数理解析研究所講究録
第1799巻 2012年 171-178

2011年度冬のLAシンポジウム [S6]

mm-GNATにおける分割点集合の選択手法に関する研究

謝 評芳 *　　大西 建輔 †

1 はじめに

時系列データやマルチメディアデータに対して類似検索を行う場合, これらのデータから特徴量と呼ばれる多次元ベクトルを抽出し, この特徴量の空間で問い合わせに"近い"点を出力する. 近い点を出力するために行われるのがε近傍検索である. ε近傍検索とは, 問い合わせ点qと問い合わせ半径εを与えたとき, qから距離がε以下の点をすべて出力するものである. ε近傍検索を効率的に行うための索引構造は, 数多く提案されており, それらの索引構造に関しては, Bohmによる概説 [1], Chavezによる概説 [2] などがある. これらの索引構造のほとんどは, 特定の距離 (通常はユークリッド距離) を用いての検索を想定しており, 索引もその距離を用いて構築されている. したがって, 索引構造は使用する距離により変化することになる.

Yiらは, 多モデル対応 (multi-modality support) という概念を提案した [3]. この概念は, データ空間には様々な類似度モデル (もしくは, 距離) が導入される可能性があり, その空間での索引構造は, どのモデルに対しても, 対応が可能であるように索引構造を構築するという考え方である. 最も単純な実現方法としては, 幾つかのモデルに対する索引構造を, それぞれ構築しておくという方法である. しかし, この方法は, モデルが増える度に索引構造を増やす必要があり, モデル数に応じた記憶領域や構築時間が必要となってしまう. もし, ある1つ索引構造だけで, 多くのモデルでの問い合わせが行える構造が構築できれば, 索引構造の構築コストや記憶領域の削減が可能となる.

*東海大学大学院理学研究科
†東海大学理学部

Yiらは, L_p距離空間を対象に, 多モデル対応のための手法を提案した [3]. この手法は, ε近傍検索の問い合わせ半径を拡大することで実現されている. 本論文で対象としている索引構造mm-GNAT[7]も同じ機能をもっている. mm-GNATでは, 索引構造として保持される距離の範囲を拡大することでL_p距離空間への多モデル対応を実現している.

多くの類似検索の索引構造では, 分割点と呼ばれる点の集合を使って, 全空間をいくつかの部分空間に分割する. この部分空間をクラスタと呼ぶ. そして, 問い合わせ点と分割点の距離により, 一部のクラスタを検索対象から取り除く. この処理を枝刈りと呼ぶ. 最後に, 問い合わせ点と残ったクラスタに含まれるすべての点との距離を計算し, 距離がε以下の点を出力する. つまり, 分割点集合の選択方法は, 索引構造の検索性能に大きく影響している. よい分割点集合を用いることで, 検索対象から取り除けるクラスタが多くなり, 距離計算やディスクアクセスといった検索コストを削減できる.

分割点集合の選択手法はいくつか提案されている. 既に決定した分割点から遠い点を分割点としていくGNAT法 [4], サンプリング点からの最大距離が遠い点を分割点としていくD-index法 [5], 点間距離が最大距離のα倍 ($0 < \alpha < 1$) 以上になるような点集合を作成するSSS法 [6], クラスタが規則的になるような点集合を生成する格子配置法 [8] などがある. 本稿では, これらの選択手法で作成した分割点集合を用いてmm-GNATを構築し, mm-GNATでのε近傍検索の性能, 特に距離計算回数を調べる.

本稿の構成は, 以下の通りである. 第2節では, 関連研究として, 索引構造mm-GNATと分割点の選択手法GNAT法, D-index法, SSS法と格子配置法に

ついて説明する. 第3節では, 計算機実験の結果を示す. 第4節では, 実験結果についての議論を述べる. 最後の第5節では, まとめを述べる.

2 関連研究

本節では, 本研究で対象とした索引構造 mm-GNAT といくつかの分割点集合の選択手法を紹介する.

