量子化を利用した次元縮小射影Simple-Mapの中心点探索に関する研究

NGUYEN THI BICH HAU[1,a)]　篠原 武[1]

概要：画像や音楽といった大量かつ高次元のマルチメディアデータの高速な近似検索を実現するために，特徴データを低次元空間に射影し，R-tree や M-tree などの空間索引構造を構築し検索を行う．ここでは，次元縮小法として Simple-Map(S-Map) を用いる．S-Map は中心点とオブジェクトの実距離を用いて射影を行う．射影距離は必ず実距離以下になり，射影空間上での任意のオブジェクトの 2 点間の距離が縮まる「距離の縮み」が生じる．S-Map の中心点がその距離の縮みに大きく影響するため，中心点の選び方が重要となる．S-Map の中心点探索に用いる量子化法とは，データベース内のオブジェクトの座標値を閾値により予め空間の最大値又は最小値に量子化する方法である．本研究では，データの基本統計量を用いてより良い閾値を選定し，量子化法の性能を高める．

キーワード：近似検索，R-tree，次元縮小射影，Simple-Map，量子化

Pivot Selection in Dimension Reduction Projection Simple-Map with Quantization

Nguyen Thi Bich Hau[1,a)]　Takeshi Shinohara[1]

Abstract: Similarity search of multi-dimensional data, such as images and music, is a method to retrieve by using a spatial index structure, such as method R-tree and method M-tree. To avoid the curse of dimensionality, we project feature data into low-dimensional space. We use a dimension reduction projection named Simple-Map(S-Map for short). S-Map projects objects by using the actual distances between pivots and objects as the coordinate values. Projective distance is shorter than actual distance by all means. The pivots of S-Map greatly influence the shrinkage of the distance, so how to choose pivots becomes important. We use the quantization method to search the pivots of S-Map. It is a method to quantize the coordinate values of objects to the maximum or the minimum of the space according if the value is larger than the threshold. In this paper, we use the fundamental statistics of data to improve the quantization method.

Keywords: similarity search, R-tree, dimension reduction mapping, Simple-Map, quantization

1. はじめに

大量の映像，音楽といったマルチメディアデータを取扱う Web サイトやエンジニアリング等が急速に普及するこの時代では，混在する非常に多くのデータから必要なものだけを探し出す情報検索技術が不可欠となっている．

マルチメディアデータの検索においては，検索処理はデータの持つ特性から非常に時間のかかる処理であり，検索処理精度も低下するという問題もある．また，質問点と完全に一致する情報のみを検索する完全一致検索は一つ一つ特性を調べる必要がある．しかし，人間の目から見て同じと判断する情報であっても，計算機にとって全く異なる情報としてとらえるケースが多い．そのため，完全一致検索よりも，似ている情報を検索するという近似検索の方が重要性が高い．

[1] 九州工業大学情報工学部知能情報工学科
Department of Artificial Intelligence, Kyushu Institute of Technology
[a)] k231093t@mail.kyutech.jp

膨大な量のマルチメディアデータを対象に高速な近似検索を行う場合は，B-tree[5] を多次元に拡張した R-tree[1] や M-tree[2] 等の索引構造が一般的に用いられる．しかし，高次元の特徴空間に対して R-tree を構築すると，次元の呪いという現象により検索の性能が悪化してしまう性質がある．そのため，特徴空間をより低次元空間に射影する次元縮小を用いて検索効率の悪化を防ぐ．

本論文では，次元縮小射影法の一つとして Simple-Map(S-Map)[3] を用いる．S-Map は任意の距離空間に対して適用可能であり，中心点とオブジェクトの実距離を用いて射影を行う．しかし，S-map では射影距離は必ず実距離以下になり，射影空間上での任意のオブジェクトの 2 点間の距離が縮まる「距離の縮み」が生じてしまう．そのため，本来検索範囲の外にあるはずのオブジェクトも検索範囲に含まれ，その分計算コストがかかってしまう．S-Map の中心点がその距離の縮みに大きく影響する．よって中心点の選び方が重要となる．

