数理解析研究所講究録
第1879巻 2014年 15-27

モジュラリティ最大化問題に対する切除平面法に基づく発見的解法

伊豆永 洋一（Yoichi Izunaga）* 山本 芳嗣（Yoshitsugu Yamamoto）†

概要

モジュラリティとは，Newman and Girvan によって提案されたグラフ分割の質を表す評価関数である．本稿では，モジュラリティ最大化問題を集合分割問題として定式化し，線形計画緩和に基づくアルゴリズムを提案する．線形計画緩和問題の双対問題を解くために，切除平面法を用いる．その際に，切除平面法を高速化するために，互いに補完し合うような切除平面群を同時に追加する手法を提案する．

1 はじめに

SNS の普及によって，グラフ上でのクラスタリングが注目を集めている．Newman and Girvan [11] によってグラフ・クラスタリングの評価関数としてモジュラリティが提案されて以来，モジュラリティ最大化問題はこの分野における中心課題の1つとなっている．また，Brandes *et al.* [3] によってモジュラリティ最大化問題の NP-困難性が証明されたことによって，様々な発見的解法が提案されている．例えば，貪欲アルゴリズムに基づくアルゴリズム [8]，グラフ・スペクトルの理論に基づくアルゴリズム [12]，線形計画法とラウンディング法に基づくアルゴリズム [1] などがある．

様々な発見的解法が提案されている一方で，厳密解法に関する研究はあまり多くない．厳密解法に関するアプローチは，主に2つに大別される．1つは，混合整数2次計画問題に定式化する方法である [14]．この定式化では，連続緩和することによって，凸2次計画問題となるので，連続緩和問題であれば比較的容易に解くことが可能である．しかし，整数解を求めるためには分枝限定法を用いる必要があり，頂点数 n が60程度の小規模な問題しか解くことができていない．

もう1つのアプローチは，0-1 整数線形計画問題に定式化するものである．Aloise *et al.* [2] は，クリーク分割問題に定式化し，Grötschel and Wakabayashi [10] によって提案された切除平面法を利用することで，$n = 115$ の問題を解いている．また，モジュラリティ最大化問題を集合分割問題として定式化する方法もある．この定式化では頂点集合のすべての部分集合を考慮するため，$\mathcal{O}(2^n)$ の変数を有する．したがって，n が大きくなった場合に問題を保持するための計算機資源を確保することでさえ困難となる．このような種類の問題に対して適用される手法に列生成法がある．しかし，追加すべき“列”を決定するために解くべき補助問題が非凸な2次計画問題となるため，これを最適に解くことは容易ではない．さらに，列生成法は非常に収束が遅いことが知られている．この収束の遅さを克服するために，du Merle *et al.* [9] は，stabilized column generation と呼ばれる高速化手法を提案している．

本稿では，集合分割問題による定式化に対して切除平面法に基づくアルゴリズムを提案する．最も良い切除平面を見つけることは，分離問題を厳密に解くという意味で NP-困難であるので，我々は追加すべき切除平面を見つけるために発見的解法を使用する．また，切除平面法を高速化させるために複数の切除平面を同時に追加する手法を提案する．

*筑波大学大学院 システム情報工学研究科（Graduate School of Systems and Information Engineering, University of Tsukuba）
†筑波大学 システム情報系（Faculty of Engineering, Information and Systems, University of Tsukuba）

本稿の構成は以下の通りである．2節で，モジュラリティの定義を与え，モジュラリティ最大化問題を集合分割問題として定式化する．3節では，緩和問題とその双対問題を紹介し，4節で，切除平面を生成する際に発見的解法を用いた切除平面法を提案する．5節では，提案アルゴリズムの内部で用いるいくつかの発見的解法を説明し，6節で，提案アルゴリズムの性能を数値実験によって検証する．

2 モジュラリティ最大化問題

$G=(V,E)$ を頂点集合 $V=\{1,2,\ldots,n\}$ と枝集合 $E=\{1,2,\ldots,m\}$ を持つ無向グラフとする．頂点集合の部分族 $\Pi=\{C_1,C_2,\ldots,C_k\}$ が，$V=\cup_{p=1}^{k}C_p$，任意の相異なる $p,q\in\{1,2,\ldots,k\}$ に対して $C_p\cap C_q=\emptyset$，任意の $p\in\{1,2,\ldots,k\}$ に対して $C_p\neq\emptyset$ を満たすとき，Π を V の分割と呼ぶ．本稿では，分割のそれぞれの要素 C_p をコミュニティと呼ぶ．また枝の一方の端点が C_p に，もう一方の端点が C_q に属するような枝集合を $E(C_p,C_q)$ とし，$C_p=C_q$ の場合には単純に $E(C_p)$ と表記する．

以上の定義の下で，分割 Π におけるモジュラリティ $\mu(\Pi)$ は以下のように定義される．

$$\mu(\Pi)=\sum_{p=1}^{k}\left(\frac{|E(C_p)|}{m}-\left(\frac{|E(C_p)|+\sum_{q=1}^{k}|E(C_p,C_q)|}{2m}\right)^2\right).$$

ただし，$|\cdot|$ は，対応する集合の要素数を表す．ここで，G の隣接行列の (i,j) 成分を e_{ij}，頂点 i の次数を d_i，頂点 i の属するコミュニティの添字を $\pi(i)$ とすると，モジュラリティは，

$$\mu(\Pi)=\frac{1}{2m}\sum_{i\in V}\sum_{j\in V}\left(e_{ij}-\frac{d_id_j}{2m}\right)\delta(\pi(i),\pi(j)),$$

と書く事ができる．δ はクロネッカーのデルタである．モジュラリティ最大化問題 (MM) は，$\mu(\Pi)$ を最大にするような V の分割を求める問題である．$(e_{ij}-\frac{d_id_j}{2m})$ を w_{ij} とおくと，w_{ij} の対称性によりモジュラリティ最大化問題は次のように定式化される．

$$(MM)\quad\left|\begin{array}{ll}\text{maximize} & \dfrac{1}{m}\displaystyle\sum_{i\in V}\sum_{j\in V;i<j}w_{ij}\delta(\pi(i),\pi(j))+\frac{1}{m}\sum_{i\in V}w_{ii}\\ \text{subject to} & \Pi\text{ is a partition of }V.\end{array}\right.$$

