類似検索における遅延評価を用いたピボット選択の高速化

三津山 雅規† 武藤 伸明† 斉藤 和巳† 池田 哲夫†
†静岡県立大学経営情報学部 〒422-8526 静岡県静岡市駿河区谷田 52-1

E-mail: †{b7090,muto,k-saito,t-ikeda}@u-shizuoka-ken.ac.jp

あらまし 範囲検索を含む類似検索では一般に、ピボットを用いることで検索効率を大きく向上させることができる。Bustos らはピボット選択の手法として貪欲法を提案したが、貪欲法ではピボット選択に大量の時間を必要とする。本論文では、ピボット選択の高速化法を提案する。はじめに、Bustos らのピボットの良さの評価基準となる関数がサブモジュラの性質を持つことを証明する。次に、サブモジュラの性質を持つ問題においては遅延評価が導入可能であり、遅延評価を用いて貪欲法を高速化する。最後に、計算機実験により遅延評価の導入による高速化の効果を明らかにする。

キーワード 類似検索, 遅延評価

Speed-up of Pivot Selection on Similarity Search by Lazy Evaluation

Masaki MITSUYAMA† Nobuaki MUTOH† Kazumi SAITO† and Tetsuo IKEDA†
† University of Shizuoka 52-1 Yata, Suruga-ku, ,Shizuoka-shi, Shizuoka, 422-8526 Japan

E-mail: †{b7090,muto,k-saito,t-ikeda}@u-shizuoka-ken.ac.jp

Abstract Introducing the use of pivots into similarity search algorithm makes it efficient. Bustos et. al. showed that selection of pivots highly affects the performance of algorithm, and proposed incremental selection method as a good algorithm for the selection of pivots. But their incremental selection requires long computational time in proportion to the number of pivots. In this paper, we show that the objective function which is their measure of the goodness of pivots has a submodular property. And we improve the efficiency of their incremental search by incorporating a technique called lazy evaluation on this submodular problem. Finally, we show the efficiency of our proposed method by some experiments.

Keyword Similarity Search, Lazy Evaluation

1. はじめに

近年，インターネットの発達などにより，マルチメディアデータが大量に蓄積され，与えられたクエリから類似したオブジェクトを検索する類似検索の研究は非常に重要になってきている．

一般に類似検索を行う場合，オブジェクトはメトリック空間，あるいは多次元空間に存在しているため，オブジェクト間の距離を計算するには膨大な計算コストがかかる．そのため，類似検索では一般にピボットを用いることで検索効率を大きく向上させている．

ピボットは通常ランダムに選んでも検索効率を向上させることができるが，より検索の高速化を実現するピボットの選び方が存在する．Bustos らは効率的なピボットの選び方である貪欲法を提案したが，ピボットを選択するための時間が非常にかかる[1]．

本論文では，ピボットの選択にかかる時間を改善するため，貪欲法に遅延評価という技術を導入する．最初に，Bustos らのピボットの良さの評価基準となる目的関数がサブモジュラの性質[2]を持つことを証明する．次に，サブモジュラの性質を持つ問題においては遅延評価が導入可能であり[3][4]，遅延評価を用いて貪欲法を高速化する．最後に，計算機実験により遅延評価の導入による高速化の効果を明らかにする．

2. 従来の手法とその問題点

2.1 範囲検索

本論文では，類似検索の 1 つである範囲検索を扱う．範囲検索とは，クエリとの距離が定められた一定の範囲内にあるオブジェクトを全て列挙するタスクである．一般にオブジェクト間の距離 d は距離関数で表される．距離関数は，オブジェクト空間 M に定義された距離関数 $d : M \times M$ よって定まった距離空間$(M; d)$であり，$\{x, y \in M\}$に対し，d は以下の距離公理を満たす．

1. 非負性 $d(x,y) \geq 0$，$d(x,y) = 0$ は $x = y$ と同値．
2. 対称性 $d(x,y) = d(y,x)$
3. 三角不等式 $d(x,y) + d(y,x) \geq d(x,z)$

距離空間$(M; d)$にデータベース U が存在し，U は N 個のオブジェクト集合 $U = \{u_1, \cdots, u_N; u \in M\}$を持っている．範囲検索は，クエリ $q \in M$ が与えられたとき，q と u との距離 $d(q,u)$が定められた一定の範囲 r 以内にある u を全て列挙する．

