情報処理学会論文誌　数理モデル化と応用　Vol. 3　No. 1　62–72 (Jan. 2010)

遅延評価導入による局所改善クラスタリング法の高速化

斉藤和巳[†1]　武藤伸明[†1]　池田哲夫[†1]
入月卓也[†1]　永田大[†1]　伊藤かの子[†1]

有限ピボット集合に基づくオブジェクトのクラスタリング問題における代表的な解法として，分割改善法，貪欲改善法，および，局所改善法があげられる．局所改善法を用いれば，他の 2 手法と比較して，一般に望ましい品質の解を安定して求められるが，その計算量は大幅に増大する傾向がある．本論文では，このクラスタリング問題がサブモジュラ性と呼ばれる数理構造を持つことを示すとともに，この構造を利用した遅延評価と呼ばれる手法の導入により，局所改善クラスタリングを高速化する新たなアルゴリズムを提案する．3 種の実データを用いた評価実験では，反復改善法や貪欲改善法に匹敵する処理時間で，提案法により安定して優れた解が求まることを示す．

Speed-up of Local Improvement Clustering Method by Incorporating Lazy Evaluation

Kazumi Saito,[†1] Nobuaki Mutoh,[†1] Tetsuo Ikeda,[†1]
Takuya Iriduki,[†1] Dai Nagata[†1] and Kanoko Ito[†1]

We address the problem of clustering objects based on a finite set of candidate pivots. For this problem, the divided improvement method, the greedy improvement method and the local improvement method are representative. Compared with the other two methods, the local improvement method is expected to stably produce better results, but it surely requires a large amount of computational load. In this paper, after showing that this clustering problem has a submodular property, we newly propose an efficient local improvement method by incorporating a technique called lazy evaluation on the submodular problem. In our experiments using three real data sets, we show that the proposed method can stably produce desirable results with computational efficiency comparable to those of the divided improvement method and the greedy improvement method.

1. はじめに

オブジェクト集合のクラスタリングは，統計分析，機械学習，データマイニングなどにおける基本問題であり，データ解析に加えて多様な応用展開が広がる．各オブジェクトがユークリッド空間のベクトルとして与えられるケースでは，K-平均（K-means）法[2] が代表的な解法であり，ガウス混合分布の推定問題として定式化すれば EM（Expectation-Maximization）アルゴリズム[1] を用いて，妥当な精度の結果を効率良く求めることができる．しかしながら，たとえば，ネットワーク上のノード群をオブジェクト集合とし，最短パス長に基づきクラスタリングする問題など，ある種の構造を持つオブジェクトに対しては，ユークリッド空間のベクトルとしてクラスタ中心を求めることはできない．

特に本論文では，クラスタ中心が有限ピボット候補集合に限定され，オブジェクトとピボット間の類似度に基づいたクラスタリング問題を対象にする．この枠組みは K-メディアン（K-median または K-medoids）問題[10] とも呼ばれ，K-平均法と類似した手法でクラスタ集合を求めることもできる．すなわち，各ピボットとの類似度でオブジェクト集合をクラスタ分割し，クラスタごとに最良ピボットの選択を繰り返す方法である．以下では，この方法を分割改善法と呼ぶ．また，すべてのオブジェクト集合を対象にして，各ピボットを順番に最良ピボットに置き換えてクラスタ集合を求めることもできる．以下では，この方法を局所改善法と呼び，詳細は後述する．

本クラスタリング問題を一般に離散最適化の観点で考えれば NP-完全クラスに属し，大規模になれば妥当な計算時間で厳密解を求めることが一般に困難な問題となる．ただし，本問題はサブモジュラ[7] と呼ばれる構造を持つと想定できる．サブモジュラ構造を持つ離散最適化問題は，潜在的に幅広い応用が存在し，たとえば，多変量データから有用な変数集合を選択する問題[5]，社会ネットワーク上の情報伝播で影響を最大にするノード集合を選択する問題[3] などが知られている．また，その望ましい性質として，いわゆる貪欲改善法で効率良く求まる近似解により，ある程度妥当な精度で，最悪ケースの解品質を理論的に保証することができる[8]．さらに，遅延評価と呼ばれる手法の導入で貪欲改善法をさらに効率化することができ[6]，非常に大規模かつ複雑な問題でも妥当な計算時間で精度保証付き近似解を得ることが可能になる．

†1 静岡県立大学
University of Shizuoka

しかしながら，上述した従来解法に対して以下のような問題点を指摘できる．一般に，分割改善法は高速であるが，その初期値設定や問題構造に依存し，望ましい品質の解を得られない場合が起こる．一方，貪欲改善法については，上述した遅延評価の導入により，分割改善法と同程度の高速化が実現できるものの，クラスタリング問題において，妥当な計算時間でさらに高品質の解を求めることが課題といえる．これらと比較して，局所改善法を用いれば，望ましい品質の解を安定して求められると期待できるが，その計算量は大幅に増大する傾向があり，その高速化手法が必要となる．

本論文では，各種クラスタリング法の計算量を評価するために，オブジェクトとピボットとの類似度計算の回数に着目する．一般に，オブジェクト数 M とピボット候補数 N の問題への適用では，これら任意のペアの類似度を格納するのに $M \times N$ オーダの記憶領域が必要となるが，大規模問題ではこの領域の確保が困難な場合も想定されるからである．このとき，問題に依存して類似度計算には，ある程度の計算コストが必要となるので，これを単位として計算量を評価する．一方，オブジェクトとピボット候補に対する $M \times N$ の類似度行列が格納できる場合には，各類似度を得る処理はメモリ参照のみで高速に実現できる．この場合でも，たとえば，本提案法で必要なソーティングに必要な計算量などを加えることで，本論文の解析結果と同様に議論を展開することができる（詳細は 5.1 節で述べる）．

本論文の構成は以下である．まず，有限ピボット候補集合に基づくオブジェクトのクラスタリング問題を定式化し，代表的な従来法として分割改善法，貪欲改善法，および，局所改善法のアルゴリズムについて述べる．次に，本クラスタリング問題がサブモジュラ性を持つことを示すとともに，遅延評価付き貪欲改善法のアルゴリズムについて述べる．次いで，高品質の結果を妥当な計算コストで安定して求めることを目的として提案する遅延評価付き局所改善法について述べるとともに，既存法との関係を解品質と探索効率の観点で考察する．最後に，3 種の実ネットワークのノードをクラスタリングする問題での実験により，代表的な従来法と比較評価して提案法の有効性を検証する．

2. 問題設定

オブジェクト集合 $X = \{x_i : 1 \le i \le M\}$ とピボット候補集合 $Y = \{y_j : 1 \le j \le N\}$ を考える．ここで，$M = |X|$ はオブジェクト数，$N = |Y|$ はピボット候補数である．任意のオブジェクト $x \in X$ とピボット $y \in Y$ のペアに対して，適切な類似度関数 $\rho(x, y)$ が定義されているとする．自然な設定として，類似度は値域が実数区間 $[0, 1]$ で交換律を満たし，自分自身との類似度は 1 でそのときに限るとする．つまり，$\forall x, \forall y,\ 0 \le \rho(x, y) \le 1$ で $\rho(x, y) = \rho(y, x)$，および，$\rho(x, y) = 1$ と $x = y$ は同値とする．本論文では，与えられた正の整数 $K < N$ に対して，以下の目的関数 f を最大にする K 個のピボット集合 $Z_K \subset Y, |Z_K| = K$ を求める離散最適化問題を考える．

$$f(Z_K) = \sum_{x \in X} \max_{y \in Z_K} \{\rho(x, y)\}. \tag{1}$$

以下では，この最適化問題を単にクラスタリング問題と呼ぶ．

実際，$Z_K = \{z_1, \cdots, z_k, \cdots, z_K\} \subset Y$ に対し，以下のように，オブジェクトのクラスタ分割を定義することができる．

$$C(z_k; Z_K) = \left\{ x \in X : z_k = \arg \max_{y \in Z_K} \{\rho(x, y)\} \right\}. \tag{2}$$

