3M1-1

一般化ピボットによる L_1 距離に基づく K-medoids クラスタリングの高速化

Acceleration of K-medoids Clustering Based on L_1 Distance Using Generalized Pivots

伏見 卓恭 *1　斉藤 和巳 *2　風間 一洋 *3
Takayasu Fushimi　Kazumi Saito　Kazuhiro Kazama

*1東京工科大学 Tokyo University of Technology
*2静岡県立大学 University of Shizuoka
*3和歌山大学 Wakayama University

With the explosive increase of multimedia objects represented as high-dimensional vectors, clustering techniques for these objects have received much attention in recent years. However, clustering methods usually require a large amount of computational cost when calculating the distances between these objects. In this paper, for accelerating the greedy K-medoids clustering algorithm with L_1 distance, we propose a new method consisting of the fast 1st medoid selection, lazy evaluation, and pivot pruning techniques, where the efficiency of the pivot construction is enhanced by our new pivot generation method called PGM2. In our experiments using real image datasets where each object is represented as a high-dimensional vector and L_1 distance is recommended as their dissimilarity, we show that our proposed method achieved a reasonably high acceleration performance.

1. はじめに

近年，Web 上には大量のマルチメディアデータが蓄積されており，これらのデータに含まれるオブジェクト間の距離に基づいてオブジェクト群をいくつかのグループにクラスタリングする技術が注目を浴びている．クラスタリング手法の代表例として，理論的に解品質が保証されている [Leskovec 07] K-medoids 法の貪欲解法があげられるが，大規模データに対して多大な計算量が必要となる．本稿では，S 次元のベクトルで表現される N オブジェクト群に対して，時間計算量 $O(N^2S)$ で距離を計算したのち，$O(KN^2)$ で K-medoids アルゴリズムを実行する．メモリに十分な余裕がある場合は，空間計算量 $N(N-1)/2$ の距離を保持できるが，オブジェクト数 N が大規模になった場合，メモリ上に保持するのは困難となるため，K-medoids アルゴリズムの各ステップで $N(N-1)/2$ の距離を再計算しなければならない．従って，時間計算量は $O(KN^2S)$ となる．本稿の実験では，$K = 100$，$N = 1{,}000{,}000$，$S = 340$ のデータを用いるため，非常に計算時間がかかることになる．

K-medoids アルゴリズムの貪欲解法により得られる解を変えることなく距離計算の回数を減らすために，遅延評価（Lazy Evaluation）技術 [Leskovec 07] と ピボット枝刈り（Pivot Pruning）技術 [Hjaltason 03] を組み合わせる．本稿では，ピボット生成アルゴリズムである PGM (Pivot Generation based on Manhattan distance) 法 [Kobayashi 14] に着目する．L_1 (マンハッタン) 距離は，カラー画像のヒストグラム間の非類似度 [?] のように，多くの場合に L_2 より自然に用いられる．PGM 法の大きな利点は，ピボットベクトルの各次元の要素の値が，少ない計算コストの反復により独立かつ最適に更新されることである．さらに，有限回の反復により収束することが保証されている．しかし，オブジェクトとピボットの全ペア間の距離を計算後，ピボット生成の目的関数を計算するために $O(N^2)$ の計算量がかかる．この計算量を $O(N \log N)$ に減らすために，2 つのピボットのみを用いて効率的に目的関数値を計算する PGM2 (Pivot Generation on Manhattan distance using 2 pivots) 法と呼ぶ拡張したピボット生成法を用いる．

上述したように，本稿ではメモリ領域を十分に確保できない状況において，多くのオブジェクト群をクラスタリングするタスクを扱う．そのために，遅延評価技術とピボット枝刈り技術を組み合わせ，L_1 距離に基づく K-medoids 法の貪欲解法を高速化する新たなアルゴリズムを提案する．L_1 距離が標準的に用いられる画像検索データセットである CoPhIR [Bolettieri 09] を対象とした評価実験により，計算効率の観点から提案手法のパフォーマンスを評価する．

