数理解析研究所講究録
第1931巻 2015年 1-17

モジュラリティ最大化問題に対するラグランジュ緩和法

稲葉 言史 *　　伊豆永 洋一 †　　山本 芳嗣 ‡

概 要

グラフの構造からリンクが密な部分グラフを抽出することをコミュニティ抽出と呼ぶ．コミュニティ抽出を評価する有用な指標としてNewmanとGirvanが提案したモジュラリティがある．我々はモジュラリティを最大化する問題を集合分割問題に定式化し，ラグランジュ緩和に基づくアルゴリズムを提案する．また，計算負荷を軽減するために，アルゴリズムに列生成法を適用する．

1 はじめに

ソーシャルネットワークサービスが普及したことにより，グラフ上のクラスタリングが注目を集めている．NewmanとGirvan [17] が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を提案して以来，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一つである．モジュラリティ最大化問題はNP-困難 [5] であるため，様々な発見的解法が提案されてきた．例えば，AgarwalとKempeのラウンディング法 [1]，Clausetらの階層凝集法 [7]，GuimeràとAmaralの焼きなまし法 [14] などがある．このような背景から，最適なモジュラリティの上界を求めることは非常に重要である．なぜならば，最適なモジュラリティが不明なネットワークに対してその上界を求めることができれば，発見的解法で得られた解の精度を評価することができるからである．MiyauchiとMiyamoto [16] はモジュラリティ最大化問題をクリーク分割問題に定式化し，その線形計画緩和問題を解くことで上界を求めるアルゴリズムを提案した．彼らのアルゴリズムはGrötschelとWakabayashi [13] の切除平面法に基づくものであり，最大で頂点数が4941のインスタンスの上界を求めることに成功した．

一方厳密解法に関する研究は，大きく分けて以下の二つがある．一つ目は，整数二次計画問題に定式化する方法である．この定式化は，非常に強い対称性を持っており，多数の等価な解が存在する．その結果，探索空間が拡大し，多くの計算時間を要する．Xuら [19] は，対称性を取り除く妥当不等式を導入することにより問題を解いている．二つ目は，集合分割問題に定式化する方法である．この定式化は，頂点集合のすべての空でない部分集合を考慮するので，頂点数の指数オーダーの変数を必要とする．そのため，頂点数が増加した場合には，問題を保持するための計算機資源を確保することさえ困難となる．この種の問題に対する有効な解法に列生成法があるが，モジュラリティ最大化問題に列生成を適用すると，追加すべき列を決定する補助問題が整数制約の課された二次計画問題に定式化されてしまう．また，列生成法は非常に収束が遅いことが知られている．Aloiseら [2] は，補助問題に対してVNS [10] と呼ばれる発見的解法を適用し，さらにdu Merleらによる列生成法の高速化手法 [9] を利用することで，頂点数が500程度のインスタンスの最適解を求めている．

*筑波大学大学院 システム情報工学研究科，s1320486@sk.tsukuba.ac.jp
†筑波大学大学院 システム情報工学研究科，s1130131@sk.tsukuba.ac.jp
‡筑波大学 システム情報系，yamamoto@sk.tsukuba.ac.jp

本稿では，集合分割問題による定式化に対して，ラグランジュ緩和と列生成法を組合わせたアルゴリズムを提案する．さらに，アルゴリズムの各反復において最適なモジュラリティの上界値と下界値を見積もるための手法を紹介する．

本稿の構成は以下のとおりである．2節ではモジュラリティの定義を与えた後，モジュラリティ最大化問題を紹介する．3節では，モジュラリティ最大化問題の異なる二つの定式化を紹介する．4節では，LP緩和問題とラグランジュ緩和問題に関する概要を説明する．5節ではラグランジュ緩和問題に列生成法を適用したアルゴリズムを説明する．6節では，上界値と下界値を見積もる手法について説明し，提案アルゴリズムの全体像の記述を行う．7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 m 本の無向枝からなる枝集合 E で構成される無向グラフとする．頂点集合 V の部分集合の族 $\Pi=\{C_1,C_2,\ldots,C_k\}$ が以下を満たすとき V の分割と呼ぶ．

$$
\begin{aligned}
&V=\cup_{p=1}^{k}C_p,\\
&\text{任意の相異なる } p,q\in\{1,2,\ldots,k\} \text{ に対して } C_p\cap C_q=\emptyset,\\
&\text{任意の } p\in\{1,2,\ldots,k\} \text{ に対して } C_p\neq\emptyset.
\end{aligned}
$$

以降では，分割の要素 C_p をコミュニティと呼ぶ．また枝の両端点が C_p に属すような枝の部分集合を $E(C_p)$ と表記する．このとき，分割 Π に対するモジュラリティ $Q(\Pi)$ は以下で定義される．

$$
Q(\Pi)=\sum_{p=1}^{k}\left(\frac{|E(C_p)|}{m}-\left(\frac{\sum_{i\in C_p}d_i}{2m}\right)^2\right),
$$

ただし，$|\cdot|$ は対応する集合の要素数，d_i は頂点 i の次数を表す．ここで，グラフ G の隣接行列の (i,j) 成分を e_{ij}，頂点 i の属するコミュニティの添字を $\pi(i)$，δ をクロネッカーデルタとすると，モジュラリティは以下のように書く事ができる．

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

任意の分割 Π に対して，モジュラリティ $Q(\Pi)$ は $-1/2$ 以上 1 以下の値をとることが知られている (Brandesら [5])．モジュラリティ最大化問題とは，$Q(\Pi)$ を最大にするような分割 Π を求める問題である．

$$
(MM)\quad\left|\begin{array}{ll}\text{最大化} & \dfrac{1}{2m}\displaystyle\sum_{i\in V}\sum_{i\in V}\left(e_{ij}-\frac{d_id_j}{2m}\right)\delta(\pi(i),\pi(j))\\ \text{制約} & \Pi\text{は}V\text{の分割.}\end{array}\right.
$$

3 定式化

本節では，モジュラリティ最大化問題に対する二つの定式化を紹介する．一つ目の定式化は整数線形計画問題によるもの，二つ目の定式化は整数二次計画問題によるものである．

3.1 整数線形計画問題による定式化

まず$\mathcal{P}$をVの空でない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．任意の$C\in\mathcal{P}$に対して0-1変数z_Cを以下で定義する．

$$z_C=\begin{cases}1 & (C\in\Pi)\\ 0 & (C\notin\Pi).\end{cases}$$

また各$i\in V$と$C\in\mathcal{P}$に対して，定数a_{iC}を頂点iがコミュニティCに含まれれば1，そうでなければ0と定義すると，ベクトル$\boldsymbol{a}_C=(a_{1C},\ldots,a_{nC})^\top$はコミュニティ$C$のインシデンスベクトルとなる．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

$$f_C=\frac{1}{2m}\sum_{i\in V}\sum_{j\in V}\left(e_{ij}-\frac{d_id_j}{2m}\right)a_{iC}a_{jC}, \tag{3.1}$$

とする．定数f_CはコミュニティCが分割の要素に選ばれたときの目的関数へのCの貢献度を表している．以上の定義の下で，モジュラリティ最大化問題は以下の集合分割問題として定式化される．

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

以降では，(P)の一つ目の制約条件を集合分割制約と呼ぶこととする．この問題は頂点集合のすべての空でない部分集合を考慮しているので，頂点数nが増加するにつれ問題を保持するための計算資源を確保することすら困難となる．