S-Map の評価関数としては，データベースからランダムに選んだ複数個のオブジェクト対の射影距離の平均を用いる．素朴な中心点探索法では，データベース内のオブジェクトを中心点候補として，評価関数を大きくするものを選ぶ．素朴法により得られた中心点から始めて局所探索を行うとより良い S-Map が得られる．局所探索法では探索に計算時間がかかるが，これを解決する量子化法 [4] が提案された．量子化法は，比較的に短い探索時間で，局所探索法には及ばないものの，ある程度良い中心点を求めることができる．本研究では，データの基本統計量の調査により量子化法の閾値の選定法を提案し，量子化法の性能を高めることを目指す．

2 章では近似検索と R-tree について述べる．3 章では次元縮小法 S-Map および各中心点探索法について述べる．4 章では各閾値選定法の性能比較検証の実験を記述する．5 章ではまとめる．

2. 近似検索

2.1 近似検索

近似検索とはデータ間の非近似度 (距離) を用いて，質問点と近似するデータを取り出すことである．近似検索のデータベース (DB) が対象とする特徴空間全体を $\mathcal{U} = \mathbb{R}^n$ とする．ここで，$\mathbb{R}$ は実数全体，n は特徴データの次元数である．任意の 2 点のオブジェクト間の非近似度の指標を示す距離関数を $d : \mathcal{U} \times \mathcal{U} \to \mathbb{R}^+$ とし，$\mathcal{D} = (\mathcal{S}, d)$ を距離空間とする．本論文で扱う近似検索では，距離関数 d は**距離の公理**である次の条件を満たすものとする．

(1) $d(X,Y) \geq 0$ (非負性)
(2) $d(X,Y) = d(Y,X)$ (対称性)
(3) $d(X,Y) \leq d(X,Z) + d(Z,Y)$ (三角不等式)
(4) $d(X,Y) = 0 \Leftrightarrow X = Y$ (同一性)

ここで，$X, Y, Z \in \mathcal{D}$ である．上の条件の中で最も重要なのは，三角不等式である．

次に，本実験で用いる距離計測について説明する．特徴空間内の任意のオブジェクトを x とする．x の特徴は n 個の実数の組 $(x_{(1)}, x_{(2)}, \ldots, x_{(n)})$ で表される．以下の二つの距離計測関数は距離の公理を満たすものである．

L_1距離 : $D(x,y) = \sum_{i=1}^{n} |x_{(i)} - y_{(i)}|$
L_∞距離 : $D(x,y) = \max_{i=1}^{n} |x_{(i)} - y_{(i)}|$

本論文では実距離計算を L_1 距離，射影距離計算を L_∞ 距離で行う．

2.2 質問方法

近似検索では，主に範囲質問および近傍質問の 2 種類の方法が用いられている．本論文における実験では，近傍質問の中でも，質問点 Q から距離が最も小さいオブジェクトを取得する**最近傍質問** $NN(\mathcal{D}, Q)$ を採用する．

$$NN(\mathcal{D}, Q) = \{O_i \in \mathcal{S} \mid O_i \text{ は } d(O_i, Q) \text{ が最小のもの } \}$$

最近傍質問では初期検索範囲を無限大に設定する．そこから検索を始めて，検索範囲内のオブジェクトを見つけた場合，そのオブジェクトを暫定解として登録し，検索範囲を暫定解と質問点との距離に収縮させる．以上の操作を検索範囲内に新たなオブジェクトが見つからなくなるまで繰り返す．

