1}^2.$$  

(31)

Therefore, the final expression of the calculation time of SAA is:

$$T_{SAA} = \log_{\frac{T_0}{T_0}} m (L_R L_S + L_R L_{C1} L_{C2} L_S + L_{C1}^2).$$

(32)

The calculation time of the SAA algorithm with different numbers of cells and sectors is shown in Figure 21. The calculation time of the SAA algorithm increased with the increase of the numbers of cells and sectors. When the number of cells was large, the difference of the computing time caused by the increase of the number of sectors became larger.

![Figure 21. Calculation time of the SAA algorithm with different numbers of cells and sectors.](image-url)
The cycle times of the MC-CLFV algorithm are a random constant $\xi$. The circulation time for one execution of the MC-CLFV algorithm consists of three major parts: Monitor the workload calculation time ($L_R \cdot L_S$), calculation the time of the structural adjustment of sectors ($L_{VF} \cdot L_{D_{min}} \cdot L_S$) and calculation the time of the coordination workload calculation ($L_R \cdot L_{C1} \cdot L_{C2} \cdot L_S$). The $L_{VF}$ is very short, because it is the time that all flexible vertices are located. The $L_{D_{min}}$ is the detection time for the minimum distance constraint (from the key intersection to sector boundary). The theoretic calculation time of the MC-CLFV algorithm could be expressed as the following:

$$T_{MC-CLFV} = \xi \cdot (L_R + L_R \cdot L_{C1} \cdot L_{C2} + L_{VF} \cdot L_{D_{min}}) \cdot L_S. \quad (33)$$

Although the MC-CLFV algorithm also consists of three major parts as the SAA algorithm, the factual runtime decreases dramatically because of a new sector adjustment method by introducing flexible vertices. The calculation time of the MC-CLFV algorithm increased with sector numbers as shown in Figure 22.

![Figure 22. Calculation time of the MC-CLFV algorithm with different sector numbers.](image)

The reason for the increase of the calculation time is not only the increase of the number of sectors, but also the reduction of the area of each sector caused by the increase of the number of sectors, which makes it difficult to meet the minimum distance constraint between critical intersections and intersections. This leads to the continuous generation of infeasible sector boundaries, which increases the computational time. As shown in Figure 23, when the number of sectors was constant, the computational time increased with the minimum distance constraint, which proved our reasoning.

The theoretical calculation time of the MC-RC algorithm is similar to that of the MC-CLFV algorithm. Since it only needs to adjust the radius of the central circular sector and the angle of the
rays that make up the other sectors, the reconstruction time is shorter. The theoretic circulation time of the MC-RC algorithm could be expressed as the following:

\[ T_{MC-CLFV} = \xi \cdot (L_R + L_{C1} \cdot L_{C2} + L_G \cdot L_{D_{min}}) \cdot L_S. \] (34)

The \( L_C \) is the reconstruction time of sectors for just once.

When the number of sectors is equal to 4, the average calculation time of the MC-RC algorithm is very short. The worst case consumes only about 140 s. However, when the number of sectors is greater than 4, the program will hardly terminate without changing the termination condition (standard deviation of sector workload). In order to test the computational speed of more than four sectors, we increased the standard deviation from 100 to 400. The worst case could consume about 1700 s. It could be inferred that the MC-RC algorithm was not suitable for constructing more than four sectors in Taiyuan airspace.

![Figure 23. Calculation time of the MC-CLFV algorithm with different minimum distances.](image)

5. Discussion

From the result of SAA in Table 5, the sector generated by SAA method was more balanced than the current sector workload, and the total workload was smaller. We also found that the solution space of SAA varied with the seeds generating method and the number of seeds. Three Voronoi diagram generation methods, combined with the number of variable seed points, improved the reliability of airspace design meeting the required performance standards. This revealed that there were defects in some previous literatures, which only depended on one method to discretize the airspace. Although simulated annealing is a very mature algorithm, it must be pointed out that adding the re-entry sector penalty and short dwell time penalty to the objective function will have an impact on the optimization search. If the weight of these two penalties is too large, the algorithm will be limited...
to local optimization, and if the settings are too small, there will be a re-entry sector and short dwell. However, because the simulated annealing algorithm accepts infeasible solutions by itself with a certain probability, it allows the re-entry sector and short residence to occur in the exploration process, which makes it possible to explore the global optimization solution, and makes it possible for the region growth method to produce sectors that meet the performance requirements. Using SAA to find a good initial solution lays a foundation for a warm start in post-processing.

