# 17. 空域編成に対する２つの最適化アプローチ

筑波大学　猿渡康文，伊豆永洋一，防衛大学校　鵜飼孝盛
航空交通管理領域　蔭山康太

## 1　研究の目的

航空需要の増加により航空管制の現行運用は，近い将来に限界を迎えると予想される。これに対して，国土交通省では空域の抜本的再編により業務負荷の低減などを図ることで管制処理能力の向上を目指している。陸域（レーダ空域）への利用者選択経路（User Preferred Route : UPR）の導入は，飛行の効率性の向上などをもたらす一方で航空管制官の作業負担の増加が懸念される。航空管制を取り巻く，このような環境へ対応するためには UPR 導入を考慮した我が国の陸域の空域編成の実現が必要不可欠である。空域編成においては運用上の制約を考慮しながら，交通の効率性や円滑性を最大にすることが求められる。

空域はセクタと呼ばれる単位に分割されて運用される。各セクタには効率性や円滑性を考慮して適正な管制処理容量（容量）が設定され，その容量を超過しないように交通量が調整される。現在，各セクタの境界線は原則として固定されているが，経路構成が変動する UPR の導入時には各経路構成に応じた柔軟なセクタ境界線の設定が必要とされる。空域編成は，この境界線の設定に相当する。

本論文では，我が国の航空管制における環境の変化に対応した航空路管制空域を編成する 2 つの最適化アプローチを提案し，得られる航空路管制空域の評価を行う。

## 2　既往の研究

空域編成問題は，所与の複数の条件を満たすセクタの集合を求める問題として定義される。いったんセクタを定めると，そのセクタを担当する航空管制官の作業量（管制作業量）が定まる。このため，一般には，管制作業量のセクタ間での均等化や低減に対する考慮が必要とされる。

空域編成問題は，主として，欧米の研究者によって研究が進められている。この問題に対するサーベイ論文としては，文献 [2] がある。[2] にまとめられているように，先行研究は，取り上げる制約条件や目的関数が，研究ごとに異なることが多く，統一的なモデルに基づいた定式化が与えられているわけではない。しかしながら，先行研究で取り上げられたモデルの多くは，NP 困難であることが示されている。

既往の研究では，セクタの集合を求めるためのさまざまな制約条件が検討されている。たとえば，管制作業量の均等化 [3]，管制作業量の上限 [4]，セクタの大きさの上限 [3]，ギザギザ状のセクタ境界線の禁止 [4] などが挙げられる。本研究では以下の 4 つの制約条件を取り上げる。

- 最小滞在時間:航空機の安全を確保するための管制指示が実現できる時間を確保すること [5]。
- 最小距離：セクタ境界線と飛行方向（針路）を変更できるウェイポイントとの間には管制指示が実現できるだけの距離を確保すること [5]。
- セクタの凸性：同一の航空機が 2 度以上同一のセクタに進入しないようなセクタの形状を確保すること [4]。
- 連結性：あるセクタに進入する飛行経路が複数存在し，かつ，それらの経路がある特定のウェイポイントで交差することなく，当該セクタから離脱するようなセクタの形状でないことを確保すること。

## 3　空域分割に対するグラフ分割アプローチ

### 3.1　空域分割モデル

各飛行は出発前に作成される計画経路に基づいて出発空港から目的空港に到達するまでに複数のセクタを通過し，セクタごとに配置された航空管制官が安全で円滑な飛行を実現するように航空管制を行う。

本論文では，[2] と同様に計画経路の集合を重

み付きグラフとしてモデル化する。計画経路は航空無線施設などの特定の地点（ウェイポイント）の列で表現することができる。ウェイポイントを頂点，計画経路において隣接するウェイポイント間に枝を与える。このグラフをもとに，例えば各枝に対して通過する飛行数を与えたり，各頂点に飛行が近接する頻度を与えることで作業量を加味することができる。以降では，グラフを複数の連結部分グラフに分割する 2 つの手法を提案する。

3.2 では，全てのセクタにおける作業量の総和の最小化を目的として列生成法と呼ばれる手法を適用する。3.3 では，セクタ間の作業量の平準化を目的として，Normalized-cut [6] と呼ばれる評価関数を最小化することで，空域編成を実現する方法を考える。提案する手法は部分グラフへの分割であるため，各セクタについて大まかな領域は決定されるが境界線は取得されない。そこでセクタ境界線の算出手法を 3.4 で提案する。

## 3.2　セクタ作業量和の最小化問題と列生成法アプローチ

頂点集合 $V$ と有向枝の集合 $E$ からなる有向グラフを $G = (V, E)$ とする。また，$E$ 上で定義された非負重み（作業量）関数を $w : E \to \mathbb{R}_+$，$E$ 上の距離関数を $d : E \to \mathbb{R}_+$ とする。グラフ $G$ の部分グラフ $T \subset G$ が連結かつ閉路を持たないとき，$T$ を（有向）木と呼ぶ。木 $T$ を構成する頂点集合と枝集合を，それぞれ $V(T)$ と $E(T)$ で表記する。また，木 $T$ の重み $c_T$ を

$$c_T = (1-\theta) \sum_{\substack{(i,j)\in E;\\ i\in V\setminus V(T),\, j\in V(T)}} w_{ij} + \theta \sum_{(i,j)\in E(T)} w_{ij} d_{ij} \tag{3.1}$$

と定義する。(3.1) の第 1 項はセクタ間（管制移管）の作業量，第 2 項はセクタ内の作業量を表している。ただし $\theta$ は $[0, 1]$ の値をとるパラメータであり，セクタ間とセクタ内の作業量の重要度を表している。

