Chinese Journal of Aeronautics, (2014),27(4): 903–912

Chinese Society of Aeronautics and Astronautics
& Beihang University

Chinese Journal of Aeronautics

cja@buaa.edu.cn
www.sciencedirect.com

Dynamic airspace configuration method based on a weighted graph model

Chen Yangzhou *, Zhang Defu

School of Electronic Information and Control Engineering, Beijing University of Technology, 100124 Beijing, China

Received 25 March 2013; revised 8 June 2013; accepted 1 April 2014
Available online 5 July 2014

KEYWORDS

Airspace sector;
Air traffic control;
Dynamic airspace configuration;
Graph partitioning;
Graph theory;
Weighted graph;
Workload

Abstract This paper proposes a new method for dynamic airspace configuration based on a weighted graph model. The method begins with the construction of an undirected graph for the given airspace, where the vertices represent those key points such as airports, waypoints, and the edges represent those air routes. Those vertices are used as the sites of Voronoi diagram, which divides the airspace into units called as cells. Then, aircraft counts of both each cell and of each air-route are computed. Thus, by assigning both the vertices and the edges with those aircraft counts, a weighted graph model comes into being. Accordingly the airspace configuration problem is described as a weighted graph partitioning problem. Then, the problem is solved by a graph partitioning algorithm, which is a mixture of general weighted graph cuts algorithm, an optimal dynamic load balancing algorithm and a heuristic algorithm. After the cuts algorithm partitions the model into sub-graphs, the load balancing algorithm together with the heuristic algorithm transfers aircraft counts to balance workload among sub-graphs. Lastly, airspace configuration is completed by determining the sector boundaries. The simulation result shows that the designed sectors satisfy not only workload balancing condition, but also the constraints such as convexity, connectivity, as well as minimum distance constraint.

1. Introduction

In air traffic management, since only several controllers are impossible to put all aircraft simultaneously flying in the whole airspace of a nation under surveillance, the airspace is usually divided into smaller regions referred to as sectors, and each sector is observed by one or more controllers. In this way, the aircraft count of each sector is supposed to be not beyond the controller's ability to monitor. Current sectors are largely determined by historical effects and in an empirical way. And such situation has never changed for a long time. For instance, approximately 600 sectors over USA airspace designed in 1960 have been in use up to now. The configuration of the fixed sectors corresponds to the way that relatively few aircraft fly along the fixed air routes. The airspace characterized by fixed air routes and fixed sectors is referred to as a structured and static one.

* Corresponding author. Tel.: +86 10 67391632.
E-mail addresses: yzchen@bjut.edu.cn (Y. Chen), defuzhang@emails.bjut.edu.cn (D. Zhang).

Peer review under responsibility of Editorial Committee of CJA.

ELSEVIER Production and hosting by Elsevier

http://dx.doi.org/10.1016/j.cja.2014.06.009

With the development of air traffic, routes structure and demand profiles have changed a lot over the years. While an increasing number of aircraft fly simultaneously along fixed air routes, more and more air traffic delays may arise for some reasons, such as bad weather and traffic congestion. The situation can be improved by the way the aircraft changes its air routes,[1,2] rather than following certain fixed air routes at all time. Accordingly, sector counts and boundary vary with traffic change. This is a dynamic airspace configuration problem.

Dynamic airspace configuration (DAC)[3] is an encouraging concept proposed to convert airspace sectorization from the structured and static airspace to a dynamic one capable of accommodating dynamically changing traffic demand. A lot of research into DAC has been carried out, and most scholars completed DAC by describing the airspace as a model and then adopting a proper algorithm to partition the airspace into sectors. The airspace models proposed in literature can be summarized as follows: cell model,[4–7] flight trajectory model,[8,9] Voronoi diagram model,[10,11] and graph model.[12–16] In cell model, the airspace is first discretized into cells, i.e., hexagonal grids, and then some algorithms were used to cluster those cells into sectors. For example, Yousefi et al.[4] thought of DAC problem as a standard facility location problem to cluster the cells into sectors, Klein[5] solved the problem via seed growth algorithm, Drew[6] and Tien et al.[7] applied mixed integer programming to cluster cells into sectors. However, the designed sectors may have undesirable shapes for the boundaries which were "jagged". In virtue of flight trajectory model, Briton et al.[8] clustered the flight trajectories into sectors by k-means algorithm, and Basu et al.[9] developed geometric algorithms for DAC. The available literature told us that final sectors based on geometric algorithms still had undesired shapes. By means of Voronoi diagram, Delahaye et al.[10] proposed initial sectors arbitrarily and then optimized them by evolutionary algorithm. Furthermore, Xue[11] improved Delahaye's scenario using iterative deepening algorithm. It should be noted that a common limitation on the three above models is that they have not made use of information on airspace structure. This might lead to the case that the designed sectors might dissatisfy those geometric constraints, such as convex constraint, minimum distance constraint, and so on. At the same time, static airspace structure was taken sufficiently into account in graph model, where vertices represent airports, waypoint and crossing points while edges represent air routes. Applying graph model, Trandac et al.,[12] Martinez et al.[13] Zhang et al.[14,15] and Li et al.[16] implemented airspace sectorization using a constraint algorithm, spectral bisection algorithm, graph partitioning algorithm and spectral clustering respectively. In addition, Klein et al.[17] developed a method that divided a current sector into several dynamic Fix Posting Areas and then reallocated those Areas to achieve DAC.

Due to the graph model being embedded with information on underlying topological structure of the airspace, it usually helps to consider the factors such as air routes and key points, i.e., airports, crossing points as well as waypoints for DAC. Therefore, the graph model is preferred in this paper. Furthermore, we also consider traffic flows along air routes which are used to compute the workloads. The workloads can be assigned as the edge weights and the vertex weights. Such topological structure with traffic flows can be described as a weighted graph mathematically. Thus, the weighted graph model is adopted for DAC here. And it is different from the traditional weighted graph that only edges are assigned with weights, but an undirected graph with the weights on both vertices and edges, where traffic information as much as possible is used. This is the key feature of our graph model.

From the above literature on DAC, we know that several constraints should be taken into account when it comes to the design of sectors. The first is workload constraint. The constraint points out that the workload of each sector should be below a threshold and the workloads of those sectors are balanced, and ensures that workload of each sector does not exceed the controller's capacity to control the aircrafts while the workloads are evenly distributed among designed sectors. The second is geometric constraints consisting of convexity constraint, connectivity constraint and minimum distance constraint. The convexity indicates that an aircraft should not enter the same sector twice, the connectivity is that a sector does not be fragmented, and the minimum distance constraint means that the distance between the sector boundaries and the key points as well as the distance between the boundaries and the air routes ought not to be less than a given minimum value. The geometric constraints ensure that the controller have adequate time to control the aircraft and to solve conflicts which may happen. These constraints are critical to ensure the safety of aircrafts. Hence, the above constraints are considered thoroughly in this paper.