3.2 整数二次計画問題による定式化

通常，最適なモジュラリティにおけるコミュニティの個数を事前に知る事はできない．しかしここでは，与えられた定数tに対して，頂点集合Vをt個以下のコミュニティに分割する問題を考える．ただし定数tは2以上の整数とする．まず$T=\{1,2,\ldots,t\}$とし，各枝$l=\{i,j\}\in E$と$p\in T$に対して，0-1変数x_{lp}を以下のように定義する．

$$x_{lp}=\begin{cases}1 & (l\in E(C_p))\\ 0 & (l\notin E(C_p)).\end{cases}$$

また頂点$i\in V$と$p\in T$に対して，0-1変数y_{ip}を次で定義する．

$$y_{ip}=\begin{cases}1 & (i\in C_p)\\ 0 & (i\notin C_p).\end{cases}$$

このとき，個数制限の課されたモジュラリティ最大化問題は以下の二次計画問題として定式化される．

$$
(QP)\quad
\left|\begin{array}{ll}
\text{最大化} & \displaystyle\sum_{p\in T}\left(\frac{1}{m}\left(\sum_{l\in E}x_{lp}\right)-\frac{1}{4m^2}\left(\sum_{i\in V}d_iy_{ip}\right)^2\right)\\
\text{制約} & \displaystyle\sum_{p\in T}y_{ip}=1 \quad (\forall i\in V)\\
& x_{lp}\le y_{ip} \quad (\forall l=\{i,j\}\in E,\forall p\in T)\\
& x_{lp}\le y_{jp} \quad (\forall l=\{i,j\}\in E,\forall p\in T)\\
& x_{lp}\in\{0,1\} \quad (\forall l\in E,\forall p\in T)\\
& y_{ip}\in\{0,1\} \quad (\forall i\in V,\forall p\in T).
\end{array}\right.
$$

一つ目の制約は各頂点が必ず一つのコミュニティに属すことを表し，残りの制約は枝 l の両端点 i,j がコミュニティ C_p に属したときに限り，$l\in E(C_p)$ となることを表している．この定式化は，非常に強い対称性を持っている．つまり，コミュニティの添字の付け変えによって全く等価な解を構成することができる．このような対称な解の存在によって，分枝限定法における限定操作が上手く働かず，結果として非常に計算時間が長くなってしまう．Xu ら [19] は，対称な解を除去するためのいくつかの妥当不等式を提案している．定数 t として最適なコミュニティ数よりも大きな値を設定すると，明らかに問題 (QP) は問題 (P) と等価である．本稿では，(P) の最適値の自明でない上界を求めるために (QP) を利用する．詳細については 6.1 節で述べる．

4 緩和問題

本節では，問題 (P) に対する線形計画緩和（LP 緩和）とラグランジュ緩和を紹介し，これらの緩和問題の間の関係について述べる．

4.1 線形計画緩和

問題 (P) に対する LP 緩和問題 (RP) は，変数 z_C に関する 0-1 制約を $0\le z_C\le 1$ と緩和することによって得られる．

$$
(RP)\quad
\left|\begin{array}{lll}
\text{最大化} & \displaystyle\sum_{C\in\mathcal{P}}f_Cz_C & \\
\text{制約} & \displaystyle\sum_{C\in\mathcal{P}}a_{iC}z_C=1 & (\forall i\in V)\\
& z_C\ge 0 & (\forall C\in\mathcal{P})
\end{array}\right.
$$

ただし，変数の上限制約 $z_C\le 1$ は集合分割制約により冗長となる．(RP) の双対問題 (RD) は以下で与えられる．

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

以降では，問題 $(\cdot)$ の実行可能領域と最適値をそれぞれ $\mathcal{F}(\cdot)$ と $\omega(\cdot)$ と表すこととする．(RP) は (P) の緩和問題なので $\omega(P)$ の上界を与える．また線形計画問題における強双対定理から $\omega(RP)=\omega(RD)$

という関係が成立するので，(RP) または (RD) を解くことにより $\omega(P)$ の上界を得ることができる．しかし，(RP) と (RD) は，それぞれ指数オーダーの変数と制約を持つ問題であるので依然として計算困難な問題である．

4.2 ラグランジュ緩和

前節で述べたとおり LP 緩和問題は元問題 (P) の最適値の上界を与え，得られる上界は非常にタイトであることが知られている．しかし，集合分割制約の存在のために問題が退化現象に陥ってしまう．この退化を克服するために様々な手法が提案されている [3, 4, 9]．本稿では，問題 (P) にラグランジュ緩和を適用することで最適解に関する有用な情報を得る．

まず，ラグランジュ乗数 $\boldsymbol{\lambda} = (\lambda_1, \lambda_2, \ldots, \lambda_n) \in \mathbb{R}^n$ を導入し，集合分割制約をペナルティとして目的関数に加えると，0-1 制約だけを持つ以下のラグランジュ緩和問題 $(LR(\mathcal{P}, \boldsymbol{\lambda}))$ を得る．

$$(LR(\mathcal{P}, \boldsymbol{\lambda})) \quad \left| \begin{array}{ll} \text{最大化} & \displaystyle\sum_{C \in \mathcal{P}} f_C z_C + \sum_{i \in V} \lambda_i (1 - \sum_{C \in \mathcal{P}} a_{iC} z_C) \\ \text{制約} & z_C \in \{0,1\} \qquad (\forall C \in \mathcal{P}). \end{array} \right.$$

目的関数を簡潔に表現するために，目的関数における変数 z_C の係数を

$$\gamma_C(\boldsymbol{\lambda}) = f_C - \sum_{i \in V} \lambda_i a_{iC},$$

と定義すると，$(LR(\mathcal{P}, \boldsymbol{\lambda}))$ の目的関数は以下のように書ける．

$$L(\boldsymbol{z}, \boldsymbol{\lambda}) = \sum_{C \in \mathcal{P}} \gamma_C(\boldsymbol{\lambda}) z_C + \sum_{i \in V} \lambda_i.$$

任意の $\boldsymbol{\lambda} \in \mathbb{R}^n$ に対して，問題 $(LR(\mathcal{P}, \boldsymbol{\lambda}))$ の最適解 $\boldsymbol{z}(\boldsymbol{\lambda}) = (z_C(\boldsymbol{\lambda}))_{C \in \mathcal{P}}$ は，$\gamma_C(\boldsymbol{\lambda}) > 0$ ならば $z_C(\boldsymbol{\lambda}) = 1$，そうでなければ $z_C(\boldsymbol{\lambda}) = 0$ と設定することで簡単に求める事ができる．そのとき，最適値 $\omega(LR(\mathcal{P}, \boldsymbol{\lambda}))$ は任意の $\boldsymbol{\lambda}$ に対して $\omega(P)$ の上界を与える．問題 $(LR(\mathcal{P}, \boldsymbol{\lambda}))$ によって与えられる上界は，ラグランジュ乗数の選び方によって大きく異なる．そこで最も良い上界を求める問題をラグランジュ双対問題と呼び，(LD) と表記する．

$$(LD) \quad \left| \begin{array}{ll} \text{最小化} & \omega(LR(\mathcal{P}, \boldsymbol{\lambda})) \\ \text{制約} & \boldsymbol{\lambda} \in \mathbb{R}^n. \end{array} \right.$$