2. 関連研究

最もシンプルな K-medoids アルゴリズムとして，PAM [Kaufman 86] が挙げられる．PAM は K 個のメドイドオブジェクトを非メドイドオブジェクトとスワップすることで反復的に目的関数を最適化するシンプルな手法であるが，計算量がかかるため大規模データへの適用は困難である．Park と Jun は高速な K-medoids アルゴリズムとして，全オブジェクトペア間の距離をメモリ上に保持する手法 [Park 09] を提案したが，膨大なメモリ空間が必要になるため，大規模データへの適用は困難である．また，大規模データに適用するために，様々なサンプリングアルゴリズム [Jiang 02, Aggarwal 09] が提案されているが，オリジナルデータセットから抽出したサンプルオブジェクトによる近似的な目的関数に基づきクラスタリングするため，サンプル数に依存した不安的な結果になる場合がある．

Paterlini ら [Paterlini 11] は，本稿の提案法と同様に，ピボットを用いた高速な K-medoids アルゴリズムを提案している．この手法では，初期メドイドを効率的に選ぶことで収束までの反復回数を削減している．本稿では，目的関数のサブモジュラ性により高い解品質が理論的に保証され，かつ，一意な解を出力する K-medoids アルゴリズムの貪欲解法に焦点を当てるため，単純に Paterlini らの手法を適用することはできない．しかし，提案する PGM2 法は，貪欲解法だけでなく様々な既存手法に適用することが可能であり，より高速なアルゴリズムの導出も期待できる．

本稿で用いるピボット枝刈り技術は，距離の三角不等式を利用して距離計算不要なオブジェクト群を枝刈りする手法である．以下に，三角不等式に基づくクラスタリングの高速化に関する研究について述べる．Elkan アルゴリズム [Elkan 03]

は，Lloyd アルゴリズム高速化の代表例であり，三角不等式から導出される距離の下限と上限を効果的に用いることで，不要な距離計算を省き K-means アルゴリズムを高速化している．Hamerly アルゴリズム [Hamerly 10] は，Elkan アルゴリズムを拡張した手法で，各オブジェクトに対して距離の下限値のみを保持することで空間計算量を削減している．

3. 提案手法

3.1 K-medoids 法の再訪

各オブジェクトがベクトルで表現され，各オブジェクトペア間に距離が定義されているとき，K-medoids アルゴリズムは全オブジェクトを K 個の類似オブジェクトからなるグループに分割する．オブジェクトベクトルを $\mathcal{X} = \{\mathbf{x}_1, \cdots, \mathbf{x}_N\}$ とすると，オブジェクトベクトル間の L_1 距離 $d(\mathbf{x}_n, \mathbf{x}_m)$ は以下のように定義される：

$$d(\mathbf{x}_n, \mathbf{x}_m) = \sum_{s=1}^{S} |x_{n,s} - x_{m,s}|. \quad (1)$$

ここで，$x_{n,s}$ は S 次元のオブジェクトベクトル $\mathbf{x}_n$ の s 次元目の要素の値を意味する．K-medoids 法では，メドイド集合 $\mathcal{R} \subset \mathcal{X}$ に対して，以下の目的関数を最小化する：

$$F(\mathcal{R}) \quad = \quad \sum_{n=1}^{N} \min_{\mathbf{r}_k \in \mathcal{R}} d(\mathbf{x}_n, \mathbf{r}_k). \quad (2)$$

目的関数 $F(\mathcal{R})$ を最小化するために，本稿では貪欲解法に着目する．貪欲法では，既に選定済みのメドイドの集合 $\mathcal{R}$ を固定し，メドイド候補オブジェクト $\mathbf{x}_n$ に対して以下の Marginal Gain（以下，MG）を計算する：

$$\begin{aligned} g(\mathbf{x}_n; \mathcal{R}) \quad &= \quad F(\mathcal{R} \cup \{\mathbf{x}_n\}) - F(\mathcal{R}) \\ &= \quad \sum_{\mathbf{x}_m \in \mathcal{X} \setminus \mathcal{R}} \min\{d(\mathbf{x}_m, \mathbf{x}_n) - \mu(\mathbf{x}_m; \mathcal{R}), 0\} \quad (3) \end{aligned}$$