Moreover, from literature we can also know there are several metrics for workload, such as traffic mass, aircraft count, dynamic density, and so on. Computing workload metrics other than aircraft count might have taken more factors into accounts. However, there is no evidence that Traffic Mass and dynamic density are more effective than aircraft count for DAC. Workload metric other than aircraft count might be prohibitive in practical application. Thus, aircraft count is adopted as workload metric in this paper.

This paper applies itself to develop a DAC method based on a weighted graph model. Firstly, we set up a weighted graph model for a given airspace which accurately describes the airspace structure information and traffic data. The procedure begins with constructing an undirected graph model for the given airspace, of which the vertices represent the key points such as airports, waypoints, and the edges represent the air routes. Then, those vertices are used as the sites of Voronoi diagram[18] which divides the airspace into units called cells, and aircraft counts of both each cell and each air route are computed. By assigning both the vertices and the edges with those aircraft counts, an accessorial graph model is built up. Furthermore, in order to facilitate the discussion, the accessorial graph model is simplified into a weighted graph model whose vertices have a one-to-one relationship with Voronoi cells. Accordingly the airspace configuration problem is described as a weighted graph partitioning problem. Secondly, the paper develops a graph partitioning algorithm that divides the weighted graph model into sub-graphs. The algorithm mixes general weighted graph cuts (GWGC) algorithm,[19] an optimal dynamic load balancing (ODLB) algorithm,[20] and a heuristic algorithm inspired from K-L algorithm[21] together. After the cuts algorithm partitions graph model into sub-graphs, the load balancing algorithm together with the heuristic algorithm transfers aircraft count to achieve workload balancing among the sub-graphs. Lastly, the cells corresponding to each sub-graph are combined together into a sector. In all, the method attempts to design the sectors with the objective of balancing workload, minimizing coordination workload as well as satisfying geometric constraints.

The contribution of this paper lies in two aspects. One aspect is that the given airspace is described as a weighted graph model with weights on both vertex and edge. Our model differs from the graph model without weights on both vertex and edge in Ref.[13] and is also different from general weighted graph model with weight only on edge in Ref.[16]. Our model is loaded up with necessary information on air traffic for DAC. The other is that we develop a graph partitioning algorithm to solve DAC problem. Our algorithm takes accurate quantitative analysis as a basis to balance the workloads rather than rough estimates.

The rest of the paper is organized as follows. In Section 2, the method for DAC is described in detail. Section 3 discusses the application and gives the simulation result analysis. Section 4 concludes this paper.

2. Dynamic airspace configuration method

In this section, a method is developed for DAC on the basis of a weighted graph model. First of all, a weighted graph model will be set up to describe the given airspace according to its structure information and traffic data. The static structure information mainly includes air routes and key points such as airports, waypoints and crossing points. While the airspace is described as an undirected graph, these key points are represented as the vertices and the air routes are also represented as the edges. Aircraft count will be adopted as the workload metric, and the equalized aircraft count among the sectors means balancing workloads. Similarly, the aircraft count flying along the air route will be used to describe the coordination workload. Both vertices and edges in the undirected graph can be assigned with those aircraft counts, and an accessorial graph model forms. The procedure for the construction of the accessorial graph will be accomplished in virtue of Voronoi diagram. Precisely, the vertices is used as the sites of Voronoi diagram, and the graph is divided into a series of units called Voronoi cells using a Voronoi diagram algorithm.[18] Then, the aircraft counts are calculated both in each cell and along each edge. Finally, both aircraft counts are added onto the vertices and the edges of the graph respectively, and an accessorial graph model comes into being. To facilitate the discussion, the weighted graph is simplified into a weighted graph model whose vertices have one-to-one relationship with Voronoi cells.

Next, a graph partitioning algorithm is proposed that mixes a GWGC algorithm, an ODLB algorithm and a heuristic algorithm inspired from K-L algorithm together organically to partition the weighted graph model into a series of sub-graphs. In advance, by means of total aircraft count of the given airspace and the maximum of the aircraft count in a sector, the number of the sectors can be determined which gives how many sub-graphs need to be gotten from the weighted graph model. Then GWGC algorithm partitions the graph model into the sub-graphs. Since the aircraft counts of those sub-graphs may be not equal, ODLB algorithm combining with a heuristic algorithm inspired from the gain of K-L algorithm is applied to improve the workload balancing by transferring vertices from sub-graphs with large aircraft count to sub-graph with small aircraft count.

Lastly, Voronoi cells corresponding to each sub-graph are combined together to form the sector. Fig. 1 shows the flow chart of the above steps, and each of them is particularized as follows.

2.1. Construction of a weighted graph model

Fig. 2 gives the construction of a weighted graph model. In Fig. 2(a)–(c), thick lines represent borders of cells, and thin lines represent air routes. At the same time, thin lines represent edges of the weighted graph model in Fig. 2(d). In Fig. 2, 1,2,...,9 represents vertex index, (1),(2),...,(8) represents the vertex weight, and [1],[2],...,[8] represents the edge weight.

For a given airspace, we assume that the static structure information includes air routes and key points such as airports, waypoints and crossing points is known in advance. According to the structure information, we set up an undirected graph $G = G(V, E)$, where the vertex set $V = \{1, 2,\ldots,n\}$ consists of the key points and the edge set $E = \{(ij){:}i, j \in V\}$ represents the air routes. Fig. 2(a) shows a simple example of the construction of the undirected graph model.

For the vertices and edges of the graph being assigned with weights, a Voronoi diagram D is built, whose sites are the vertices of the undirected graph. D decomposes by its borders the airspace into a series of units called as Voronoi cells C_i ($i = 1, 2,\ldots,n$). As one can see that each cell corresponds only to one site, and the convexity of sectors will be satisfied when the cells are combined into the sectors. And also, a part of the borders will be the sector boundaries while the designed sectors come into being. Fig. 2(b) shows how the Voronoi diagram divides the airspace into the cells.

From Fig. 2(b) one can see that there may be a case that some of the sites or the air routes are close to the cell borders. This leads to a result that some of the designed sectors will not satisfy the minimum distance constraint if the sector boundaries coincide with those borders, so the borders have to be removed, and the cells that are adjacent to those borders are combined into new cells. We assume that there will be r ($r \leqslant n$) cells in the pretreated Voronoi diagram. Fig. 2(c) shows the example of the preprocessing Voronoi diagram, and it is easily seen that two cells corresponding to the sites numbered 7 and 8 are combined into a new cell.

After the decomposition of airspace into cells via the Voronoi diagram, according to the traffic data, the aircraft count flying in each cell at peak-traffic time over a period is computed. Here, we consider the aircraft count in cell C_i as w_i. Likewise, aircraft count flying long each air route is computed. The less aircraft flying across the sector boundaries means the less coordination workload for controllers. Thus, the minimum of the coordination workload is preferable and is adopted as the sectorization objective. When two aircraft counts are assigned onto both the corresponding vertices and the corresponding edges respectively, an accessorial graph model is built up. Fig. 2(c) also shows the example of the construction of an accessorial graph model.