木の頂点集合 $V(T)$ を用いて，$G$ の頂点集合 $V$ の最小重みの分割を求める問題として空域分割問題を捉える。

### 3.2.1　定式化

任意の木 $T \in \mathcal{T}$ に対して，決定変数 $x_T$ を

$$x_T = \begin{cases} 1 & (T \in \mathcal{T}^*) \\ 0 & (T \notin \mathcal{T}^*) \end{cases}$$

と定義する。空域分割問題は以下の最適化問題として定式化できる。

$$\mathrm{P}: \left| \begin{array}{ll} \text{min.} & \sum_{T\in\mathcal{T}} c_T x_T \\ \text{s.t.} & \sum_{T\in\mathcal{T}} a_{iT} x_T = 1, \quad \forall i \in V, \\ & x_T \in \{0,1\}, \quad \forall T \in \mathcal{T}. \end{array} \right.$$

ただし，$a_{iT}$ は頂点 $i$ が木 $T$ の頂点集合 $V(T)$ に含まれるならば 1，そうでなければ 0 となる定数である。このとき，ベクトル $\boldsymbol{a}_T = (a_{iT} \mid i \in V)^\top$ は，木 $T$ の頂点集合 $V(T)$ の特性ベクトルである。

一般に集合分割問題は強 NP 困難である。問題 P の線形計画緩和問題（LP 緩和）を解くことで，問題 P の最適解に関する有用な情報を得ることを考える。

$$\mathrm{LP}: \left| \begin{array}{ll} \text{min.} & \sum_{T\in\mathcal{T}} c_T x_T \\ \text{s.t.} & \sum_{T\in\mathcal{T}} a_{iT} x_T = 1, \quad \forall i \in V, \\ & x_T \geq 0, \quad \forall T \in \mathcal{T}. \end{array} \right.$$

変数 $x_T$ の上限制約は冗長である。上述の問題とその双対問題は，$G$ のサイズが大きくなるにつれて，集合族 $\mathcal{T}$ の要素数が指数的に増加するため，効率良く解くための工夫が必要である。

### 3.2.2　列生成法のフレームワーク

莫大な個数の変数や制約をもつ問題に対しては，列生成法や切除平面法と呼ばれる手法が有効である [7]。列生成法では，一部の集合族 $\mathcal{S} \subseteq \mathcal{T}$ 上で定義される以下の限定主問題 LP($\mathcal{S}$) を考える。

$$\mathrm{LP}(\mathcal{S}): \left| \begin{array}{ll} \text{min.} & \sum_{T\in\mathcal{S}} c_T x_T \\ \text{s.t.} & \sum_{T\in\mathcal{S}} a_{iT} x_T = 1, \quad \forall i \in V, \\ & x_T \geq 0, \quad \forall T \in \mathcal{S}. \end{array} \right.$$

LP($\mathcal{S}$) の最適解は元の線形計画問題 LP の最適解とは限らない。必要に応じて列を逐次生成しながら，LP の最適解を求める。

問題 LP($\mathcal{S}$) の最適解を $\boldsymbol{x}^*_{(\mathcal{S})} = (x^*_T \mid T \in \mathcal{S})^\top$，対応する双対最適解を $\boldsymbol{\lambda}^*_{(\mathcal{S})} = (\lambda^*_i \mid i \in V)^\top$ とすると，(3.2) で定義される既約費

用 $\gamma(T, \boldsymbol{\lambda}^*_{(\mathcal{S})})$ が任意の $T \in \mathcal{T} \setminus \mathcal{S}$ に対して非負であれば，$\boldsymbol{x}^*_{(\mathcal{S})}$ は問題 LP の最適解となる。

$$\gamma(T, \boldsymbol{\lambda}^*_{(\mathcal{S})}) = c_T - \sum_{i \in V} a_{iT} \lambda^*_i. \tag{3.2}$$

一方で，既約費用が負になる木 $T \in \mathcal{T} \setminus \mathcal{S}$ が存在するならば，その木を集合族 $\mathcal{S}$ に追加することで目的関数値を改善できる可能性がある。このような木が存在するかどうかの判定に関しては，

$$\min \left\{ \gamma(T, \boldsymbol{\lambda}^*_{(\mathcal{S})}) \mid T \in \mathcal{T} \setminus \mathcal{S} \right\} \tag{3.3}$$

を解くことで実現できる。

(3.3) を列生成子問題として定式化することを考える。まず，元のグラフ $G$ にダミー頂点 0 と，その頂点を始点とし各頂点 $i$ を終点とする枝を加えた拡張グラフを $\bar{G} = (\bar{V}, \bar{E})$ とする。また，任意の頂点 $i \in \bar{V}$ に対して以下を定義する。

$$\delta(i) = \{\, j \in V \mid (i,j) \in \bar{E} \,\},$$
$$\rho(i) = \{\, j \in V \mid (j,i) \in \bar{E} \,\}.$$

列生成子問題は以下のように定式化できる。

$$
\mathrm{SP}(\boldsymbol{\lambda}^*_{(\mathcal{S})}):\left|
\begin{array}{ll}
\text{min.} & (1-\theta) \displaystyle\sum_{(i,j) \in E} w_{ij}(1-y_i)y_j \\
& \quad + \theta \displaystyle\sum_{(i,j) \in E} w_{ij} d_{ij} z_{ij} - \sum_{i \in V} \lambda^*_i y_i \\
\text{s.t.} & \displaystyle\sum_{j \in \delta(0)} z_{0j} = 1, \\
& \displaystyle\sum_{j \in \rho(i)} z_{ji} = y_i, \quad \forall i \in V, \\
& \displaystyle\sum_{j \in \delta(i)} z_{ij} \le |V| \times y_i, \quad \forall i \in V, \\
& l_i + 1 \le l_j + |V|(1 - z_{ij}), \\
& \qquad \forall (i,j) \in E, \\
& \displaystyle\sum_{(i,j) \in E} z_{ij} = \sum_{i \in V} y_i - 1, \\
& (\boldsymbol{y}, \boldsymbol{z}) \in F, \\
& y_i \in \{0,1\}, \quad \forall i \in V, \\
& z_{ij} \in \{0,1\}, \quad \forall (i,j) \in \bar{E}, \\
& l_i \in \mathbb{R}, \quad \forall i \in V.
\end{array}
\right.
$$