問題 (LD) の目的関数 $\omega(LR(\mathcal{P}, \boldsymbol{\lambda}))$ は，$\boldsymbol{\lambda} \in \mathbb{R}^n$ に関する区分線形凸関数であり区分点において微分可能でない．このような問題に対する良く知られた解法として劣勾配法がある．アルゴリズムの詳細は，5.1 節で説明する．

ここから，問題 (RD) と問題 (LD) の理論的な関係について述べる．ラグランジュ緩和問題 $(LR(\mathcal{P}, \boldsymbol{\lambda}))$ を LP 緩和した問題、つまり整数制約 $z_C \in \{0,1\}$ を $0 \leq z_C \leq 1$ を置き換えた問題を考える．この問題を $(\overline{LR}(\mathcal{P}, \boldsymbol{\lambda}))$ と表記する．

$$(\overline{LR}(\mathcal{P}, \boldsymbol{\lambda})) \quad \left| \begin{array}{ll} \text{最大化} & \displaystyle L(\boldsymbol{z}, \boldsymbol{\lambda}) = \sum_{C \in \mathcal{P}} \gamma_C(\boldsymbol{\lambda}) z_C + \sum_{i \in V} \lambda_i \\ \text{制約} & 0 \leq z_C \leq 1 \qquad (\forall C \in \mathcal{P}). \end{array} \right.$$

$(\overline{LR}(\mathcal{P}, \boldsymbol{\lambda}))$ の最適解が $(LR(\mathcal{P}, \boldsymbol{\lambda}))$ の最適解でもあるということは明らかである．この性質は整数性と呼ばれている．したがって (LD) の最適値と (RD) の最適値は一致する．詳細については，Geoffrion [11] を参照してほしい．

5 列生成法

問題 $(LR(\mathcal{P},\boldsymbol{\lambda}))$ の最適解 $\boldsymbol{z}(\boldsymbol{\lambda})$ は係数 $\gamma_C(\boldsymbol{\lambda})$ の符号判定だけで求める事ができるが，係数 $\gamma_C(\boldsymbol{\lambda})$ の個数が莫大であるため，そのすべてを計算する事は実用上困難である．しかし，問題 (P) の最適解において正の値を取る変数の個数は高々n 個である．そこで，列生成法を適用する．

5.1 限定主問題と劣勾配法

まず $\mathcal{P}$ の中から十分小さい部分族 $\mathcal{S}$ を選び，少数の変数だけを持つ限定主問題を以下で定義する．

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

さらに，問題 $(P(\mathcal{S}))$ のラグランジュ緩和問題を $(LR(\mathcal{S},\boldsymbol{\lambda}))$ とする．

$$(LR(\mathcal{S},\boldsymbol{\lambda})) \quad \left| \begin{array}{lll} \text{最大化} & \displaystyle\sum_{C\in\mathcal{S}} \gamma_C(\boldsymbol{\lambda}) z_C + \sum_{i\in V} \lambda_i & \\ \text{制約} & z_C \in \{0,1\} & (\forall C \in \mathcal{S}). \end{array} \right.$$

前節での議論と同様に，乗数 $\boldsymbol{\lambda}$ が与えられたとき，問題 $(LR(\mathcal{S},\boldsymbol{\lambda}))$ の最適解 $\boldsymbol{z}(\mathcal{S},\boldsymbol{\lambda}) = (z_C(\boldsymbol{\lambda}))_{C\in\mathcal{S}}$ は以下のルールによって得られる．

$$z_C(\boldsymbol{\lambda}) = \begin{cases} 1 & (\gamma_C(\boldsymbol{\lambda}) > 0) \\ 0 & (\gamma_C(\boldsymbol{\lambda}) \le 0). \end{cases} \tag{5.1}$$

また，$(P(\mathcal{S}))$ に対するラグランジュ双対問題を $(LD(\mathcal{S}))$ とする．

$$(LD(\mathcal{S})) \quad \left| \begin{array}{ll} \text{最小化} & \omega(LR(\mathcal{S},\boldsymbol{\lambda})) \\ \text{制約} & \boldsymbol{\lambda} \in \mathbb{R}^n. \end{array} \right.$$

ここからは，問題 $(LD(\mathcal{S}))$ を解くために使用する劣勾配法について説明する．

定義 1. $\boldsymbol{g}(\boldsymbol{\lambda}_0) \in \mathbb{R}^n$ が，関数 $\omega(LR(\mathcal{S},\boldsymbol{\lambda}))$ の $\boldsymbol{\lambda}_0$ における劣勾配であるとは，任意の $\boldsymbol{\lambda} \in \mathbb{R}^n$ に対して，$\boldsymbol{g}(\boldsymbol{\lambda}_0)^\top(\boldsymbol{\lambda} - \boldsymbol{\lambda}_0) \le \omega(LR(\mathcal{S},\boldsymbol{\lambda})) - \omega(LR(\mathcal{S},\boldsymbol{\lambda}_0))$ が成立することをいう．

劣勾配法は，現在のラグランジュ乗数 $\boldsymbol{\lambda}$ を劣勾配方向に沿って更新する最急降下法の一種である．次の補題は，ラグランジュ緩和問題の最適解から容易に劣勾配を求める事ができる事を示している．

補題 1. $\boldsymbol{z}(\mathcal{S},\boldsymbol{\lambda}) = (z_C(\boldsymbol{\lambda}))_{C\in\mathcal{S}}$ をラグランジュ緩和問題 $(LR(\mathcal{S},\boldsymbol{\lambda}))$ の最適解とする．このとき，以下で定義される $\boldsymbol{g}(\boldsymbol{\lambda})$ は，$\omega(LR(\mathcal{S},\boldsymbol{\lambda}))$ の $\boldsymbol{\lambda}$ における劣勾配である．

$$\boldsymbol{g}(\boldsymbol{\lambda}) = (1 - \sum_{C\in\mathcal{S}} a_{iC} z_C(\boldsymbol{\lambda}))_{i\in V}.$$

本稿では，ステップ幅 μ の決定にアルミホの規準を用いる．具体的には以下の不等式を満たすように μ を選択する．

$$\omega(LR(\mathcal{S},\boldsymbol{\lambda}-\mu\boldsymbol{g}(\boldsymbol{\lambda})))-\omega(LR(\mathcal{S},\boldsymbol{\lambda}))\leq-\xi\mu\|\boldsymbol{g}(\boldsymbol{\lambda})\|^2, \tag{5.2}$$

ただし，ξ は開区間 $(0,1)$ から選ばれたパラメータである．

Procedure Armijo($\boldsymbol{\lambda},\boldsymbol{g}(\boldsymbol{\lambda})$)

Step 1 : ステップ幅を $\mu\leftarrow 1$ と初期化し，$\tau\leftarrow 0.5$，$\xi\leftarrow 0.01$ と設定する．

Step 2 : μ が不等式 (5.2) を満たすならば，μ を出力し終了．

Step 3 : そうでなければ $\mu\leftarrow\tau\mu$ と設定し，Step 2 へ行く．

劣勾配法を以下に記述する．

Procedure Sub-Gradient($\mathcal{S},\boldsymbol{\lambda}$) (SG($\mathcal{S},\boldsymbol{\lambda}$))