From the accessorial graph model, it is seen that there may be several or more edges between any two cells, as well as possible several vertices in a cell, this leads to a little difficulty in further analysis. In order to facilitate the discussion, the weighted graph model is further simplified into a weighted graph model G_{w}, in which the vertex v_i represents the cell C_i, and the weight on the vertex v_i represents the aircraft count w_i in the corresponding cell C_i. Likewise, the edge e_{ij} represents

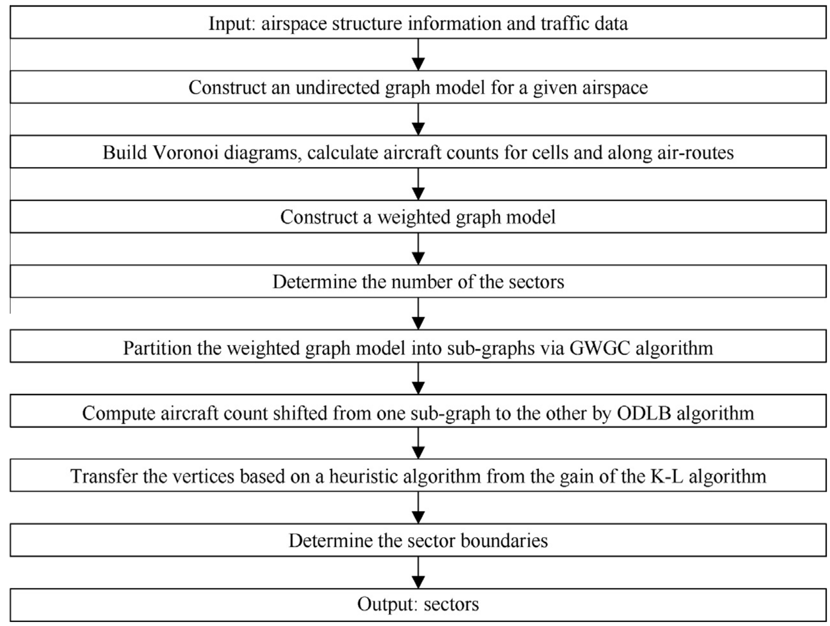


Fig. 1 Dynamic airspace configuration method.

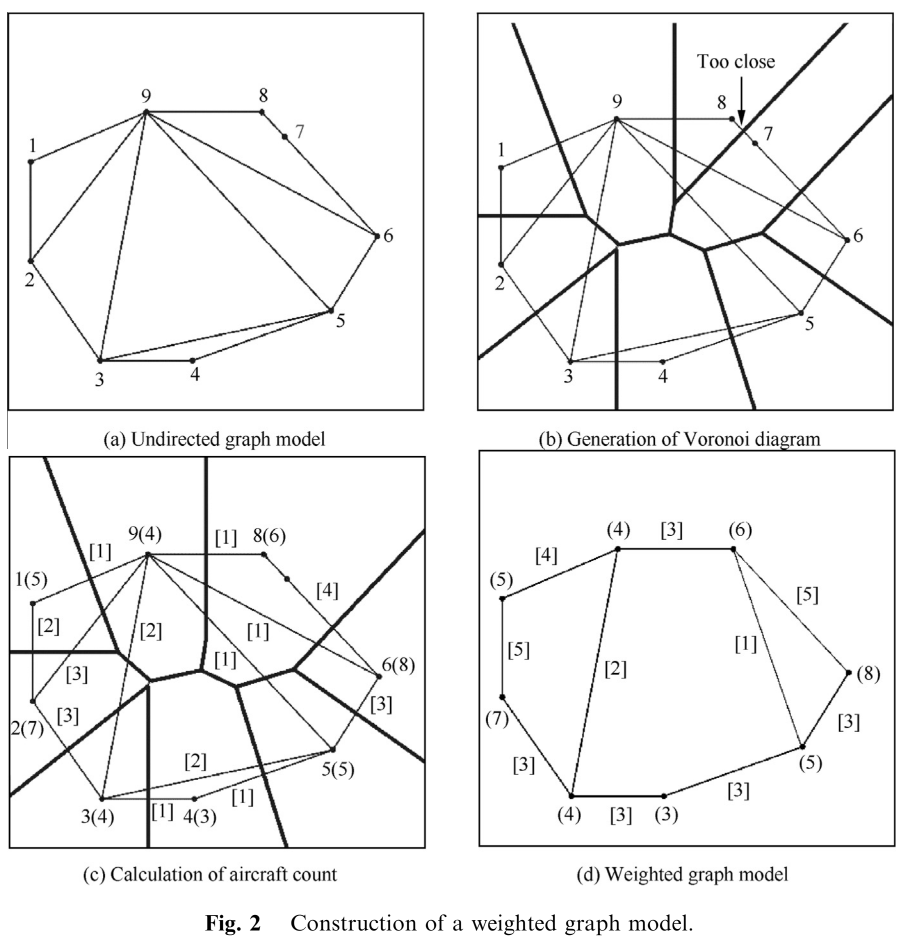


Fig. 2 Construction of a weighted graph model.

all the air routes between cells C_i and C_j, and accordingly the edge weight w_{ij} describes the sum of aircraft counts along all the air routes between the cells C_i and C_j. Furthermore, for G_{w} all aircraft counts on vertices are represented as a vector $\boldsymbol{w}$, and all aircraft counts along the edges among the cells are described by a matrix $\boldsymbol{W}$ as follows:

$$\boldsymbol{w} = [w_1, w_2, \ldots, w_r]^{\mathrm{T}} \tag{1}$$

$$\boldsymbol{W} = [w_{ij}]_{r\times r}, w_{ij} = w_{ji} \tag{2}$$

From the construction of G_{w}, one can know that there is an exact one-to-one relationship between the vertices of G_{w} and the cells. The relationship offers real convenience for the property obtained from operation on G_{w} being propagated back to the pretreated Voronoi diagram. Fig. 2(d) shows an example of the weighted graph model, accordingly the weight vector $\boldsymbol{w}$ and the weight matrix $\boldsymbol{W}$ are written as follows:

$$\boldsymbol{w} = [5, 7, 4, 3, 5, 8, 6, 4]^{\mathrm{T}}$$

$$\boldsymbol{W} = \begin{bmatrix} 0 & 5 & 0 & 0 & 0 & 0 & 0 & 4 \\ 5 & 0 & 3 & 0 & 0 & 0 & 0 & 0 \\ 0 & 3 & 0 & 3 & 0 & 0 & 0 & 2 \\ 0 & 0 & 3 & 0 & 3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 & 3 & 1 & 0 \\ 0 & 0 & 0 & 0 & 3 & 0 & 5 & 0 \\ 0 & 0 & 0 & 0 & 1 & 5 & 0 & 3 \\ 4 & 0 & 2 & 0 & 0 & 0 & 3 & 0 \end{bmatrix}$$