非線形項 $y_i y_j$ に対して適当な変換を施すと列生成子問題を混合整数線形計画問題に帰着でき，既存の最適化ソルバーを適用することが可能である。また，$F$ は木に対して何らかの実務的な制約条件を課す集合とする（詳細については省略する）。**Algorithm 1** に列生成法の概要をまとめる。

**Algorithm 1** 列生成法

1: 初期の部分集合族 $\mathcal{S}$ を設定する。
2: 問題 LP($\mathcal{S}$) を解いて，最適値 $v(\mathrm{LP}(\mathcal{S}))$ と対応する双対変数 $\boldsymbol{\lambda}^*_{(\mathcal{S})}$ を得る。
3: 問題 $\mathrm{SP}(\boldsymbol{\lambda}^*_{(\mathcal{S})})$ を解いて，最適解 $\boldsymbol{y}^*$ と最適値 $v(\mathrm{SP}(\boldsymbol{\lambda}^*_{(\mathcal{S})}))$ を得る。
4: **if** $v(\mathrm{SP}(\boldsymbol{\lambda}^*_{(\mathcal{S})})) \ge 0$ **then**
5: 　　LB $\leftarrow v(\mathrm{LP}(\mathcal{S}))$ とし，Line 8 へ進む。
6: **else**
7: 　　$\mathcal{S} \leftarrow \mathcal{S} \cup \{\{i \in V \mid y^*_i = 1\}\}$ として，Line 2 へ戻る。
8: 問題 P($\mathcal{S}$) を解き，最適値 $v(\mathrm{P}(\mathcal{S}))$ を得る。

## 3.3 セクタ作業量の平準化問題と Normalized-cut アプローチ

Normalized-cut の最小化に対しては，様々な発見的解法が提案されている。本論文では，スペクトル法と呼ばれる手法を応用する。Normalized-cut を空域分割問題に適用するにあたり，モデルを簡略化し，対象とするグラフは無向グラフであり，各枝には単一の重みが付されているとする*。

頂点集合 $V = \{1, 2, \ldots, n\}$ と無向枝の集合 $E$ からなる無向グラフを $G = (V, E)$ とし，任意の頂点対上で定義される重み関数を $w : V \times V \to \mathbb{R}_+$ とする。ただし，枝の張られていない頂点対の重みは 0 とする。また，各頂点の（重み付き）次数を $d_i = \sum_{j \in V} w_{ij}$ と定義し，頂点次数を対角要素に持つ対角行列を $D$，各頂点対の重みを要素に持つ行列を $W$ とする。任意の頂点部分集合 $A \subset V$ に対して，

$$\mathrm{cut}(A, A') = \sum_{i \in A} \sum_{j \in A'} w_{ij}, \quad \mathrm{vol}(A) = \sum_{i \in A} d_i$$

と定義する。ただし，$A' = V \setminus A$ である。頂点集合 $V$ の分割 $\{V_1, V_2, \ldots, V_k\}$ に対する Normalized-cut は以下で定義される。

$$\mathrm{Ncut}(V_1, V_2, \ldots, V_k) = \sum_{j=1}^{k} \frac{\mathrm{cut}(V_j, V'_j)}{\mathrm{vol}(V_j)}.$$

Normalized-cut の各項

$$\frac{\mathrm{cut}(V_j, V'_j)}{\mathrm{vol}(V_j)} \tag{3.4}$$

はセクタ $V_j$ に関するカット重みと $V_j$ に含まれる頂点次数の和の比率を表している。したがって，この関数を最小化することで，セクタ間の管制移管に関する作業量が小さく，かつ各セクタの作業量が平準化された分割を得ることが期待される。

*3.2 節のモデルでは，各枝に作業量に関する重みと距離に関する重みが与えられていた。

### 3.3.1 Normalized-cut 最小化とラプラシアン

通常のラプラシアン $L$ と正規化ラプラシアン $L_N$ を導入する。（詳細は，[8] を参照のこと。）

$$
\begin{aligned}
L &= D - W,\\
L_N &= D^{-1/2} L D^{-1/2}.
\end{aligned}
$$

Normalized-cut の最小化問題がラプラシアンを用いて記述できることを示す。頂点集合 $V$ の分割 $\{V_1, V_2, \ldots, V_k\}$ が与えられたときに，変数 $x_i^j$ を

$$
x_i^j = \begin{cases} 1/\sqrt{\mathrm{vol}(V_j)} & (i \in V_j) \\ 0 & (i \notin V_j) \end{cases} \tag{3.5}
$$

と定義し，

$k$ 本の変数ベクトル $\boldsymbol{x}^j$ を並べた行列を $X$ とすると，

$$
\mathrm{Ncut}(V_1, V_2, \ldots, V_k) = \sum_{j=1}^{k} (\boldsymbol{x}^j)^\top L \boldsymbol{x}^j = \mathrm{Tr}(X^\top L X)
$$

が成り立つ。ただし，Tr は行列の対角和を表す線形オペレータである。したがって，Normalized-cut の最小化問題はラプラシアン $L$ を用いて，以下の形式で表すことができる。

$$
\mathrm{NC}: \left| \begin{array}{ll} \text{min.} & \mathrm{Tr}(X^\top L X) \\ \text{s.t.} & X \text{ の各要素は (3.5) を満たす.} \end{array} \right.
$$