ただし，オブジェクトに対して，類似度を最大にする Z_K の要素が複数存在する場合，適当にタイブレークを行うとする．よって，オブジェクト集合 X とピボット候補集合 Y の要素間の類似度 ρ が与えられれば，式 (1) で定義した目的関数の最大化問題を解くことにより，式 (2) で規定するクラスタリング結果を得ることができる．

3. 従来解法

本クラスタリング問題における代表的な 3 種の従来解法について述べる．以下では，これらの手法を分割改善（DI: Divided Improvement）法，貪欲改善（GI: Greedy Improvement）法，および，局所改善（LI: Local Improvement）法と呼ぶ．

3.1 分割改善法

上述したように，ピボット集合 $Z_K = \{z_1, \cdots, z_K\}$ が与えられれば，オブジェクトをクラスタ分割した集合 $\{C(z_k; Z_K) : k = 1, \cdots K\}$ が定まる．分割改善（DI）法は，K-平均法[2] と類似したアイデアで，目的関数 (1) を改善させるために，ピボット z_k ごとにクラスタ $C(z_k; Z_K)$ の範囲で，次式に基づき最良ピボット $\hat{z}_k$ を求めることを繰り返す．

$$\hat{z}_k = \arg \max_{y \in Y} \left\{ \sum_{x \in C(z_k; Z_K)} \rho(x, y) \right\} \tag{3}$$

以下に，DI 法のアルゴリズムの詳細を示す．

D1. 異なる K 個のピボットを Y からランダムに選定し $Z_K = \{z_1, \cdots, z_K\}$ を初期化する；
D2. 式 (2) に基づきクラスタ分割 $C(z_1; Z_K), \cdots, C(z_K; Z_K)$ を求める；
D3. ピボット z_k（$1 \le k \le K$）のそれぞれで，式 (3) に基づきクラスタごとの最良ピボッ

ト $\hat{z}_k$ を求める；
D4. すべての k（$1 \leq k \leq K$）で $z_k = \hat{z}_k$ ならば $\hat{Z}_K = \{\hat{z}_1, \cdots, \hat{z}_K\}$ を出力し終了する；
D5. すべての k（$1 \leq k \leq K$）で $z_k \leftarrow \hat{z}_k$ と更新し D2. へ戻る．

明らかに DI 法において，結果のピボット集合 $\hat{Z}_K$ は，ステップ D1 で選定する初期ピボット集合に依存して変わりうる．

DI 法の主たる計算量について述べる．ステップ D2. では，各オブジェクトについて最類似ピボットを求めてクラスタ分割を行うので，その計算量は $K \times M$ となる．一方，ステップ D3. では，各クラスタ $C(z_k; Z_K)$ を対象にオブジェクト候補集合 Y から最良ピボットを選択するので，その計算量は $|C(z_1; Z_K)||Y| + \cdots + |C(z_K; Z_K)||Y| = M \times N$ となる．よって，ステップ D2. から D5. の反復を R_D 回行うとすれば，一般に $K \ll N$ より，DI 法の主たる計算量は $R_D \times M \times N$ となる．

なお，X のオブジェクトがユークリッド空間のベクトルとして与えられ，類似度関数 ρ が通常のユークリッド距離の自乗と等価であり，さらに，$Z_K \subset Y$ の制約をなくして任意のユークリッド空間のベクトルまで許容すれば，DI 法は標準的な K-平均法に帰着される．すなわち，ステップ D3. では，クラスタ $C(z_k; Z_K)$ に属するベクトル（オブジェクト）の重心を計算することにほかならない．ここで，$Z_K \subset Y = X$ の制約を採用すれば，いわゆるユークリッド空間での K-メディアンと呼ばれる問題となる．一般に，K-メディアン問題として定式化すれば，外れ値などにロバストになることが知られている[10]．

3.2　貪欲改善法

貪欲改善（GI）法は，$k = 1$ から順に K まで，すでに選定したピボット集合を固定し，目的関数値を最大にするピボットを Y から求めて追加することで結果のピボット集合を求める．いま，各オブジェクト x のピボット集合 Z における最大類似度を以下で表す．

$$\mu(x; Z) = \max_{y \in Z} \{\rho(x, y)\}. \tag{4}$$

ただし，$\mu(x; \emptyset) = 0$ と定義する．このとき，目的関数は以下のように計算できる．

$$f(Z \cup \{y\}) = \sum_{x \in X} \max\{\mu(x; Z), \rho(x, y)\}. \tag{5}$$

以下に，GI 法のアルゴリズムの詳細を示す．
G1. $k \leftarrow 1$，$Z_0 \leftarrow \emptyset$ とし，各オブジェクト $x \in X$ に対し $\mu(x; \emptyset) \leftarrow 0$ と初期化する；
G2. 式 (5) で $\hat{z}_k = \arg\max_{y \in Y \setminus Z_{k-1}} \{f(Z_{k-1} \cup \{y\})\}$ を求め，$Z_k \leftarrow Z_{k-1} \cup \{\hat{z}_k\}$ とする；
G3. $k = K$ ならば $\hat{Z}_K = \{\hat{z}_1, \cdots, \hat{z}_K\}$ を出力し終了する；
G4. 各オブジェクト $x \in X$ に対し $\mu(x; Z_k)$ を求め，$k \leftarrow k + 1$ としステップ G2. へ戻る．

ここで，ステップ G2. における最良追加要素のタイブレークを一意に設定すれば，GI 法で求まる結果のピボット集合 $\hat{Z}_K$ はつねに同じである．

GI 法の主たる計算量について述べる．ステップ G2. では，集合 $Y \setminus Z_{k-1}$ の各ピボットについて，各オブジェクトとの類似度を求めて最良ピボット $\hat{z}_k$ を求めるが，式 (5) を用いれば，その計算量は $(N - k) \times M$ となる．一方，ステップ G4. では，各オブジェクトに対し $\mu(x; Z_k)$ を求めるとき，$\mu(x; Z_k) = \max\{\mu(x; Z_{k-1}), \rho(x, \hat{z}_k)\}$ の関係を用いれば，その計算量は M となる．よって，ステップ G2. から G4. の反復を K 回行うので，一般に $K \ll N$ より，GI 法の主たる計算量は $K \times M \times N$ となる．

3.3　局所改善法

局所改善（LI）法は，ピボット集合 Z_K から，あるピボット $z \in Z_K$ を取り除き，目的関数値を最大にするピボット $\hat{z} \in Y$ に変更することを繰り返す．以下に，LI 法のアルゴリズムの詳細を示す．
L1. $k \leftarrow 1$，$\eta \leftarrow 0$ とし，K 個のピボットを Y からランダムに選定し Z_K を初期化する；
L2. $Z_{K-1} \leftarrow Z_K \setminus \{z_k\}$ とし，各オブジェクト $x \in X$ に対し $\mu(x; Z_{K-1})$ を求める：
L3. 式 (5) で $\hat{z}_k = \arg\max_{y \in Y \setminus Z_{K-1}} \{f(Z_{K-1} \cup \{y\})\}$ を求める；
L4. $z_k \neq \hat{z}_k$ ならば $z_k \leftarrow \hat{z}_k$，$Z_K \leftarrow Z_{K-1} \cup z_k$，$\eta \leftarrow 0$ とし，さもなければ $\eta \leftarrow \eta + 1$ とする；
L5. $\eta = K$ ならば $Z_K = \{z_1, \cdots, z_K\}$ を出力し終了する；
L6. $k = K$ ならば $k \leftarrow 1$ とし，さもなければ $k \leftarrow k + 1$ としてステップ L2. へ戻る．