When the airspace is described as G_{w}, the DAC problem with the objective of balancing sector workloads and minimizing the coordination workload is converted into the graph partitioning problem of maximizing the sub-graph weight balance and minimizing the edge weight among the sub-graphs $G_{\mathrm{w}}^{i\,\prime}(i = 1, 2, \ldots, k)$. The DAC objective can be described mathematically as a graph partitioning objective by the following function:

$$\min_{G_{\mathrm{w}}^{1\,\prime}, G_{\mathrm{w}}^{2\,\prime}, \ldots, G_{\mathrm{w}}^{k\,\prime}} \sum_{c=1}^{k} \frac{\mathrm{cut}(G_{\mathrm{w}}^{i\,\prime}, G_{\mathrm{w}}/G_{\mathrm{w}}^{i\,\prime})}{w(G_{\mathrm{w}}^{i\,\prime})} \tag{3}$$

$$\text{subject to} \quad w(G_{\mathrm{w}}^{i\,\prime}) = w(G_{\mathrm{w}}^{j\,\prime}) \tag{4}$$

$$\text{where} \quad w(G_{\mathrm{w}}^{i\,\prime}) = \sum_{v_c \in G_{\mathrm{w}}^{i\,\prime}} w_c,$$

$$\mathrm{cut}(G_{\mathrm{w}}^{i\,\prime}, G_{\mathrm{w}}/G_{\mathrm{w}}^{i\,\prime}) = \sum_{v_c \in G_{\mathrm{w}}^{i\,\prime}, v_d \notin G_{\mathrm{w}}^{i\,\prime}} w_{cd} \tag{5}$$

Here, $w(G_{\mathrm{w}}^{i\,\prime})$ is the weight of ith sub-graph, k is the number of the sector that can be determined by the total aircraft count of the given airspace A_{count} and the maximum aircraft count of a sector S_{count} as follows:

$$k = \left\lceil \frac{A_{\mathrm{count}}}{S_{\mathrm{count}}} \right\rceil \tag{6}$$

where $\lceil\ \rceil$ is a symbol which denotes that a decimal can be rounded up to a integer.

Next, according to the objective function, the weighted graph model will be partitioned into a series of sub-graphs by a graph partitioning algorithm we develop.

2.2. Partition of the weighted graph model

Assume that the weighted graph model obtained in Subsection 2.1 is $G_{\mathrm{w}} = \{V_{\mathrm{w}}, E_{\mathrm{w}}, w, W\}$, where $V_{\mathrm{w}} = \{v_1, v_2, \ldots, v_n\}$ is a vertex set, $E_{\mathrm{w}} = \{e_{ij}{:}v_i,\ v_j \in V_{\mathrm{w}}\}$ is an edge set in which e_{ij} is the edge connecting v_i and v_j, $\boldsymbol{w}$ is the vector describing aircraft counts on all vertices as Eq. (1), and $\boldsymbol{W}$ is the matrix describing aircraft counts along all edges as Eq. (2). G_{w} is expected to be partitioned into k disjoint sub-graphs $G_{\mathrm{w}}^{i\,\prime}(i = 1, 2, \ldots, k)$ according to the objective function as Eqs. (3)–(4). For simpler description, the rest of the paper also refers to the aircraft counts on the vertex as the vertex weight and refers to the aircraft count along the edge as the edge weight.

Aimed at the objective function, the paper develops an algorithm to implement the graph partitioning. The algorithm consists of two steps, namely (1) partitioning the graph into sub-graphs by GWGC algorithm to achieve the objective as Eq. (3), (2) transferring the vertices with weights via ODLB algorithm together with a heuristic algorithm to achieve the objective as Eq. (4).

2.2.1. Partitioning G_w by GWGC algorithm

This subsection will apply GWGC algorithm to partition G_w into k sub-graphs $G_{\mathrm{w}}^{i}(i = 1, 2, \ldots, k)$. From the macroscopic view, this is a global approach to the problem, and the weights for the sub-graphs are roughly balanced. The algorithm can achieve an objective defined as Eq. (3):

$$J = \min_{G_{\mathrm{w}}^{1}, G_{\mathrm{w}}^{2}, \ldots, G_{\mathrm{w}}^{k}} \sum_{i=1}^{k} \frac{\mathrm{cut}(G_{\mathrm{w}}^{i}, G_{\mathrm{w}}/G_{\mathrm{w}}^{i})}{w(G_{\mathrm{w}}^{i})} \tag{7}$$

This is a general weighted graph cuts problem and a solution to the problem is designed as follows:

Step 1. Create a diagonal matrix $\boldsymbol{T}$ with diagonal entries obtained by summing all entries in the corresponding column of the matrix $\boldsymbol{W}$, and compute the Laplacian matrix $\boldsymbol{L} = \boldsymbol{T} - \boldsymbol{W}$.

Step 2. $s_i = \dfrac{1}{\sqrt{w_i}} (i = 1, 2, \ldots, r)$, $\boldsymbol{S} = \mathrm{diag}(s_1, s_2, \ldots, s_r)$, and calculate $\boldsymbol{C} = \boldsymbol{S} \times \boldsymbol{L} \times \boldsymbol{S}$.

Step 3. Calculate the eigenvalues of $\boldsymbol{C}$, assume that they are $\lambda_1 = 0 \leqslant \lambda_2 \leqslant, \ldots, \leqslant \lambda_r$ in ascending sort. Then, use k-means algorithm to cluster k vectors corresponding to eigenvalues from λ_1 to λ_k. From k clusters, we can get a series of $G_{\mathrm{w}}^{i}(i = 1, 2, \ldots, k)$ that holds connectivity.

When G_{w} is partitioned into a series of sub-graphs by GWGC algorithm, the sub-graphs meet the property which follows the objective as Eq. (7). Now, we analyze the property qualitatively from two aspects. (1) The smaller $\sum_{i=1}^{k} \mathrm{cut}(G_{\mathrm{w}}^{i}, G_{\mathrm{w}}/G_{\mathrm{w}}^{i})$, the smaller J. (2) With given $\sum_{i=1}^{k} \mathrm{cut}(G_{\mathrm{w}}^{i}, G_{\mathrm{w}}/G_{\mathrm{w}}^{i})$, J is minimized while balancing the weight for sub-graphs $w(G_{\mathrm{w}}^{i})(i = 1, 2, \ldots, k)$. In fact, GWGC algorithm proposes a scheme for partitioning G_{w} that takes into account both balancing $w(G_{\mathrm{w}}^{i})$ and minimizing $\sum_{i=1}^{k} \mathrm{cut}(G_{\mathrm{w}}^{i}, G_{\mathrm{w}}/G_{\mathrm{w}}^{i})$. In other words, J is achieved by the common contribution of balancing $w(G_{\mathrm{w}}^{i})$ and minimizing $\sum_{i=1}^{k} \mathrm{cut}(G_{\mathrm{w}}^{i}, G_{\mathrm{w}}/G_{\mathrm{w}}^{i})$. Thus, the sub-graph weights may not be equal under the conditions of minimizing J, i.e. $w(G_{\mathrm{w}}^{i}) \neq w(G_{\mathrm{w}}^{j})$. Such situation requires further measures to balance $w(G_{\mathrm{w}}^{i})(i = 1, 2, \ldots, k)$. That is to say, it is necessary to transfer the vertices with the weights among the sub-graphs to achieve the objective as Eq. (4).