(MM) の目的関数の第2項目は定数であるので，以降では省略する．ここからは，モジュラリティ最大化問題を集合分割問題として定式化する．そのために，いくつかの記号を導入する．$\mathcal{P}$ を V の非空なすべての部分集合の族とする．したがって，$\mathcal{P}$ の要素数は 2^n-1 となる．$\mathcal{P}$ のそれぞれの要素 C に対して，0-1変数 z_C を以下のように定義する．

$$z_C=\begin{cases}1 & \text{when } C\in\Pi\\ 0 & \text{otherwise.}\end{cases}$$

また，各 $i\in V$ と $C\in\mathcal{P}$ について，定数 a_{iC} を以下のように定義する．

$$a_{iC}=\begin{cases}1 & \text{when } i\in C\\ 0 & \text{otherwise.}\end{cases}$$

このとき，$\boldsymbol{a}_C=(a_{iC},\ldots,a_{nC})^\top$ はコミュニティCの *incidence vector* となる，つまり $C=\{i\in V\mid a_{iC}=1\}$ と表す事ができる．各 $C\in\mathcal{P}$ に対して f_C を

$$f_C=\frac{1}{m}\sum_{i\in C}\sum_{j\in C;i<j}w_{ij},\tag{2.1}$$

と定義すると，(2.1) は以下のように書き換える事ができる．

$$= \frac{1}{m} \sum_{i \in V} \sum_{j \in V; i<j} w_{ij} a_{iC} a_{jC}.$$

定数 f_C はコミュニティC が分割の要素に採用されたときの，目的関数への C の貢献度を表している．すると，(MM) は以下の整数計画問題 (P) として定式化される．

$$(P) \quad \left| \begin{array}{lll} \text{maximize} & \displaystyle\sum_{C \in \mathcal{P}} f_C z_C & \\ \text{subject to} & \displaystyle\sum_{C \in \mathcal{P}} a_{iC} z_C = 1 & (\forall i \in V) \\ & z_C \in \{0,1\} & (\forall C \in \mathcal{P}). \end{array} \right.$$

問題 (P) は莫大な個数の変数を持つ問題であるので，頂点数 n が増加するにつれて簡単に計算機資源の容量を超えてしまう．

3 緩和問題と双対問題

この節では，まず前節で定式化した問題 (P) に対する緩和問題とその双対問題について説明する．(P) に関しては，変数の個数だけではなく，変数に課された整数制約も問題 (P) を計算困難にしている要因である．このような場合に (P) の最適解についての有用な情報を得るための手段として，(P) の緩和問題を考える事が挙げられる [4, 5, 13]．そこで，(P) の整数制約 $z_C \in \{0,1\}$ を $0 \leq z_C \leq 1$ に置き換えた LP 緩和（*Linear Programming relaxation*）を考える．(P) の LP 緩和問題 (RP) は次で与えられる．

$$(RP) \quad \left| \begin{array}{lll} \text{maximize} & \displaystyle\sum_{C \in \mathcal{P}} f_C z_C & \\ \text{subject to} & \displaystyle\sum_{C \in \mathcal{P}} a_{iC} z_C = 1 & (\forall i \in V) \\ & z_C \geq 0 & (\forall C \in \mathcal{P}). \end{array} \right.$$

変数の上限制約 $z_C \leq 1$ は 1 つ目の制約条件により冗長な制約となるので，取り除いても問題は等価である．問題 (RP) の双対問題 (RD) は以下で与えられる．

$$(RD) \quad \left| \begin{array}{lll} \text{minimize} & \displaystyle\sum_{i \in V} \lambda_i & \\ \text{subject to} & \displaystyle\sum_{i \in V} a_{iC} \lambda_i \geq f_C & (\forall C \in \mathcal{P}) \\ & \lambda_i \in \mathbb{R} & (\forall i \in V). \end{array} \right.$$

(2.1) より，コミュニティC が 1 点集合であるとき $f_C = 0$ となることに注目すると，(RD) の不等式制約は双対変数 λ_i の非負制約を含んでいる．したがって (RD) は，変数に対する非負制約の課された以下の問題と等価である．

$$(RD) \quad \left| \begin{array}{lll} \text{minimize} & \displaystyle\sum_{i \in V} \lambda_i & \\ \text{subject to} & \displaystyle\sum_{i \in V} a_{iC} \lambda_i \geq f_C & (\forall C \in \mathcal{P}) \\ & \lambda_i \geq 0 & (\forall i \in V). \end{array} \right.$$

ここで，問題 $(\cdot)$ の実行可能領域と最適値をそれぞれ $\mathcal{F}(\cdot)$ と $\omega(\cdot)$ と表す．(RP) は (P) の緩和問題であるので，$\omega(P) \leq \omega(RP)$ が成立する．また線形計画問題における強双対定理を適用することで，$\omega(P) \leq \omega(RD)$ という関係を得る．つまり，(RD) の最適解を得る事で，$\omega(P)$ の上界値を得る事ができる．(RD) は n 個の変数だけを持つ問題であるが，制約の本数が非常に多い問題となり依然として計算困難な問題である．

4 切除平面法

(RD) は変数の個数をはるかに上回る本数の制約を有している．したがって，大部分の制約は最適解において有効ではないはずである．このような種類の線形計画問題に対して，広く使用される解法の1つとして切除平面法がある．そこで，本節では，まず切除平面法の概略を説明し，我々の提案するアルゴリズムを説明する．