However, irregular sector boundaries in Figures 14–16 were not satisfactory. Any method based on discrete airspace block reorganization would have such problems as zigzag, c-shaped, trapezoidal and so on. Therefore, effective post-processing is very important. The result in Table 6 shows that the two proposed post-processing methods could achieve a more balanced partitioning scheme than the existing sector. The MC-CLFV algorithm could get more balance configuration than MC-RC, but the MC-RC algorithm could get smaller total task load. The choice between the two performance criteria, the minimum total workload and the balanced task load of each sector, seems to be contradictory. Of course, we expected the minimum total task load and the balanced task load of each sector at the same time. However, when we cannot have both of them, how do we choose? Answering this question needs to be traced back to the purpose of AS. The fundamental purpose of AS is to increase traffic throughput. Especially when the current throughput of airspace is mainly limited by the task load of controllers, the purpose of AS is to reduce the average task load of controllers, regardless of whether the number of sectors remains unchanged or by increasing the number of sectors. When the number of sectors and the demand of controllers remain unchanged, a more balanced scheme is preferred for the fundamental purpose of AS. Since only in this way, each sector has more buffers to cope with fluctuating traffic flow and traffic growth. Therefore, we concluded that the MC-CLFV algorithm was superior to the MC-RC algorithm in Shanxi airspace for increasing traffic throughput. However, compared with Figures 18 and 20 and Figure 19, it was found that there were no narrow blocks in the sector generated by MC-RC, and MC-RC seemed to show some advantages when it was necessary to utilize the space of narrow blocks. It must be mentioned that in order to evaluate the computing time of the MC-RC algorithm, we relaxed the sector workload balancing constraints. When the number of sectors is more than 4, there may be no feasible solution without relaxing the sector workload balance constraints. This means that the MC-RC algorithm is not as applicable as MC-CLFV algorithm. We believed that any algorithm has its use value in an appropriate environment.

Comparing the results of Tables 5 and 6, we could find that although the sector generated by the SAA algorithm had irregular boundaries, the total task load was smaller than that after irregular boundaries were smoothed, which indicates that appropriately allowing some non-convex boundaries was a measure to reduce the total workload, and blindly pursuing convex boundaries would increase the total workload.

6. Conclusions

Partitioning the airspace into optimal sectors, even 2D, is still a leading or forefront research issue. It is a multi-objective optimization problem, and the contradiction between multiple objectives or constraints makes the problem complicated. Many technologies and methods, such as Voronoi diagrams, k-mean clustering, graphic reshaping technology, graph coding technology, visualization, task load calculation, the simulated annealing algorithm (SAA) and variable neighborhood search strategies, post-processing of sector boundary based on the Monte Carlo method and morphology, were synthetically used to realize automatic AS in this paper. The modeling approach and algorithm solution presented in this paper were tested and compared to existing sectorization. The provided results demonstrated that the proposed sector design method framework was able to provide very satisfying sectorization with regards to sector load balancing, as well as to total task load minimization, minimum distance to boundary, the number of re-entering and short-dwell criteria.

The graph cutting algorithms proposed by us had two theoretical contributions. The first was to eliminate the inherent problem of irregular sector boundaries generated by region growth algorithm.
Second, because of the results of the previous region growing algorithm as a hint, how many sectors and the number of flexible vertices in the graph cutting algorithm became known parameters. Therefore, the AS problem became the problem of determining the location of flexible vertices. This greatly simplifies the complexity of the graph cutting algorithm.

The proposed method framework of AS and software could provide reliable assistant design and analysis tools for airspace planners to design airspace, improve the reliability and efficiency of design and reduce the burden of airspace planners while guaranteeing the key performance indicators in the process of AS. In addition, this laid the foundation for reconstructing airspace with a more intelligent method. The presented methods when appropriately adapted, could be applied in another partitionable system, e.g., in land partitioning, power network partitioning, communication network partitioning and other transport sectors, e.g., in technological processes scheduling, rescue actions, crisis management, etc.