2.2.2. Calculating the vertex weights to be transferred by ODLB algorithm

Multiple algorithms have come forward with the ideas to discuss the shift of the vertex weights for the sake of the objective as Eq. (4). Here, we will introduce an ODLB algorithm to deal with such balance problem. The ODLB algorithm gives the direction in which the vertex is moved as well as the amount of the vertex weight to be transferred. Next, let us demonstrate an example to illustrate the algorithm.

Provided that by means of GWGC algorithm G_{w} is partitioned into four sub-graphs, G^1_{w}, G^2_{w}, G^3_{w}, G^4_{w}, and accordingly the weights for four sub-graphs are shown, see Fig. 3, the dotted lines represent the sub-graph boundaries.

In order to achieve $w(G^i_{\mathrm{w}}) = w(G^j_{\mathrm{w}})$, it is necessary to transfer a part of the weights denoted as x_{ij} from G^i_{w} to G^j_{w}. In Fig. 3, two schemes are presented, Obviously, the scheme in Fig. 3(b) is better than the other in Fig. 3(a) in terms of the amount of the weight to be transferred. The method always intends to transfer as less weight as possible among the sub-graphs. The ODLB algorithm gives an optimal solution. Then, let us discuss the algorithm.

For four sub-graphs, the average weights per sub-graph should be $\overline{w} = \dfrac{w(G^1_{\mathrm{w}}) + w(G^2_{\mathrm{w}}) + w(G^3_{\mathrm{w}}) + w(G^4_{\mathrm{w}})}{4}$. The following equations hold true:

$$\begin{cases} x_{12} + x_{13} + x_{14} = w(G^1_{\mathrm{w}}) - \overline{w} \\ -x_{12} + x_{23} = w(G^2_{\mathrm{w}}) - \overline{w} \\ -x_{13} - x_{23} + x_{34} = w(G^3_{\mathrm{w}}) - \overline{w} \\ -x_{14} - x_{34} = w(G^4_{\mathrm{w}}) - \overline{w} \end{cases}$$

Furthermore, the equation can be described as

$$\boldsymbol{Fx} = \boldsymbol{b}$$

where $\boldsymbol{b} = [w(G^1_{\mathrm{w}}) - \overline{w}, w(G^2_{\mathrm{w}}) - \overline{w}, w(G^3_{\mathrm{w}}) - \overline{w}, w(G^4_{\mathrm{w}}) - \overline{w}]$,

$$\boldsymbol{x} = [x_{12}, x_{13}, x_{14}, x_{23}, x_{34}]^{\mathrm{T}},$$

$$\boldsymbol{F} = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ -1 & 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & -1 & 1 \\ 0 & 0 & -1 & 0 & -1 \end{bmatrix}$$

We suppose that the weight will be shifted along the direction from the sub-graph with small index to the sub-graph with large index. The former is called the head of the direction and the latter is called the tail. So, $\boldsymbol{F}$ is defined as

$$F_{ij} = \begin{cases} 1 & \text{if sub} - \text{graph } i \text{ is the head of the direction of shift} \\ -1 & \text{if sub} - \text{graph } j \text{ is the tail of the direction of shift} \\ 0 & \text{Otherwise} \end{cases}$$

This is a system of linear inhomogeneous equations to be solved, where there are five variables and four equations. The knowledge of linear algebra verifies that there are infinite solutions to the equations. Among solutions, the solution for the migration of less weight is preferred. Here, the Euclidean norm of the data movement is used as a metric which minimize x_{ij}, and the norm is expressed by $\frac{1}{2}\boldsymbol{x}^T\boldsymbol{x}$. Thus, the following problem need to be solved

$$\text{Minimize } \frac{1}{2}\boldsymbol{x}^T\boldsymbol{x} \tag{8}$$

$$\text{subject to} \quad \boldsymbol{Fx} = \boldsymbol{b} \tag{9}$$

This is typically a minimization problem, and is easily extended to more general form. It can be solved via the following procedure:

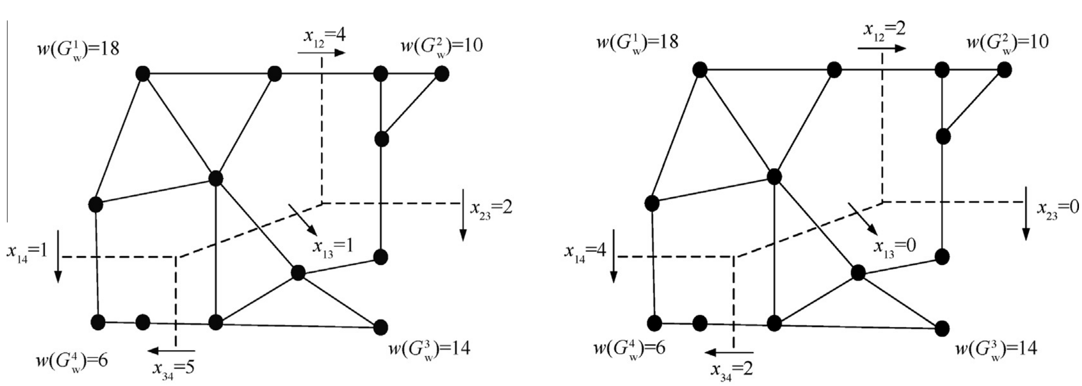
Step 1. Calculate $\bar{w} = \frac{1}{k}\sum_{i=1}^{k} w(G^i_{\mathrm{w}})$, and $\boldsymbol{b} = [w(G^1_{\mathrm{w}}) - \bar{w}, w(G^2_{\mathrm{w}}) - \bar{w}, \ldots, w(G^k_{\mathrm{w}}) - \bar{w}]^{\mathrm{T}}$.

Step 2. Construct the matrix of sub-graphs $\boldsymbol{F}$, and calculate matrix $\boldsymbol{L} = \boldsymbol{FF}^{\mathrm{T}}$.

Step 3. Construct a linear equation system $\boldsymbol{Ld} = \boldsymbol{b}$ and solve it for d. One can conclude that from the sub-graph i to j is obtained by $\boldsymbol{Fd}^{\mathrm{T}}$.

From the ODLB algorithm, we know a fact that x_{ij} may be a negative value or positive value. The positive value means the weights from G^i_{w} to G^j_{w} while a negative value means the weights from G^j_{w} to G^i_{w}. In addition, x_{ij} may not be the integers, but decimals. However, x_{ij} is the aircraft count, this means x_{ij} is integer. So the measure must be taken to deal with the problem by rounding x_{ij} to be a whole number $[x_{ij}]$. Here, [ ] is the symbol denoting that a decimal is rounded to an integer.

In what follows, we are going to discuss how to minimize $\sum_{i=1}^{k} \mathrm{cut}(G^i_{\mathrm{w}}, G_{\mathrm{w}}/G^i_{\mathrm{w}})$ when transferring the vertices to balance the weighs of sub-graphs.



(a) A scheme of transferring more weights (b) A scheme of transferring less weights

Fig. 3 Two schemes for transferring the weights among the sub-graphs.