### 3.3.2 スペクトル法

問題 NC には変数の離散制約が存在するため解くことが困難である。制約条件 (3.5) を緩和して解くことを考える。また，$X^\top X$ の対角成分が一定値になるように行列 $X$ の左から対角行列 $D^{1/2}$ を掛けた行列 $Y (= D^{1/2} X)$ を変数とする問題を考える。

$$
\mathrm{RNC}: \left| \begin{array}{ll} \text{min.} & \mathrm{Tr}(Y^\top D^{-1/2} L D^{-1/2} Y) \\ & \quad = \mathrm{Tr}(Y L_N Y) \\ \text{s.t.} & Y^\top Y = I, \\ & Y (= D^{1/2} X) \in \mathbb{R}^{n \times k}. \end{array} \right.
$$

問題 RNC は正規化ラプラシアンに関する問題であり，元問題 NC の緩和問題となっている。このとき問題 RNC は，目的関数を最小化するような正規直交基底を選択する問題と捉えることができ，正規化ラプラシアン $L_N$ の半正定値性から，最適値は $L_N$ の小さい方から $k$ 個の固有値 $\lambda_j$ の和となる。また，最適解は $\lambda_j$ に対応する固有ベクトルである。

**定理 3.1** 問題 RNC の最適値は $\sum_{j=1}^{k} \lambda_j$，最適解 $Y^*$ は次で与えられる。

$$
Y^* = \begin{bmatrix} \boldsymbol{v}^1 & \boldsymbol{v}^2 & \cdots & \boldsymbol{v}^k \end{bmatrix} \in \mathbb{R}^{n \times k}.
$$

ただし，$\lambda_1, \lambda_2, \ldots, \lambda_k$ は $L_N$ の小さい方から $k$ 個の固有値，$\boldsymbol{v}^1, \boldsymbol{v}^2, \ldots . \boldsymbol{v}^k$ は対応する固有ベクトルである。さらに，$X^* = D^{-1/2} Y^*$ は，一般化固有値問題 $L\boldsymbol{u} = \lambda D \boldsymbol{u}$ の固有値 $\lambda_1, \lambda_2, \ldots, \lambda_k$ に対応する固有ベクトル $\boldsymbol{u}^1, \boldsymbol{u}^2, \ldots, \boldsymbol{u}^k$ を並べた行列で与えられる。

定理 3.1 より，ラプラシアン $L$ に関する一般化固有値問題を解けば，緩和問題 RNC を解くことができる。しかし，これにより得られる $X^*$ は一般に制約条件 (3.5) を満たすとは限らないので，何らかの方法でこの緩和解をラウンディングする必要がある。本論文では $k$-means 法を利用する。この方法は，$k$ 個の固有ベクトル $\boldsymbol{u}^j (j = 1, 2, \ldots, k)$ の第 $i$ 成分からなる $k$ 次元ベクトル $\boldsymbol{z}^i = (u_i^1, u_i^2, \ldots, u_i^k)$ を頂点 $i$ の座標と見なし，その $k$ 次元ベクトルに対して $k$-means クラスタリングを適用し，分割を求める方法である。Normalized-cut 最小化に対するスペクトル法は以下のように記述できる。

**Algorithm 2** スペクトル法

ラプラシアン $L$ に関する一般化固有値問題 $L\boldsymbol{u} = \lambda D \boldsymbol{u}$ を解き，小さい方から $k$ 個の固有値 $\lambda_j$ と対応する固有ベクトル $\boldsymbol{u}^j$ を求める。LB $\leftarrow \sum_{j=1}^{k} \lambda_j$.
**for** $i \in V$ **do**
    $\boldsymbol{z}^i \leftarrow (u_i^1, u_i^2, \ldots, u_i^k) \in \mathbb{R}^k$
データ $\{\boldsymbol{z}^1, \boldsymbol{z}^2, \ldots, \boldsymbol{z}^n\}$ に対して $k$-means クラスタリングを実行し，分割 $\{V_1, V_2, \ldots, V_k\}$ を得る.
UB $\leftarrow$ Ncut$(V_1, V_2, \ldots, V_k)$ として LB と UB を出力する。

### 3.4　制約付き三角形分割に基づく空域分割

#### 3.4.1　制約付きドロネー三角形分割

本節では，与えられた頂点集合の分割をもとに，セクタの幾何学的な形状を求める可視化方法を提案する。

$P$ を平面上の点の集合とし，その凸包，すなわち $P$ を包含する最も小さな凸図形を考える。$P$ の要素（点）を線分で結び，$P$ の凸包を多数の三角形に分割する。ここで，分割によって得られる三角形の外接円の内部に，$P$ に属する点は含まれてはならない。このような操作を三角形分割という。

ある $P$ に対して，その三角形分割はいくつも存在する。これらのうち，三角形分割によって得られる任意の三角形の外接円の内部に，$P$ に属する点が含まれないようなものをドロネー三角形分割という。$P$ に対してドロネー三角形分割は一意に定まり，またドロネー三角形分割によって得られる三角形の内角の大きさの最小値が最も大きい三角形分割である。

次に，$P$ の2点を結ぶ線分の集合 $A$ を考える。ただし，$A$ の任意の2線分は，端点以外で交わることがないものとする。このとき，三角形分割により得られる三角形の辺の中に，$A$ の全ての線分が含まれるようなものを制約付き三角形分割と呼ぶ。制約付き三角形分割のうち，上述のドロネー三角形分割のように，得られる三角形の外接円内に含まれる点の数が最も少なくなるものを制約付きドロネー三角形分割と呼ぶ [11]。

#### 3.4.2　基本的な定式化と計算の方針

グラフ $G=(V,E)$ の頂点集合 $V$ の $n$ 個の部分集合 $\{V_k; k=1,2,\dots,n\}$ への分割が得られているとする。このとき，頂点 $v$ が属する部分集合の添字を $[v]$ とする。すなわち，$v \in V_{[v]}$ である。さらに，頂点 $v$ に対応する平面上の点の座標を $(x_v, y_v)$ とし，頂点の部分集合 $V_k$ に対応する平面上の領域（セクタ）を $R_k$ とする。このとき，頂点 $V$ の分割 $\{V_k\}$ に対応する，求めたい空域の分割は，平面の $\{R_k\}$ による分割である。