ここで η は，あるピボットを取り除き別のピボットへの更新を試みても，元のピボットが最良となり，実際にピボット更新の起こらないことが連続して続いている回数を表す変数である．よって，ステップ L5. において $\eta = K$ となれば，連続してすべてのピボットが更新されなかったことになる．明らかに LI 法において，結果のピボット集合 $\hat{Z}_K$ は，ステップ L1 で選定する初期ピボット集合に依存して変わりうる．

LI 法の主たる計算量について述べる．ステップ L2. では，各オブジェクトに対し $\mu(x; Z_{K-1})$ を求めるので，その計算量はたかだか $M(K - 1)$ となる．一方，ステップ L3. では，集合 $Y \setminus Z_{k-1}$ の各ピボットにおいて，各オブジェクトとの類似度を求めて最良ピボット $\hat{z}_k$ を求めるので，GI 法のステップ G2. と同様に式 (5) を用いれば，その計算量は $(N - k) \times M$ となる．よって，ステップ L2. から L6. の反復において，$k = 1, \cdots K$ までのピボット z_k それぞれの変更を R_L 回行うとすれば，一般に $K \ll M$ より，GI 法の主たる計算量は

$R_L \times K \times M \times N$ となる．

4. サブモジュラ最大化問題としての定式化

集合を定義域とする実数値関数 F が与えられ，集合の任意の要素 z と包含関係 $U \subset V$ を満たす任意の集合ペアに対して，以下のサブモジュラ不等式が成り立つとき，関数 F はサブモジュラ関数と呼ばれる．

$$F(U \cup \{z\}) - F(U) \geq F(V \cup \{z\}) - F(V) \tag{6}$$

以下では，式 (1) で定義したクラスタリング問題の目的関数がサブモジュラ関数となることを示す．定義より，任意のオブジェクト $x \in X$ に対して，任意のピボット $z \in Y$ と $U \subset V \subset Y$ となる任意のピボット集合で，以下の関係を満たせば，サブモジュラ不等式が成り立つことが示せる．

$$\max_{y \in U \cup \{z\}} \{\rho(x,y)\} - \max_{y \in U} \{\rho(x,y)\} \geq \max_{y \in V \cup \{z\}} \{\rho(x,y)\} - \max_{y \in V} \{\rho(x,y)\}. \tag{7}$$

いま，ピボット集合 $V \cup \{z\}$ を対象とすれば，オブジェクト x の最類似ピボットは，集合 V か $\{z\}$ のどちらかに含まれる．まず，V のケースでは，$\max_{y \in V \cup \{z\}}\{\rho(x,y)\} = \max_{y \in V}\{\rho(x,y)\}$，すなわち，式 (7) の右辺は 0 であり，$\max_{y \in U \cup \{z\}}\{\rho(x,y)\} \geq \max_{y \in U}\{\rho(x,y)\}$ より，式 (7) の不等式は成り立つ．一方，$\{z\}$ のケースでは，$\max_{y \in V \cup \{z\}}\{\rho(x,y)\} = \max_{y \in U \cup \{z\}}\{\rho(x,y)\} = \rho(x,z)$ であり，$\max_{y \in U}\{\rho(x,y)\} \leq \max_{y \in V}\{\rho(x,y)\}$ より，式 (7) の不等式が成り立つ．よって，式 (1) の目的関数 f はサブモジュラ関数であり，我々のクラスタリング問題はサブモジュラ最大化問題として定式化できる．

サブモジュラ最大化問題として定式化できることより，3.2 節で述べた貪欲改善法での結果は，厳密解ではないものの，ある程度妥当な精度で最悪ケースの解品質が理論的に保証される．詳細には，厳密解を Z_K^* とすると，貪欲改善法で求まる近似解 $\hat{Z}_K$ の精度は，関係式 $f(\hat{Z}_K) \geq (1 - 1/e) f(Z_K^*)$ で抑えられることが証明されている[8]．ここで，e は自然対数の底であり，貪欲改善法の結果は，最悪でも厳密解の 63%程度以上の性能が保証されることになる．

また，遅延評価（lazy evaluation）と呼ばれる手法の導入により，3.2 節で述べた GI 法の探索効率を向上させることができる[6]．いま，あるピボット集合 Z に，ピボット $y \in Y$ を追加したときの目的関数の差分値を以下で表す．

$$g(y;Z) = f(Z \cup \{y\}) - f(Z) = \sum_{x \in X} \left(\max\{\mu(x;Z), \rho(x,y)\} - \mu(x;Z)\right). \tag{8}$$

ただし，$f(\emptyset) = 0$ と定義する．GI 法の各ステップ k で求まる解リストを $Z_1, \cdots, Z_K$ とすれば，サブモジュラ不等式より，差分値 $g(y;Z_k)$ は $0 \leq k \leq K$ において非増加関数となる．したがって，$s < k$ となる正の整数 s に対し，差分値 $g(y;Z_s)$ は $g(y;Z_k)$ の上限値となる．この関係式を利用して，探索効率を向上させるのが遅延評価付き貪欲改善（GILE: Greedy Improvement with Lazy Evaluation）法である．以下では，ピボット $y \in Y$ に対する差分上限値を $\zeta(y)$ で表し，差分上限値でピボットを降順にソートして構成したリストの上位 n 番目のピボットを r_n とする．以下に，GILE 法のアルゴリズムの詳細を示す．

GL1. $k \leftarrow 1$，$Z_0 \leftarrow \emptyset$ とし，各オブジェクト $x \in X$ に対し $\mu(x;\emptyset) \leftarrow 0$，各ピボット $y \in Y$ に対し $\zeta(y) \leftarrow \infty$ に初期化し，$n = 1, \cdots, N$ で $r_n \leftarrow y_n$ に設定する；

GL2. $G \leftarrow 0$ とし，$n = 1, \cdots, N$ の順で GL2-1 から GL2-2 の処理を繰り返す；

GL2-1. $\zeta(r_n) \leq G$ なら，$\zeta(\hat{z}_k) \leftarrow 0$，$Z_k \leftarrow Z_{k-1} \cup \{\hat{z}_k\}$ とし，ステップ GL3 に進む；

GL2-2. $\zeta(r_n) \leftarrow g(r_n; Z_{k-1})$ とし，$G < \zeta(r_n)$ なら，$\hat{z}_k \leftarrow r_n$，$G \leftarrow \zeta(r_n)$ とする；

GL3. $k = K$ ならば $Z_K = \{z_1, \cdots, z_K\}$ を出力し終了する；

GL4. 各オブジェクト $x \in X$ に対し $\mu(x;Z_k)$ を求め，差分上限値 $\zeta(y)$ でピボットを降順にソートしたものを $r_1, \cdots, r_N$ とし，$k \leftarrow k+1$ としステップ GL2. へ戻る．

ここで G は，処理過程の時点で最良差分値を格納する変数である．GILE 法では，ステップ GL2 の第 k 反復目の最良ピボット選定において，差分上限値 $\zeta(y)$ を用いて Y の各要素を降順にソートしたリストを利用する．このリストの先頭から順に探索する過程で，あるピボット $y \in Y$ での実際の改善値と比較して，探索リスト上で未探索の先頭要素での上限が小さくなれば，その時点で探索が終了し最良要素 $\hat{z}_k$ が求まる．明らかに，GILE 法の出力結果は GI 法と一致する．