2.2.3. Transferring the vertices using a heuristic algorithm

The preceding GWGC algorithm partitions G_w into a series of sub-graphs, and the ODLB algorithm gives $[x_{ij}]$ between two sub-graphs. If $[x_{ij}]$ can be satisfied by transferring the vertices optionally, it is impossible to minimize $\sum_{i=1}^{k}\mathrm{cut}(G_w^i, G_w/G_w^i)$ in original scheme. For example, in Fig. 4, there are two sub-graphs, and one sub-graph is separate from the other by a dotted line. Since the sub-graph weights are not balanced, we intend to transfer five weights from the left sub-graph to the right. When both weights of v_1 and v_2 are 5, which vertex should be transferred, v_1 or v_2?

When v_1 is moved to the right sub-graph, the edge weight connecting two sub-graphs will decrease. The change in the edge weight can be gotten from the following equation:

$$8-(1+2+1)=4$$

This means that coordination workload will be reduced. Similarly, when v_2 is moved, the edge weight does not change any more. Compared to v_2, v_1 is preferable to moving to the right sub-graph. The change in the edge weight for the vertex migration is called the gain which comes from K-L algorithm. The equation is generalized to calculate the gain g_d for the vertex v_d:

$$g_d = \sum_{v_d \in G_w^i, v_f \in G_w^j} w_{df} - \sum_{v_d, v_e \in G_w^i} w_{de} \tag{10}$$

Certainly, it is also true that the gain of a vertex being a negative value means an increase in coordination workload when the vertex is transferred.

Here, on the basis of the concept of the gain a heuristic algorithm is proposed to ensure the minimum of $\sum_{i=1}^{k}\mathrm{cut}(G_w^i, G_w/G_w^i)$ while transferring the vertices from one sub-graph to the other.

Firstly, according to $[x_{ij}]$, we can determine the migration direction of the vertices. Let the set of vertices in G_w^i adjacent to G_w^j be denoted as B_{ij}, the sum of the weights corresponding to the vertices in B_{ij} be a_{ij} and the gain g_d of v_d in B_{ij} be determined by Eq. (10).

And the vertices in B_{ij} are sorted according to their gains by a descending order. The heuristic algorithm is described as follows. The vertex in B_{ij} with the largest gains is transferred to G_w^j, and the procedure is repeated according to the descending order. If $a_{ij} < [x_{ij}]$, after transferring all vertices in B_{ij}, the procedure above can continue until the required $[x_{ij}]$ has been satisfied for new vertices in G_w^i adjacent to G_w^j will appear after migrating all vertices in B_{ij}. In this way, we can get a series of new sub-graphs $G_w^{i\prime}(i=1,2,\ldots,k)$ from G_w^i with equalized $w(G_w^{i\prime})$.

The new sub-graphs $G_w^{i\prime}(i=1,2,\ldots,k)$ satisfy the properties as follows:

$$w(G_w^{i\prime}) = w(G_w^{j\prime}), G_w = \bigcup_{i=1}^{k} G_w^{i\prime}, G_w^{i\prime} \cap G_w^{j\prime} = \varnothing \tag{11}$$

Certainly, it is obvious that absolute balanced $w(G_w^{i\prime})$ for sub-graphs is always impossible due to the non-unitary aircraft counts corresponding to the cells. Let $L_{\max}$ and $L_{\min}$ denote the defined maximum and the given minimum of aircraft count for all sub-graphs $G_w^{i\prime}(i=1,2,\ldots,k)$, and the balanced $w(G_w^{i\prime})$ can be given by

$$L_{\min} \leqslant w(G_w^{i\prime}) \leqslant L_{\max} \tag{12}$$

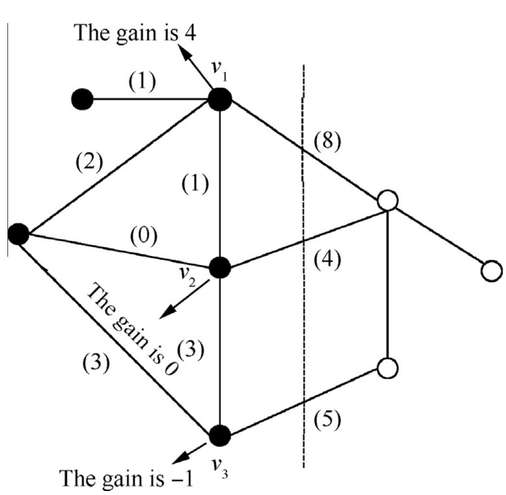


Fig. 4 Computing the gains of the vertices.

2.3. Determination of the sectors

From the construction of the weighted graph G_w, we know there is an exact one-to-one relationship between its vertices and the corresponding Voronoi cells, so the vertices in each new sub-graph among $G_w^{1\prime}, G_w^{2\prime}, \ldots, G_w^{k\prime}$ can be mapped back to the Voronoi cells. Therefore, we combine those cells with respect to each sub-graph $G_w^{i\prime}$ together to form a sector S_i, and the property described as Eqs. (3)–(4) is propagated back to the sectors. Finally, we obtain k sectors satisfying the workload constrain.

So far, we take the following measure to keep the constraints held. Firstly, the decomposition of airspace by the Voronoi diagram keeps the sectors the convexity constraint. Secondly, the graph partitioning algorithm we develop is applied to ensuring the workload constraint and the connectivity constraint. Thirdly, the removal of the borders of the cells being close to the key points ensures the minimum distance constraints. Therefore, the designed sectors in this paper satisfy the preceding constraints.

3. Experiment and simulation

In this section, our DAC method is validated with real air traffic data. Beijing air traffic area (BJA) is used for simulation, for BJA is one of the three Chinese busiest areas in air traffic and so it is representative, see Fig. 5(a). Here, we are aimed at the airspace above 18000 feet (1 feet = 0.3048 m) altitude. From the figure, we know that there are seven sectors for current air traffic management.

Some parameters are set as follows. (1) The aircraft count of each sector is set to 10, especially the redundancy of 20% is adopted for the sake of the reliability of DAC and the safety of aircraft. (2) The minimum distance between the airports and the sector boundaries, between the waypoints and the boundaries, between the waypoints and the boundaries is set to 15 nm, 9 nm and 3 nm respectively. We take the air traffic of every two hours into account.