2.1 mm-GNAT

mm-GNAT は GNAT[4] を拡張したものである. GNAT ではすべての点をそれぞれの点が最も近い分割点 SP_i に所属させ, クラスタ D_{SP_i} を構成する. 索引構造として分割点からそれぞれのクラスタへの最小距離と最大距離を range として保存する. GNAT を用いた ε 近傍検索を行う時, 問い合わせ点と分割点との距離 r を計算し, 条件

$$[r-\varepsilon, r+\varepsilon] \cap \mathrm{range}(SP_i, D_{SP_j}) = \emptyset$$

を満たしたクラスタ D_{SP_j} を枝刈りすることができる. また, L_p 距離関数には, 次の性質がある.

$$\mathrm{dist}_\infty(x,y) \leq \mathrm{dist}_p(x,y) \leq \mathrm{dist}_1(x,y).$$

mm-GNAT はこの性質を利用し, range の最小距離を L_∞ 距離で, 最大距離を L_1 距離で計算する. つまり, 索引構造として保持される range を適切に拡大することで, 任意の L_p 距離関数に対応可能としている.

2.2 分割点集合の選択手法

2.2.1 GNAT 法

この方法は Brin[4] により提案された GNAT のための分割点集合の選択手法である. そのため本稿では GNAT 法と呼ぶ. GNAT 法では, サンプリングをおこない, 大きさ $3k$ の点集合を作る. この点集合から決定した分割点との距離和が最大となる点を選出し, 分割点とする手法である. k 個の分割点を選択する GNAT 法をアルゴリズム 1 に示す. この手法の時間計算量は $O(k^3)$ 時間 である.

アルゴリズム 1 GNAT 法

Input S: データ集合, k: 分割点の数;
Output T: 点数が k の分割点集合;

1. S から $3k$ 個の点を選び, 集合 S_{3k} を作る;
2. $p_0 \in S_{3k}$ をランダムにとる. p_0 から S_{3k} の中で最も遠い点を p_1 とする, $T := \{p_1\}$;
3. $i := 2$;
   while $|T| < k$ **do**
   (a) $p_i \in S_{3k}$ s.t. $\max_S \sum_{q \in T} \mathrm{dist}(p_i, q)$,
   $T := T \cup \{p_i\}$;
   (b) $i := i + 1$;

2.2.2 D-index 法

Bustos らは [5] で平均値を用いる分割点の選択手法 D-index 法を提案した. D-index 法はサンプリングした点集合までの距離の最大値の平均値を元に, 分割点集合に点を追加している. k 個の分割点を選択する D-index 法をアルゴリズム 2 に示す. この手法の時間計算量は $O(mAk)$ 時間である. ただし, m はサンプリングの点数, A はペアの数である.

[5] では, ペアの数 A を大きく, サンプリング集合のデータ点の数 m を小さくした分割点集合がよいと主張している. また, 10 万点における検索に, $A = 100000, m = 50$ を用いている. そこで本研究の計算機実験でもこの値を用いた.

2.2.3 SSS 法

Brisaboa らは [6] で点集合のすべての点間の最大距離 M を元にした分割点集合の選択手法 SSS 法を

アルゴリズム 2 D-index 法

Input S: データ集合, k: 分割点の数;
Output T: 点数が k の分割点集合;

1. S からランダムで A ペアのデータ点 $(x_1, y_1), ..., (x_A, y_A)$ を選ぶ;
2. **for** $i := 1$ **to** A **do** $D[i] := 0$;
3. $i := 0, T := \{\}$;
   while $i < k$ **do**
   (a) サンプリングをおこない, S から m 個のデータ点を選び, 集合 S_m を作る;
   (b) すべての $p_i \in S_m$ に対して
   for $j := 1$ **to** A **do**
   $DD_{p_i}[j] :=$
   $\max(|\mathrm{dist}(p_i, x_j) - \mathrm{dist}(p_i, y_j)|, D[j])$;
   (c) $\mu_{p_i} := (DD_{p_i}[1] + ... + DD_{p_i}[A])/A$;
   (d) $p_i : \mu_{p_i}$ が最大のものとする.
   $T := T \cup \{p_i\}, D := DD_{p_i}$;
   (e) $i := i + 1$;

アルゴリズム 3 SSS 法

Input S: データ集合, α: パラメータ;
Output T: 分割点集合;