4.1 切除平面法

切除平面法では，直接 (RD) を解く代わりに $\mathcal{P}$ の中の部分族 $\mathcal{C}$ のみを扱い，以下の問題 $(RD(\mathcal{C}))$ を反復的に解くことで (RD) の最適解を得る手法である．

$$(RD(\mathcal{C})) \quad \left| \begin{array}{lll} \text{minimize} & \displaystyle\sum_{i\in V} \lambda_i & \\ \text{subject to} & \displaystyle\sum_{i\in V} a_{iC}\lambda_i \geq f_C & (\forall C \in \mathcal{C}) \\ & \lambda_i \geq 0 & (\forall i \in V). \end{array} \right.$$

ここで，$\boldsymbol{\lambda}^*(\mathcal{C})$ を問題 $(RD(\mathcal{C}))$ の最適解とする．$C \in \mathcal{P} \setminus \mathcal{C}$ に関しては，制約 $\sum_{i\in V} a_{iC}\lambda_i \geq f_C$ は考慮されていないので，$\boldsymbol{\lambda}^*(\mathcal{C})$ は (RD) の実行可能解であるとは限らない．(RD) における $\boldsymbol{\lambda}^*(\mathcal{C})$ の実行可能性を判定するために，C に対応する制約の違反度 $\gamma_C(\mathcal{C})$ を定義する．

$$\gamma_C(\mathcal{C}) = \sum_{i\in V} a_{iC}\lambda_i^*(\mathcal{C}) - f_C. \tag{4.1}$$

任意の $C \in \mathcal{P} \setminus \mathcal{C}$ に対して，$\gamma_C(\mathcal{C}) \geq 0$ が成り立つならば，$\boldsymbol{\lambda}^*(\mathcal{C})$ は (RD) の実行可能解であることが保証されるので，したがって (RD) の最適解であることが分かる．一方で，ある $C \in \mathcal{P} \setminus \mathcal{C}$ に対して，

$$\gamma_C(\mathcal{C}) < 0$$

が成り立つならば，上式を満たす C を $\mathcal{C}$ に加えることで，問題 $(RD(\mathcal{C}))$ の最適値を改善できる可能性がある．すなわち，$\omega(RD(\mathcal{C} \cup \{C\})) \geq \omega(RD(\mathcal{C}))$ が成立する．(2.1) を，(4.1) に代入することによって次式を得る．

$$\gamma_C(\mathcal{C}) = \sum_{i\in V} a_{iC}\lambda_i^*(\mathcal{C}) - \frac{1}{m}\sum_{i\in V}\sum_{j\in V;i<j} w_{ij}a_{iC}a_{jC},$$

したがって，$\mathcal{P}$ 上で $\gamma_C(\mathcal{C})$ を最小化する問題は，以下の 0-1 変数の制約下での 2 次関数の最小化問題に定式化される．この問題は，一般に分離問題（*Separation Problem*）と呼ばれる．

$$(SP(\mathcal{C})) \quad \left| \begin{array}{lll} \text{minimize} & \displaystyle\sum_{i\in V} \lambda_i^*(\mathcal{C})y_i - \frac{1}{m}\sum_{i\in V}\sum_{j\in V;i<j} w_{ij}y_iy_j & \\ \text{subject to} & y_i \in \{0,1\} & (\forall i \in V). \end{array} \right.$$

$(SP(\mathcal{C}))$ の最適解 $\boldsymbol{y}^*$ は，$\mathcal{P}$ 上で $\gamma_C(\mathcal{C})$ を最小にするコミュニティの *incidence vector* となる．$(SP(\mathcal{C}))$ の最適値が負であるような $\boldsymbol{y}^*$ が見つかったならば，以下の制約を問題 $(RD(\mathcal{C}))$ に追加する．

$$\sum_{i\in V} y_i^*\lambda_i \geq f^*.$$

ただし，$f^* = \frac{1}{m}\sum_{i\in V}\sum_{j\in V;i<j} w_{ij}y_i^*y_j^*$ である．

上の議論から，切除平面法のアルゴリズムは以下のように与えられる．

Prototype of the Cutting Plane Algorithm

Step 0
　$\mathcal{C}$ を V の非空な部分集合の初期部分族とする.
Step 1
　最適解 $\boldsymbol{\lambda}^*(\mathcal{C})$ と最適値 $\omega(RD(\mathcal{C}))$ を得るために $(RD(\mathcal{C}))$ を解く.
Step 2
　$(SP(\mathcal{C}))$ を解いて，最適解 $\boldsymbol{y}^*$ を得る.
Step 3
　If $\omega(SP(\mathcal{C})) \geq 0$ then
　　$\mathcal{C}^* \leftarrow \mathcal{C}$，$\omega^* \leftarrow \omega(RD(\mathcal{C}))$ として，$\mathcal{C}^*$ と ω^* を出力し，アルゴリズム終了.
　else
　　$C \leftarrow \{i \in V \mid y_i^* = 1\}$ と設定し，$\mathcal{C} \leftarrow \mathcal{C} \cup \{C\}$ と更新する．Step 1 へ戻る.
　end if

切除平面法のアルゴリズムが終了すると，(RD) の最適解が得られたことになり，したがって (RP) の最適解も得られる．もし，この最適解が整数解であったならば，その解は元問題 (P) の最適解である．しかし，一般には整数解である保証はないので，あらためて整数解を得る必要がある．アルゴリズム終了時に得られる部分族 $\mathcal{C}^*$ は，元問題 (P) の最適解において正の値を取る可能性の高い変数に対する部分集合の族である．そこで，$C \in \mathcal{C}^*$ に関する変数 z_C だけを考慮した，以下の問題 $(P(\mathcal{C}))$ を解くことによって整数解を得ることにする.

$$
(P(\mathcal{C}^*)) \quad \left| \begin{array}{lll} \text{maximize} & \displaystyle\sum_{C \in \mathcal{C}^*} f_C z_C + \frac{1}{m} \sum_{i \in V} w_{ii} & \\ \text{subject to} & \displaystyle\sum_{C \in \mathcal{C}^*} a_{iC} z_C = 1 & (\forall i \in V) \\ & z_C \in \{0,1\} & (\forall C \in \mathcal{C}^*). \end{array} \right.
$$