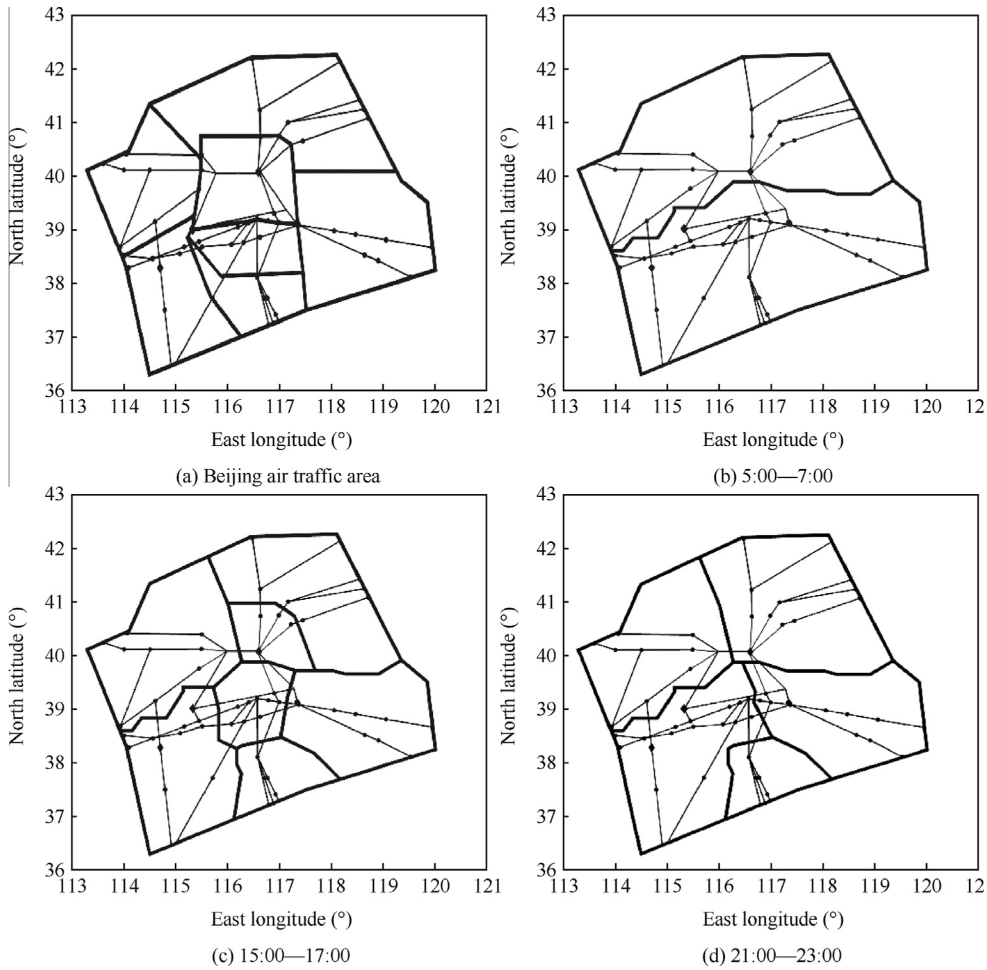


Fig. 5 Beijing air traffic area and new sectors of BJA via our method for three different time intervals.

Especially, the peak traffic of every stage is applied for DAC, for it takes into accounts the controller's maximum ability to monitor and provide traffic flow control within the confines of each sector.

Fig. 5 gives different sectorization of BJA computed by our DAC method for three different time intervals. In Fig. 5, thick lines represent the sector boundaries, and thin lines represent the air routes; the horizontal axis represents the east longitude, and the vertical axis represents the north latitude. It is seen that (1) all sectors satisfy the geometric constraints, (2) the number of the sectors computed by our method varies over time. Such variability can be reasonably explained, for the air traffic varies over time, and there are few sectors with low air traffic while there are more sectors with high traffic. This shows that our DAC method offers a much greater degree of flexibility in the airspace sectorization when the air traffic varies.

Next, we will analyze the performance of new sectors of BJA computed by the method and compare new sectors with current sectors.

3.1. Average performance

Fig. 6 describes the sector aircraft count for current sectors and new sectors. In Fig. 6, squares represent new sectors, and circles represent current sectors. For the average performance, there are three performance indicators, namely the mean of the average aircraft count of the sector, the standard deviation of the average aircraft count and the coefficient of aircraft count balancing c_{bal}. They can be calculated from Fig. 6 which describes, and the result are summarized in Table 1, where c_{bal} is defined as

$$c_{\mathrm{bal}} = (L_{\mathrm{smax}} - L_{\mathrm{smin}})/L_{\mathrm{smax}} \times 100\% \tag{13}$$

where L_{smax} and L_{smin} are the maximum and minimum aircraft count of the designed sector.

The standard deviation (Stdev) indicates the degree of the aircraft count deviating from the mean aircraft count while c_{bal} gives the minimum difference. So the less the standard deviation and c_{bal}, the more balanced the aircraft count. From Table 1, we know that the standard deviation and c_{bal} of new sectors are smaller than those of current sectors. This shows that new sectors by our method have more balanced aircraft count than current sectors.

It should be noted that the designed sectors being balanced by means of the peak of the air traffic do not mean that the aircraft counts are evenly distributed among the sectors at any time over two hours. However, this can ensure at no time is the aircraft count beyond the maximum of aircraft count of the sector.

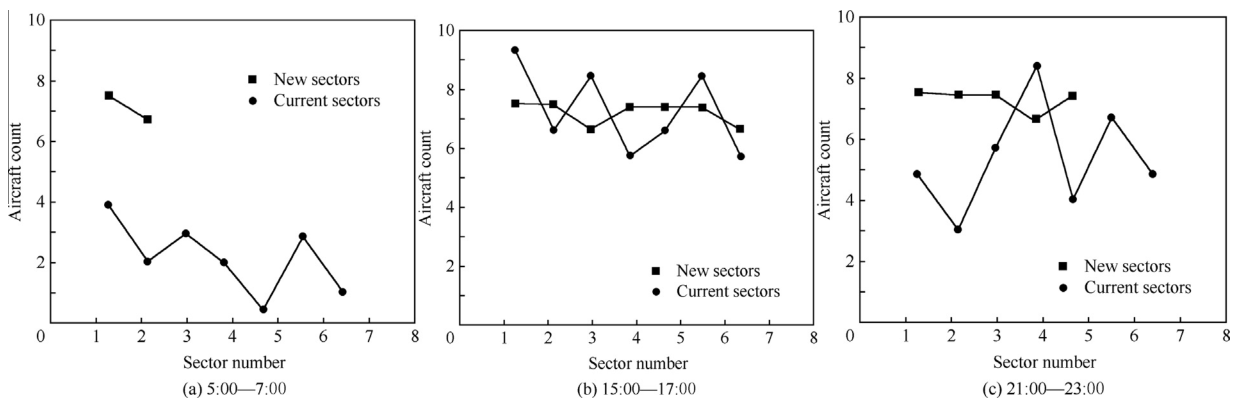


Fig. 6 Aircraft count for both new sectors and current sectors during different time intervals.

Table 1 Average performance.

Sectors type	Time	Mean	Stdev	c_{bal} (%)
Current sectors	5:00–7:00	2.1	1.34	100
	15:00–17:00	7.7	1.40	40
	21:00–23:00	5.57	1.99	66
New sectors	5:00–7:00	7.5	0.7	12.5
	15:00–17:00	7.7	0.49	12.5
	21:00–23:00	7.8	0.45	12.5

3.2. Number of the sectors

From Table 2, the new sectorization has fewer sectors than current sectors. The degree of the reduction can be measured by p_{save} defined as

$$p_{save} = (N_c - N_n)/N_c \times 100\% \tag{14}$$

where N_c is the number of current sectors and N_n is the number of new sectors.

From Table 2, we know that p_{save} is positive and new sectors are less than current sectors. The consequent result is a promising reduction in the number of controllers which cut down the administrative expenses.