1. S において, すべてのペアの距離を計算し, その最大値を M とする;
2. $p_0 \in S$ をランダムでとる. $T := \{p_0\}$;
3. **for** $i := 1$ **to** $|S|$ **do**
   (a) $p_i \in S$;
   (b) **if** p_i と T のすべての点との距離が αM 以上 **then** $T := T \cup \{p_i\}$;

提案した. SSS 法ではまず M を計算する. その上で, 分割点集合のすべての点との距離が $\alpha M (0 < \alpha < 1)$ 以上となる点を追加していく. この手法は, 決めた個数の分割点を選択することはできないが, パラメータ α を用いてある程度は制御できる. SSS 法をアルゴリズム 3 に示す. この手法の時間計算量は $O(n^2)$ 時間である. ただし, n はデータ集合の点数である.

2.2.4 立方格子配置法 (SQUARE 法)

立方格子配置は, 点を立方体に配置する手法で, d 次元空間では次のように定義される [8].

$$\{(\overbrace{2\mathbb{Z}a, 2\mathbb{Z}a, ..., 2\mathbb{Z}a}^{d})|a > 0, \mathbb{Z} : \text{整数全体}\}.$$

図 1 は, 2 次元空間におけるの立方格子配置の例で

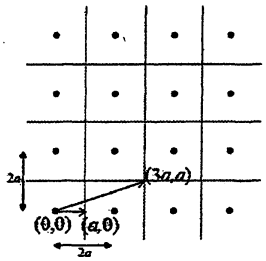

図 1: 2 次元空間における立方格子配置

ある. このように分割点を配置した場合, 各クラスタは立方体となる.

SQUARE 法を使って, 分割点集合を作成する方法を説明する. データ集合から次元の最小値 min と最大値 max を計算し, 等分数 $2c$ で, 各次元の min と max の間を分割する. このとき, 奇数番目の点を分割点とする. また, 作成される分割点の数は c^d となる.

SQUARE 法を用いると, mm-GNAT の range の範囲をおさえることができる. 図 1 の例を用いて説明する. 図 1 の左下にあるクラスタの分割点の座標を $(0, 0)$ とする. この分割点から隣接するクラスタへの L_∞ 距離での最小距離と L_1 距離での最大距離は点 $(a, 0)$ と点 $(3a, a)$ との距離となるため, それぞれ a と $4a$ となる. つまり, 隣接するクラスタ

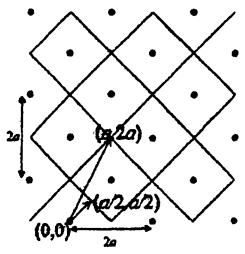

図 2: 2 次元空間における面心立方格子配置

の range の範囲は $[a, 4a]$ に収まる. d 次元で考えると, 分割点から隣接するクラスタへの最小距離と最大距離はそれぞれ a と $(d+2)a$ となる. つまり, range の範囲は $[a, (d+2)a]$ に収まる. a と c の間には $2ac = \max - \min$ の関係が成立する.

2.2.5 面心立方格子配置法 (FC 法)

面心立方格子配置は次のように定義される [8].

$$\left\{ (n_1, n_2, ..., n_d) \left| \sum_i n_i = (2\mathbb{Z}+1)a, n_i = \mathbb{Z}a \right. \right\} \quad (1)$$

図 2 は, 2 次元空間における面心立方格子配置の例である.

FC 法を使って, 分割点集合を作成する方法を説明する. データ集合から次元の最小値 min と最大値 max を計算し, 等分数 $2c-1$ で, 各次元の min と max の間を分割する. できた点を候補点とし, 式 (1) を満たす点を分割点とする. このとき, 作成される分割点の数は $\frac{(2c)^d}{2}$ となる.

このように分割点を配置した場合, 検索に用いる range の範囲をおさえることができる. 図 2 の例を用いて説明する. 図 2 の左下にあるクラスタの分割点の座標を $(0,0)$ とする. この分割点から隣接するクラスタへの L_∞ 距離での最小距離と L_1 距離での最大距離は点 $(\frac{1}{2}a, \frac{1}{2}a)$ と点 $(a, 2a)$ との距離なので, それぞれ $\frac{1}{2}a$ と $3a$ となる. つまり, 隣接するクラスタの range の範囲は $[\frac{1}{2}a, 3a]$ に収まる. d 次元で考えると, 分割点から隣接するクラスタへの最小距離と最大距離はそれぞれ $\frac{1}{2}a$ と $(d+1)a$ となる. つまり, range の範囲は $[\frac{1}{2}a, (d+1)a]$ に収まる. a と c の間には $a(2c-1) = \max - \min$ の関係が成立する.