点集合 $P$ のある三角形分割に対し，その双対として平面の分割を構成することができる。すなわち，点 $p \in P$ を平面内の面に対応させ，三角形分割において線分で結ばれた点は辺で隣り合うようにすることで，平面の分割ができる。ここで，グラフ $G$ の分割で，同一の部分集合 $V_k$ に属する頂点 $u, v$ に対応する点 $(x_u, y_u), (x_v, y_v)$ について，対応する面の和集合を考えることで，頂点集合の分割 $\{V_k\}$ に対応する平面の分割 $\{R_k\}$ を得ることができる。このとき，$R_k$ の境界 $\partial R_k$ は，$V_k$ と異なる部分集合 $V_{k'}$ に属する頂点に対応する面と隣り合う部分のみで構成される。したがって $\partial R_k$ は，$[u] \neq [v]$ であるような線分 $(x_u, y_u)$-$(x_v, y_v)$ とのみ共有点をもつ。

また，$\partial R_k$ は $V_k$ とそれ以外とを分ける。したがって，その一部を $j \neq k$ である $\partial R_j$ と，別の一部を $l \neq j, k$ である $\partial R_l$ と共有する。$\partial R_j, \partial R_k, \partial R_l$ が1点を共有する状況を考えよう。この共有点から伸びる境界線は $R_j \cap R_k, R_k \cap R_l, R_l \cap R_j$ のいずれかの一部である。これらの境界線は，両端点が異なる部分集合に属する頂点である線分と交わりをもつため，この共有点は3つの頂点が相異なる3つの部分集合に属する三角形の内部に存在する。

以上より，$\partial R_k$ を線分と半直線で構成する。まず，グラフ $G=(V,E)$ の辺集合 $E$ に対応する線分を含むような，点集合 $\{(x_v, y_v); v \in V\}$ の制約付きドロネー三角形分割を行う。これによって得られた三角形のうち，線分の両端点 $u, v$ が異なる部分集合に属する（$[u] \neq [v]$ となる）ような線分上と，3頂点 $v_1, v_2, v_3$ が相異なる部分集合に属する（$[v_1] \neq [v_2] \neq [v_3]$ となる）三角形の内部に節点を設け，その位置を修正することで平面の分割を得る。

#### 3.4.3　初期分割の生成

部分集合 $V_k$ に対応するセクター $R_k$ の境界 $\partial R_k$ を折れ線の集合として表現し，節点の集合を $\Pi$，節点の隣接リストの集合を $\Sigma$ とする。

まず，点集合 $\{(x_v, y_v); v \in V\}$ の $\{(x_u, y_u)$-$(x_v, y_v) \mid (u,v) \in E\}$ の制約付きドロネー三角形分割を行う。この三角形分割により得られる三角形の集合を $\mathcal{T}$，線分の集合を $\mathcal{S}$ とする。制約付きドロネー三角形分割の生成法から，$\mathcal{S} \subseteq E$ である。また，$S^{\mathrm{T}}(t)$，$P^{\mathrm{T}}(t)$ をそれぞれ三角形 $t \in \mathcal{T}$ の辺，頂点の集合とする。なお，$\bigcup_{t \in \mathcal{T}} S^{\mathrm{T}}(t) = \mathcal{S}$，$\bigcup_{t \in \mathcal{T}} P^{\mathrm{T}}(t) = V$

である。さらに，$P^{\mathrm{S}}(s)$ を辺 $s \in \mathcal{S}$ の両端点とする。

ここで，線分 $s \in \mathcal{S}$ のうち，両端点が異なる部分集合に属するものを $\mathcal{S}_2$ と，三角形 $t \in \mathcal{T}$ のうち 3 頂点が相異なる部分集合に属するものを $\mathcal{T}_3$ とする。そして，線分 $s \in \mathcal{S}_2$ 上と三角形 $t \in \mathcal{T}_3$ 内部に節点を設け，分割空域の節点集合 $\Pi$ を構成する。本論文では，$s \in \mathcal{S}_2$ 上の点として，線分 $s$ の中点を，三角形 $t \in \mathcal{T}_3$ 内部の点として，三角形 $t$ の 3 頂点の重心を用いる。また節点 $\pi \in \Pi$ は，線分 $s \in \mathcal{S}_2$，あるいは三角形 $t \in \mathcal{T}_3$ に対応づけられる。そこで，以下では $\pi[s], \pi[t]$ によって $s \in \mathcal{S}$，$t \in \mathcal{T}$ に対応する節点を表すことにする。

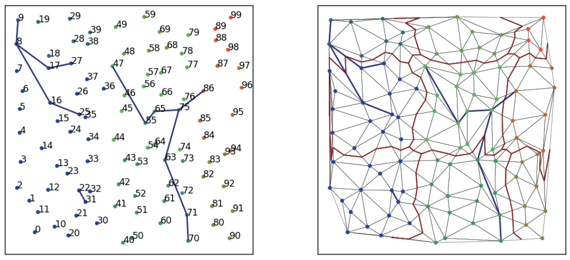
節点集合 $\Pi$ を構成したのち，それらの隣接関係を定めることで線分の集合 $\Sigma$ を表現する。節点のときと同様に，$\sigma[s], \sigma[t]$ によって $s \in \mathcal{S}$，$t \in \mathcal{T}$ に対応する節点に隣接する節点のリストを表すことにする。ただし，これらのリストの要素も，節点そのものではなく，節点に対応する線分，三角形といった幾何学的オブジェクトとする。