GILE 法の主たる計算量について述べる．ステップ GL2-2. では，ピボット r_n に対して，各オブジェクトとの類似度を求めて改善値 $g(r_n; Z_{k-1})$ を得るのに，式 (8) を用いれば，その計算量は M となる．よって，ステップ GL3 の終了条件を満たすまでの反復で探索したピボット数の平均割合を α_G（$0 < \alpha_G < 1$）とすれば，ステップ GL2 の計算量は $\alpha_G \times M \times N$ となる．一方，ステップ GL4. では，GI 法のステップ G4. と同じ計算量 M に加えて，差分上限値による N 個のピボットのソートを行うので $N \log N$ の計算量が必要となるが，このソーティング処理には既知の差分上限値リストを用いるので，類似度計算の処理単位の積は不要なため無視できる．よって，ステップ G2. から G4. の反復を K 回行うので，GILE

法の主たる計算量は $\alpha_G \times K \times M \times N$ となる．すなわち，GILE 法の計算効率は α_G に依存するが，この量を解析的に求めることは一般に困難なため，本論文では実験により評価する．

5. 提案法と考察

高品質の結果を妥当な計算コストで安定して求めることを目的として新手法を提案する．また，各手法間の関係を解品質と探索効率の観点で考察する．

5.1 遅延評価付き局所改善法

上述した遅延評価の考え方を土台として，LI 法の探索効率を向上させることを目的とした遅延評価付き局所改善（LILE: Local Improvement with Lazy Evaluation）法を新たに提案する．すなわち，ピボット集合 Z に対して，あるピボット $u \in Z$ を取り除き，別のピボット $v \in Y$ を追加するときに，差分上限値 $\zeta(v)$ を用いて高速化する方法である．以下では，読解性を考慮して記法を簡略化するため，ピボット集合 Z と，除去するピボットを $u \in Z$ に固定して議論する．このとき，ピボット候補集合 Y の中で最良の追加ピボット $\hat{z}$ は次式で求まる．

$$\hat{z} = \arg\max_{v \in Y}\{g(v; Z \setminus \{u\})\} = \arg\max_{v \in Y}\{f((Z \setminus \{u\}) \cup \{v\}) - f(Z \setminus \{u\})\}. \tag{9}$$

LILE 法では，式 (9) の差分値が以下のように変形できることに着目する．

$$\begin{aligned} g(v; Z \setminus \{u\}) &= g(v; Z) + g(u; Z \setminus \{u\}) - g(u; (Z \setminus \{u\}) \cup \{v\}) \qquad (10)\\ &= f(Z \cup \{v\}) - f(Z) + f(Z) - f(Z \setminus \{u\}) - (f(Z \cup \{v\}) - f((Z \setminus \{u\}) \cup \{v\})). \end{aligned}$$

ここで，変化量 $\Delta(v)$ を以下で定義しておく．

$$\Delta(v) = f(Z) - f(Z \setminus \{u\}) - (f(Z \cup \{v\}) - f((Z \setminus \{u\}) \cup \{v\})) \tag{11}$$

また，$g(v; Z)$ はピボット集合 Z にピボット v を追加したときの差分値にほかならず，$g(v; Z)$ もしくは，その差分上限値 $\zeta(v)$ が探索過程で求められているとする．このとき，$\Delta(v)$ を効率良く求めることができれば，局所改善法において，ピボット u をピボット v に置き換えた後の差分値 $g(v; Z \setminus \{u\})$ の上限値が求まり，遅延評価の導入を実現することができる．

変化量 $\Delta(v)$ の効率的な計算法について述べる．まず，式 (11) を詳細化し，各オブジェクト x に対する変化量 $\delta(x, v)$ を以下で定義する．

$$\delta(x, v) = \max_{y \in Z}\{\rho(x, y)\} - \max_{y \in Z \setminus \{u\}}\{\rho(x, y)\} - \left(\max_{y \in Z \cup \{v\}}\{\rho(x, y)\} - \max_{y \in (Z \setminus \{u\}) \cup \{v\})}\{\rho(x, y)\}\right). \tag{12}$$

ここで $\Delta(v)$ と $\delta(x, v)$ に関して，以下の関係が成り立つ．

$$\Delta(v) = \sum_{x \in X} \delta(x, v) \tag{13}$$

以下に，$\delta(x, v)$ の計算法を示す．いま，オブジェクト x に対して，$Z \setminus \{u\}$ の範囲での最類似ピボットを $w = \arg\max_{y \in Z \setminus \{u\}}\{\rho(x, y)\}$ とする．まず，$\rho(x, u) \leq \rho(x, w)$ のケースでは，$\max_{y \in Z}\{\rho(x, y)\} = \max_{y \in Z \setminus \{u\}}\{\rho(x, y)\}$ かつ $\max_{y \in Z \cup \{v\}}\{\rho(x, y)\} = \max_{y \in (Z \setminus \{u\}) \cup \{v\})}\{\rho(x, y)\}$ より，$\delta(x, v) = 0$ となる．よって，$\delta(v)$ の計算には $\rho(x, u) > \rho(x, w)$ のケースのみ考えれば十分となる．すなわち，除去するピボット u に対し，$x \in C(u; Z)$ となるオブジェクトのみで，$\Delta(v)$ を計算できる．一方，$x \in C(u; Z)$ のとき，新たに加えるピボット v の類似度 $\rho(x, v)$ については，$\rho(x, v) > \rho(x, u)$，$\rho(x, u) \geq \rho(x, v) > \rho(x, w)$ または $\rho(x, w) \geq \rho(x, v)$ の 3 通りのケースが起こる．まず，$\rho(x, v) > \rho(x, u)$ ならば，$\max_{y \in Z \cup \{v\}}\{\rho(x, y)\} = \max_{y \in (Z \setminus \{u\}) \cup \{v\})}\{\rho(x, y)\} = \rho(x, v)$ より，$\delta(x, v) = \rho(x, u) - \rho(x, w)$ となる．次に，$\rho(x, u) \geq \rho(x, v) > \rho(x, w)$ ならば，$\max_{y \in Z}\{\rho(x, y)\} = \max_{y \in Z \cup \{v\}}\{\rho(x, y)\} = \rho(x, u)$ より，$\delta(x, v) = \rho(x, v) - \rho(x, w)$ となる．最後に，$\rho(x, w) \geq \rho(x, v)$ ならば，$\max_{y \in Z}\{\rho(x, y)\} = \max_{y \in Z \cup \{v\}}\{\rho(x, y)\} = \rho(x, u)$ かつ $\max_{y \in Z \setminus \{u\}}\{\rho(x, y)\} = \max_{y \in (Z \setminus \{u\}) \cup \{v\})}\{\rho(x, y)\} = \rho(x, w)$ より，$\delta(x, v) = 0$ となる．したがって，$\delta(x, v)$ は次式で計算することができる．

$$\delta(x, v) = \begin{cases} \rho(x, u) - \rho(x, w) & \text{if } \rho(x, v) > \rho(x, u) > \rho(x, w),\\ \rho(x, v) - \rho(x, w) & \text{if } \rho(x, u) \geq \rho(x, v) > \rho(x, w),\\ 0 & \text{otherwise.} \end{cases} \tag{14}$$

遅延評価付き局所改善（LILE）法のアルゴリズムの詳細を以下に示す．

LL1. $k \leftarrow 1$，$\eta \leftarrow 0$ とし，初期ピボット集合 Z_K をランダムに初期化し，各オブジェクト $x \in X$ に対し $\mu(x; Z_K)$ を求め，各ピボット $y \in Y$ で $\zeta(y) \leftarrow g(y; Z_K)$ に設定する；