2.2 ピボットを用いた範囲検索の高速化

一般に距離空間$(M; d)$においてクエリとオブジェクト間の距離の計算は高コストである．この距離計算を，ピボットを用いることで削減することができる．以下にその手法について述べる．

クエリ q とピボット p 間の距離を $d(q,p)$，ピボット p とオブジェクト u 間の距離を $d(p,u)$とする．距離空間$(M; d)$においては距離公理が利用できるため，三角不等式より，次式が成り立つ．

$$d(q,u) \geq \left| d(q,p) - d(p,u) \right|. \quad (1)$$

適当な範囲 r を設定したとき，もし$|d(q,p) - d(p,u)| > r$ならば，式(1)より，$d(q,u) > r$ となるのは明らかである．よって，$|d(q,p) - d(p,u)| > r$ となるオブジェクト u については $d(q,u)$ の計算を減らすことができる．

ここで，$d(p,u)$は，クエリが入力される前に事前計算しておくことが可能なので，クエリの入力後に計算が必要なのは $d(q,p)$ のみである．したがって，クエリと全てのオブジェクトの距離を計算するのに比較すると，距離計算回数を大幅に減らすことができる．

ピボットを K 個に増やすことも可能であり，この場合，事前準備として，K 個のピボット集合 $Q=\{p_1, \cdots, p_K; p \in U\}$を選び，$Q$ は U 内の全オブジェクトとの距離 $d(p,u)$をあらかじめ計算しておく．この場合，距離 $d(q,u)$ の計算は K 個のピボットすべてについて $|d(q,p) - d(p,u)| \leqq r$ を満たすオブジェクトについてのみ行えばよい．すなわち，ピボットが 1 つの場合よりもさらに距離 $d(q,u)$の計算回数を減らすことができる．K 個のピボットすべてについて，計算された$|d(q,p) - d(p,u)|$の最大のものを,以下のように定義する．

$$D_{(q,u)}(Q) \stackrel{def}{=} \max_{p \in Q} \left| d(q,p) - d(p,u) \right| . \quad (2)$$

$D_{(q,u)}(Q) > r$ であるオブジェクトについては距離 $d(q,u)$を計算する必要はないということになる．

2.3 範囲検索を高速化させる目的関数

Bustos らは，$D_{(q,u)}(Q)$の値が K 個のピボットの選び方に依存し，良いピボット集合を選ぶことで $d(q,u)$の距離計算をする必要がないオブジェクト数が増え，計算速度が向上することを明らかにしている．すなわち，良いピボット集合とは $D_{(q,u)}(Q) > r$ を満足するオブジェクト数が多いピボット集合であるといえる．U 上の全てのオブジェクトペアを $B=\{(u_1,u_2),(u_1,u_3), \cdots,(u_{N-1},u_N); (u_i,u_j) \in U \times U\}$としたとき，$D_{(q,u)}(Q)$の総和を表現する目的関数を以下のように定義する．

$$F(Q) \stackrel{def}{=} \sum_{(u_i,u_j) \in B} D_{(u_i,u_j)}(Q). \quad (3)$$

ピボット集合 Q より良いピボット集合 Q'とは，$F(Q) > F(Q')$である．

式(3)のように，全てのオブジェクトを対象にすると，計算効率が悪い．そこで Bustos らは，U の中から A 個のオブジェクトペア $C=\{(a_1,a'_1), (a_2,a'_2), \ldots,(a_A,a'_A); (a_i, a_i') \in U \times U\}$を選び，以下を目的関数としている．

$$F(Q) \stackrel{def}{=} \sum_{(a_i,a_i') \in C} D_{(a_i,a_i')}(Q). \quad (4)$$

式(4)の目的関数を最適化する手法として Bustos らは貪欲法を提案している．以下に貪欲法を示す．

オブジェクト集合 U の中から T 個のサンプルオブジェクトを選び，ピボット候補集合 $S=\{s_1, \ldots, s_T; s \in U\}$とする．最初に，$s$ それぞれについて $F(s)$を計算する．$F(s)$を最大にする s を，1 つ目のピボット p_1 とする．次に，先ほどの $F(p_1)$にあらたなピボット候補を加えた $F(p_1+ s)$が最大になる s を 2 つ目のピボット p_2 とする．これを K 回繰り返す．