この問題 $(P(\mathcal{C}^*))$ は，元問題 (P) よりもはるかに少ない変数を持つ問題となることが期待されるので，IP ソルバーによって十分に実用的な計算時間で解くことが可能となる．$(P(\mathcal{C}^*))$ は $\mathcal{C}^*$ に含まれない C に関する変数 z_C を無視しているので，$(P(\mathcal{C}^*))$ は $\omega(P)$ の下界を与える．つまり以下が成立する.

$$
\omega(P(\mathcal{C}^*)) \leq \omega(P) \leq \omega(RD(\mathcal{C}^*)).
$$

このとき，$\omega(RD(\mathcal{C}^*)) - \omega(P(\mathcal{C}^*))$ は $\omega(P(\mathcal{C}^*))$ と $\omega(P)$ の差の上界を与えるので，$(P(\mathcal{C}^*))$ の最適解の精度評価が可能になる.

4.2 提案アルゴリズム

切除平面法の Step 2 において解かれる問題 $(SP(\mathcal{C}))$ は，0-1 整数制約の下での非凸 2 次計画問題であり，厳密に最適解を求めることは困難である．そこで，$(SP(\mathcal{C}))$ に対して発見的解法を適用する．$(SP(\mathcal{C}))$ に適用数する発見的解法の詳細に関しては，5.1 節で述べることにして，ここでは，まずアルゴリズムの停止条件について議論する.

命題 1. LB を $\omega(P)$ の下界とする．もし $LB > \omega(RD(\mathcal{C}))$ が成立するならば，ある $C \in \mathcal{P} \setminus \mathcal{C}$ に対して $\gamma_C(\mathcal{C}) < 0$ が成り立つ.

Proof. 問題 (RP) は (P) の緩和問題であることから，$\omega(RP) \geq \omega(P) \geq LB$ が成り立つ．もし，任意の $C \in \mathcal{P} \setminus \mathcal{C}$ に対して $\gamma_C(\mathcal{C}) \geq 0$ ならば，$\omega(RD(\mathcal{C})) = \omega(RD)$ である．以上から，$\omega(RD(\mathcal{C})) \geq LB$ を得る．したがって，対偶をとると，$LB > \omega(RD(\mathcal{C}))$ が成り立つならば，ある $C \in \mathcal{P} \setminus \mathcal{C}$ に対して $\gamma_C(\mathcal{C}) < 0$ が成立する． □

ここで，発見的解法によって得られた目的関数値を $\alpha(SP(\mathcal{C}))$ とする．このとき $\alpha(SP(\mathcal{C}))$ が非負であっても，$\omega(SP(\mathcal{C}))$ が非負であるかどうかは分からない．命題 1 から，$LB > \omega(RD(\mathcal{C}))$ が成立している限りは，切除平面法を停止させることはできない．さらに，たとえ $LB \leq \omega(RD(\mathcal{C}))$ かつ $\alpha(SP(\mathcal{C})) \geq 0$ が成立したとしても，(RD) の最適解が得られたと結論付けることはできない．したがって，$LB \leq \omega(RD(\mathcal{C}))$ の下で，一定回数連続して $\alpha(SP(\mathcal{C}))$ が非負の値を取り続けた場合にアルゴリズムを停止することにする．なお $\omega(P)$ の下界 LB に関しては，$(RD(\mathcal{C}))$ の最適解 $\boldsymbol{\lambda}^*(\mathcal{C})$ の情報を利用することで入手する．詳細に関しては 5.2 節で述べる．

我々の提案アルゴリズムを以下に記述する．ただし，k_{max} は $(SP(\mathcal{C}))$ を連続して解く回数を表している．このアルゴリズムを，*Single-Cutting-plane-at-a-Time Algorithm* (SCP) と呼ぶことにする．

Single-Cutting-Plane-at-a-Time Algorithm (SCP)

Step 0
　$\mathcal{C}$ を V の非空な部分集合の初期部分族とする．
　自然数 k_{max} を決定し，$LB \leftarrow -\infty$，$k \leftarrow 0$ と設定する．
Step 1
　最適解 $\boldsymbol{\lambda}^*(\mathcal{C})$ と最適値 $\omega(RD(\mathcal{C}))$ を得るために $(RD(\mathcal{C}))$ を解く．
Step 2
　元問題 (P) の実行可能解を構成し，その目的関数値を $\ell(P)$ とする．(詳細に関しては，5.2 節を見よ.)
　if $\ell(P) > LB$ then $LB \leftarrow \ell(P)$.
Step 3
　$(SP(\mathcal{C}))$ に発見的解法を適用する．(詳細に関しては，5.1 節を見よ.)
　発見的解法によって得られた解を $\boldsymbol{y}^*$，その目的関数値を $\alpha(SP(\mathcal{C}))$ とする．
Step 4
　if $\alpha(SP(\mathcal{C})) \geq 0$ then
　　$k \leftarrow k + 1$.
　else
　　$C \leftarrow \{ i \in V \mid y_i^* = 1 \}$ と設定し，$\mathcal{C} \leftarrow \mathcal{C} \cup \{C\}$，$k \leftarrow 0$ と更新する．Step 1 へ戻る．
　end if
Step 5
　if $k > k_{max}$ then
　　$\mathcal{C}^* \leftarrow \mathcal{C}$，$\omega^* \leftarrow \omega(RD(\mathcal{C}))$ と設定し，$\mathcal{C}^*$ と ω^* を出力する．アルゴリズム終了．
　else
　　Step 3 へ戻る．
　end if

SCP を用いたいくつかの予備実験を通して，$\mathcal{C}$ が更新されても，最適値 $\omega(RD(\mathcal{C}))$ が一定であるという現象が頻繁に観察された．SCP のアルゴリズムが進行しても $\omega(RD(\mathcal{C}))$ の収束が非常に遅い現象を，表 1