上記のような約束の下で，$\Sigma$ は次のような関係を満たす。いま，三角形 $t \in \mathcal{T}$ のうち 3 頂点が異なる 2 つの部分集合に属するものを $\mathcal{T}_2$ とする。三角形 $t \in \mathcal{T}_2$ の 3 辺のうち，頂点が異なる部分集合に属するものは 2 つ存在し，これらの辺には対応する節点がある。$t \in \mathcal{T}_2$ であるので，$t$ の内部には節点は存在しないため，辺上の 2 つの節点は互いに隣接する。次に，三角形 $t \in \mathcal{T}_3$ は，3 辺全てが，その端点が異なる部分集合に属し，辺上に 3 つの節点が存在する。また，この三角形 $t$ の内部にも節点が存在し，辺上の 3 節点と $t$ 内部の節点は互いに隣接する。これらをまとめ，**Algorithm 3** により $\Sigma$ を構成できる。

図 1 に，正方形内に一様かつランダムに発生させた頂点集合に対する計算例を示す。(a) は，入力サンプルであり，頂点の色が属する部分集合に対応しており，また青色の線は制約付きドロネー三角形分割に必ず含まれるべき線分を示している。また，(b) は入力サンプルに対して，上述の初期分割生成を行なった結果であり，領域の境界が赤色の線で示されている。これを見ると，領域の分割自体はできているものの，凹凸が散見される。

**Algorithm 3** 初期分割の生成

$\Sigma \leftarrow \{\sigma[s],\, s \in \mathcal{S}_2\} \cup \{\sigma[t],\, t \in \mathcal{T}_3\}$
**for** $\sigma \in \Sigma$ **do**
  $\sigma \leftarrow \{\}$
**for** $t \in \mathcal{T}_2$ **do**
  $\{s_0, s_1\} \leftarrow \{s \in S^{\mathrm{T}}(t) \mid \#\{[v],\, v \in P^{\mathrm{S}}(s)\} = 2\}$
  $\sigma[s_0] \leftarrow \sigma[s_0] \cup \{s_1\}$
  $\sigma[s_1] \leftarrow \sigma[s_1] \cup \{s_0\}$
**for** $t \in \mathcal{T}_3$ **do**
  **for** $s \in S^{\mathrm{T}}(t)$ **do**
    $\sigma[t] \leftarrow \sigma[t] \cup \{s\}$
    $\sigma[s] \leftarrow \sigma[s] \cup \{t\}$



(a) 入力サンプル　(b) サンプル入力に対する初期分割

**図 1 正方形内の一様かつランダムな頂点での例**

### 3.4.4　競合学習法による改善

分割領域の形状を改善する。領域形状の改善は節点 $p \in \Pi$ を隣接構造は保ったまま移動させることで実現する。本論文では，教師なし学習の一種である競合学習法を利用する。

分割領域の形状は，可能な限り単純であることが望ましい。ここで単純とは，境界にあまり凹凸がなく，それぞれの領域が凸に近い形状といった意味であり，3 つの節点を結んだ折れ線が，できる限り直線に近くなるよう節点を移動させればよい。しかしながら，ある一つの領域に着目して，このような変形を行おうとすると，これに隣接する領域の形状が悪化する恐れがある。このような副作用を加味した上で，領域を望ましい形状へ変形することは難しい。

本論文では，ランダムな順番で節点を選択し，

選択された節点が隣接する節点より影響を受け，上述のように折れ線が直線に近づくような方向へ移動させるといった操作を繰り返し行うことを考える。このとき，前述のようにある一つの領域については望ましい変形であっても，別の領域にとってはそうではないかもしれない。しかし，このような局所的な操作を，節点を選択する順番をランダムに変更しながら繰り返すことにより，大局的（平均的）には望ましい方向へと変形することが期待できる。

いま，$\tau$ 回目の反復において，ある節点が選択されたとする。この節点は，$\mathcal{S}_2 \cup \mathcal{T}_3$ の要素に対応している。節点が線分 $g \in \mathcal{S}_2$ に対応するものであるとするこのとき，線分 $g$ の両端点を $p_0, p_1$ とする。ここで，$g$ に対応する節点 $\pi[g]$ を，隣接する節点 $\pi[q]$, $q \in \sigma[g]$ に近づくよう移動させる。ただし，$\pi[g]$ は線分 $g$ 上にあるようにする。このとき，$\pi[g]$ の移動量は，$\tau$ の増加とともに次第に小さくなるようにする。一方，節点が三角形 $g \in \mathcal{T}_3$ に対応するものならば，節点 $\pi[g]$ は 3 つの節点と隣接する。これらの $\pi[g]$ と隣接する 3 節点との長さの総和を最小にする点は，3 節点を頂点とする三角形のフェルマー点となる。ここでは計算の簡便のため，$\pi[g]$ を 3 頂点の重心に移動させる。

上記の手順を **Algorithm 4** に示す。

**Algorithm 4** 競合学習法による分割領域の調整

**for** $\tau = 1, 2, \ldots$ **do**
  $\mathcal{G} \leftarrow \{\mathcal{S}_2 \cup \mathcal{T}_3$をランダムに並べ替えたもの$\}$
  **for** $g \in \mathcal{G}$ **do**
    **if** $g \in \mathcal{S}_2$ **then**
      $\{p_0, p_1\} \leftarrow P^{\mathrm{S}}(g)$
      $\pi[g] \leftarrow \pi[g] + \sum_{q \in \sigma[g]} \frac{(\pi[q]-\pi[g]) \cdot (p_1 - p_0)}{||\pi[q]-\pi[g]|| \cdot ||p_1 - p_0||}(p_1 - p_0) \cdot \frac{1}{a\tau}$
    **else**
      $\pi[g] \leftarrow \sum_{q \in \sigma[g]} \frac{q}{3}$