ここで，$\mathcal{R} \neq \emptyset$ なら $\mu(\mathbf{x}_m\ ; \mathcal{R}) = \min_{\mathbf{r} \in \mathcal{R}} d(\mathbf{x}_m, \mathbf{r})$ であり，それ以外の場合は $\mu(\mathbf{x}_m\ ; \emptyset) = 0$ である．貪欲解法を以下に示す：

1. 初期化：$k \leftarrow 1$，$\mathcal{R}_0 \leftarrow \emptyset$；
2. $k \leq K$ の間，以下を繰り返す：
   (a) 選定：$\hat{\mathbf{r}}_k = \underset{\mathbf{x}_n \in \mathcal{X} \setminus \mathcal{R}_{k-1}}{\arg\min}\ g(\mathbf{x}_n; \mathcal{R}_{k-1})$；
   (b) 追加：$\mathcal{R}_k \leftarrow \mathcal{R}_{k-1} \cup \{\hat{\mathbf{r}}_k\}$；
   (c) $k \leftarrow k + 1$
3. 分割：$\mathcal{X}^{(k)} = \{\mathbf{x}_m \in \mathcal{X}; \mathbf{r}_k = \underset{\mathbf{r} \in \mathcal{R}}{\arg\min}\{d(\mathbf{x}_m, \mathbf{r})\}\}$；

3.2 高速化アルゴリズム

アルゴリズム 1 に提案法の疑似コードを示す．ここで，mg は MG $g(\mathbf{x}_n; \mathcal{R}_{k-1})$ を保持する一時変数であり，$LB(\mathbf{x}_n, \mathbf{x}_m; \mathcal{P})$ は後述する距離 $d(\mathbf{x}_n, \mathbf{x}_m)$ の下限値を表す．提案アルゴリズムでは，後述する高速選択技術（FastSelection）により第 1 メドイド $\mathbf{r}_1$ を計算量 $O(SN \log N)$ で高速に求める．次いで，一般化ピボット生成技術（PivotGeneration）により 2 つのピボット $\mathcal{P} = \{\mathbf{p}_1, \mathbf{p}_2\}$ を計算量 $O(TN \log N)$ で効率的に求める．以降，遅延評価技術とピボット枝刈り技術を用いて，第 2 メドイドから第 K メドイドまでを求めていく．

1. 高速選択技術（FastSelection：FS）
   本稿で適用する L_1 距離に基づく目的関数は，以下に示すように次元ごとに独立して計算できるため，その利点を活かし高速に第 1 メドイドを求める．目的関数における n 番目のオブジェクトベクトルの第 s 次元の値に関する部分 $F_s(n) = \sum_{m=1}^{N} |x_{n,s} - x_{m,s}|$ を抜き出す．ここで説明の便宜上，$x_{1,s} \leq \cdots \leq x_{N,s}$ とする．オブジェクト $\mathbf{x}_n$ に関する目的関数値は $F(\{\mathbf{x}_n\}) = \sum_{s=1}^{S} F_s(n)$ と書ける．ここで，係数 $b(n) = N - 2n + 1$ と $\beta(n) = \sum_{m=n+1}^{N} x_{m,s} - \sum_{m=1}^{n-1} x_{m,s}$ を用いると，$F_s(n) = -b(n)x_{n,s} + \beta(n)$ と書き直せる．従って，$\beta(1) = \sum_{m=2}^{N} x_{m,s}$ と初期化すると，以降 $\beta(n+1) = \beta(n) - x_{n,s} - x_{n+1,s}$ と計算できるため，$n = 1$ から N まで逐次的に $F_s(n) = -b(n)x_{n,s} + \beta(n)$ を $O(N)$ で計算できる．従って，オブジェクトベクトルの各次元の値を $x_{1,s} \leq \cdots \leq x_{N,s}$ となるように $O(SN \log N)$ で昇順ソートすれば，全ての n に対する目的関数値 $F(\{\mathbf{x}_n\})$ を $O(SN)$ で計算できる．最終的に，計算量 $O(SN \log N)$ で第 1 メドイドを選択できるため，単純に求めた場合 $O(SN^2)$ と比較して大幅に高速化できる．
2. 遅延評価技術（LazyEvaluation：LE）
   第 k メドイド選択（$k \geq 2$）のステップで適用される遅延評価技術 [Leskovec 07] は，第 k メドイドの候補オブジェクト $\mathbf{x}_n \in \mathcal{X}$ に関する MG $g(\mathbf{x}_n; \mathcal{R})$ の上限値 $UB(\mathbf{x}_n)$ を