3. サブモジュラ性

貪欲法は目的関数を最適化するのに有効的な手法であるが，ピボットを選択するのに膨大な時間がかかるというデメリットがある．本論文では，ピボット選択にかかる時間を高速化するために遅延評価という技術を導入する．遅延評価は，サブモジュラの性質を持つ問題において導入可能である．本節では，Bustos らの目的関数 $F(Q)$がサブモジュラの性質を持つことを証明する．

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 p と包含関係 $L \subset R$ 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．

$$f(L \cup \{p\}) - f(L) \geq f(R \cup \{p\}) - f(R). \quad (5)$$

以下にBustosらが定義した目的関数$F(Q)$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ペア$(a_i, a_i') \in U \times U$に対して，任意のピボット$p \in S$と$L \subset R \subset S$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．

$$D_{(a_i,a_i')}(L \cup \{p\}) - D_{(a_i,a_i')}(L) \geq D_{(a_i,a_i')}(R \cup \{p\}) - D_{(a_i,a_i')}(R). \quad (6)$$

ピボット集合$R \cup \{p\}$を対象とすれば，$D_{(a_i,a_i')}(R \cup \{p\})$の値を決めるピボットは，集合$R$か$\{p\}$どちらかに含まれる．まず，$R$のケースでは，

$$D_{(a_i,a_i')}(R\cup\{p\})=D_{(a_i,a_i')}(R)$$

であるので，式(6) の右辺は0 であり，

$$D_{(a_i,a_i')}(L\cup\{p\})\geq D_{(a_i,a_i')}(L)$$

より，式(6) の不等式は成り立つ.一方，$\{p\}$ のケースでは，

$$D_{(a_i,a_i')}(R\cup\{p\})=D_{(a_i,a_i')}(L\cup\{p\})=D_{(a_i,a_i')}(\{p\})$$

であり，

$$D_{(a_i,a_i')}(L)\leq D_{(a_i,a_i')}(R)$$

より，式(6)の不等式が成り立つ．よって， Bustosらの目的関数$F(Q)$は，サブモジュラ関数であり，サブモジュラ最大化問題として定式化できる．

4. 遅延評価導入による高速化

サブモジュラ最大化問題として定式化したことにより，遅延評価という技術が導入できる．あるピボット集合Qにピボット候補集合Sのピボット$s\in S$を追加したとき，の$F(Q)$の差分値を以下のように表す．

$$g(s;Q)=F(Q\cup\{s\})-F(Q). \tag{7}$$

式(7)はサブモジュラ不等式より，非増加関数であり，$g(s,Q)$の上限値を$\zeta(s)$とすると，貪欲法の各ステップkにおいてk=0のとき$\zeta(s)=\infty$であり，$1\leq k\leq K$のとき$\zeta(s)=g(s,Q_{k-1})$である．この関係を利用し，検索効率を向上されるのが遅延評価である．以下に遅延評価のアルゴリズムを示す．

以下では，ピボット候補集合$S=\{s_1,\cdots,s_T\}$に含まれるピボット$s\in S$に対する差分上限値を$\zeta(s)$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．Gは，処理過程の時点で最良差分値を格納する変数とする．

Step 1. k=1, $Q=\phi$ とする．ピボット候補$s\in S$について，$\zeta(s)=\infty$ と初期化し， n=1,…, Tで$r_n\leftarrow s_n$とする．

Step 2. $G\leftarrow 0$とし， n=1,…, Tの順でStep 2-1からStep 2-2の処理を繰り返す．

Step 2-1. $\zeta(r_n)\leq G$なら，$\zeta(p_k)\leftarrow 0$， $Q_k\leftarrow Q_{k-1}\cup\{p_k\}$とし，Step 3に進む;

Step 2-2. $\zeta(r_n)\leftarrow g(r_n;Q_{k-1})$とし，$G<\zeta(r_n)$ なら，$p_k\leftarrow r_n$，$G\leftarrow\zeta(r_n)$ とする．

Step 3. $k=K$なら$Q_k=\{p_1,\cdots,p_k\}$を出力し終了する．

Step 4. 差分上限値$\zeta(s)$ でピボットを降順にソートしたものをr_1，… , r_Tとし， $k\leftarrow k$+1としStep 2へ戻る．