図 2 に，図 1 の入力および初期分割に対し，上述の計算を行なった結果を示している。図 1(b) と同様に，領域の境界が赤い線で示されている。全ての領域が凸とはならないものの，初期分割と比較すると領域境界線は直線に近くなっており，視認性に優れた分割が得られている。

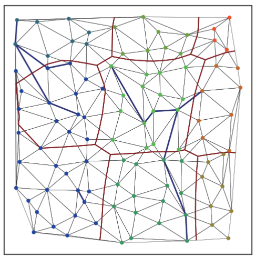


**図 2 正方形内の一様かつランダムな頂点に対する領域の修正**

## 4　計算機実験

3.2 項と 3.3 項で提案した 2 つのアプローチを，実データに基づき作成した空域ネットワークに対して適用し，得られた空域分割（頂点分割）の比較を行う。

空域ネットワークは以下のとおり作成した。日本国内で比較的旅客便数が多く，かつ発着空港が適度に分散するように，いくつかの路線を選択する。次に，各路線を飛行する際に想定される航路上の（発着空港を含む）ウェイポイントの列を求め，各ウェイポイント間を線分で結ぶ。最後に，航路がウェイポイント以外で交差する場所を求め，交点をダミーのノードとして交差するそれぞれの航路に挿入する。

具体的な路線として羽田–福岡，羽田–伊丹，伊丹–仙台，羽田–高松，羽田–金沢の各路線を選択し，それぞれについて出発・到着空港を入れ替えた 10 通りとした。その結果，空域ネットワークは，ノード数が 84，エッジ数が 99 となった。

次に，セクタ間とセクタ内の作業量を次のとおり定めた。セクタ間の作業は，主に管制の移管に付随することから，セクタを出入りする航空機の数を用いることでその作業量をはかることとする。すなわち，(3.1) の第 1 項について，$w_{ij}$ を枝 $(i, j)$ に対応するウェイポイント間を通過する 1 日あたりの航空機数とする。また，セクタ内の作業量は，当該セクタを飛行する航空機に対する監視と指示が主なものである。セクタ内の作業量を表す (3.1) の第 2 項について，$w_{ij}$ を枝 $(i, j)$ に対応するウェイポイント間を通過する 1 日あたりの航空機数，$d_{ij}$ をその直線距離 (m) とした。

数値実験は CPU Intel Core i7 1.4GHz，メモリ 16GB の計算機上で行い，実験プログラムは Python 3.6.3 を用いて実装した。列生成法における限定主問題と列生成子問題の求解には Gurobi Optimizer 8.0.1 の Python API，Normalized-cut アプローチにおける線形計算アルゴリズムと $k$-means 法については，それぞれ SciPy と scikit-learn を用いた。

### 4.1　列生成アプローチの計算結果

セクタ作業量 (3.1) における調整パラメータは $\theta = 0.5$，各セクタ作業量の上下限は $(l, u) = (12000, +\infty)$, セクタ境界線とウェイポイント間の制約においては $d_{\min} = 4 \times 1800$，初期部分集合族のサイズについては $s = 4$，とそれぞれ設定した。なお，列生成子問題を解く際には，各反復で厳密には解かずに目的関数値が負となる実行可能解が見つかった時点で列を追加する戦略を採用した。また，限定主問題の求解に対してはクロスオーバー機能を無効にした内点法を用いた。

列生成法の計算結果を表 1 にまとめる。

表 1 計算結果（列生成法）

| | |
|---|---|
| 反復回数 | 1647 |
| LB | 188244.809 |
| UB | 188244.809 |
| セクタ数 | 7 |
| 計算時間（秒） | 1389.82 |

図 3 は分割された頂点をセクタごとに色分けしたものであり，表 2 は得られた 7 個のセクタの詳細についてまとめたものである。表 2 を見ると，各セクタを構成する頂点数や作業量のばらつきが大きいことが観察される。特にセクタ作業量については最大で 4 倍程度の差が存在する。

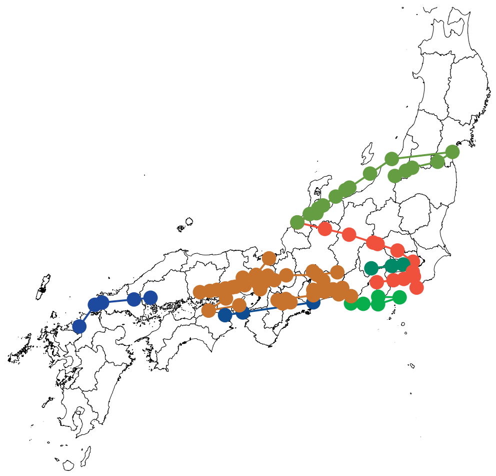


図 3 分割結果（列生成法）

表 2 各セクタの概要（列生成法）

| セクタ番号 | 頂点数 | 作業量（式 (3.1) |
|---|---|---|
| 1 | 5 | 24510.48 |
| 2 | 3 | 26251.33 |
| 3 | 3 | 12832.47 |
| 4 | 5 | 22575.55 |
| 5 | 15 | 17978.81 |
| 6 | 40 | 53199.33 |
| 7 | 13 | 30896.79 |
| 最大値/最小値 | 13.33 | 4.14 |

### 4.2　Normalized-cut アプローチの計算結果

Normalized-cut アプローチにおいては予め分割するセクタ数を固定する必要があるため，列生成アプローチによって得られた分割数である $k = 7$ に設定し比較を行う。表 3 に Normalized-cut アプローチの計算結果をまとめる。

表 3 計算結果（Normalized-cut）

| | |
|---|---|
| LB | 0.2562 |
| UB | 0.8939 |
| セクタ数 | 7 |
| 計算時間（秒） | 0.28 |

図 4 は分割された頂点をセクタごとに色分けしたものであり，表 4 は得られた 7 個のセクタの詳細についてまとめたものである。表 4 から，Normalized-cut アプローチにより得られた各セクタの頂点数や作業量のばらつきは，列生成アプローチによるものと比較して小さくなっていることが観察される。特にセクタ作業量については最大で 2 倍程度の差に抑えられている。