Step 1 : 自然数 k_{max} を決定し，$k\leftarrow 0$ と設定する．

Step 2 : (5.1) に従い $z_C(\boldsymbol{\lambda})$ を決めて $(LR(\mathcal{S},\boldsymbol{\lambda}))$ を解く．
$\boldsymbol{z}(\mathcal{S},\boldsymbol{\lambda})$ と $\omega(LR(\mathcal{S},\boldsymbol{\lambda}))$ をそれぞれ得られた解と最適値とする．

Step 3 : $\boldsymbol{z}(\mathcal{S},\boldsymbol{\lambda})$ から劣勾配 $\boldsymbol{g}(\boldsymbol{\lambda})$ を計算する．
$\|\boldsymbol{g}(\boldsymbol{\lambda})\|<0.001$ であれば，$\boldsymbol{\lambda}$ と $\omega(LR(\mathcal{S},\boldsymbol{\lambda}))$ を出力し終了．

Step 4 : Procedure Armijo($\boldsymbol{\lambda},\boldsymbol{g}(\boldsymbol{\lambda})$) を呼び出しステップ幅 μ を決める．
$\boldsymbol{\lambda}\leftarrow\boldsymbol{\lambda}-\mu\boldsymbol{g}(\boldsymbol{\lambda})$，$k\leftarrow k+1$ と設定する．

Step 5 : $k\geq k_{max}$ であれば，$\boldsymbol{\lambda}$，$\omega(LR(\mathcal{S},\boldsymbol{\lambda}))$ を出力し終了．
そうでなければ Step 2 へ行く．

劣勾配法によって得られる解は，問題 $(LD(\mathcal{S}))$ の最適解であるとは限らない．そこで，劣勾配法によって得られた目的関数値を $\bar{\omega}(LD(\mathcal{S}))$ と書く事にする．

5.2 列生成法

問題 $(LR(\mathcal{S},\boldsymbol{\lambda}))$ において $C\in\mathcal{P}\setminus\mathcal{S}$ に関する変数 z_C は考慮されていないので，最適解 $\boldsymbol{z}(\mathcal{S},\boldsymbol{\lambda})$ は問題 $(LR(\mathcal{P},\boldsymbol{\lambda}))$ の最適解とは限らない．したがって，$\omega(LR(\mathcal{S},\boldsymbol{\lambda}))$ および $\omega(LD(\mathcal{S}))$ が $\omega(P)$ の上界であるという保証はない．もし，任意の $C\in\mathcal{P}\setminus\mathcal{S}$ に対して $\gamma_C(\boldsymbol{\lambda})\leq 0$ が成り立てば，$\boldsymbol{z}(\mathcal{S},\boldsymbol{\lambda})$ は $(LR(\mathcal{P},\boldsymbol{\lambda}))$ の最適解であり，したがって $\omega(LR(\mathcal{S},\boldsymbol{\lambda}))$ および $\omega(LD(\mathcal{S}))$ は $\omega(P)$ の上界であることが結論付けられる．一方で，ある $C\in\mathcal{P}\setminus\mathcal{S}$ が存在して，

$$\gamma_C(\boldsymbol{\lambda})>0$$

を満たすならば，その C を部分族 $\mathcal{S}$ に加える事によって $\omega(LR(\mathcal{S},\boldsymbol{\lambda}))$ を改善できる．すなわち，$\omega(LR(\mathcal{S}',\boldsymbol{\lambda}))>\omega(LR(\mathcal{S},\boldsymbol{\lambda}))$ が成り立つ．ただし $\mathcal{S}'=\mathcal{S}\cup\{C\}$ である．ここで，$\boldsymbol{\lambda}$ は $(LD(\mathcal{S}))$ に対する近似解であって，$(LD(\mathcal{S}'))$ に対しては妥当な近似解ではない可能性があることに注意せよ．

そこで，$\mathcal{S}$ を $\mathcal{S}'$ へと更新した後，再び劣勾配法を呼び出し $(LD(\mathcal{S}'))$ を解き，近似解 $\boldsymbol{\lambda}'$ を得る．ここからは，どのようにして $\gamma_C(\boldsymbol{\lambda}) > 0$ となるコミュニティ C を探索するかについて説明する．(3.1) を $\gamma_C(\boldsymbol{\lambda})$ に代入すると，

$$\gamma_C(\boldsymbol{\lambda}) = \frac{1}{m}\sum_{i\in V}\sum_{j\in V}\left(e_{ij} - \frac{d_i d_j}{2m}\right)a_{iC}a_{jC} - \sum_{i\in V}\lambda_i a_{iC}$$

が得られる．したがって $\gamma_C(\boldsymbol{\lambda})$ を最大にするコミュニティを求める問題は以下の凸二次関数を目的関数に持つ最適化問題に定式化される．以降ではこの問題を補助問題 $(AP(\boldsymbol{\lambda}))$ と呼ぶ．

$$(AP(\boldsymbol{\lambda})) \quad \left|\begin{array}{lll} \text{最大化} & \dfrac{1}{m}\displaystyle\sum_{l\in E} x_l - \frac{1}{4m^2}\left(\sum_{i\in V} d_i y_i\right)^2 - \sum_{i\in V}\lambda_i y_i & \\ \text{制約} & x_l \le y_i & (\forall l = \{i,j\} \in E) \\ & x_l \le y_j & (\forall l = \{i,j\} \in E) \\ & x_l \in \{0,1\} & (\forall l \in E) \\ & y_i \in \{0,1\} & (\forall i \in V). \end{array}\right.$$

決定変数 x_l は，枝 $l \in E$ の両端点 i, j が $\gamma_C(\boldsymbol{\lambda})$ を最大にするコミュニティに属すとき 1，そうでないとき 0 を取る変数であり，決定変数 y_i は頂点 $i \in V$ がそのコミュニティに属すとき 1，そうでないとき 0 を取る変数である．つまり最適解 $\boldsymbol{y}^*$ はコミュニティのインシデンスベクトルである．最適値が正である $\boldsymbol{y}^*$ が見つかったならば，係数 $(f^* - \sum_{i\in V}\lambda_i y_i^*)$ を持つ 0-1 変数を問題 $(LR(\mathcal{S}, \boldsymbol{\lambda}))$ に加える．ただし，

$$f^* = \frac{1}{2m}\sum_{i\in V}\sum_{j\in V}\left(e_{ij} - \frac{d_i d_j}{2m}\right)y_i^* y_j^*.$$

である．以上の議論から，本稿で提案するラグランジュ緩和に基づく列生成法は以下のように記述される．

Algorithm Prototype of Lagrangian-based-Column-Generation

Step 1 : $\mathcal{S}$ を V の非空な部分集合族の初期部分族とする．
$\boldsymbol{\lambda}$ を初期ラグランジュ乗数とする．

Step 2 : Procedure SG$(\mathcal{S}, \boldsymbol{\lambda})$ を呼び出し $(LD(\mathcal{S}))$ を解く．
$\boldsymbol{\lambda}$ と $\bar{\omega}(LD(\mathcal{S}))$ をそれぞれ得られた解と目的関数値とする．

Step 3 : $(AP(\boldsymbol{\lambda}))$ を解き，最適解 $\boldsymbol{y}^*$ を得る．

Step 4 : $\omega(AP(\boldsymbol{\lambda})) \le 0$ ならば，$\mathcal{S}^* \leftarrow \mathcal{S}$，$\omega^* \leftarrow \bar{\omega}(LD(\mathcal{S}))$ に設定する．$\mathcal{S}^*$ と ω^* を出力し終了．
そうでなければ，Step 5 へ．