We did not claim that we had completed the full-automatic design of the airspace. Since airspace design is a complicated process, we had only solved the problem that the initial airspace design was almost completed manually in China in the past. This scheme has to be further simulated and evaluated by SIMMOD, AIRTOP and other tools [78]. Finally, it has to be evaluated by human in the loop before it can be implemented [6]. Some modifications will even be made based on feedback in a field test. This reflects the fact that there is a huge gap for improvement in the current task load model and airspace sectorization model. It is also a great opportunity and challenge for researchers to automate the whole process of airspace design.

In general, the proposed method framework not only satisfied a variety of key performance indicators, but also eliminated the irregular sector shape generated by the regional growth method, which laid the foundation for improving the reliability and acceptability of the airspace design and enriches the automatic airspace reconstruction method.

**Author Contributions:** Conceptualization, Z.Y. and J.M.; methodology, Z.Y.; software, Z.Y. and F.K.; validation, Z.Y. and F.K.; formal analysis, Z.Y.; investigation, Z.Y.; resources, W.G.; data curation, Z.Y.; writing—Original draft preparation, Z.Y.; writing—Review and editing, Z.Y. and B.Z.; visualization, F.K.; supervision, J.M.; project administration, Z.Y.; funding acquisition, B.Z. and J.M.

**Funding:** This work was supported in part by the National Natural Science Foundation of China under Grant 71571182, in part by the National Natural Science Youth Foundation of China under Grant 61603396, in part by the National Natural Science Foundation of China and the Civil Aviation Grant under Grant U1633124 and Grant U1733102, in part by Shenzhen Science and Technology Innovation Committee under Grant ZDSYS20170725140921348, and in part by the Air Traffic Management Research Institute, NTU and the Civil Aviation Authority of Singapore under grant M4061216.057.

**Conflicts of Interest:** The authors declare no conflict of interest.

**References**

1. Cui, Q.; Wei, Y.M.; Li, Y.; Li, W.X. Exploring the differences in the airport competitiveness formation mechanism: Evidence from 45 Chinese airports during 2010–2014. *Transp. B* 2017, 5, 330–346. [CrossRef]
2. Chen, Y.X.; Yu, J.; Tsai, S.B.; Zhu, J.F. An Empirical Study on the Indirect Impact of Flight Delay on China’s Economy. *Sustainability* 2018, 10, 357. [CrossRef]
3. Guo, Q.L.; Sun, H.B.; Zhang, B.M.; Wen-Chuan, W.U. Power Network Partitioning Based on Clustering Analysis in MVar Control Space. *Autom. Electr. Power Syst.* 2005, 29, 36–40.
4. Tang, J.; Alam, S.; Lokan, C.; Abbass, H.A. A multi-objective evolutionary method for dynamic airspace re-sectorization using sectors clipping and similarities. In Proceedings of the 2012 IEEE Congress on Evolutionary Computation, Brisbane, Australia, 10–15 June 2012.
5. Ehrmanntraut, R.; McMillan, S. Airspace design process for dynamic sectorisation. In Proceedings of the 2007 IEEE/AIAA 26th Digital Avionics Systems Conference, Dallas, TX, USA, 21–25 October 2007.
6. Conker, R.S.; Moch-Mooney, D.A.; Niedringhaus, W.P.; Simmons, B.T. New process for “clean sheet” airspace design and evaluation. In Proceedings of the 7th US/Europe ATM Seminar, Barcelona, Spain, 2–5 July 2007.
7. Demetriou, D.; See, L.; Stillwell, J. Integrating GIS and genetic algorithms for automating land partitioning. In Proceedings of the International Conference on Remote Sensing & Geoinformation of the Environment, Paphos, Cyprus, 7–10 April 2014.

8. Yan, X.; He, F.; Chen, Y. Elimination particle swarm optimization with hill climbing algorithm for solving large-scale hardware/software partitioning problem. *J. Southeast Univ.* 2017, 47, 802–816. [CrossRef]

9. Danan, E.; Awduche, D.O.; Jabbari, B. Analytical framework for dynamic traffic partitioning in MPLS networks. In Proceedings of the 2000 IEEE International Conference on Communications, New Orleans, LA, USA, 18–22 June 2000.