Algorithm 1 Proposed algorithm

Input: $\mathcal{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}, K$
Output: $\mathcal{R}_K = \{\mathbf{r}_1, \ldots, \mathbf{r}_K\} \subset \mathcal{X}$
Initialize: $\mathcal{R}_0 \leftarrow \emptyset$, $g(\mathbf{x}; \mathcal{R}_0) \leftarrow 0$ for each object $\mathbf{x}$
$\mathbf{r}_1 \leftarrow$FastSelction($\mathcal{X}$), $\mathcal{R}_1 \leftarrow \mathcal{R}_0 \cup \{\mathbf{r}_1\}$ ▷ $O(SN \log N)$
$\mathcal{P} \leftarrow$PivotGeneration($\mathcal{X}$) ▷ $O(TN \log N)$
for $k = 2$ to K **do**
  for $\rho = 1$ to N **do**
    $n = rank(\rho)$
    if $g(\mathbf{x}_n; \mathcal{R}_{k-2}) < g_k^*$ **then**
      continue ▷ (Lazy evaluation)
    end if
    Initialize: $mg \leftarrow 0$
    for $m = 1$ to N **do**
      if $LB(\mathbf{x}_n, \mathbf{x}_m; \mathcal{P}) \geq \mu(\mathbf{x}_m; \mathcal{R}_{k-1})$ **then**
        continue ▷ (Pivot pruning)
      end if
      $d(\mathbf{x}_n, \mathbf{x}_m) \leftarrow \sum_{s=1}^{S} |x_{n,s} - x_{m,s}|$
      if $d(\mathbf{x}_n, \mathbf{x}_m) < \mu(\mathbf{x}_m; \mathcal{R}_{k-1})$ **then**
        $mg \mathrel{+}= \mu(\mathbf{x}_m; \mathcal{R}_{k-1}) - d(\mathbf{x}_n, \mathbf{x}_m)$
      end if
    end for
    $g(\mathbf{x}_n; \mathcal{R}_{k-1}) \leftarrow mg$
    if $g_k^* < mg$ **then**
      $g_k^* \leftarrow mg$
    end if
  end for
  $\mathbf{r}_k \leftarrow \underset{\mathbf{x}_n \in \mathcal{X} \setminus \mathcal{R}_{k-1}}{\arg\max}\ g(\mathbf{x}_n; \mathcal{R}_{k-1})$, $\mathcal{R}_k \leftarrow \mathcal{R}_{k-1} \cup \{\mathbf{r}_k\}$
  for $n = 1$ to N **do**
    $d(\mathbf{x}_n, \mathbf{r}_k) \leftarrow \sum_{s=1}^{S} |x_{n,s} - r_{k,s}|$
    $\mu(\mathbf{x}_n; \mathcal{R}_k) \leftarrow \min\{d(\mathbf{x}_n, \mathbf{r}_k), \mu(\mathbf{x}_n; \mathcal{R}_{k-1})\}$
  end for
  rank $= Sort(\{g(\mathbf{x}_1; \mathcal{R}_{k-1}), \ldots, g(\mathbf{x}_N; \mathcal{R}_{k-1})\})$
end for