Step 5 : $C^* \leftarrow \{i \in V \mid y_i^* = 1\}$ と設定し，$\mathcal{S} \leftarrow \mathcal{S} \cup \{C^*\}$ と更新する．Step 2 へ戻る．

このアルゴリズムが終了したら，得られた部分集合族 $\mathcal{S}^*$ から問題 $(P(\mathcal{S}^*))$ を構成する．この問題 $(P(\mathcal{S}^*))$ は元問題 (P) よりも少ない変数を持つ問題であることが期待されるため，既存の IP ソルバで現実的な計算時間内に解くことができるだろう．$(P(\mathcal{S}^*))$ の最適値 $\omega(P(\mathcal{S}^*))$ は $\omega(P)$ の下界になる．さらに，$\omega(AP(\lambda)) \le 0$ が満たされると $\omega(P)$ の上界が得られ，次の不等式が得られる．

$$\omega(P(\mathcal{S}^*)) \le \omega(P) \le \bar{\omega}(LD(\mathcal{S}^*)).$$

$\bar{\omega}(LD(\mathcal{S}^*)) - \omega(P(\mathcal{S}^*))$ は，$\omega(P)$ と $\omega(P(\mathcal{S}^*))$ の差の上界を与えるので，$(P(\mathcal{S}^*))$ の精度評価が可能になる．

もし劣勾配法によって得られたラグランジュ乗数 $\boldsymbol{\lambda}$ が，各 $C \in \mathcal{S}$ に対して $f_C - \sum_{i \in V} \lambda_i a_{iC} \leq 0$ という条件を満たすならば，補助問題 $(AP(\boldsymbol{\lambda}))$ の最適解 $\boldsymbol{y}^*$ から構成されるコミュニティは $\mathcal{P} \setminus \mathcal{S}$ の要素であることが保証される．しかし，一般には各反復で得られる $\boldsymbol{\lambda}$ が上記の条件を満たしているとは限らないので，生成されたコミュニティが $\mathcal{P} \setminus \mathcal{S}$ の要素であるかに関しては不明である．したがって，本稿で提案するアルゴリズムは，すでに $\mathcal{S}$ に含まれているコミュニティを再び生成する可能性がある．予備実験においてはこのような現象はほとんど観測されなかったが，この問題は提案アルゴリズムに残された解決すべき点である．

6 提案アルゴリズム

列生成法は大規模な整数計画問題に対して非常に有効なテクニックであるが，収束が遅いという欠点がある．この節では，列生成法の各反復において上界値と下界値を見積もる方法を紹介した後，各反復で得られた上・下界値を利用した二つの停止条件を提案する．

6.1 上・下界値の見積もり

列生成法を適用するにあたり，$\boldsymbol{\lambda}$ が問題 (LD) の最適解に近づくにつれて，$(LD(\mathcal{S}))$ の目的関数値の改善が鈍化するという現象が観察される．この現象は *tailing-off effect* [8, 15] と呼ばれている．このような場合でも，$\omega(P)$ の非自明な上界を見積もることができれば，$(LD(\mathcal{S}))$ の目的関数値と上界値の差を見ることによって，各反復において現在の目的関数値の質を評価することが可能となる．したがって，この差が十分に小さくなった時点でアルゴリズムを停止させれば，アルゴリズムの反復数を少なくできることが期待される．

元問題 $\omega(P)$ の最適値の上界を求めるために，3.2 節で紹介した問題 (QP) を利用する．まず (QP) の一つ目の制約をラグランジュ緩和すると，以下の緩和問題 $(LRQP(\boldsymbol{\lambda}))$ が得られる．

$$
(LRQP(\boldsymbol{\lambda})) \left| \begin{array}{ll} \text{最大化} & \displaystyle\sum_{p \in T} \left(\frac{1}{m} \sum_{l \in E} x_{lp} - \frac{1}{4m^2} \left(\sum_{i \in V} d_i y_{ip} \right)^2 \right) + \sum_{i \in V} \lambda_i \left(1 - \sum_{p \in T} y_{ip} \right) \\ \text{制約} & x_{lp} \leq y_{ip} \quad (\forall l = \{i,j\} \in E, \forall p \in T) \\ & x_{lp} \leq y_{jp} \quad (\forall l = \{i,j\} \in E, \forall p \in T) \\ & x_{lp} \in \{0,1\} \quad (\forall l \in E, \forall p \in T) \\ & y_{ip} \in \{0,1\} \quad (\forall i \in V, \forall p \in T). \end{array} \right.
$$

一つ目の制約を緩和したことにより，問題 $(LRQP(\boldsymbol{\lambda}))$ は t 個のコミュニティ毎の子問題に分解することができる．各 $p \in T$ に対する子問題を $(LRQP(\boldsymbol{\lambda}, p))$ と表すこととする．

$$
(LRQP(\boldsymbol{\lambda}, p)) \left| \begin{array}{ll} \text{最大化} & \displaystyle\frac{1}{m} \sum_{l \in E} x_{lp} - \frac{1}{4m^2} \left(\sum_{i \in V} d_i y_{ip} \right)^2 - \sum_{i \in V} \lambda_i y_{ip} \\ \text{制約} & x_{lp} \leq y_{ip} \quad (\forall l = \{i,j\} \in E) \\ & x_{lp} \leq y_{jp} \quad (\forall l = \{i,j\} \in E) \\ & x_{lp} \in \{0,1\} \quad (\forall l \in E) \\ & y_{ip} \in \{0,1\} \quad (\forall i \in V). \end{array} \right.
$$

$(LRQP(\boldsymbol{\lambda}))$ の最適値は $(LRQP(\boldsymbol{\lambda},p))$ の最適値を用いて以下のように表すことができる．

$$\begin{aligned}\omega(LRQP(\boldsymbol{\lambda})) &= \max_{\boldsymbol{x},\boldsymbol{y}} \left\{ \sum_{p\in T} \left(\frac{1}{m}\sum_{l\in E} x_{lp} - \frac{1}{4m^2}\left(\sum_{i\in V} d_i y_{ip}\right)^2 \right) - \sum_{i\in V}\lambda_i y_{ip} + \sum_{i\in V}\lambda_i \right\} \\ &= \sum_{p\in T}\omega(LRQP(\boldsymbol{\lambda},p)) + \sum_{i\in V}\lambda_i.\end{aligned}$$

ここで，添え字 $p\in T$ に関係なく各子問題 $(LRQP(\boldsymbol{\lambda},p))$ の最適値が等しいこと，また $(LRQP(\boldsymbol{\lambda},p))$ はアルゴリズム中で解く補助問題 $(AP(\boldsymbol{\lambda}))$ と等価な問題であることに注目すると，結局，問題 $(LRQP(\boldsymbol{\lambda}))$ の最適値は以下で与えられる．

$$\omega(LRQP(\boldsymbol{\lambda})) = t\cdot\omega(AP(\boldsymbol{\lambda})) + \sum_{i\in V}\lambda_i. \tag{6.1}$$

定数 t として問題 (P) の最適解におけるコミュニティ数の上界を与えると，明らかに問題 (QP) は問題 (P) と等価になる．したがって，問題 (QP) のラグランジュ緩和問題 $(LRQP(\boldsymbol{\lambda}))$ の最適値は，任意の $\boldsymbol{\lambda}\in\mathbb{R}$ に対して $\omega(P)$ の上界となる．以上を命題 1 にまとめておく．