表 1: *SCP* による実験結果

iteration	$\omega(RD(\mathcal{C}))$		
	Dolphins	Football	Jazz
1	-0.0213	-0.0087	-0.0070
500	0.4241	0.4430	0.3276
1000	0.4241	0.4470	0.3276
2000	0.4241	0.4549	0.3276
3000	0.4241	0.4587	0.3276
4000	0.4241	0.4605	0.3276
5000	0.4241	0.4605	0.3276
6000	0.4241	0.4609	0.3276

表 2: 問題例

name	n	m	$\omega(P)$
Dolphins	62	159	0.5285
Football	115	613	0.6045
Jazz	198	2742	0.4448

に示す．“Dolphins” と “Jazz” に関しては，各 500 反復ごとに $\omega(RD(\mathcal{C}))$ を見ても，全く変化が無いことが観察される．なお，実験に用いたインスタンス “Dolphins”, “Football” と “Jazz” は，以下のサイトから入手することができる．

`http://www.cc.gatech.edu/dimacs10/archive/clustering.shtml`

各インスタンスの問題サイズと既知の最適値 $\omega(P)$ を，表 2 に掲載する．

この収束の遅さは，問題 $(RD(\mathcal{C}))$ の特殊な構造に原因があると考えられる．$(RD(\mathcal{C}))$ の目的関数の係数はすべて 1 であり，また制約条件の係数行列の値は 0 もしくは 1 のみである．この特殊な構造によって，目的関数の等高線と実行可能領域 $\mathcal{F}(RD(\mathcal{C}))$ のある面（face）が平行になることで，その面全体が最適解となりうる．その結果，大量の切除平面が追加されるにも関わらず，$\omega(RD(\mathcal{C}))$ が一定になってしまう．そこで，そのような面全体を切り落とすために，お互いに補完し合うような複数の切除平面を同時に追加する方法を提案する．まず，1 本目の切除平面は，これまでと同様に $(SP(\mathcal{C}))$ を解いて得られる $\boldsymbol{y}^*$ と f^* によって定義されるものを生成する．その後，$y_i^* = 1$ となっているすべての i に関して $y_i = 0$ と固定して，$(SP(\mathcal{C}))$ を考える．より詳細に述べると，$V^{(1)} = \{ i \in V \mid y_i^* = 1 \}$ と定義し，2 本目の切除平面を生成するために，すなわち $\boldsymbol{y}^{(1)}$ と $f^{(1)}$ を得るために，以下の問題 $(SP(\mathcal{C}, V^{(1)}))$ を解く．

$$
(SP(\mathcal{C}, V^{(1)})) \quad \left| \begin{array}{lll} \text{minimize} & \displaystyle\sum_{i \in V} \lambda_i^*(\mathcal{C}) y_i - \frac{1}{m} \sum_{i \in V} \sum_{j \in V; i<j} w_{ij} y_i y_j & \\ \text{subject to} & y_i \in \{0,1\} & (\forall i \in V \setminus V^{(1)}) \\ & y_i = 0 & (\forall i \in V^{(1)}). \end{array} \right.
$$

これを一般化し，$V^{(h)} = \{ i \in V \mid y_i^{(l)} = 1 \text{ for some } l < h \}$ として，以下の問題を繰り返し解く．ただし，$\boldsymbol{y}^{(0)} = \boldsymbol{y}^*$, $V^{(0)} = \emptyset$ である．

$$
(SP(\mathcal{C}, V^{(h)})) \quad \left| \begin{array}{lll} \text{minimize} & \displaystyle\sum_{i \in V} \lambda_i^*(\mathcal{C}) y_i - \frac{1}{m} \sum_{i \in V} \sum_{j \in V; i<j} w_{ij} y_i y_j & \\ \text{subject to} & y_i \in \{0,1\} & (\forall i \in V \setminus V^{(h)}) \\ & y_i = 0 & (\forall i \in V^{(h)}). \end{array} \right.
$$

この問題を制限付き分離問題と呼ぶことにする．$\alpha(AP(\mathcal{C}, V^{(h)}))$ が負である限りは，h をインクリメントし $(SP(\mathcal{C}, V^{(h)}))$ を解いて切除平面の生成を続ける．そして，$\alpha(AP(\mathcal{C}, V^{(h)}))$ が非負となった時点で，それまでに生成した切除平面群を $\mathcal{C}$ に追加する．アルゴリズム *SCP* の Step 4 が以下のように修正される．このアルゴリズムを，*Multiple-Cutting-Planes-at-a-Time Algorithm*(MCP) と呼ぶことにする．

Step 4 of the Multiple-Cutting-Planes-at-a-Time Algorithm (MCP)

Step 4
  if $\alpha(SP(\mathcal{C})) \geq 0$ then
    $k \leftarrow k+1$.
  else
    $\alpha(AP(\mathcal{C}, V^{(h)}))$ が非負となるまで，切除平面を生成する．
    生成した切除平面群を $\mathcal{C}$ に追加する．$k \leftarrow 0$ と設定し，Step 1 へ戻る．
  end if

---

5 発見的解法

この節では，SCP および MCP の内部で用いる発見的解法について紹介する．まず，両アルゴリズムの Step 3 において ($SP(\mathcal{C})$) に対して使用する発見的解法について説明し，次に LP 緩和問題 ($RD(\mathcal{C})$) の最適解から元問題 (P) の実行可能解を構成する発見的解法について説明する．

5.1 評価関数摂動法

提案アルゴリズムでは ($SP(\mathcal{C})$) に対して，Charon and Hudry [6, 7] によって提案された評価関数摂動法 (Noising method) を使用する．

まず，評価関数摂動法の内部で用いる近傍探索法について説明する．($SP(\mathcal{C})$) の実行可能解 $\boldsymbol{y}$ が与えられたときに，その近傍 $N(\boldsymbol{y})$ を以下で定義する．

$$N(\boldsymbol{y}) = \{\, \boldsymbol{y}' \mid \|\boldsymbol{y}' - \boldsymbol{y}\|_1 \leq 1 \,\}.$$