用いる．各オブジェクトの上限値を $UB(\mathbf{x}_n) \leftarrow F(\{\mathbf{x}_n\})$ と初期化し，第 h メドイド選択ステップにおいて実際に $g(\mathbf{x}_n;\mathcal{R}_h)$ が計算された場合，$UB(\mathbf{x}_n) \leftarrow g(\mathbf{x}_n;\mathcal{R}_h)$ と更新する．サブモジュラ性により $g(\mathbf{x}_n;\mathcal{R}_k) \leq UB(\mathbf{x}_n)$ が保証されている（$k > h$）．従って，第 k メドイドを選択するステップの時点で最良の MG を g_k^* とすると，$UB(\mathbf{x}_n) \leq g_k^*$ を満たすオブジェクト $\mathbf{x}_n$ に関して MG $g(\mathbf{x}_n;\mathcal{R}_k)$ の計算を省略することができる．つまり，$\mathbf{x}_n$ は目的関数増加に貢献しないため，MG を計算しても意味がなく省略できるということである．メドイド候補の各オブジェクトに関して条件 $UB(\mathbf{x}_n) \leq g_k^*$ を満たすかチェックするが，$UB(\mathbf{x}_n)$ の値で候補オブジェクトを降順にソートすれば，比較的早期により良い g_k^* を得られる．したがって，ソート済みの候補オブジェクトリストを順に走査し，条件 $UB(\mathbf{x}_n) \leq g_k^*$ を満たすオブジェクトが得られた時点で，それ以降のオブジェクトに関して MG 計算を一括して省略できる．

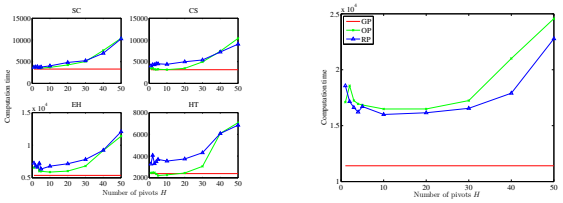
3. ピボット枝刈り技術（PivotPruning）
ピボット枝刈り技術 [Zezula 06] は，MG $g(\mathbf{x}_n;\mathcal{R}_k)$ を計算する前に，距離 $d(\mathbf{x}_n,\mathbf{x}_m)$ の下限距離 $LB(\mathbf{x}_n,\mathbf{x}_m;\mathcal{P})$ を用いて枝刈り条件 $d(\mathbf{x}_n,\mathbf{x}_m) \geq \mu(\mathbf{x}_n;\mathcal{R})$ をチェックする．本稿では，PGM 法を拡張し，2 個のピボットを生成する PGM2 法を提案し，得られたピボット集合 $\mathcal{P} = \{\mathbf{p}_1,\mathbf{p}_2\}$ から下限距離を計算する．式 (3) を見ると，枝刈り条件 $d(\mathbf{x}_n,\mathbf{x}_m) \geq \mu(\mathbf{x}_n;\mathcal{R})$ を満たすオブジェクト $\mathbf{x}_n$ に関して，ゲインを加算する必要がないことがわかる．下限距離は，距離の三角不等式から $LB(\mathbf{x}_n,\mathbf{x}_m;\mathcal{P}) = \max_{\mathbf{p}\in\mathcal{P}} |d(\mathbf{x}_n,\mathbf{p}) - d(\mathbf{x}_m,\mathbf{p})| \leq d(\mathbf{x}_n,\mathbf{x}_m)$ により計算する．以降，PivotGeneration により生成したピボット集合による枝刈り技術を GP と表記する．

提案法では，FS 技術，LE 技術，GP 技術の順で適用する．

4. 評価実験

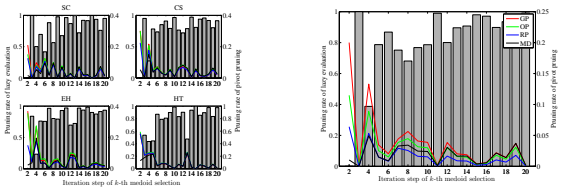
提案手法の計算効率を評価するために，CoPhIR (Content-based Photo Image Retrieval) テストコレクション [Bolettieri 09] に提案手法を適用し，計算時間や枝刈り率をベースラインと比較する．このテストコレクションから，L_1 距離を採用している 4 つの記述子：scalable color (SC), color structure (CS), edge histogram (EH), and homogeneous texture (HT) をオブジェクトの特徴ベクトルとして用いる．それぞれのベクトルの次元数は，64, 64, 12, 80, 62 である．さらに，5 つの記述子を結合した 340 次元のベクトルを MIX として評価実験に用いる．そして，各記述子に対して，オブジェクト数 $N = 1,000,000$ までのオブジェクトをランダムに選び，複数のデータセットを用意する．