命題 1. *t を問題 (P) の最適解におけるコミュニティ数の上界とする．このとき，任意の $\boldsymbol{\lambda}\in\mathbb{R}^n$ に対して (6.1) は $\omega(P)$ の上界である．*

命題 1 における $\boldsymbol{\lambda}$ の任意性から，この命題は提案アルゴリズムの各反復で得られる $\boldsymbol{\lambda}$ に対しても成立する．そのため，計算負荷の追加なしに $\omega(P)$ の上界を見積もることができる．この上界と $\omega(LD(\mathcal{S}))$ の差が小さくなれば，$\omega(AP(\boldsymbol{\lambda}))\leq 0$ が成立しなくてもアルゴリズムを停止する．停止条件の詳細については 6.2 節にて説明する．

次に問題 (P) の実行可能解を作り $\omega(P)$ の下界を得るための貪欲法に基づく発見的解法について説明する．アルゴリズムの各反復において，手元にある部分集合族 $\mathcal{S}$ のなかから $\gamma_C(\boldsymbol{\lambda})$ が大きい順にコミュニティ C を選び，分割 Π に採用していくことを考える．ただし分割を構成するため，$\mathcal{S}$ に含まれるコミュニティのなかで，すでに選ばれたコミュニティと互いに素であるような要素を選ばなければならないことに注意せよ．以上の議論から，各反復で $\omega(P)$ の下界を得る発見的解法は以下のように記述できる．

Procedure Greedy-Heuristics$(\mathcal{S},\boldsymbol{\lambda})$ (GH$(\mathcal{S},\boldsymbol{\lambda})$)

Step 1 : $\mathcal{S}$ と $\boldsymbol{\lambda}$ をそれぞれ各反復時の部分集合族とラグランジュ乗数とする．
$\ell(P)\leftarrow 0$ と設定する．

Step 2 : $C^*\leftarrow \mathrm{argmax}\{\gamma_C(\boldsymbol{\lambda}) \mid C\in\mathcal{S}\}$，$\ell(P)\leftarrow \ell(P)+f_{C^*}$ と設定する．

Step 3 : $V'\leftarrow V\setminus C^*$，$S'\leftarrow\{C\in\mathcal{S}\mid C\subseteq V'\}$ と更新する．

Step 4 : $V'=\emptyset$ であれば，$\ell(P)$ を出力し終了．
そうでなければ，頂点集合と部分集合族をそれぞれ $V\leftarrow V'$，$S\leftarrow S'$ と更新し Step 2 へ戻る．

6.2 停止条件

前節でも言及したとおり，列生成法は，目的関数値 $\omega(LD(\mathcal{S}))$ が $\omega(P)$ の上界に達した後にも，多くの反復を要する現象が頻繁に観測される．また，列生成法において最も計算時間を要するのは補助問

題を解く部分である．したがって，列生成法全体の反復回数を減らすことができれば，補助問題を解く回数も減り，大幅な計算時間の短縮が期待される．ここで，予め定められたパラメータ ε を用いた以下の停止条件を考える．

$$\frac{UB - \omega(LD(\mathcal{S}))}{UB} \leq \varepsilon,$$

ただし，UB はこれまでに得られている最良の上界値とする．つまり，$\omega(LD(\mathcal{S}))$ が十分に $\omega(P)$ の上界値に近づいたときに，アルゴリズムを停止させるというものである．ただし，アルゴリズムの各反復で得られる目的関数値 $\bar{\omega}(LD(\mathcal{S}))$ は必ずしも $\omega(LD(\mathcal{S}))$ と一致するとは限らない．そこでこの不都合を克服するために，本稿では二つの停止条件を提案する．

一つ目の停止条件は，$\bar{\omega}(LD(\mathcal{S}))$ の代わりに $\omega(P)$ の下界値 LB を利用する．上界値 UB と下界値 LB の差が十分に小さくなったら，この下界値は元問題の最適値の十分な近似値となっていることが結論付けられるため，アルゴリズムを停止する．

二つ目の停止条件では，問題 $(LD(\mathcal{S}))$ の最適値を用いる．4.2 節で見たように，ラグランジュ双対問題の最適値と LP 緩和問題の最適値は一致することが知られている．ここで，$(LD(\mathcal{S}))$ の最適値を計算するために，この関係を利用する．限定主問題 $(P(\mathcal{S}))$ の LP 緩和問題とその双対問題を $(RP(\mathcal{S}))$ および $(RD(\mathcal{S}))$ と表記すると，上で述べた関係から，$\omega(LD(\mathcal{S})) = \omega(RD(\mathcal{S}))$ を得る．したがって，$(UB - \bar{\omega}(LD(\mathcal{S})))/UB \leq \varepsilon$ が成立したときのみ，$\omega(RD(\mathcal{S}))$ を得るために LP 緩和問題 $(RD(\mathcal{S}))$ を解いて，

$$\frac{UB - \omega(RD(\mathcal{S}))}{UB} \leq \varepsilon$$

が満たされているかを確認する．

6.3 アルゴリズムの全体像

これまでの議論をまとめると，提案アルゴリズムは以下のように記述される．以降では，このアルゴリズムを *Lagrangian-based-Column-Generation Algorithm* (**LCG**) と呼ぶ．LCG は，列生成法により有望な変数を集め上・下界値を求める Phase 1（Step 1 から Step 7）と，整数解を求める Phase 2 (Step 8) から構成されている．

Algorithm Lagrangian-based-Column-Generation (LCG)

Step 1 : 許容値パラメータ ε を決める．
$UB \leftarrow 1$，$LB \leftarrow 0$ と設定する．
$\mathcal{S}$ を V の非空な部分集合族の部分族とする．
$\boldsymbol{\lambda}$ を初期ラグランジュ乗数とする．

Step 2 : Procedure SG$(\mathcal{S}, \boldsymbol{\lambda})$ を呼び出し，$(LD(\mathcal{S}))$ を解く．
$\boldsymbol{\lambda}$ と $\bar{\omega}(LD(\mathcal{S}))$ をそれぞれ得られた解と目的関数値とする．
Procedure GH$(\mathcal{S}, \boldsymbol{\lambda})$ を呼び出し，$\omega(P)$ の下界値 $\ell(P)$ を得る．
$LB < \ell(P)$ であれば $LB \leftarrow \ell(P)$ と設定する．

Step 3 : $(AP(\boldsymbol{\lambda}))$ を解き，最適解 y^* を得る．
命題 1 に従い上界値 $u(P)$ を計算する．
$UB > u(P)$ であれば $UB \leftarrow u(P)$ と設定する．

Step 4 : $\omega(AP(\boldsymbol{\lambda})) \leq 0$ または $(UB-LB)/UB \leq \varepsilon$ であれば, $\mathcal{S}^* \leftarrow \mathcal{S}$, $\omega^* \leftarrow \bar{\omega}(LD(\mathcal{S}))$, $LB^* \leftarrow LB$ および $UB^* \leftarrow UB$ と設定し, Step 8 へ行く.

Step 5 : $(UB - \bar{\omega}(LD(\mathcal{S})))/UB \leq \varepsilon$ であれば, $(RD(\mathcal{S}))$ を解き $\omega(RD(\mathcal{S}))$ を得る.
そうでなければ Step 7 へ行く.