上の近傍の定義の下では，$N(\boldsymbol{y})$ の任意の要素は，ある要素 y_q を $1 - y_q$ に置き換えることで得られる．ここで，$N(\boldsymbol{y})$ の要素を $\boldsymbol{y}^{(q)}$ と書くことにする．$\boldsymbol{y}$ を $\boldsymbol{y}^{(q)}$ に置き換えたときの，($SP(\mathcal{C})$) の目的関数の差分 $v(\boldsymbol{y}, \boldsymbol{y}^{(q)})$ は，次で与えられる．

$$v(\boldsymbol{y}, \boldsymbol{y}^{(q)}) = (1 - 2y_q)\left(\lambda_q^*(\mathcal{C}) - \frac{1}{m}\sum_{i \in V \setminus \{q\}} w_{iq} y_i\right). \tag{5.1}$$

差分 (5.1) は $N(\boldsymbol{y})$ の各要素に対して計算される．もし，それらの差分がすべて非負であった場合は，近傍探索を停止し，局所的最適解として $\boldsymbol{y}$ を出力する．そうでなければ，$N(\boldsymbol{y})$ 上で $v(\boldsymbol{y}, \boldsymbol{y}^{(q)})$ を最小にする解に移動する．以上をまとめると，近傍探索法は以下のように記述される．

Conventional Local Search

Step 0
  ($SP(\mathcal{C})$) の初期実行可能解を $\boldsymbol{y}$ とする．
Step 1
  for $q = 1$ to n do
    $v(\boldsymbol{y}, \boldsymbol{y}^{(q)})$ を計算する．

Step 2
$v^* \leftarrow \min\{v(\boldsymbol{y}, \boldsymbol{y}^{(q)})\}$, $\boldsymbol{y}^* \leftarrow \mathrm{argmin}\{v(\boldsymbol{y}, \boldsymbol{y}^{(q)})\}$ と設定する.
if $v^* \geq 0$ then
$\boldsymbol{y}$ を出力し, アルゴリズム終了.
else
$\boldsymbol{y} \leftarrow \boldsymbol{y}^*$ と設定し, Step 1 へ戻る.
end if

上の近傍探索法では, 局所的最適解に陥る危険性があることがよく知られている. この欠点を克服するために, Charon and Hudry は評価関数摂動法と呼ばれる手法を提案した. この手法は近傍探索法を基礎としているが, ノイズと呼ばれる乱数を加えることで目的関数を摂動させるという点において, 通常の近傍探索法と異なる.

ここから, 評価関数摂動法について説明する. まず, 初期実行可能解 $\boldsymbol{y}$ を与え, 近傍探索法によって $N(\boldsymbol{y})$ 内で最も良い解 $\boldsymbol{y}^*$ を見つける. その後, 区間 $[-r, r]$ からランダムに選んだ ρ を差分 $v(\boldsymbol{y}, \boldsymbol{y}^*)$ に加えることで, 次の摂動された差分 $\tilde{v}(\boldsymbol{y}, \boldsymbol{y}^*)$ を計算する.

$$\tilde{v}(\boldsymbol{y}, \boldsymbol{y}^*) = v(\boldsymbol{y}, \boldsymbol{y}^*) + \rho.$$

もし, $\tilde{v}(\boldsymbol{y}, \boldsymbol{y}^*)$ が負であったならば, $\boldsymbol{y}$ から $\boldsymbol{y}^*$ へ移動する. 予め定めた回数だけ反復を終えた後に, 区間 $[-r, r]$ を $[-r + d, r + d]$ へと縮小させる. そして, この区間が $\{0\}$ になったときに, アルゴリズムは停止する. 以下に, 評価関数摂動法のアルゴリズムを記述する.

Noising Method

Step 0
$\boldsymbol{y}$ を初期実行可能解とする.
$r \leftarrow 100$, $d \leftarrow 1$, FixedIter $\leftarrow 10$, $t \leftarrow 0$ と設定する.
Step 1
$t \leftarrow t + 1$ と設定し, $N(\boldsymbol{y})$ における局所最適解 $\boldsymbol{y}^*$ を得るために近傍探索法を呼び出す.
Step 2
ノイズ ρ を区間 $[-r, r]$ からランダムに選び, $\tilde{v}(\boldsymbol{y}, \boldsymbol{y}^*)$ を計算する.
if $\tilde{v}(\boldsymbol{y}, \boldsymbol{y}^*) < 0$ then $\boldsymbol{y} \leftarrow \boldsymbol{y}^*$ と設定する.
Step 3
if $t \equiv 0 \pmod{\text{FixedIter}}$ then $r \leftarrow r - d$ と設定し, Step 1 へ戻る.
Step 4
if $r = 0$ then
$\boldsymbol{y}$ を出力し, アルゴリズム終了.
else
Step 1 へ戻る.
end if

5.2 ラウンディング法

ここでは，SCP および MCP の各反復において得られる LP 緩和問題 $(RD(\mathcal{C}))$ の最適解の情報を利用して，元問題 (P) の実行可能解を構成する発見的解法について説明する．

SCP，MCP の各反復の Step 1 において，$(RD(\mathcal{C}))$ の最適解が得られる．この最適解 $\boldsymbol{\lambda}^*(\mathcal{C})$ を用いて，以下に示す相補性条件と呼ばれる不等式系を解くことで，主問題 $(RP(\mathcal{C}))$ の最適解 $z^*(\mathcal{C})$ を得ることができる．

$$z_C(\mathcal{C})\left(\sum_{C\in\mathcal{C}} a_{iC}\lambda_i^*(\mathcal{C}) - f_C\right) = 0 \quad (\forall C \in \mathcal{C}),$$
$$\lambda_i^*(\mathcal{C})\left(1 - \sum_{C\in\mathcal{C}} a_{iC} z_C(\mathcal{C})\right) = 0 \quad (\forall i \in V).$$