LL2. $Z_{K-1} \leftarrow Z_K \setminus \{z_k\}$ とし，各オブジェクト $x \in X$ に対し $\mu(x; Z_{K-1})$ を求めるとともに，$x \in C(z_k; Z_K)$ となるオブジェクトを選定する；

LL3. オブジェクト $x \in C(z_k; Z_K)$ に対し，式 (14) で任意の $y \in Y$ について $\delta(x, y)$ を求め，差分上限値を $\zeta(y) \leftarrow \zeta(y) + \sum_{x \in C(z_k; Z_K)} \delta(x, y)$ で更新し，これらの値でピボットを降順にソートしたものを $r_1, \cdots, r_N$ とする；

LL4. $G \leftarrow 0$ とし，$n = 1, \cdots, N$ の順で LL4-1 から LL4-2 の処理を繰り返す：
LL4-1. $\zeta(r_n) \leq G$ なら，$\zeta(\hat{z}_k) \leftarrow 0$，$Z_k \leftarrow Z_{k-1} \cup \{\hat{z}_k\}$ とし，ステップ LL5. に進む；
LL4-2. $\zeta(r_n) \leftarrow g(r_n; Z_{k-1})$ とし，$G < \zeta(r_n)$ なら，$\hat{z}_k \leftarrow r_n$，$G \leftarrow \zeta(r_n)$ とする；
LL5. $z_k \neq \hat{z}_k$ ならば $Z_K \leftarrow Z_{K-1} \cup \hat{z}_k$，$\eta \leftarrow 0$ とし，さもなければ $\eta \leftarrow \eta + 1$ とする；
LL6. $\eta = K$ ならば $Z_K = \{z_1, \cdots, z_K\}$ を出力し終了する；
LL7. $k = K$ ならば $k \leftarrow 1$ とし，さもなければ $k \leftarrow k + 1$ としてステップ L2. へ戻る．
ここで η は，LI 法と同様に，あるピボットを取り除き別のピボットへの更新を試みても，元のピボットが最良となり，実際にピボット更新の起こらないことが連続して続いている回数を表す変数である．また G は，GILE 法と同様に，処理過程の時点で最良差分値を格納する変数である．これまに述べてきた他手法と対比させ，LILE 法のアルゴリズムを説明する．ステップ LL1 と LL2 では，LI 法のステップ L1 と L2 での処理に加え，遅延評価を実行するため，$\zeta(y) \leftarrow g(y; Z_K)$ の初期化と $x \in C(z_k; Z_K)$ となるオブジェクト選定を行っている．ステップ LL3 は，上述した差分値計算のために必要で，他手法に存在しない新たな処理である．また，ステップ LL4 とそのサブステップ LL4-1 と LL4-2 は，遅延評価の実行部であり，この部分は，GILE 法のステップ GL2 とそのサブステップ GL2-1 と GL2-2 と等価である．最後に，ステップ LL5 から LL7 の処理は，LI 法のステップ LL4 から LL6 とほぼ符合する処理である．明らかに，同一初期ピボット集合 Z_K に対し，LILE 法の出力結果は LI 法と一致する．

LILE 法の主たる計算量について述べる．ステップ LL1. では，LI 法のステップ L2. と同様の $K \times M$ の計算量に加えて，差分値 $g(y; Z_K)$ の計算に $M \times N$ の計算量が必要となる．また，ステップ LL2. はたかだか $K \times M$ の計算量で実行できる．一方，ステップ LL3. では，オブジェクト $x \in C(z_k; Z_K)$ に対して，全ピボットの差分上限値を更新するので，その計算量は $|C(z_k; Z_K)| \times N$ となる．また，GILE 法のステップ G4 と同様に，差分上限値による N 個のピボットのソートするので $N \log N$ の計算量が必要となるが，このソーティング処理には既知の差分上限値リストを用いるので，類似度計算の処理単位の積は不要なため無視できる．ステップ LL4. については，ステップ LL4-1. の終了条件を満たすまでの反復で探索したピボット数の平均割合を α_L（$0 < \alpha_L < 1$）とすれば，GILE 法のステップ GL2 と同様に，その計算量は $\alpha_L \times M \times N$ となる．いま，ステップ L2. から L6. の反復において，$k = 1, \cdots, K$ までピボット z_k それぞれの更新を R_L 回行うとする．このときステップ LL3. における全体の計算量は，DI 法に類似して近似的に $R_L \times (|C(z_1; Z_K)| \times N + \cdots + |C(z_K; Z_K)| \times N) \approx R_L \times M \times N$ となる．よって，LILE 法の主たる計算量は $R_L \times M \times N + R_L \times \alpha_L \times K \times M \times N$ となる．すなわち，LILE 法の計算効率は R_L と α_L に依存するが，これらの量を解析的に求めることは一般に困難なため，本論文では実験により評価する．

ここで，オブジェクトとピボット候補に対する $M \times N$ の類似度行列が格納でき，各類似度を得る処理がメモリ参照のみで高速に実現できる場合について考察する．つまり，ステップ LL3. において，N 個のピボットのソートのための計算量 $N \log N$ を考慮する．上記と同様に，$k = 1, \cdots, K$ までピボット z_k それぞれの更新を R_L 回行うとすれば，ソートに必要な全計算量は $R_L \times K \times N \times \log N$ となる．上述した LILE 法の主たる計算量 $R_L \times M \times N + R_L \times \alpha_L \times K \times M \times N$ と比較すると，このソートの計算量が支配的かどうかは，M と $K \log N$，または $\alpha_L M$ と $\log N$ の大小関係により定まる．ただし，LI 法の主たる計算量 $R_L \times K \times M \times N$ と比較すれば，一般の問題設定で $\log N \ll M$ となることより，類似度行列が格納できる場合でも，LILE 法により大幅な計算量の削減を実現することが期待できる．

5.2　解品質と探索効率に関する考察

これまでに述べた各手法について，出力結果の解品質とアルゴリズムの計算量の観点で考察する．解品質については，あるピボット集合 Z_K に対して，各手法を用いて，目的関数値を改善する $f(\hat{Z}_K) > f(Z_K)$ となるような $\hat{Z}_K$ が求まるか考える．まず，GI 法については，$k = 1$ から K まで順次結果を求めるので，このようなピボット集合の改善は考えていない．一方，あるピボット集合 Z_K を GI 法の出力結果とすれば，DI 法や LI 法により，改善した $\hat{Z}_K$ を得られる場合が起こる．なぜなら，GI 法の第 s（$s < K$）反復目で選定されたピボット z_s については，その後に追加されるピボット集合について考慮されず選定されているので，ピボット集合 Z_K における最適性が保証されないからである．

次に，あるピボット集合 Z_K を DI 法で改善できれば LI 法でも改善できるが，この逆は起こらないことを示す．いま，DI 法で改善できるなら，少なくとも 1 つのピボット z_s が $\hat{z}_s$ に更新されて，$f(Z_K \setminus z_s \cup \hat{z}_s) > f(Z_K)$ が成り立つ．明らかに，このような $\hat{z}_s$ が存在するなら，LI 法の $k = s$ の処理において改善が起こる．逆に，LI 法で改善できて $f(Z_K \setminus z_s \cup \hat{z}_s) > f(Z_K)$ となる $\hat{z}_s$ が存在しても，$\sum_{x \in C(z_s; Z_K)} \rho(x, z_s) > \sum_{x \in C(z_s; Z_K)} \rho(x, \hat{z}_s)$ となる場合は起こりうる．直感的には，クラスタ $C(\hat{z}_s; Z_K)$ と $C(z_s; Z_K)$ が大幅に異なる場合である．すなわち，あるピボット集合 Z_K を LI 法で改善できても DI 法では改善できない場合が起こる．これより，DI 法において望ましくない局所解が結果となるような場合でも，LI 法により克服できる可能性が期待できる．したがって，潜在的には LI 法により，最も高品質の結果を