5. 遅延評価導入の評価実験

本節では，貪欲法でピボットを選択した場合と，遅延評価を導入した貪欲法でピボットを選択した場合を比較するための計算機実験を行う．貪欲法と遅延評価付き貪欲法は必ず同じピボットを選択するため，遅延評価付き貪欲法はピボット選択にかかる時間のみを短縮する．実験で用いるデータ及び値の設定は，Bustosらが自身の論文で行った実験のデータを参考にしている．また実験結果は，ピボットを選択するためにかかる，ピボットの数に応じた距離計算回数を示す．

5.1 ユークリッド距離による実験

実験のデータは，8 次元ベクトルのデータと 14 次元ベクトルのデータを扱う．

8 次元，14 次元ベクトルのデータは，各クエリ及び各オブジェクトは，0～1 までの値を次元数保持しており，距離はユークリッド距離を用いて計算する．データの設定は，オブジェクト数は 1 万，クエリ数は 1 万，ペア数は 1 万，サンプル数は 250，ピボットは 1~50 まで変化させている．範囲 r は範囲内のオブジェクト数が全オブジェクト数の 0.01%になるように設定している．範囲内のオブジェクト数が H 個となるような範囲の求め方は，各クエリについて，データ用オブジェクト集合から，クエリとの距離が H 番目に小さいオブジェクトとの距離を求め，求めた距離の総和をクエリ数で割った値を範囲 r とする．

ユークリッド距離は以下のように計算を行う．E 次元ベクトル空間において，$\{\boldsymbol{x}_v: v=1,\cdots,V\}$, $\{y_w: w=1,\cdots,W\}$とし，距離計算を行うとき，次式のように計算する．

$$d(\mathbf{x},\mathbf{y})=\sqrt{\sum_{e=1}^{E}(x_e-y_e)^2}.$$

図 1 は 8 次元ベクトルのデータの結果であり，貪欲法は，ピボットが 50 個のとき距離計算回数は 2 億 2650 万回であり，遅延評価付き貪欲法は，ピボットが 50 個のとき距離計算回数は 4644 万回であり，約 80%の距離計算の削減をしている．貪欲法は，ピボット数に比例して計算量が増大していくのがわかる．一方遅延評価付き貪欲法は，アルゴリズムからわかるように，最初のピボットを選ぶときは，貪欲法と同様の計算回数であるが，2 つ目以降のピボットでは前回の $\zeta(s)$から$\zeta(r_n)\leq G$ となれば，r_n 以降の s の計算を削減する．

また図 2 は 14 次元ベクトルのデータの結果である．貪欲法は，ピボットが 50 個のとき距離計算回数は 2 億 2650 万回であり，遅延評価付き貪欲法は，ピボットが 50 個のとき距離計算回数は 5445 万回であり，約 76%の距離計算の削減をしている．わずかではあるが，低次元である 8 次元データの方が遅延評価がより効果を発揮している．