表 4 各セクタの概要（Normalized-cut）

| セクタ番号 | 頂点数 | 作業量（式 (3.4)） |
|---|---|---|
| 1 | 19 | 0.1081 |
| 2 | 19 | 0.1631 |
| 3 | 12 | 0.0833 |
| 4 | 12 | 0.1226 |
| 5 | 6 | 0.1666 |
| 6 | 8 | 0.1250 |
| 7 | 8 | 0.1250 |
| 最大値/最小値 | 3.16 | 2.00 |

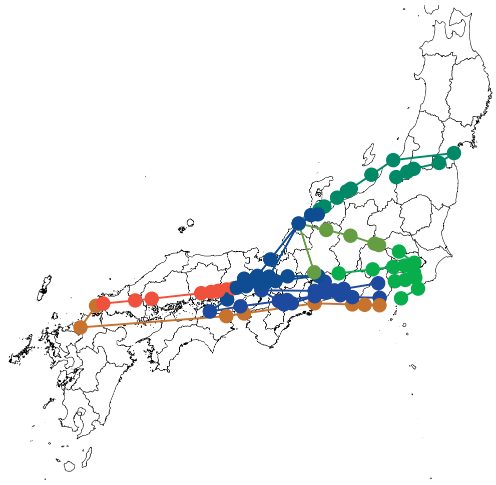


図 4 分割結果（Normalized-cut）

### 4.3　セクタ作業量の評価の差異による比較

4.2 節の計算結果により，Normalized-cut アプローチによって得られた各セクタ間の作業量のばらつきが小さいことが確認された。しかし，Normalized-cut を空域分割問題に適用するにあたり，ネットワークの枝上の重みとしてその距離を考慮していない。そこで，Normalized-cut アプローチにより得られたセクタ作業量を，列生成アプローチでのセクタ作業量 (3.1) に変換した結果を表 5 に示す。

## 5　結論

本研究では，我が国の空域編成をシステマチックに得ることを目的に，数理最適化法の適用を検討した。空域の編成は，空域をセクタとよぶ単位に分割することで実現される。航空管制官の管制作業量をセクタ間で平準化することが期待されている。あるセクタにおける航空管制官の管制作業量は，セクタ内作業量と隣接するセクタとの間のセクタ間作業量によって規定される。本研究では，これらの管制作業量に着目した 2 つの空域編成のための数理最適化アプローチを提案した。また，セクタを GIS（Geographic Information System）のような地図上に “見える化” する 2 つの方法を提案した。

本研究で提案した方法は，空域編成をシステマチックに実現することができるものである。今後導入が想定される UPR など，新たな空域編成が求められる環境においても，本研究の提案はその基盤として活用することが可能である。

表 5 セクタ作業量の変換結果

| セクタ番号 | 作業量（式 (3.4)） | 作業量（式 (3.1)） |
|---|---|---|
| 1 | 0.1081 | 51387.52 |
| 2 | 0.1631 | 44067.28 |
| 3 | 0.0833 | 14654.84 |
| 4 | 0.1226 | 57939.01 |
| 5 | 0.1666 | 7157.75 |
| 6 | 0.1250 | 91178.40 |
| 7 | 0.1250 | 37090.46 |
| 最大値/最小値 | 2.00 | 12.73 |

## 参考文献

[1] P. Kopardekar, K. Biilimoria and B. Sridhar, “Initial Concepts for Dynamic Airspace Configuration”, 7th AIAA Aviation Technology , Integration and Operations Conference (ATIO), Belfast, Northern Ireland, Sep. 2007.

[2] P. Flener and J. Pearson, “Automatic Airspace Sectorization: A Survey”, http://arxiv.org/abs/1311.0653,（参照， Apr. 2019）.

[3] P. Jägare, P. Flener and J. Pearson, “Airspace Sectorisation using Constraint-Based Local Search”, 10th Air Traffic Management Research and Development Seminar, Chicago, Illinois, June 2013.

[4] S. Kulkani, R. Gansan and L. Sherry, “Static Sectorization Approach to Dynamic Airspace Configuration using Approximate Dynamic Programming”, 2011 Integrated Communication Navigation and Surveillance (ICNS) Conference, Herndon, Virginia, April, 2011.

[5] T.K. Venugopalan, C. Wong, S. Suresh and N. Sundarajan(2018), “Simultaneous Optimization of Airway and Sector Design for Air Traf-

fic Management", *Journal of Air Transportation*, Vol. 26, No. 1, pp. 8–22.

[6] J. Shi and J. Malik (2000), "Normalized Cuts and Image Segmentation", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 22, No. 8, pp.888–905.

[7] G. Desaulniers, J. Desrosiers and M. M. Solomon(2005), "Column generation", Springer.

[8] U. von Luxburg (2007), "A Tutorial on Spectral Clustering", *Statistics and Computing*, Vol. 17, No. 4, pp.395–416.

[9] T.A. Granberg, T. Polishchuk, V. Polishchuk and C. Schmidt, "A Novel MIP-based Airspace Sectorization for TMAs", 12th USA/Europe Air Traffic Management Research and Development Seminar, Seattle, Washington, June 2017.

[10] 吉田幸司, 井上亮（2017），"管制地点を母点としたボロノイ図を用いたセクター設定手法の提案", 土木学会論文集 D3, Vol. 73, No. 5, pp. 1023–1031.

[11] 杉原厚吉 (1995), "グラフィックスの数理'(情報数学講座 (13))', pp.226, 共立出版.

[12] 中野良平（2005), "ニューラル情報処理の基礎数理（情報システム工学）", pp.238. 数理工学社.