得られることが想定できる．ただし，初期ピボット集合の与え方によっては，望ましくない局所解が LI 法の結果となる場合も起こるので，これらの点については実験により評価する．

以下では，計算量について考察する．まず，GI 法と LI 法については，計算量がピボット数 K にそのまま比例するので，多数のピボットを求める問題では，単調に処理時間が増大することになる．特に LI 法では，反復回数 R_L にも比例するので，最も多くの処理時間を要すると考えられる．一方，DI 法については，ピボット数 K に影響されず，反復回数 R_D に依存した処理時間となる．これらに対して，GILE 法と LILE 法については，計算量がピボット数 K に比例するものの，その項には遅延評価による効率化ファクタ α_G と α_L が効いてくる．一般に遅延評価の性質より，探索が進むに従って差分上限値 $\zeta(y)$ の評価が正確になると期待できる．よって，ピボット数 K に対し単調な比例による処理時間の増大は抑えられると考えられる．特に，LILE 法については，R_L に関する反復においても差分上限値の精緻化による効果が期待できる．概していえば，LILE 法の主たる計算量は $R_L \times M \times N + R_L \times \alpha_L \times K \times M \times N$ となることより，前者の $R_L \times M \times N$ は DI 法の処理時間に近く，後者の $R_L \times \alpha_L \times K \times M \times N$ は GILE 法の処理時間に匹敵すると予想できる．つまり，LILE 法の処理時間は，DI 法と GILE 法の処理時間の和程度になると考えられる．これらの点の詳細についても実験により評価する．

6. 評 価 実 験

6.1 評価データと実験設定

ネットワークのノード群をクラスタリングする問題で各種手法を評価する．実験では，3 種の実ネットワークを用いた．詳細には，Watts らがスモールワールドに関する論文で用いた電力ネットワーク[12]（ノード数 4,941，リンク数 6,594），ブログのエントリとトラックバックから構成される有向グラフに対し，トラックバックは相互承認が必要なことから無向化して構築したブログネットワーク[11]（ノード数 12,047，リンク数 39,960），および，ウィキペディア内の人名一覧に登場する人物を対象に，記事中に 6 回以上共起した 2 人の人物をリンクすることから得られる無向グラフの最大連結成分を抽出した人名ネットワーク[4]（ノード数 9,481，リンク数 122,522）を用いた．これらのネットワークでは，リンクの意味付けとともに，ある程度の幅でリンク密度（ノード数とリンク数の比）も異なるので，それぞれに固有の性質を持ったネットワークと考えられる．

実験では，オブジェクト集合 X とピボット候補集合 Y をネットワークの全ノードとして設定した．すなわち，$X = Y$ である．オブジェクト x とピボット y の類似度については，ノード x とノード y 間のネットワーク上での最短パス長を $d(x, y)$ を求め，類似度関数を $\rho(x, y) = 1/(1 + d(x, y))$ で定義した．また，本論文の基本設定に従い，各ノード間の最短パス長はつねに再計算するようにした．なお，このような最短パス長の逆数の利用は，非連結グラフへも適用可能となり，より汎用性の高い手段であることが指摘されている[9]．各種手法の性能評価では，結果のピボット数を $K = 5$，10，15，20，25，30 に設定した．

6.2 解品質の比較結果

表 1，**表 2**，**表 3** には，式 (1) で定義した目的関数値に基づき，各手法での解品質の比較結果を示す．ここで，LILE 法の結果は LI 法と実質等価であり，GILE 法の結果は GI 法と完全に一致するので省略した．また，LI 法と DI 法の結果については，初期値を変えた 10 回の試行結果における平均と標準偏差を示すとともに，これら試行における最良値を括弧内に参考値として示している．

これら表より，LI 法の結果は，どのピボット数 K においても，DI 法や GI 法と比較して優れた解品質であることが分かる．すなわち，5.2 節で述べた考察と符合した実験結果が得られている．詳細には，DI 法と比較して，LI 法の標準偏差は実質的に小さいことが見てとれる．つまり，LI 法を用いれば，初期値依存性は軽減され安定して高品質の結果を得られることが示唆される．また，GI 法と比較して，LI 法の結果は向上していることも見てとれる．すでに述べているように，GI 法では，サブモジュラ性により最悪ケースが保証され，ある程度妥当な精度の結果となっているが，これらと比較して LILE 法の解品質が優っていることが分かる．特に電力とブログネットワークにおいて，LILE 法での結果の改善度が大きくなっている．本実験より，本論文で対象としたクラスタリング問題において，局所改善アプローチの有望性が示唆されたと考える．

6.3 探索効率の比較結果

表 4，**表 5**，**表 6** に，出力結果を得るまで処理時間（sec.）の比較結果を示す．ここで，各手法のプログラムは C 言語で実装し，実験には Intel 1.86 GHz CPU のパソコンを用いた．また，LILE 法，LI 法および DI 法の結果については，初期値を変えた 10 回の試行結果における平均と標準偏差を示している．

この実験結果より，LILE 法は LI 法と比較して大幅な処理時間の短縮が実現されていることが分かる．また，5.2 節で述べた考察の妥当性も示唆されたと考えられる．詳細には，まず，GI 法と LI 法については，ピボット数 K に比例して，単調に処理時間が増大していることが分かる．特に LI 法では，反復回数 R_L にも比例するので，最も多くの処理時間を要している．一方，DI 法については，ピボット数 K に影響されず，ほぼ一定の処理時間

表 1　電力ネットワークでの解品質
Table 1　Solution quality for the electrical power grid.

K	LI (LILE)	DI	GI (GILE)
5	714.7 ± 1.3　(715.4)	693.9 ± 14.7　(715.4)	703.0
10	849.1 ± 0.8　(849.4)	797.3 ± 15.0　(822.4)	834.9
15	950.6 ± 0.0　(950.6)	893.2 ± 14.3　(927.8)	934.1
20	1020.8 ± 0.8　(1023.1)	962.4 ± 18.1　(987.2)	1005.2
25	1079.0 ± 0.3　(1079.2)	1020.8 ± 25.2　(1050.0)	1064.0
30	1125.9 ± 0.7　(1126.8)	1056.3 ± 15.0　(1072.9)	1111.6

表 2　ブログネットワークでの解品質
Table 2　Solution quality for the blog network.

K	LI (LILE)	DI	GI (GILE)
5	2922.3 ± 0.0　(2922.3)	2817.5 ± 60.4　(2913.1)	2917.2
10	3347.0 ± 0.9　(3347.6)	3191.3 ± 74.6　(3335.4)	3316.7
15	3551.8 ± 3.0　(3554.1)	3343.7 ± 59.5　(3433.9)	3523.0
20	3724.0 ± 2.8　(3725.0)	3520.6 ± 48.2　(3582.8)	3684.5
25	3850.4 ± 0.9　(3851.5)	3637.9 ± 34.6　(3718.8)	3814.0
30	3947.5 ± 1.0　(3950.1)	3725.2 ± 71.4　(3824.8)	3920.0

表 3　人名ネットワークでの解品質
Table 3　Solution quality for the Wikipedia network.