評価実験のためのベースライン比較手法として，LE 技術のみを用いた LE 法，LE 技術と FS 技術を合わせた FS 法，外れ値オブジェクトをピボットとして選択しピボット枝刈り技術を適用する OP 法，ランダムに選んだオブジェクトをピボットとして選択し枝刈りする RP 法，選出済みのメドイドをピボットとする MD 法を採用する．OP 法における外れ値オブジェクトは，第 1 メドイドを選出後，$\hat{\mathbf{q}}_1 = \arg\max_{\mathbf{v}\in\mathcal{X}} d(v,\mathbf{r}_1)$，$P \leftarrow P \cup \{\hat{\mathbf{q}}_1\}$ として第 1 外れ値オブジェクトを選出し，つづいて $\hat{\mathbf{q}}_h = \arg\max_{\mathbf{v}\in\mathcal{X}} \min_{\mathbf{p}\in P} d(\mathbf{v},\mathbf{p})$，$P \leftarrow P \cup \{\hat{\mathbf{q}}_h\}$ として h 番目の外れ値オブジェクトを選出する手続きを H 個の外れ値オブジェクトを選出するまで繰り返す．上記の比較手法の命名規則に従い，以後，提案手法を GP 法と呼ぶことにする．



(a) Each descriptor. (b) MIX.

図 1: ピボット数を変化させた際の計算時間

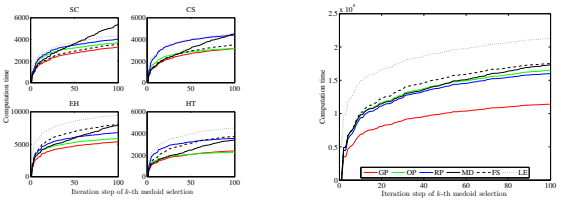


(a) Each descriptor. (b) MIX.

図 2: 貪欲法 k 反復目での枝刈り率

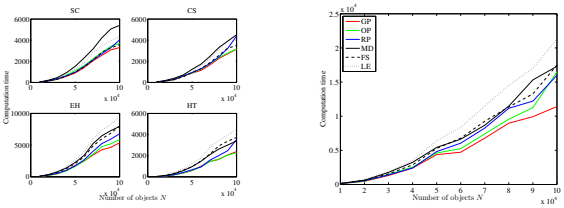
まず最初に，3 つのピボット枝刈り技術に基づく手法：GP 法，OP 法，RP 法について，ピボット数を $1 \leq H \leq 50$ と変化させた際の計算時間について評価する．対象とするオブジェクト数は $N = 100,000$，メドイド数は $K = 100$ と固定する．また，提案手法である GP 法のピボット数は $H = 2$ である．図 1 に，各ディスクリプタに対する計算時間をプロットした．図 1 を見ると，GP 法は いずれのピボット数での OP 法と RP 法と比較しても高速に解を得られることがわかる．さらに，OP 法と RP 法では，ピボット数を増やすにつれて計算時間が増加する傾向にあることもわかる．一般論として，OP 法と RP 法では適切なピボット数を与えることで，良いパフォーマンスを得られることがわかるが，現実問題では最適なピボット数を前もって知ることは不可能である．一方 GP 法では，ピボット数を 2 に固定するため，前もってパラメータであるピボット数を指定する必要がない．これらの結果より，2 個のピボットを生成し枝刈りする提案手法の有効性が示唆された．