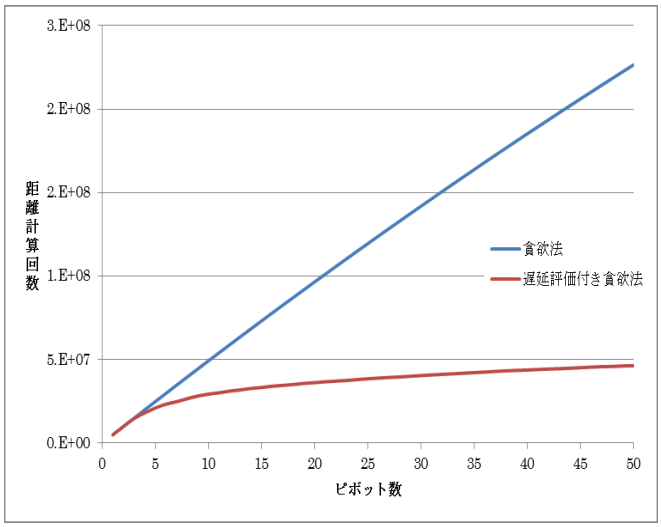

図 1.　8 次元ベクトルのデータ

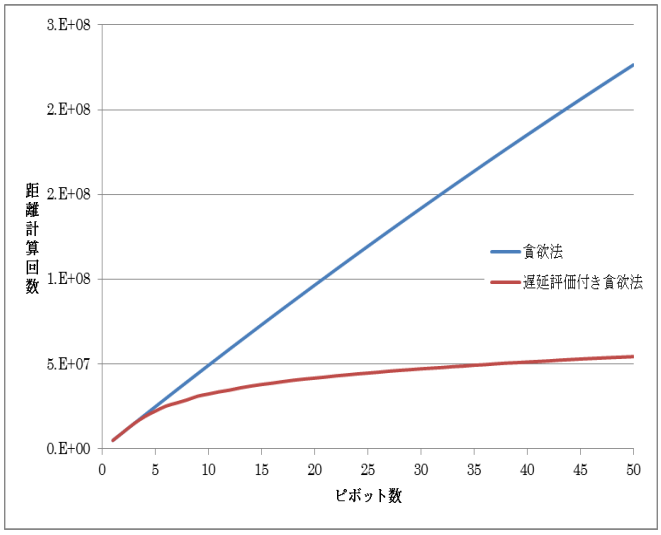

図 2.　14 次元ベクトルのデータ

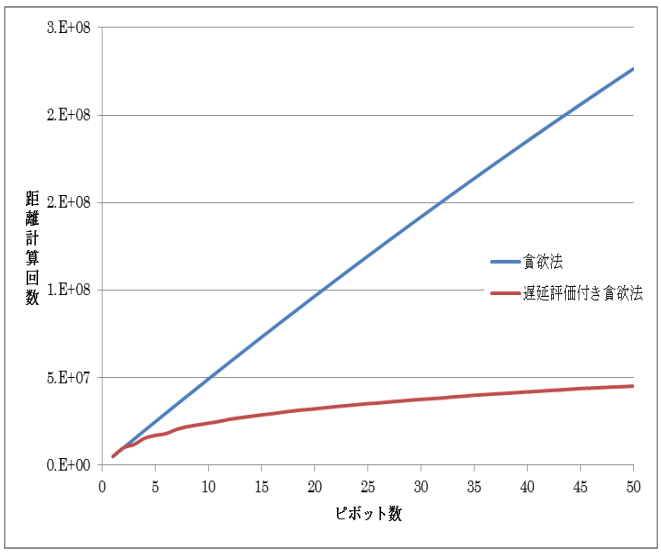

図 3.　単語データ

5.2 編集距離による実験

実験のデータは，emacs のスペルチェックのための辞書の単語データを使う．オブジェクトとクエリは単語データから抽出する．最初に，オブジェクトを単語データからランダムに 1 万個選び，クエリは先ほど選ばれたオブジェクト以外からランダムに 1 万個選ぶ．範囲は 2 に設定している．距離は編集距離を用いて計算する．編集距離とは，文字の追加，削除，置換をそれぞれ距離 1 として距離を定義．2 つの文字列 A と B の編集距離とは，A を B に変形するのに必要な文字操作の最小回数ことである．

図 3 は単語データの結果である．貪欲法は，ピボットが 50 個のとき距離計算回数は 2 億 2650 万回であり，遅延評価付き貪欲法は，ピボットが 50 個のとき距離計算回数は 4530 万回であり，約 80%の距離計算の削減をしている．遅延評価の効果は多次元ベクトルのデータの場合とほぼ変わらない効果を発揮している．

6. おわりに

本論文では,ピボットを選択する時間を短縮するために，Bustos らのピボットの良さの評価基準となる関数がサブモジュラの性質を持つことを証明し，貪欲法に遅延評価を導入した．遅延評価を導入したことにより，14 次元ベクトルのデータにおいて通常の貪欲法より約 76%の距離計算を削減し，8 次元ベクトルのデータ及び，emacs のスペルチェックのための辞書の単語データを用いたデータにおいては，距離計算を約 80%削減した.今後はさらなる高速化の実現を目指す．

参　考　文　献

[1] Bustos, B., Navarro, G. and Ch´avez, E.: Pivot selection techniques for proximity searching in metric spaces., Pattern Recognition Letters, Vol.24, No.14, pp.2357–2366 (2003).

[2] 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).

[3] J. Leskovec, A. Krause, C. Guestrin, C. Faloutsos, J. VanBriesen, and N. Glance. Cost-effective outbreak detection in networks, Proceedings of the 13th International Conference on Knowledge Discovery and Data Mining, pp.420–429 (2007).

[4] 斉藤 和巳，武藤 伸明，池田 哲夫，入月 卓也，永田 大，伊藤 かの子，"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，" 情報処理学会論文誌，数理モデル化と応用，Vol.3, No.1, pp.62－－72, 2010.