K	LI (LILE)	DI	GI (GILE)
5	3224.0 ± 0.0　(3224.0)	3067.1 ± 42.9　(3137.6)	3221.0
10	3420.7 ± 0.0　(3420.7)	3285.4 ± 40.3　(3325.6)	3417.4
15	3536.3 ± 0.3　(3536.8)	3364.4 ± 58.5　(3407.0)	3536.2
20	3632.4 ± 0.0　(3632.4)	3433.7 ± 35.1　(3496.5)	3629.3
25	3710.9 ± 0.1　(3710.9)	3513.3 ± 22.3　(3550.4)	3706.2
30	3770.8 ± 0.3　(3771.0)	3516.8 ± 32.9　(3569.9)	3766.1

を要しているが，K が大きくなるにつれ，反復回数の減少により，平均処理時間も減少する傾向も見てとれる．また，GILE 法と LILE 法については，計算量がピボット数 K に比例するものの，遅延評価による効率化が十分に機能していることも分かる．すなわち，一般に遅延評価の性質より，探索が進むに従って差分上限値の評価が正確になっていることが示唆される．特に，LILE 法については，R_L に関する反復においても差分上限値の精緻化による大幅な処理時間の短縮が実現できたと考えられる．最後に，今回の実験の範囲で概していえば，LILE 法の処理時間は，DI 法と GILE 法の処理時間の和程度であることも分かる．

表 4　電力ネットワークでの計算効率（sec.）
Table 4　Processing efficiency for the electrical power grid.

K	LILE	LI	DI	GILE	GI
5	3.34 ± 0.33	9.70 ± 1.07	1.61 ± 0.37	1.91	4.24
10	3.41 ± 0.68	19.89 ± 4.11	1.32 ± 0.40	2.11	8.46
15	3.71 ± 0.58	30.58 ± 4.90	1.35 ± 0.26	2.25	12.68
20	3.39 ± 0.29	40.45 ± 4.61	1.05 ± 0.19	2.42	16.90
25	4.37 ± 0.64	59.85 ± 3.72	0.95 ± 0.22	2.48	21.11
30	4.67 ± 0.41	79.44 ± 11.56	0.94 ± 0.23	2.60	25.31

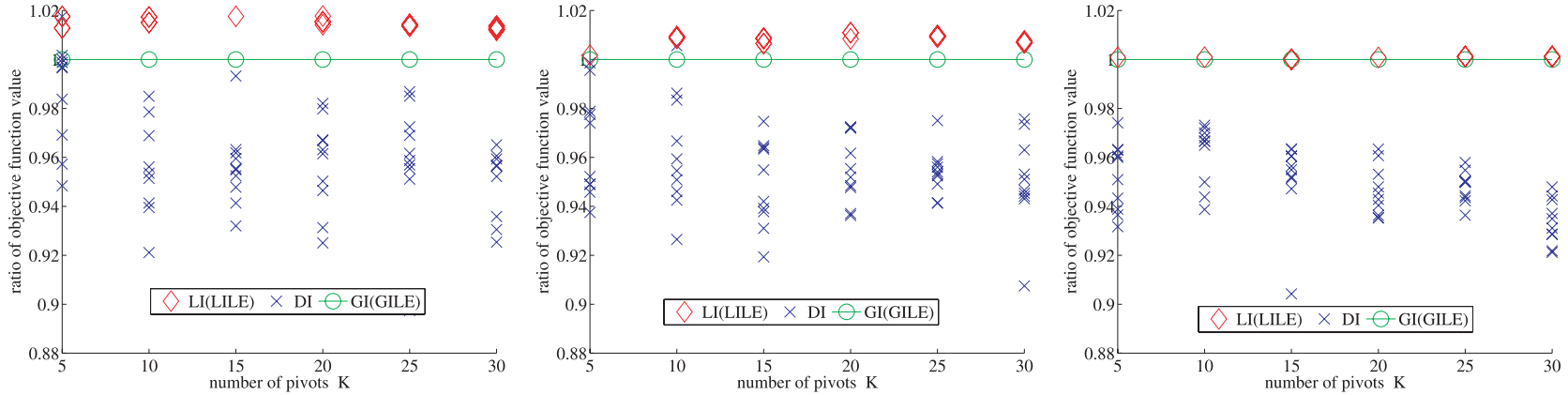
表 5　ブログネットワークでの計算効率（sec.）
Table 5　Processing efficiency for the blog network.

K	LILE	LI	DI	GILE	GI
5	22.65 ± 2.35	64.74 ± 5.21	14.21 ± 3.15	12.41	28.73
10	24.43 ± 2.34	132.00 ± 13.41	14.44 ± 4.32	14.26	57.41
15	26.51 ± 2.22	211.71 ± 22.94	15.24 ± 3.95	14.89	86.07
20	29.74 ± 2.51	323.53 ± 26.13	10.83 ± 2.52	15.48	114.74
25	29.52 ± 1.06	367.55 ± 26.41	13.76 ± 3.42	15.66	143.36
30	28.39 ± 2.51	458.91 ± 58.40	10.05 ± 1.70	16.22	171.96

表 6　人名ネットワークでの計算効率（sec.）
Table 6　Processing efficiency for the Wikipedia network.

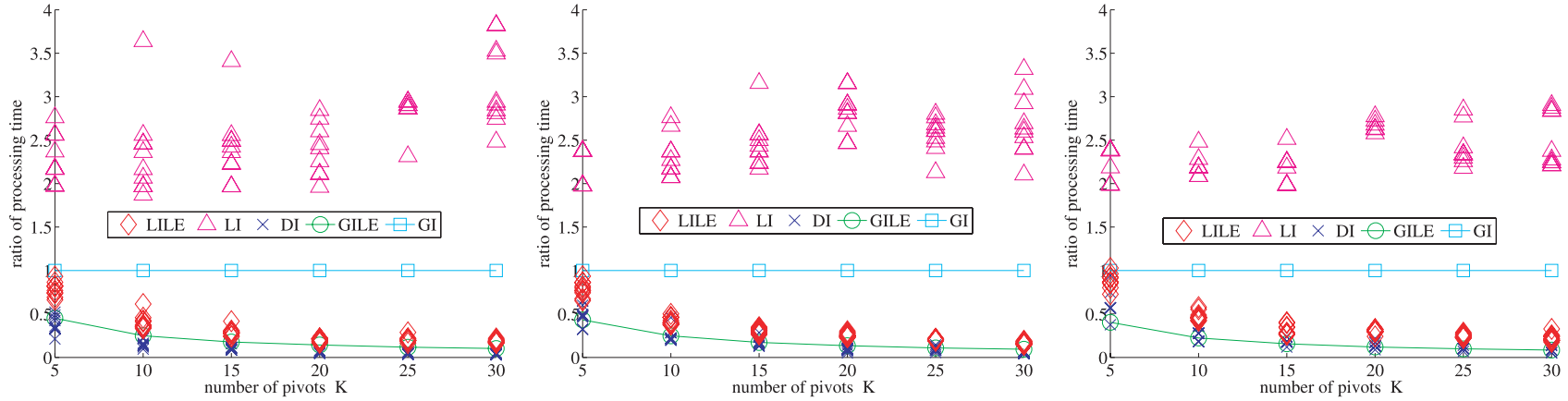
K	LILE	LI	DI	GILE	GI
5	30.20 ± 2.73	78.29 ± 5.49	20.75 ± 4.79	13.84	34.30
10	32.58 ± 3.36	150.30 ± 7.71	18.25 ± 4.33	15.32	68.52
15	32.14 ± 5.44	219.48 ± 17.72	19.07 ± 3.37	16.07	102.75
20	41.11 ± 3.05	363.14 ± 7.52	19.83 ± 4.64	16.39	137.00
25	41.79 ± 3.27	412.93 ± 35.75	17.53 ± 2.71	16.97	171.25
30	46.30 ± 8.46	513.76 ± 61.38	18.58 ± 5.67	17.34	205.51