次に，メドイド数を $2 \leq k \leq 20$ と変化させた際の枝刈り率について評価する．$LE(k)$, $GP(k)$, $OP(k)$, $RP(k)$, $MD(k)$ をそれぞれ LE 技術，GP 技術，OP 技術，RP 技術，MD 技術により距離計算が省略されたノードペアの集合とする．そして，各技術による枝刈り率 $\alpha(\cdot)$ を $\alpha(LE(k)) = |LE(k)|/N^2$, $\alpha(GP(k)) = (|LE(k) \cup GP(k)| - |LE(k)|)/N^2$, $\alpha(OP(k)) = (|LE(k) \cup OP(k)| - |LE(k)|)/N^2$, $\alpha(RP(k)) = (|LE(k) \cup RP(k)| - |LE(k)|)/N^2$, $\alpha(MD(k)) = (|LE(k) \cup MD(k)| - |LE(k)|)/N^2$ とする．各手法において，LE 技術は各ピボット枝刈り技術より先に適用されることに注意されたい．対象とするオブジェクト数は $N = 100,000$，OP 法と RP 法のピボット数は $H = 10$ とした．図 2 に，各ディスクリプタに対する枝刈り率をプロットした．グレーの棒グラフ（左側縦軸）は枝刈り率 $\alpha(LE(k))$ を表し，赤，緑，青，黒の折れ線グラフ（右側縦軸）は $\alpha(GP(k))$, $\alpha(OP(k))$, $\alpha(RP(k))$, $\alpha(MD(k))$ を表す．図 2 を見ると，$k = 2, 4$ の時，LE 技術は目的関数の MG 計算における距離計算省略ができず低い枝刈り率となっている．これは $k = 2, 4$ の時，各オブジェクトの上限 $UB(\mathbf{w})$ が目的関数の MG $g(\mathbf{w};R)$ の非常に粗い近似になっていることが原因と考えられる．さらに，$k = 2$ の時，MD 技術はピボット数が 1 になるため，枝刈り率が比較的小さいことがわかる．一方，GP 技術，OP 技術，RP 技術による枝刈りは，LE 技



(a) Each descriptor. (b) MIX.

図 3: 貪欲法 k 反復目での計算時間



(a) Each descriptor. (b) MIX.

図 4: オブジェクト数を変化させた際の計算時間

術による枝刈りの後で実行されるにもかかわらず，高い枝刈り率を実現できていることが見て取れる．すなわち，これらのピボット枝刈り技術は LE 技術に対して補完的に働いていることが示唆される．

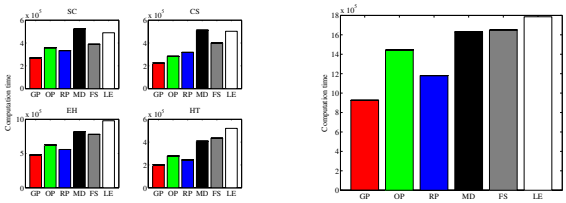
図 3 に，メドイド数を $2 \leq k \leq 100$ と変化させた際の計算時間をプロットした．各ディスクリプタに対する計算時間を見ると，GP 法は他の手法と比較して十分に高速であることがわかる．特に本稿の評価実験では，最先端技術である LE 法より 2 倍から 3 倍の速度を実現している．また，MD 法はメドイド数が増えるにつれて計算時間がかかることがわかる．

図 4 に，オブジェクト数を $10,000 \leq N \leq 100,000$ と変化させた際の計算時間をプロットした．メドイド数は $K = 100$，OP 法と RP 法におけるピボット数は $H = 10$ とした．図 4 から，各ディスクリプタに対して，最先端技術である LE 法と比較して，提案手法である GP 法は 2 倍程度高速であることがわかる．さらに，比較に用いた手法と GP 法では，オブジェクト数が増加するにつれ計算時間の差が大きくなる傾向にあることもわかる．図 5 に，オブジェクト数 $N = 1,000,000$ の時の計算時間をプロットした．図 5 からも，大規模なオブジェクト群に対して比較手法より大幅に高速に解を得られることがわかる．MIX ディスクリプタに対する LE 法の計算時間はおよそ 21 日なのに対して，GP 法では 11 日で解が得られており，より大規模なデータに対してはこの差は大きく働くと考えられる．

5. おわりに

本稿では，遅延評価技術とピボット枝刈り技術を組み合わせることで，L_1 距離に基づく K-medoids クラスタリングの貪欲解法を高速に求めるアルゴリズムを提案した．特に，新たに提案した PGM2 法が生成する 2 個のピボットを利用して効率的な枝刈りを実現する点に新規性がある．L_1 距離を採用する CoPhIR データセットのディスクリプタを用いた評価実験では，提案手法により高速に K-medoids 法の貪欲解を得られることを確認した．今後は，さらに多様なデータを用いて提案手法の有効性を確認していくつもりである．

謝辞 本研究は，JSPS 科研費 16K16154 の助成を受けたものです．



(a) Each descriptor. (b) MIX.