10. Melikov, A.Z.; Ponomarenko, L.A. Analysis of models of the integral cellular communication network with different schemes of channel partitioning. *Autom. Remote Control* 2016, 77, 802–816. [CrossRef]

11. Chu, K.C.; Lin, F.Y.S.; Wang, C.S. Modeling of Adaptive Load Balancing with Hybrid F/CDMA and Sectorization Schemes in Mobile Communication Networks. *J. Netw. Syst. Manag.* 2010, 18, 395–417. [CrossRef]

12. Peli, G.; Nooteboom, B. Market Partitioning and the Geometry of the Resource Space1. *Am. J. Sociol.* 1999, 104, 1132–1153. [CrossRef]

13. Krile, S. Optimization approach in multi-stop routing of small islands. *Zesz. Nauk. Akad. Mor. W Szczec.* 2018, 54, 9–16.

14. Guze, S. Review of Methods and Algorithms for Modelling Transportation Networks Based on Graph Theory. *Sci. J. Gdyn. Marit. Univ.* 2018, 25–39. [CrossRef]

15. Wiśnicki, B.; Milewski, D.; Chybowski, L.; Hełczyński, I. The Concept of the Development of Intermodal Transport Network Illustrated by Polish Market. *NAŠE MORE Znan. StručniˇCasopis Za More I Pomor.* 2017, 64, 33–37.

16. Sen, A.; Ghosh, P.; Vittal, V.; Bo, Y. A new min-cut problem with application to electric power network partitioning. *Int. Trans. Electr. Energy Syst.* 2010, 19, 778–797. [CrossRef]

17. Delahaye, D.; Alliot, J.; Schoenauer, M.; Farges, J. Genetic algorithms for partitioning air space. In Proceedings of the Tenth Conference on Artificial Intelligence for Applications, San Antonio, TX, USA, 1–4 March 1994; pp. 291–297.

18. John, R.M.; Robert, G.R.; David, B.F. Genetic Algorithms for Automatic Regrouping of Air Traffic Control Sectors. In *Evolutionary Programming IV: Proceedings of the Fourth Annual Conference on Evolutionary Programming, San Diego, CA, USA, 1–3 March 1995*; MIT Press: Cambridge, MA, USA, 1995; p. 1. [CrossRef]

19. Babić, O.; Krstić, T. Airspace daily operational sectorization by fuzzy logic. *Fuzzy Sets Syst.* 2000, 116, 49–64. [CrossRef]

20. Xue, M. Airspace Sector Redesign Based on Voronoi Diagrams. *J. Aerosp. Comput. Inf. Commun.* 2009, 6, 624–634. [CrossRef]

21. Chen, Y.Z.; Zhang, D.F. Dynamic airspace configuration method based on a weighted graph model. *Chin. J. Aeronaut.* 2014, 27, 903–912. [CrossRef]

22. Kulkarni, S.; Ganesan, R.; Sherry, L. Dynamic Airspace Configuration Using Approximate Dynamic Programming Intelligence-Based Paradigm. *Transp. Res. Rec.* 2012, 2266, 31–37. [CrossRef]

23. Gerdes, I.; Temme, A.; Schultz, M. Dynamic airspace sectorisation for flight-centric operations. *Transp. Res. Part C Emerg. Technol.* 2018, 95, 460–480. [CrossRef]

24. Michalek, D.; Balakrishnan, H. Dynamic Reconfiguration of Terminal Airspace During Convective Weather. In Proceedings of the 49th IEEE Conference on Decision and Control (CDC), Atlanta, GA, USA, 15–17 December 2010; pp. 4875–4881. [CrossRef]

25. Zou, X.; Cheng, F.; An, B.; Song, J.Y. Sectorization and Configuration Transition in Airspace Design. *Math. Prob. Eng.* 2016, 2016, 6048326. [CrossRef]

26. Flener, P.; Pearson, J. Automatic Airspace Sectorisation: A Survey. *arXiv* 2013, arXiv:1311.0653.

27. Bichot, C.-E.; Durand, N. A tool to design functional airspace blocks. In Proceedings of the ATM 2007, 7th USA/Europe Air Traffic Management Research and Development Seminar, Barcelona, Spain, 2–5 July 2007.