3 計算機実験

我々は, 人工データ集合と楽曲データ集合 (表 1) を用いて計算機実験をおこなった. 人工データ集合は, 4 次元, 8 次元, 16 次元の一様分布データである. 楽曲データ集合は, 楽曲から抽出した LSP 係数をデータ点と見なしたものであり, データ依存の 20 次元のデータである. これらを順に DB_1, DB_2, DB_3, DB_4 とする.

表 1: 実験に用いたデータ集合

名前	次数	点数	分布	種類
DB_1	4	10 万	一様分布	人工データ
DB_2	8	10 万	一様分布	人工データ
DB_3	16	10 万	一様分布	人工データ
DB_4	20	10 万	データ依存	楽曲データ

それぞれのデータ集合に対して, 6 つの分割点集合の選択手法 GNAT 法, D-index 法, SSS 法と格子配置法の SQUARE 法, FC 法とデータ集合からランダムに点をとる手法 RAND 法を用いて, 分割点集合を作る. この分割点集合を元に距離関数を用いて, クラスタを構成し, 索引構造を構築する. 索引構造の構築に用いた距離関数を構築距離と呼ぶ. 本研究では, 構築距離が L_1 距離, L_2 距離, L_∞ 距離の 3 種類の mm-GNAT 索引構造を構築した. それぞれの索引構造を用いて, 検索距離が L_1 距離, L_2 距離, L_∞ 距離の 3 種類で検索をおこなった. 検索時に問い合わせ点 q から距離 ε 以下の点の数 (正解数) を求め, これらの点をすべて出力するまでに要した距離計算回数を調べた. ただし, 検索を行うための問い合わせ点集合は, データ集合から無作為に 1000 点を選んだ. 正解数をデータ数で割った値を選択率として用いる.

データ集合の点数の 1% を分割点にすると考え, 本研究では, データ集合に対して分割点数を 1000 とし

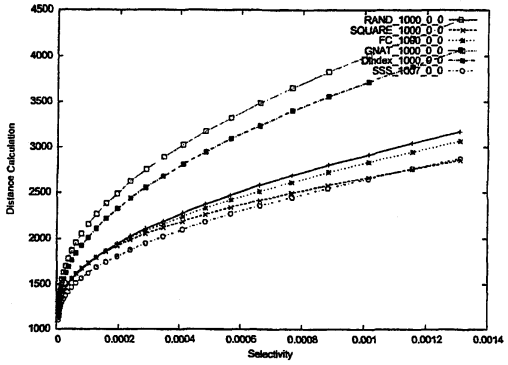

図 3: DB_1(4 次元), 分割点数 1000, 構築距離 L_∞ 距離, 検索距離 L_∞ 距離の実験結果

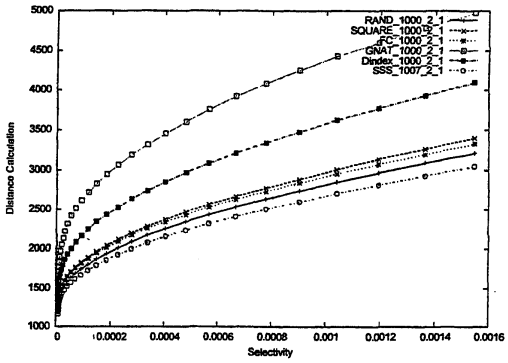

図 4: DB_1(4 次元), 分割点数 1000, 構築距離 L_2 距離, 検索距離 L_1 距離の実験結果

て実験をおこなった. 格子配置法では, 分割点数が指数的に大きくなるため, 1000 個以上の候補分割点を作成し, クラスタを構成した上で, クラスタ内の点数が多い上位 1000 点を分割点として採用した. SSS 法では, 分割点数が 1000 に近くなるように, α の値を調整した. 実験結果の一部のグラフを図 3 から図 10 に示す.