不等式系を解いて得られる $z^*(\mathcal{C})$ は，あくまでも LP 緩和問題 $(RP(\mathcal{C}))$ の最適解なので整数解である保証はない．そこで，$z^*(\mathcal{C})$ から元問題 (P) の実行可能解を構成することを考える．まず $z^*(\mathcal{C})$ を用いて，以下のルールによって $\bar{z} = (\bar{z}_C)_{C\in\mathcal{C}}$ を構成する．

$$\bar{z}_C = \begin{cases} 1 & (z_C^*(\mathcal{C}) > 0) \\ 0 & (z_C^*(\mathcal{C}) = 0). \end{cases} \tag{5.2}$$

上で構成した $\bar{z}$ の表す頂点集合の部分族は，V の被覆となる [13]．ただし，頂点集合の部分族 $\{C_1, C_2, \ldots, C_k\}$ が，$V = \cup_{p=1}^{k} C_p$，任意の $p \in \{1, 2, \ldots, k\}$ に対して $C_p \neq \emptyset$ を満たすとき V の被覆であるという．したがって，$\bar{z}$ から得られる V の部分族においては，複数のコミュニティに属する頂点が存在する可能性がある．そこで，そのような頂点をどこか 1 つのコミュニティに属するような操作を施すことで，頂点集合 V の分割を構成する．これによって，(P) の実行可能解が得られ，したがって $\omega(P)$ の下界が得られる．

6 計算機実験

ここでは，提案アルゴリズムの性能を計算機実験によって評価する．実験は，CPU：Intel Core2 Duo, 3.06GHz processor，メモリ：4.0GB の計算機上で行った．また，アルゴリズムは Java により実装し，LP および IP ソルバーとして CPLEX 12.3 を用いている．求解した問題は，4 節で紹介した 3 つのインスタンスである．

評価関数摂動法において乱数を使用してため試行によって結果が異なる可能性があるので，各インスタンスに対して 3 回アルゴリズムを実行している．初期の部分族 $\mathcal{C}$ は，すべての 1 点集合を集めた族，すなわち $\{\{1\}, \{2\}, \ldots, \{n\}\}$，に設定している．また，パラメータ k_{max} は 30 に設定している．

$\lvert\mathcal{C}^\star\rvert$	アルゴリズム終了時に得られる $\mathcal{C}^\star$ の要素数
$\omega(RD(\mathcal{C}^\star))$	アルゴリズム終了時に得られる $(RD(\mathcal{C}^\star))$ の最適値
LB	アルゴリズム終了時に得られる (P) の下界値
$\omega(P(\mathcal{C}^\star))$	$(P(\mathcal{C}^\star))$ の最適値
gap	相対誤差．gap $= \left(\frac{\omega(P)-\omega(P(\mathcal{C}^\star))}{\omega(P)}\right) \times 100$ によって計測
time	計算時間（秒）

表 3: MCP の計算結果

instance	exec.	$\|\mathcal{C}^\star\|$	$\omega(RD(\mathcal{C}^\star))$	LB	$\omega(P(\mathcal{C}^\star))$	gap (%)	time (s)
Dolphins	1	801	0.5285	0.5285	0.5285	0.000	13
	2	832	0.5285	0.5285	0.5285	0.000	12
	3	818	0.5285	0.5285	0.5285	0.000	11
Football	1	4699	0.6045	0.6045	0.6045	0.000	269
	2	4378	0.6045	0.6045	0.6045	0.000	242
	3	4628	0.6045	0.6045	0.6045	0.000	287
Jazz	1	22447	0.4441	0.4441	0.4441	0.157	13806
	2	23510	0.4442	0.4441	0.4441	0.157	14614
	3	24383	0.4442	0.4442	0.4442	0.134	18005

表 3 を見ると，Dolphins と Football に関しては MCP によって最適に解かれていることが確認される．ただし，MCP は最適性の保証を与えるわけではない．Jazz に関しては，最適解を得ることはできていないが，いずれの試行においても相対誤差は 0.2% 以下となっている．

また生成された制約の個数 $|\mathcal{C}^\star|$ は，元問題の制約の個数よりもはるかに少なくなっていることが確認される．例えば，頂点数 $n = 62$ の Dolphins に関しては，生成された制約式の個数は，元問題の制約式の総数 4.6×10^{18} の約 $1/10^{15}$ 以下となっている．

図 1,2 および 3 は，SCP と MCP の挙動を比較したものである．横軸は LP 緩和問題 ($RD(\mathcal{C})$) を解いた回数，縦軸はその目的関数値を表している．また SCP に関しては，MCP によって生成された切除平面の個数と同じ数の切除平面が生成された時点でアルゴリズムを停止している．どちらのアルゴリズムも，早い段階で $\omega(RD(\mathcal{C}))$ が急激に上昇し，アルゴリズムが進行するにつれてゆっくりとした変化となっている．しかし MCP は，SCP と比較して非常に高速に最適値へと収束していることが確認される．