28. Trandac, H.; Baptiste, P.; Duong, V. Airspace sectorization with constraints. *RAIRO Oper. Res.* 2005, 39, 105–122. [CrossRef]
29. Li, J.; Wang, T.; Hwang, I.; Hwang, I. A spectral clustering based algorithm for dynamic airspace configuration. In Proceedings of the 9th AIAA Aviation Technology, Integration, and Operations Conference (ATIO) and Aircraft Noise and Emissions Reduction Symposium (ANERS), Hilton Head, SC, USA, 21–23 September 2009; p. 7056.
30. Chen, Y.; Bi, H.; Zhang, D.; Song, Z. Dynamic airspace sectorization via improved genetic algorithm. J. Mod. Transp. 2013, 21, 117–124. [CrossRef]
31. Zhang, D.F.; Chen, Y.Z. Airspace sectorisation via a weighted graph model. Aeronaut. J. 2014, 118, 267–274. [CrossRef]
32. Yousefi, A.; Donohue, G. Temporal and spatial distribution of airspace complexity for air traffic controller workload-based sectorization. In Proceedings of the AIAA 4th Aviation Technology, Integration and Operations (ATIO) Forum, Chicago, IL, USA, 20–22 September 2004; p. 6455.
33. Martinez, S.; Chatterji, G.; Sun, D.; Bayen, A. A weighted-graph approach for dynamic airspace configuration. In Proceedings of the AIAA Guidance, Navigation and Control Conference and Exhibit, Monterey, CA, USA, 5–8 August 2002; p. 6448.
34. Drew, M. A method of optimally combining sectors. In Proceedings of the 9th AIAA Aviation Technology, Integration, and Operations Conference (ATIO) and Aircraft Noise and Emissions Reduction Symposium (ANERS), Hilton Head, SC, USA, 21–23 September 2009; p. 7057.
35. Kulkarni, S.; Ganesan, R.; Sherry, L. Static sectorization approach to dynamic airspace configuration using approximate dynamic programming. In Proceedings of the Integrated Communications, Navigation and Surveillance Conference (ICNS), Herndon, VA, USA, 10–12 May 2011; pp. J2-1–J2-9.
36. Bloem, M.; Gupta, P. Configuring airspace sectors with approximate dynamic programming. In Proceedings of the 27th Congress of International Council of the Aeronautical Sciences, Nice, France, 19–24 September 2010.
37. Jägare, P. Airspace Sectorisation Using Constraint Programming. Available online: Http://www.atmseminar.org/seminarContent/seminar10/papers/216-Flener_0125130237-Final-Paper-4-12-13.pdf. (accessed on 11 September 2019).
38. Leiden, K.; Peters, S.; Quesada, S. Flight level-based dynamic airspace configuration. In Proceedings of the 9th AIAA Aviation Technology, Integration, and Operations Conference (ATIO) and Aircraft Noise and Emissions Reduction Symposium (ANERS), Hilton Head, SC, USA, 21–23 September 2009; p. 7104.
39. Gianazza, D. Forecasting workload and airspace configuration with neural networks and tree search methods. Artif. Intell. 2010, 174, 530–549. [CrossRef]
40. Kicinger, R.; Yousefi, A. Heuristic method for 3D airspace partitioning: Genetic algorithm and agent-based approach. In Proceedings of the 9th AIAA Aviation Technology, Integration, and Operations Conference (ATIO) and Aircraft Noise and Emissions Reduction Symposium (ANERS), Hilton Head, SC, USA, 21–23 September 2009; p. 7058.
41. Drew, M. Analysis of an Optimal Sector Design Method. In Proceedings of the 2008 IEEE/AIAA 27th Digital Avionics Systems Conference, St. Paul, MN, USA, 26–30 October 2008.
42. Sabhnani, G.; Yousefi, A.; Mitchell, J.S. Flow conforming operational airspace sector design. In Proceedings of the 10th AIAA Aviation Technology, Integration, and Operations (ATIO) Conference, Fort Worth, TX, USA, 13–15 September 2010; p. 9377.
43. Basu, A.; Mitchell, J.S.; Sabhnani, G.K. Geometric algorithms for optimal airspace design and air traffic controller workload balancing. J. Exp. Algorithmics 2009, 14, 3. [CrossRef]
44. Sherali, H.D.; Hill, J.M. Configuration of airspace sectors for balancing air traffic controller workload. Ann. Oper. Res. 2013, 203, 3–31. [CrossRef]
45. Xue, M. Three-dimensional sector design with optimal number of sectors. J. Guid. Control Dyn. 2012, 35, 609–618. [CrossRef]
46. Gerdes, I.; Temme, A.; Schultz, M. Dynamic airspace sectorization using controller task load. In Proceedings of the Sixth Sesar Innovation Days, Delft, The Netherlands, 8–10 November 2016.
47. Delahaye, D.; Schoenauer, M.; Alliot, J.-M. Airspace sectoring by evolutionary computation. In Proceedings of the 1998 IEEE International Conference on Evolutionary Computation Proceedings. IEEE World Congress on Computational Intelligence, Anchorage, AK, USA, 4–9 May 1998; pp. 218–223.
48. Pawlak, W.; Goel, V.; Rothenberg, D.; Brinton, C. Comparison of algorithms for the dynamic resectorization of airspace. In Proceedings of the Guidance, Navigation, and Control Conference and Exhibit, Monterey, CA, USA, 5–8 August 2002; p. 4106.
49. Sergeeva, M.; Delahaye, D.; Mancel, C.; Zerrouki, L.; Schede, N. 3d sectors design by genetic algorithm towards automated sectorisation. In Proceedings of the 5th SESAR Innovation Days, Bologna, Italy, 1–3 December 2015.