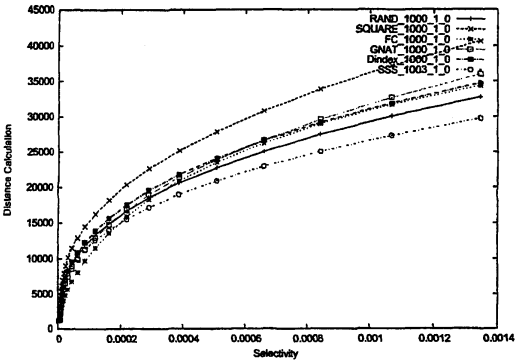

図 5: DB_2(8 次元), 分割点数 1000, 構築距離 L_1 距離, 検索距離 L_∞ 距離の実験結果

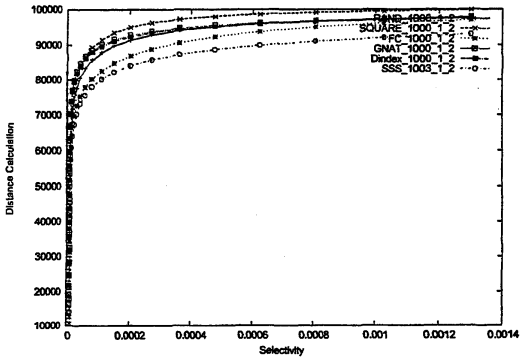

図 6: DB_2(8 次元), 分割点数 1000, 構築距離 L_1 距離, 検索距離 L_2 距離の実験結果

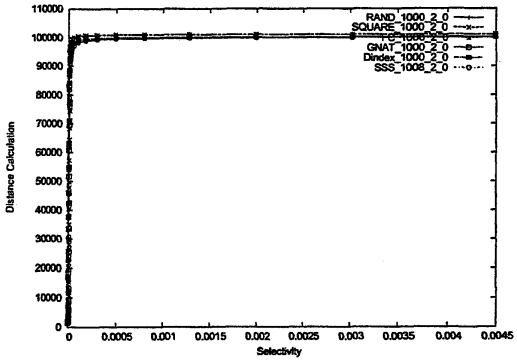

図 7: DB_3(16 次元), 分割点数 1000, 構築距離 L_2 距離, 検索距離 L_∞ 距離の実験結果

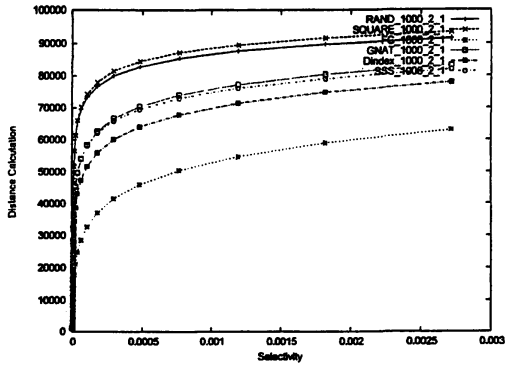

図 8: DB_3(16 次元), 分割点数 1000, 構築距離 L_2 距離, 検索距離 L_1 距離の実験結果

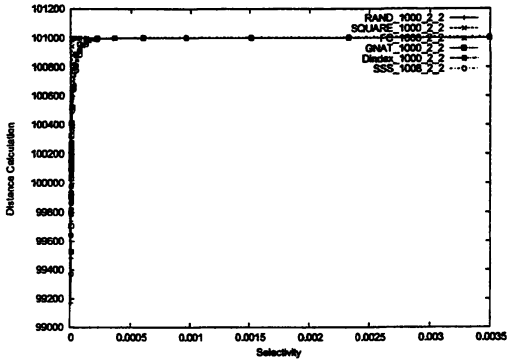

図 9: DB_3(16 次元), 分割点数 1000, 構築距離 L_2 距離, 検索距離 L_2 距離の実験結果

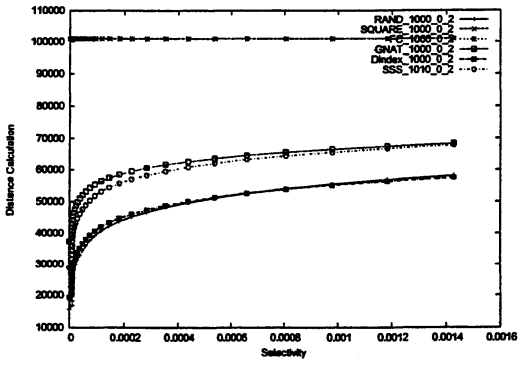

図 10: DB_4(20 次元), 分割点数が 1000, 構築距離が L_∞, 検索距離が L_2 の実験結果

4 議論

格子配置法

人工データ DB_1, DB_2, DB_3

DB_1 での実験結果 (図 3) では, SSS 法と格子配置法の距離計算回数が少ない. それについで RAND 法が少ないことがわかる. この傾向は他の DB_1 に対する実験でも同様である. しかし, 図 4 のように, 格子配置法のほうが RAND 法より距離計算回数が多い場合がある (9 種類の実験うち 2 回). 格子配置法で作成した分割点は互いに一定の距離以上を離れている. これが, DB_1 での実験で格子配置法の距離計算回数が少ない理由として考えられる.

また, FC 法の距離計算回数が SQUARE 法より多い場合がある. これは, SQUARE 法のクラスタに含まれる点の数はほぼ一定であるが, FC 法のクラスタに含まれる点の数は, ほぼ一定のものと, その半分程度のもの (境界周辺のクラスタ) があるためと考えられる.