Step 6 : $(UB - \omega(RD(\mathcal{S})))/UB \leq \varepsilon$ であれば $\mathcal{S}^* \leftarrow \mathcal{S}$, $\omega^* \leftarrow \bar{\omega}(LD(\mathcal{S}))$, $LB^* \leftarrow LB$ および $UB^* \leftarrow UB$ と設定すし, Step 8 へ行く.

Step 7 : $C^* \leftarrow \{i \in V \mid y_i^* = 1\}$ と設定し $\mathcal{S} \leftarrow \mathcal{S} \cup \{C^*\}$ と更新する. Step 2 へ戻る.

Step 8 : IP ソルバで $(P(\mathcal{S}^*))$ を解く.
$\mathcal{S}^*$, LB^*, UB^* および ω^* を出力し終了.

7 計算機実験

提案アルゴリズムの性能を計算機実験によって評価した. アルゴリズムを Python 2.7 で実装し, IP ソルバーには Gurobi 6.0.0 を用いた. 実験は, CPU : Intel Core i7, 2.67 GHz processor, メモリ : 8.0 GB の計算機上で行った. 表 1 は, 実験に用いた 6 つのベンチマーク問題のサイズと既知の最適値をまとめたものである. 表中の M は (P) の最適なコミュニティ数を表している. Karate, Dolphins, Les Miserables, Books, Football は以下のサイトから入手できる.

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

また, s838 は以下のサイトから入手できる.

`http://wws.weizmann.ac.il/mcb/UriAlon/download/collection-complex-networks`

表 1: インスタンスの名称, サイズ, 最適値, 最適なコミュニティの個数

name	n	m	$\omega(P)$	M
Karate	34	78	0.41979	4
Dolphins	62	159	0.52852	5
Les Miserables	77	254	0.56001	6
Books	105	441	0.52724	4
Football	115	613	0.60457	10
s838	512	819	0.81940	13

以降すべての実験において, 初期の部分族 $\mathcal{S}$ には, すべての一点集合を集めた族, つまり $\mathcal{S} = \{\{1\}, ..., \{n\}\}$ を, 初期ラグランジュ乗数には零ベクトルを設定している. また, 上界値を見積もる際のパラメータ t . には最適なコミュニティの個数 M を用いた.

まず, 停止条件の許容パラメータ ε を 0.00 に設定して実験を行う. これは $\omega(AP(\boldsymbol{\lambda})) \leq 0$ を満たすまで, 列生成法の反復を続けることを意味する. 表 2 は, 測定項目の一覧である. 表 3 は, LCG の Phase 1 と Phase 2 の結果をまとめたものである. 表 3 を見ると, 生成された変数の個数 $|\mathcal{S}^*|$ は元問

表 2: 測定項目

iteration	ラグランジュ双対問題 ($LD(\mathcal{S})$) を解いた回数
$\|\mathcal{S}^*\|$	アルゴリズム終了時に得られる $\mathcal{S}^*$ の要素数
ω^*	アルゴリズム終了時に得られる ($LD(\mathcal{S}^*)$) の目的関数値
LB^*	アルゴリズム終了時に得られる (P) の下界値
UB^*	アルゴリズム終了時に得られる (P) の上界値
$\omega(P(\mathcal{S}^*))$	($P(\mathcal{S}^*)$) の最適値
gap(%)	相対誤差. gap $= \left(\frac{\omega(P(\mathcal{S}^*))-\omega(P)}{\omega(P)}\right) \times 100$ によって計測
time 1	Phase 1 の計算時間 (秒)
time 2	Phase 2 の計算時間 (秒)

表 3: 実験結果 ($\varepsilon = 0.00$)

instance	iteration	$\|\mathcal{S}^*\|$	ω^*	LB^*	UB^*
Karate	35	68	0.42245	0.40952	0.42245
Dolphins	44	105	0.53109	0.52207	0.53109
Les Miserables	88	164	0.56386	0.55112	0.56386
Books	72	176	0.52922	0.52075	0.52922
Football	58	172	0.60567	0.60080	0.60567
s838	292	803	0.82103	0.77446	0.82103

instance	$\omega(P(\mathcal{S}^*))$	gap(%)	time 1	time 2
Karate	0.41741	-0.56783	12.71	0.33
Dolphins	0.52761	-0.17224	50.09	1.00
Les Miserables	0.55652	-0.62305	144.41	2.02
Books	0.52075	-1.23092	348.36	3.28
Football	0.60332	-0.20608	4283.32	3.58
s838	0.81428	-0.62426	27029.97	167.00

題 (P) の変数の個数よりもはるかに少なくなっていることが確認できる．頂点数 $n = 34$ の Karate を例にとると，生成された変数の個数は元問題の変数の総数 1.7×10^{10} の約 $1/10^8$ 以下であった．また，どのインスタンスに対しても最適解を得たことを保証することはできていないが，その相対誤差はいずれも 2% 以内であることが確認される．LCG と既存の発見的解法との性能を比較するために，既存の発見的解法によって得られた相対誤差を表 4 にまとめる．GN，CNM，SD および CHL はそれぞれ，Girvan と Newman のアルゴリズム [12]，Clauset らの階層凝集法 [7]，Newman のスペクトラル法 [18] および Cafieri らの分割法 [6] を表している．表 4 を見ると，LCG は CHL を除く発見的解法に対して，

表 4: 既存の発見的解法によって得られた相対誤差

instance	GN	CMN	SD	CHL
Karate	-4.45556	-9.2995	-6.264	-0.21444
Dolphins	-1.60833	-6.24598	-7.05771	-0.386
Les Miserables	-3.57315	-10.6087	-8.24628	-2.36603
Books	NA	-4.7676	-11.3679	-0.15367
Football	-0.59543	-4.51869	-18.523	-0.61032
s838	NA	-1.68904	-10.432	-0.33805

解の精度の観点から優れていることが確認される．次に，LCG の各反復における挙動を観察する．図 1〜6 は，各反復における上界値，$\bar{\omega}(LD(\mathcal{S}))$ および下界値をプロットしたものである．横軸は反復回数，つまりラグランジュ双対問題 $(LD(\mathcal{S}))$ を解いた回数を表している．これらの図から，いずれのインスタンスにおいても，アルゴリズムの早い段階で $\bar{\omega}(LD(\mathcal{S}))$ が急激に上昇すること，アルゴリズムが進行するにつれて変化がゆるやかとなることが確認できる．