50. Ito, T.; Demaine, E.D. Approximability of the subset sum reconfiguration problem. J. Comb. Optim. 2014, 28, 639–654. [CrossRef]

51. Cheng, C.K. The optimal partitioning of networks. Networks 1992, 22, 297–315. [CrossRef]

52. Gendreau, M.; Potvin, J.-Y. Handbook of Metaheuristics; Springer: Berlin/Heidelberg, Germany, 2010; Volume 2.

53. Johnson, D.S.; Aragon, C.R.; McGeoch, L.A.; Schevon, C. Optimization by simulated annealing: An experimental evaluation; part I, graph partitioning. Oper. Res. 1989, 37, 865–892. [CrossRef]

54. Johnson, D.S.; Aragon, C.R.; McGeoch, L.A.; Schevon, C. Optimization by simulated annealing: An experimental evaluation; part II, graph coloring and number partitioning. Oper. Res. 1991, 39, 378–406. [CrossRef]

55. Rahimian, F.; Payberah, A.H.; Girdzijauskas, S.; Jelasity, M.; Haridi, S. A Distributed Algorithm for Large-Scale Graph Partitioning. ACM Trans. Auton. Adapt. Syst. 2015, 10, 12. [CrossRef]

56. Dobrin, A. A Review of Properties and Variations of Voronoi Diagrams; Whitman College: Walla Walla, WA, USA, 2005.

57. Sridhar, B.; Sheth, K.S.; Grabbe, S. Airspace complexity and its application in air traffic management. In Proceedings of the 2nd USA/Europe Air Traffic Management R&D Seminar, Orlando, FL, USA, 1–4 December 1998; pp. 1–6.

58. Crump, J.H. Review of stress in air traffic control: Its measurement and effects. Aviat. Space Environ. Med. 1979, 50, 243–248. [PubMed]

59. Averty, P.; Athenes, S.; Collet, C.; Dittmar, A. Evaluating a new index of mental workload in real ATC situation using psychophysiological measures. In Proceedings of the 21st Digital Avionics Systems Conference, Irvine, CA, USA, 27–31 October 2002; pp. 7A4–7A4.

60. Martin, C.; Cegarra, J.; Averty, P. Analysis of mental workload during en-route air traffic control task execution based on eye-tracking technique. In Proceedings of the International Conference on Engineering Psychology and Cognitive Ergonomics, Orlando, FL, USA, 9–14 July 2011; pp. 592–597.