DB_2 での実験結果 (図 5, 図 6) では, 格子配置法は RAND 法より距離計算回数が多い場合が多くあった. DB_2 での実験では, SQUARE 法では 6561 個から, FC 手法では 32768 個から 1000 点を選んだ. そのため, DB_2 の場合は DB_1 の場合よりもクラスタに含まれる点の数が大きく変化し, 問い合わせ点もデータ集合から選んでいるので, 枝刈りの効果があまり出ていないためと考えられる.

DB_3 での実験結果では, 検索距離が L_∞ 距離 (図 7), L_2 距離 (図 9) の場合, どの方法でもほぼ全件検索になる. 検索距離が L_1 距離の場合 (図 8) は, FC 法の距離計算回数が一番少ない. 等分数 1 の FC 法で作った候補分割点が互い離れていることと, それぞれのクラスタの形が L_1 距離での球に類似していることから, 枝刈りの効果が出ていると考えられる.

楽曲データ DB_4

DB_4 に格子配置法を適用すると, データ点は 2 つのクラスタにしか入ってないので, ほぼ全件検索に

なっている (図 9). 高次元の偏りのあるデータ集合に対して, この手法はあまり適さないと考えられる. これは, 分割点数が次元の指数的に大きくなるためである.

GNAT 法と D-index 法

DB_1 での実験結果 (図 3, 図 4) では, GNAT 法と D-index 法の距離計算回数は, 他の方法より多い. DB_2, DB_3, DB_4 での実験結果では, 距離計算回数が他の方法との差が小さくなり, 逆転する場合もある (図 5, 図 8, 図 10). GNAT 法と D-index 法は, 離れた点を分割点としている. しかし, 分割点の数を増やしていくと, 選択された分割点が固まり, データ点がうまく分割されない. そのため, 枝刈りがおこなわれない. 一方で, 次元が高いと, 選択された分割点が分散するので, 枝刈りの効果が出るためと考えている.

SSS 法

選択された分割点が互いにある程度の距離以上に離れると, 枝刈りがおこなわれ, 距離計算回数が少なくなる. SSS 法は, 分割点が互いに αM 以上に離れている. DB_1, DB_2 での実験結果では, 図 3 から図 6 のように, 距離計算回数が最も少ない場合が多く, ほかの場合でも距離計算回数が 2 番目か 3 番目に少ない. DB_3, DB_4 での実験結果 (図 8, 図 10) では, 距離計算回数が 2 番目か 3 番目に少ない.