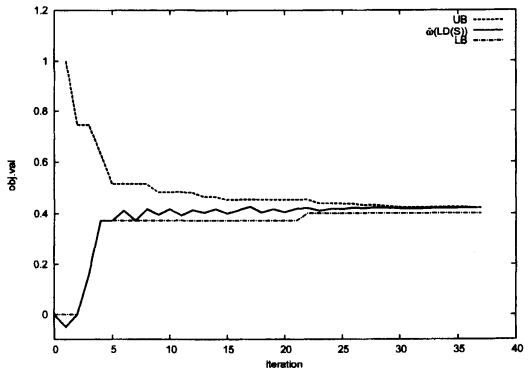

図 1: Karate

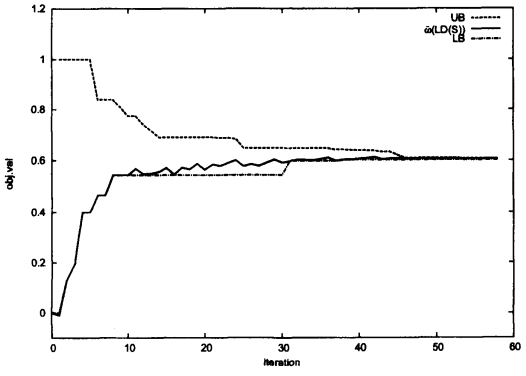

図 2: Dolphins

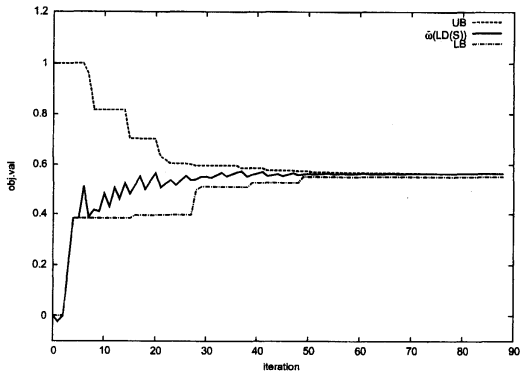

図 3: Les Miserables

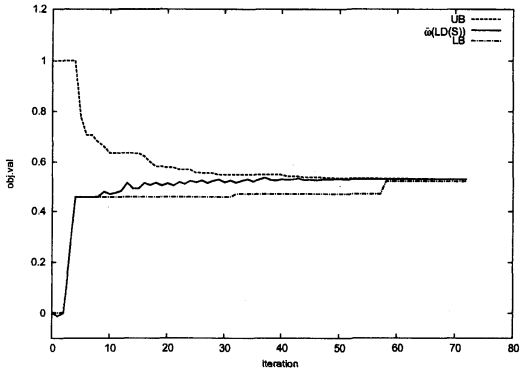

図 4: Books

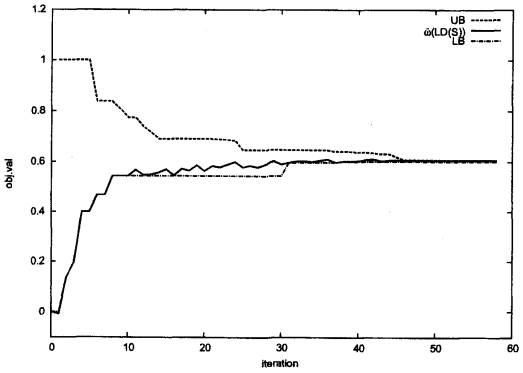

図 5: Football

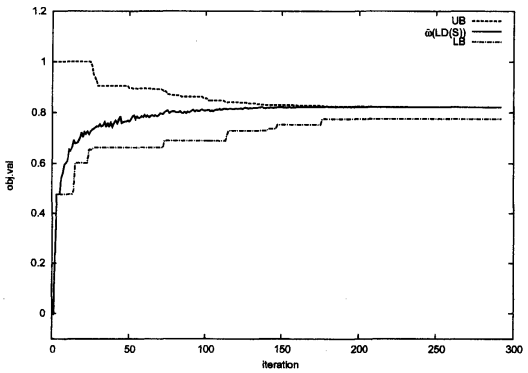

図 6: s838

次に許容パラメータ ε を 0.03 と設定して実験を行った．表 5 は，LCG の Phase 1 と Phase 2 の結果をまとめたものである．表 5 から，いずれのインスタンスにおいても $\varepsilon = 0.00$ の場合と比べて計算時

表 5: 実験結果 ($\varepsilon = 0.03$)

instance	iteration	$\lvert\mathcal{S}^*\rvert$	ω^*	LB^*	UB^*
Karate	30	63	0.42255	0.40952	0.42343
Dolphins	32	93	0.53421	0.52207	0.54295
Les Miserables	55	131	0.56228	0.55112	0.57002
Books	49	153	0.52922	0.47033	0.53436
Football	45	159	0.60734	0.60080	0.61917
s838	158	669	0.82011	0.75199	0.82929
instance		$\omega(P(\mathcal{S}^*))$	gap(%)	time 1	time 2
Karate		0.41741	-0.56783	10.38	0.30
Dolphins		0.52761	-0.17224	34.12	0.89
Les Miserables		0.55343	-1.17523	86.25	1.62
Books		0.52055	-1.26798	208.94	2.83
Football		0.60080	-0.62293	2454.76	3.32
s838		0.80050	-2.30714	11406.32	139.76

間が短縮されていることが確認できる．例えば s838 では 1/2 以下に計算時間が短縮された．解の精度に関しては Les Miserable，Books，Football および s838 に対しては悪化しているものの，いずれも相対誤差は 3% 以内であることが確認される．

8 おわりに

本稿では，モジュラリティ最大化問題に対してラグランジュ緩和法に基づくアルゴリズムを提案した．また，計算負荷を軽減するため，提案アルゴリズムに列生成法を適用した．さらに，$\omega(P)$ の自明ではない上界と下界を見積もる方法を開発し，各反復時の $(LD(\mathcal{S}))$ の最適値の精度評価を可能にした．なお，今後の課題として以下が挙げられる:

- 5 節で言及した通り，提案アルゴリズムは既に生成したコミュニティを再生成する可能性があるため，アルゴリズムの有限収束性が保証されない．そこで，劣勾配法において $(RD(\mathcal{S}))$ の実行可能解となるようなラグランジュ乗数 $\boldsymbol{\lambda}$ を得る手法を考える必要がある．
- 各反復でより効果的な下界を見積もる方法を開発する必要がある．

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, 2010.

[3] P.Benchimol, G.Desaulniers, and J.Desrosiers, "Stabilized dynamic constraints aggregation for solving set partitioning problems," *European Journal of Operational Research*, 223, pp.360-371, 2012.

[4] M.A.Boschetti, A.Mingozzi, and S.Ricciardelli, "A dual ascent procedure for the set partitioning problem," *Discrete Optimization*, 5, pp.735-747, 2008.

[5] 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.

[6] S.Cafieri, P.Hansen and L.Liberti, "Locally optimal heuristic for modularity maximization of networks," *Physical Review*, E.83, 2011.

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

[8] G.Desaulniers, J.Desrosiers, and M.M.Solomon eds., *Column generation*, Springer, Heidelberg, 2005.

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

[10] M.Gendreau and J.Y.Potvin eds., *Handbook of Metaheuristics(Second Edition)*, Springer, 2010.

[11] A.M.Geoffrion, "Lagrangean relaxation for integer programming," *Mathematical Programming Study*, 2, pp.82-114, 1974.

[12] M.Girvan and M.Newman, "Community structure in social and biological networks," *Proceedings of the National Academy of Sciences* USA, 99, 2002.

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

[14] R.Guimerà and A.Amaral, "Functional cartography of complex metabolic networks," *Nature*, 433, pp.895-900, 2005.

[15] M.E.Lübbecke and J.Desrosiers, "Selected topics in column generation," *Operations Research*, 53, pp.1007-1023, 2005.

[16] A.Miyauchi and Y.Miyamoto, "Computing an upper bound of modularity," *The European Physical Journal*, B.86, 302, 2013.

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

[18] M.Newman, "Modularity and community structure in networks," *Proceedings of the National Academy of Sciences* USA, 103, pp.8577-8582, 2006.

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