61. Baart, D. An Evaluation of Dynamic Density Metrics Using RAMS; FAA Technical Center Report: Washington, DC, USA, 2001.

62. Yousefi, A.; Donohue, G.L.; Qureshi, K. Investigation of en route metrics for model validation and airspace design using the total airport and airspace modeler (TAAM). In Proceedings of the Fifth USA/Europe Air Traffic Management R&D Seminar, Budapest, Hungary, 23–27 June 2003.

63. Majumdar, A.; Ochieng, W.; McAuley, G.; Lepadetu, C. The factors affecting airspace capacity in europe: A framework methodology based on cross sectional time-series analysis using simulated controller workload data. In Proceedings of Proceedings of the 6th USA/Europe Air Traffic Management R & D Seminar, Baltimore, MD, USA, 27–30 June 2005.

64. Oktal, H.; Yaman, K. A new approach to air traffic controller workload measurement and modelling. Aircr. Eng. Aerosp. Technol. 2011, 83, 35–42. [CrossRef]

65. Chatterji, G.; Sridhar, B. Measures for air traffic controller workload prediction. In Proceedings of the 1st AIAA, Aircraft, Technology Integration, and Operations Forum, Los Angeles, CA, USA, 16–18 October 2001; p. 5242.

66. Kopardekar, P.; Magyarits, S. Measurement and prediction of dynamic density. In Proceedings of the 5th USA/Europe Air Traffic Management R & D Seminar, Budapest, Hungary, 23–27 June 2003.

67. Gianazza, D. Learning air traffic controller workload from past sector operations. In Proceedings of the ATM Seminar, 12th USA/Europe Air Traffic Management R&D Seminar, Seattle, WA, USA, 26–30 June 2017.

68. Jorna, P. Operator workload as a limiting factor in complex systems. In Automation and Systems Issues in Air Traffic Control; Springer: Berlin/Heidelberg, Germany, 1991; pp. 281–292.

69. Council, N.R. Air Traffic Controller Staffing in the en Route Domain: A Review of the Federal Aviation Administration’s Task Load Model; Transportation Research Board: Washington, DC, USA, 2010; Volume 301.

70. FAA. Order 7210.3, Facility Operation and Administration; FAA: Washington, DC, USA, 2007.

71. Palla, N. Graph connectivity by the adjacency matrix. Far East J. Appl. Math. 2003, 12, 189–225.

72. Kanazawa, N.; Sasahira, T.; Kaneda, S.; Haga, H. Automatic generation method of relation matrix for ISM. In Proceedings of the Conference on Knowledge-Based Software Engineering, Tallinn, Estonia, 28–31 August 2006.
73. Likas, A.; Vlassis, N.; Verbeek, J.J. The global k-means clustering algorithm. *Pattern Recognit.* 2003, 36, 451–461. [CrossRef]

74. De, L.A.; Blanco, S.; Clergent, Y.; Dufresne, J.L.; El, H.A.F.R. Monte Carlo method and sensitivity estimations. *J. Quant. Spectrosc. Radiat. Transf.* 2018, 75, 529–538.

75. Wakefield, A.; Miller, S. Improving System Models by Using Monte Carlo Techniques on Plant Models. In Proceedings of the AIAA Modeling and Simulation Technologies Conference and Exhibit, Monterey, CA, USA, 5–8 August 2002; p. 7094.

76. Mogford, R.H.; Bridges, W.; Gujarl, V.; Lee, P.U.; Preston, W. Generic Airspace Survey. In Proceedings of the 2013 Aviation Technology, Integration, and Operations Conference, Los Angeles, CA, USA, 12–14 August 2013.

77. Guttman, J.A.; Stein, E.; Gromelski, S. *The Influence of Generic Airspace on Air Traffic Controller Performance*; Federal Aviation Administration Technical Center: Atlantic City, NJ, USA, 1995.

78. Schier, S.; Papenfuss, A.; Lorenz, S.; Walther, J.; Moehlenbrink, C. An approach to support controller work-place design in a multi-airport environment using fast and real-time simulations. *CEAS Aeronaut. J.* 2011, 2, 185–193. [CrossRef]

© 2019 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (http://creativecommons.org/licenses/by/4.0/).