RAND 法

RAND 法は, 分割点をデータ集合からランダムにとる. 一様分布のデータ集合の場合, 4 次元, 8 次元では, 距離計算回数が 4 番目に少ない場合が多い (図 3). 図 4, 図 5 のように, 2 番目に少ない場合もある. 16 次元 (図 8) では距離計算回数が多い. しかし, データ依存のデータ集合 DB_4(図 10) に対して, 他の方法より距離計算回数が少ないことが分かった. これは, 問い合わせ点をデータ点からランダムにとったためと考えられる.

5 まとめ

本稿では, GNAT 法, D-index 法, SSS 法, SQUARE 法, FC 法, RAND 法で分割点集合を構成し, その分割点集合により, 構築距離が L_1 距離, L_2 距離と L_∞ 距離の mm-GNAT 索引構造を構築した. そして, それぞれの索引構造を用いて, 検索距離が L_1 距離, L_2 距離と L_∞ 距離で 1000 回の ε 近傍検索をおこない, 距離計算回数を調べた.

格子配置法は, 低次元の一様分布のデータ集合に対して, SSS 法と同じ程度の距離計算回数が必要となった. この計算回数は他の手法よりも少ない. 一方, 分割点集合の構築時の時間計算量から見ると, SSS 法よりも格子配置法の方が高速である. そのため, 4 次元では格子配置法が最もよいと考えられる. 高次元の場合は, 分割点数が大きくなるため, クラスタに含まれる点数の多いものを選んで分割点とした. 4 次元では, ある程度の効果があったが, 8 次元以上の実験では, あまり効果はなかった.

GNAT 法と D-index 法は, 低次元の一様分布のデータ集合では, 他の方法より距離計算回数が多い. 高次元のデータ集合では, 距離計算回数が他の方法との差が小さくなり, 逆転した場合も出ていることが分かった. 低次元の偏りのあるデータ集合に対して, 距離計算回数がどうなるかは, 今後確認していきたいと考えている.

SSS 法は, 距離計算回数が一番少ない場合が多いことが分かった. SSS 法では, 分割点を選択するに用いたパラメータ α の値を小さくすると, 分割点数が多くなり, 大きくすると, 分割点数が少なくなる. α の値をどのように決定すればよいかについても, 今後取り組んでいきたいと考えている.

RAND 法は, 分割点集合の構築計算量が最も少ない. 一様分布の 16 次元データ集合の場合, 距離計算回数は多いが, 4 次元, 8 次元では, 4 番目に少ない場合が多く, 2 番目に少ない場合もあった. また, データ依存の DB_4 の場合, RAND 法の距離計算回数が一番少ない.

参考文献

[1] C. Bohm, S. Berchtold, D. A. Keim: Searching in High-Dimensional Spaces-Index Structures for Improving the Performance of Multimedia Databases, ACM Computing Surveys, Vol. 33, No. 3, pp.322-373, September 2001.

[2] E. Chavez, G. Navarro, R. Baeza-Yates, J. L. Marroquin: Searching in Metric Spaces, ACM Computing Surveys, Vol. 33, No. 3, pp.273-321, September 2001.

[3] B. Yi, C. Faloutsos: Fast Time Sequence Indexing for Arbitrary L_p Norms, Proc. of the 26th International Conference on VLDB, Cario, Egypt, pp. 385-394, 2000.

[4] S. Brin: Near Neighbor Search in Large Metric Spaces, Proc. of the 21st International Conference on VLDB, pp.574-584, 1995.

[5] B. Bustos, G. Navarro, E. Chavez: Pivot selection techniques for proximity searching in metric spaces, Proc. of the XXI Conference of the Chielan Computer Science Society(SCCC01), IEEE CS Press, pp.33-40, 2001.

[6] N. R. Brisaboa, A. Farina, O. Pedreira: Spatial Selection of Sparse Pivots for Similarity Search in Metric Spaces, Proc. of the 33rd International Conference on Conference on Current Trends in Theory and Practice of Informatics(SOFSEM), pp.8-13, 2007.

[7] K. Onishi, M. Kobayakawa, M. Hoshi: mm-GNAT: Index Structure for Arbitrary L_p Norm, IPSJ Transactions on Databases Vol.3, No.3, pp.88-98, 2010.

[8] K. Onishi, M. Hoshi: Cover Ratio of Absolute Neighbor, WALCOM 2008, LNCS 4921, pp.70-80, 2008.