表 1 から表 3 に示した解品質も考慮し，LILE 法と DI 法を比較する．DI 法の計算効率は LILE 法に優るものの，特にピボット数 K が多くなれば，DI 法での初期値を変えた 10 回の試行では最良結果でも，LILE 法の結果と比較して大幅に劣っている．つまり，DI 法で異なる初期値の試行を繰り返して最良結果を求めるとしても，妥当な計算量で LILE 法に匹敵する結果を得ることは一般に困難と予想される．したがって，本論文で対象としたクラスタリング問題において，高品質の結果を妥当な計算コストで安定して求める新手法として LILE 法の有望性が示唆されたと考える．



(a) 電力ネットワークでの解品質比　(b) ブログネットワークでの解品質比　(c) 人名ネットワークでの解品質比

図 1　GI 法に対する解品質比
Fig. 1　Ratio of solution quality to the GI method.



(a) 電力ネットワークでの処理効率比　(b) ブログネットワークでの処理効率比　(c) 人名ネットワークでの処理効率比

図 2　GI 法に対する処理効率比
Fig. 2　Ratio of processing efficiency to the GI method.

6.4 実験結果の考察

さらなる実験結果の考察のため，解品質と処理効率の両面で最も安定した結果となるGI法の性能を単位とし，上述した3種のネットワークにおける各手法の性能比を比較した．**図1**には，GI法に対する解品質比を，**図2**には，GI法に対する処理効率比を示す．図1の解品質での比較より，GI（GILE）法については，電力，ブログ，人名の各ネットワークの順で，LI（LILE）法の解品質に近付いていることが分かる．このことより，貪欲改善法には，ある程度妥当な品質の結果が得られる問題と，そうでない問題が存在することが示唆される．これに対しDI法については，このような傾向は実験の範囲では見てとれない．一方，図2の処理効率での比較では，LI法については，電力，ブログ，人名の各ネットワークの順で，処理効率が若干安定して改善される傾向が見られる．その他の手法については，どのネットワークにおいてもほとんど同じような傾向を示している．

7. おわりに

本論文では，有限ピボット候補集合に基づくオブジェクトのクラスタリング問題がサブモジュラ構造を持つことを示すとともに，この構造を利用した遅延評価と呼ばれる戦略の導入により，局所改善クラスタリングを高速化する新たな手法として遅延評価付き局所改善（LILE）法を提案した．実験では，3種の実ネットワークのノードをクラスタリングする問題において，分割改善（DI）法や遅延評価付き貪欲改善（GILE）法など代表的な解法と比較して，LILE法の性能を評価した．その結果，DI法やGILE法と同程度の計算時間で，安定して優れた品質の解がLILE法で求まることが示唆された．今後は，さらに多様なクラスタリング問題にLILE法を適用して，その有効性などの評価を進める予定である．

参考文献

1) Dempster, A.P., Laird, N.M. and Rubin, D.B.: Maximum likelihood from incomplete data via EM algorithm, *J. Royal Statist. Soc. Ser. B* (*methodology*), Vol.39, pp.1–38 (1977).
2) Duda, R.O. and Hart, P.E.: *Pattern classification and scene analysis*, John Wiley & Sons (1973).
3) Kempe, D., Kleinberg, J. and Tardos, E.: Maximizing the spread of influence through a social network, *Proc. 9th International Conference on Knowledge Discovery and Data Mining*, pp.137–146 (2003).
4) Kimura, M., Saito, K. and Nakano, R.: Extracting influential nodes for information diffusion on a social network, *Proc. 22nd AAAI Conference on Artificial Intelligence*, pp.1371–1376 (2007).
5) Krause, A. and Guestrin, C.: Near-optimal Nonmyopic Value of Information in Graphical Models, *Proc. 21st Conference in Uncertainty in Artificial Intelligence*, pp.324–331 (2005).
6) Leskovec, J., Krause, A., Guestrin, C., Faloutsos, C., VanBriesen, J. and Glance, N.: Cost-effective outbreak detection in networks, *Proc. 13th International Conference on Knowledge Discovery and Data Mining*, pp.420–429 (2007).
7) 室田一雄：離散凸解析の考えかた，最適化における離散と連続の数理，共立出版 (2007).
8) Nemhauser, G., Wolsey, L. and Fisher, M.: An analysis of the approximations for maximizing submodular set functions, *Mathematical Programming*, Vol.14, pp.265–294 (1978).
9) Newman, M.E.J.: The structure and function of complex networks, *SIAM Review*, Vol.45, No.2, pp.165–256 (2003).
10) Ripley, B.D.: *Pattern Recognition and Neural Networks*, Cambridge University Press (1996).
11) 斉藤和巳：ウェブサイエンス入門—インターネットの構造を解き明かす，NTT出版 (2007).
12) Watts, D.J. and Strogatz, S.H.: Collective dynamics of 'small-world' networks, *Nature*, Vol.393, pp.440–442 (1998).

(平成21年 8 月13日受付)
(平成21年 9 月20日再受付)
(平成21年10月23日採録)

斉藤　和巳（正会員）

昭和38年生．昭和60年慶應義塾大学理工学部数理科学科卒業．同年NTT入社．平成3年より1年間オタワ大学客員研究員．平成19年静岡県立大学経営情報学部教授．機械学習，複雑ネットワーク等の研究に従事．博士（工学）（東京大学）．平成8年情報処理学会論文賞等受賞．電子情報通信学会，人工知能学会，日本神経回路学会，日本応用数理学会各会員．

武藤　伸明

昭和 42 年生．平成 2 年横浜国立大学工学部電子情報工学科卒業．平成 4 年横浜国立大学大学院工学研究科博士課程前期修了．平成 7 年横浜国立大学大学院工学研究科博士課程後期修了．同年静岡県立大学経営情報学部助手．平成 19 年静岡県立大学経営情報学部准教授．専門は離散幾何学，グラフ論など．博士（工学）（横浜国立大学）．電子情報通信学会員．

池田　哲夫（正会員）

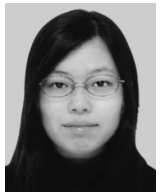
昭和 31 年生．昭和 54 年東京大学理学部情報科学科卒業．昭和 56 年東京大学大学院理学系研究科情報科学専攻修士課程修了．同年日本電信電話公社（現 NTT）電気通信研究所入所．平成 14 年岩手県立大学ソフトウェア情報学部教授．平成 18 年静岡県立大学経営情報学部教授．専門は，データベース工学，情報検索，社会情報システム等．博士（工学）（東京大学）．ACM，IEEE CS，言語処理学会各会員．

入月　卓也

昭和 63 年生．静岡県立大学経営情報学部経営情報学科在学中．複雑ネットワークのデータマイニングに興味を持つ．人工知能学会会員．

永田　　大

昭和 61 年生．平成 21 年静岡県立大学経営情報学部経営情報学科卒業．平成 21 年より静岡県立大学大学院経営情報学研究科経営情報学専攻在学中．複雑ネットワーク，データマイニング，離散幾何学に興味を持つ．



伊藤かの子

平成 21 年静岡県立大学経営情報学部経営情報学科卒業．平成 21 年より静岡県立大学大学院経営情報学研究科経営情報学専攻在学中．地理情報システム，自然言語処理に興味を持つ．