図 5: 計算時間（$N = 1,000,000$，$K = 100$）

参考文献

[Aggarwal 09] Aggarwal, A., Deshpande, A., and Kannan, R.: Adaptive Sampling for k-Means Clustering, in *Proc. of the 12th Int'l Workshop and 13th Int'l Workshop on Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pp. 15–28, Berlin, Heidelberg (2009), Springer-Verlag

[Bolettieri 09] Bolettieri, P., Esuli, A., Falchi, F., Lucchese, C., Perego, R., Piccioli, T., and Rabitti, F.: CoPhIR: a Test Collection for Content-Based Image Retrieval, *CoRR*, Vol. abs/0905.4627v2, (2009)

[Bustos 03] Bustos, B., Navarro, G., and Chavez, E.: Pivot selection techniques for proximity searching in metric spaces, *PATTERN RECOGNITION LETTERS*, Vol. 24, No. 14, pp. 2357–2366 (2003)

[Elkan 03] Elkan, C.: Using the Triangle Inequality to Accelerate k-Means., in Fawcett, T. and Mishra, N. eds., *Machine Learning, Proc. of the 12th Int'l Conf. (ICML 2003)*, pp. 147–153, AAAI Press (2003)

[Hamerly 10] Hamerly, G.: Making k-means Even Faster, in *SIAM Int'l Conf. on Data Mining*, pp. 130–140 (2010)

[Hjaltason 03] Hjaltason, G. R. and Samet, H.: Index-driven Similarity Search in Metric Spaces (Survey Article), *ACM Trans. Database Syst.*, Vol. 28, No. 4, pp. 517–580 (2003)

[Jiang 02] Jiang, C., Li, Y., Shao, M., and Jia, P.: Accelerating Clustering Methods through Fractal Based Analysis, in *Proc. of the 1st Workshop on application of self-similarity and fractals in data mining(KDD2002 Workshop)* (2002)

[Kaufman 86] Kaufman, L. and Rousseeuw, P.: *Clustering Large Data Sets (with discussion)*, pp. 425–437 (1986)

[Kobayashi 14] Kobayashi, E., Fushimi, T., Saito, K., and Ikeda, T.: Similarity Search by Generating Pivots Based on Manhattan Distance, in *Proc. of the 13th Pacific Rim International Conference on Artificial Intelligence(PRICAI2014)*, pp. 435–446, Springer International Publishing (2014)

[Leskovec 07] Leskovec, J., Krause, A., Guestrin, C., Faloutsos, C., VanBriesen, J., and Glance, N.: Cost-effective Outbreak Detection in Networks, in *Proc. of the 13th ACM SIGKDD Int'l Conf. on Knowledge Discovery and Data Mining*, pp. 420–429, New York, NY, USA (2007), ACM

[Park 09] Park, H.-S. and Jun, C.-H.: A simple and fast algorithm for K-medoids clustering, *Expert Systems with Applications*, Vol. 36, No. 2, Part 2, pp. 3336 – 3341 (2009)

[Paterlini 11] Paterlini, A. A., Nascimento, M. A., and Traina, C. J.: Using Pivots to Speed-Up k-Medoids Clustering., *Journal of Information and Data Management*, Vol. 2, No. 2, pp. 221–236 (2011)

[Zezula 06] Zezula, P., Amato, G., Dohnal, V., and Batko, M.: *Similarity Search: The Metric Space Approach*, Vol. 32 of *Advances in Database Systems*, Springer, 1st edition (2006)