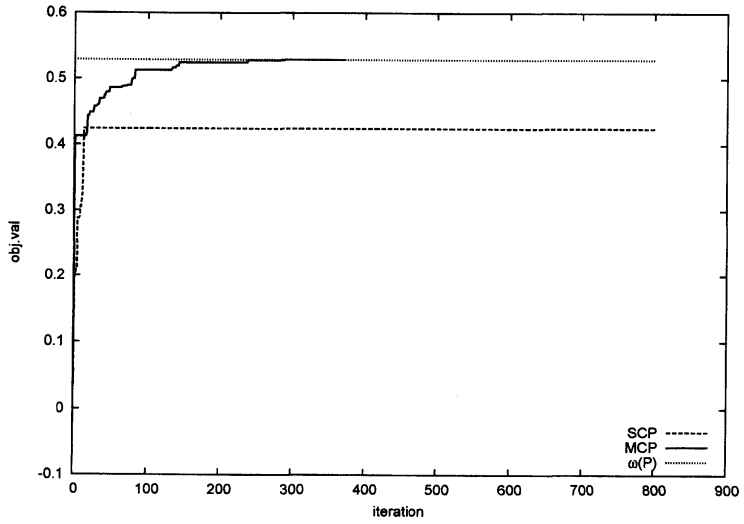

図 1: Dolphins に対する SCP と MCP の挙動

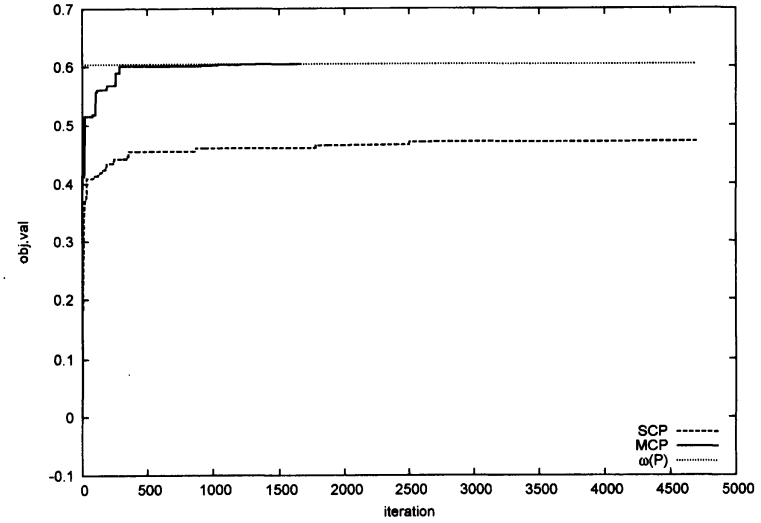

図 2: Football に対する SCP と MCP の挙動

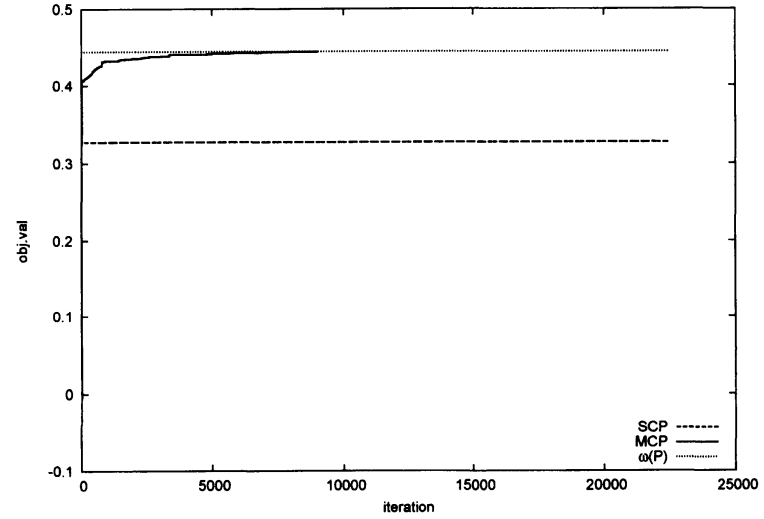

図 3: Jazz に対する SCP と MCP の挙動

7 おわりに

本稿では，モジュラリティ最大化問題に対して，切除平面法に基づくアルゴリズムを提案した．計算機実験の結果，我々の提案した MCP は，SCP に比べて少ない切除平面の生成によって最適解へと収束することが確認され，それに伴う計算時間の短縮も見られた．

今後の課題として，以下の点が挙げられる．1つは，多面体をより深く切り取る切除平面の生成である．本研究では，制約の違反度を指標として切除平面を生成しているが，それ以外の指標に基づく生成法が考えられる．また，元問題 (P) の下界を得るための発見的解法に関しても，さらなる改良の余地があると考えられる．

参考文献

[1] G. Agarwal and D. Kempe, "Modularity-maximizing graph communities via mathematical programming," *The European Physical Journal*, B.66, pp.409-418, 2008.

[2] D. Aloise, S. Cafieri, G. Caporossi, P. Hansen, L. Liberti, and S. Pellon, "Column generation algorithms for exact modularity maximization in networks," *Physical Review*, E.82, 2012.

[3] U. Brandes, D. Delling, M. Gaertler, R. Gorke, M. Hoefer, Z. Nikoloski, and D. Wagner, "On modularity clustering," *IEEE Transactions on Knowledge and Data Engineering*, 20, pp.172-188, 2008.

[4] A. Caprara, M. Fischetti, and P. Toth, "Heuristic method for the set covering problem," *Operations Research*, 47, pp.730-743, 1999.

[5] A. Caprara, P. Toth, and M. Fischetti, "Algorithms for the set covering problem," *Annals of Operations Research*, 98, pp.353-371, 2000.

[6] I. Charon and O. Hudry, "Application of the noising method to the travelling salesman problem," *European Journal of Operations Research*, 125, pp.266-277, 2000.

[7] I. Charon and O. Hudry, "The noising methods: A generalization of some metaheuristics," *European Journal of Operations Research*, 135, pp.86-101, 2001.

[8] A.Clauset, M.Newman and C.Moore, "Finding community structure in very large networks," *Physical Review*, E.70, 2004.

[9] O. du Merle, D. Villeneuve, J. Desrosiers, and P. Hansen, "Stabilized column generation," *Discrete Mathematics*, 194, pp.229-237, 1999.

[10] M. Grötschel and Y. Wakabayashi, "A cutting plane algorithm for a clustering problem," *Mathematical Programming*, 45, pp.59-96, 1989.

[11] M. E. J. Newman and M. Girvan, "Finding and evaluating community structure in networks," *Physical Review*, E.69, 2004.

[12] M. E. J. Newman, "Modularity and community structure in networks," *Proceedings of the National Academy of Science*, pp.8577-8582, 2006.

[13] S. Umetani and M. Yagiura, "Relaxation heuristics for the set covering problem," *Journal of the Operations Research Society of Japan*, 50, pp.350-375, 2007.

[14] G.Xu, S.Tsoka and L.Papageorgiou, "Finding community structures in complex networks using mixed integer optimization," *The eurpean Physical Journal*, B.50, pp.231-239, 2007.