3.3. Coordination workload

Similar to the discussion in Section 3.2, the coordination workload of new sectors is less than those of current sectors, and the degree of the reduction can be measured by cw_{red} defined as

$$\mathrm{cw}_{red} = (\mathrm{cw}_c - \mathrm{cw}_n)/\mathrm{cw}_c \times 100\% \tag{15}$$

where cw_c is the coordination workload of current sectors and cw_n is the coordination workload of new sectors. From Table 3, the coordination workload of new sectors is less than those of current sectors, and less coordination workload means less pressure for controller to control aircrafts.

Table 2 Number of the sectors.

Time	Current sector count	New sector count	p_{save} (%)
5:00–7:00	7	2	71.4
15:00–17:00	7	7	0.0
21:00–23:00	7	5	28.5

Table 3 Coordination workload.

Time	Coordination workload of current sector	Coordination workload of new sector	cw_{red} (%)
5:00–7:00	13	2	84.6
15:00–17:00	109	54	50
21:00–23:00	67	15	77.6

4. Conclusions

The paper has presented a new method for DAC based on a weighted graph model by applying GWGC algorithm and ODLB algorithm in combination with a heuristic algorithm inspired from the gain of K-L algorithm to partition given airspace into sectors achieving the objective of balancing the workloads and of minimizing the coordination workloads among the designed sectors. Simulation indicates that:

(1) The designed sectors have balanced aircraft count while coordination workload is minimized.
(2) Simulation result shows that the designed sectors satisfy geometrical constraints, such as convexity constraint, connectivity constraint and minimum distance constraint.
(3) And more importantly, the low traffic results in fewer sectors than the current airspace configuration, and the consequence is promising reduction in the number of controllers, and thereby the administrative expense is cut down.

The performance of simulation validates the feasibility and effectiveness of the method.

Acknowledgement

The authors are grateful to the anonymous reviewers for their critical and constructive review of the manuscript. This study was supported by the National Natural Science Foundation of China (No. 61079001).

References

1. Wang YY, Wei TT, Qu XJ. Study of multi-objective fuzzy optimization for path planning. *Chin J Aeronaut* 2012;**25**(1):51–6.
2. Chen BH, Jiao ZX, Ge SZS. Aircraft-on-ground path following control by dynamical adaptive backstepping. *Chin J Aeronaut* 2013;**26**(3):668–75.
3. Kopardekar P, Bilimoria K, Sridhar B. Initial concepts for dynamic airspace configuration. *Proceedings of the 7th AIAA aviation technology, integration, and operations conference*; 2007 Sep 18–20; Belfast, Northern Ireland; 2007. p. 7763–74.
4. Yousefi A, Donohue G. Temporal and spatial distribution of airspace complexity for air traffic controller workload-based sectorization. *Proceedings of the 4th AIAA aviation technology, integration, and operations conference*; 2004 Sep 20–24; Chicago, 2004. p. 6455–68.
5. Klein A. An efficient method for airspace analysis and partitioning based on equalized traffic mass. *Proceedings of the 6th USA/europe seminar on air traffic management research and development*; 2005 Jun 27–30; Baltimore Maryland, USA; 2005. p. 1–10.
6. Drew M. Analysis of an optimal sector design method. *Proceeding of the 27th digital avionics systems conference*; 2008 Oct 26–30; Saint Paul, USA; 2008. p. 3.B.4-1–3.B.4-10.
7. Tien SL, Hoffman R. Optimizing airspace sectors for varying demand patterns using multi-controller staffing. *Proceeding of the 8th USA/europe air traffic management research and development seminar*; 2009 June-July; Napa, California, USA; 2009.p. 1–10.
8. Brinton C, Pledgie S. Airspace partitioning using flight clustering and computational geometry. *Proceeding of the 27th digital avionics systems conference*; 2008 Oct 26–30; St Paul, Minnesota, USA; 2008. p. 3.B.3-1–3.B.3-10.
9. Basu A, Mitchell J, Sabhnani G. Geometric algorithms for optimal airspace design and air traffic controller workload balancing. *Proceeding of the 16th fall workshop on computational and combinatorial geometry*; 2008 Jan 19; San Francisco, CA, United states; 2008. p. 75–89.
10. Delahaye D, Schoenauer M, Alliot JM. Airspace sectoring by evolutionary computation. *Proceedings of the IEEE international congress on evolutionary computation*; 1998; Anchorage, USA; 1998. p. 218–23.
11. Xue M. Airspace sector redesign based on Voronoi diagrams. *J Aerosp Comput Info Commun* 2009;**6**:624–34.
12. Trandac H, Duong V. Optimized sectorization of airspace with constraints. *Proceedings of the 5th eurocontrol/FAA ATM R&D seminar*; 2003 June; Budapest, Hungary; 2003. p. 1–11.
13. Martinez S, Chatterji G, Sun D, Bayen A. A weighted-graph approach for dynamic airspace configuration. *Proceedings of the AIAA guidance, navigation, and control conference*; 2007 Aug 20–23; Hilton Head, South Carolina, USA; 2007. p. 6448–63.
14. Zhang DF, Chen YZ, Bi H, Song ZX. Airspace sectorization based on weighted graph spectral bisection algorithm. *Proceeding of the12th COTA international conference of transportation professionals*; 2012 Aug 3–6; Beijing, China; 2012. p. 1924–35.
15. Zhang DF, Chen YZ. Airspace sectorization via a weighted graph model. *Aeronaut J* 2014;**118**(1201):267–74.
16. Li JH, Wang T, Savai M, Hwang I. Graph-based algorithm for dynamic airspace configuration. *J Guid Control Dynam* 2010;**33**(1):1082–95.
17. Klein A, Rodfers M, Kaing H. Dynamic FPAs: a new method for dynamic airspace configuration. *Proceedings of integrated communications, navigation, and surveillance conference*; 2008 May 5–8; Bethesda, Maryland, USA; 2008. p. 1–11.
18. Fortune SJ. A sweep line algorithm for Voronoi diagrams. *Algorithmica* 1987;**2**(1–4):153–74.
19. Dhillon IS, Guan YQ. Weighted graph cuts without eigenvectors: a multilevel approach. *IEEE Trans Pattern Anal Mach Intell* 2007;**29**(11):1–14.
20. Hu YF, Blake RJ. An improved diffusion algorithm for dynamic load balancing. *J Parallel Comput* 1999;**25**(4):417–44.
21. Hendrickson B, Leland R. Multilevel algorithm for partitioning graph. *Proceedings of the 1995 ACM/IEEE conference on supercomputing*; New York, USA; 1995. p. 626–57.

Chen Yangzhou is a professor at the College of Electronic Information and Control Engineering, Beijing University of Technology, China. He received his Ph.D. degrees from St. Petersburg University, St. Petersburg, Russia, in 1994 (under the supervision of V.A. Yakubovich). His research interest covers hybrid and network dynamic systems, cooperative control of multiple vehicles/robots, and information processing and control in intelligent transportation systems (ITS).

Zhang Defu is a Ph.D. student at School of Electronic Information and Control Engineering, Beijing University of Technology. He received his B.S. degree from University of Jinan in 2010. His area of research includes hybrid and network dynamic systems, air traffic management and intelligent transportation system.