2.3 R-tree

R-tree や M-tree 等の階層的空間索引は高速な近似検索手法として知られている．本論文では代表的な空間索引構造の一つである R-tree を用いる．

R-tree はオブジェクトの追加・削除などの動的な操作が可能な多岐平衡木である．階層的に入れ子になった相互に重なり合う最小包囲矩形 Minimum Bounding Rectangle(MBR) で空間を分割する．空間的に近いデータ同士を同じ包括領域で囲い，ノードに登録する．葉ノードには DB 内のオブジェクトへのポインタを持つ．葉ノード以外の各ノードには二つのデータが格納される．一つは子ノードへのポインタであり，もう一つはそのノードの全ての子ノードを囲む MBR のデータである．高次元空間では MBR の形状が正規形から離れたり，重なりが大きくなったりすることが次元の呪いという現象を引き起こす原因となる．そのため，次元縮小法を用いて高次元の特徴空間を低次元へ射影する必要がある．また，MBR の重なりの軽減や，形状を正規形に近づけるため，Hilbert 曲線 [6] による順序付けを用いて静的に構築する．

検索の際には質問点の検索範囲と重複するノードのみを訪問することにより，参照コストを削減できる．また，次元縮小を用いて R-tree を構築する場合は，葉ノードにおい

てオブジェクトと質問点との距離を計算する前に，射影空間上の距離を調べることで，実距離計算回数を最小限に抑え，効率的な検索を実現する．

3. 次元縮小射影 Simple-Map

高次元の特徴空間に対して R-tree を構築すると次元の呪いにより検索の性能が悪化してしまう．それを緩和するため，特徴空間をより低次元空間に射影する次元縮小法がある．本論文では次元縮小法として S-Map を用いる．射影前の特徴空間を実空間，射影後の空間を射影空間と呼ぶ．

3.1 Simple-Map

S-Map は射影関数として，中心点とオブジェクトの実距離を用いる．また，S-Map は距離の公理を満たす全ての距離空間に対して適用可能である．射影空間上では任意のオブジェクトの 2 点間の距離が，元空間の距離よりも縮まる性質がある．このことを**距離の縮み**と呼ぶ．

射影に用いる中心点を p とすると，任意のオブジェクト O の射影空間上での座標値は

$$\varphi_p(O) = d(p, O)$$

と定義される．三角不等式より，射影空間上の任意の 2 点のオブジェクト X,Y 間の距離を $d'(X,Y)$ とすると，射影距離 $d'(X,Y)$ は中心点を媒介することで距離の縮みが生じることがある．

$$d'(X,Y) = |\varphi_p(X) - \varphi_p(Y)| \leq d(X,Y)$$

中心点を複数用いた場合に拡張する．S-Map による射影では，中心点の数が射影空間の次元数となる．中心点の集合を $P = \{p_1, p_2, \ldots, p_{n'}\}$ とすると，射影関数は

$$f_P(X) = (\varphi_{p_1}(X), \varphi_{p_2}(X), \ldots, \varphi_{p_{n'}}(X))$$

となる．ここで n' は射影空間の次元数である．また，複数の中心点を用いて射影された 2 点のオブジェクト間の距離は

$$d'(X,Y) = \max_{p \in P} |\varphi_p(X) - \varphi_p(Y)| \leq d(X,Y)$$

となる．このように，射影空間上での距離は L_∞ 距離で計測する．中心点の数が多いほど，射影空間上で距離の情報を多く保存できる．つまり，中心点を多くとることで距離の縮みを小さくできるが，射影空間が高次元になるため，射影距離のコストが増加し，次元の呪いの影響を強く受ける．

3.2 中心点探索法

素朴法：DB 内のオブジェクトからランダムに取ってきたオブジェクト対の間の射影距離の平均を，中心点選択用の評価関数として用いる．そして，DB 内のオブジェクトから中心点の候補を選定し評価関数によってスコアを求め，スコアが高かった中心点の候補を新たな中心点とする．

素朴法のアルゴリズム

```
p[1], p[2], …, p[n']: central points of S-map;
c: candidate for central point;
x: the number of random trials;
maxscore := 0;
for i := 1 to n' do
  repeat x times
    select c randomly from database;          (1)
    s := score using p[1], …, p[i - 1], and c;
      if s > maxscore then
        p[i] := c;
        maxscore := s;
      end if
  end repeat
end for
```

局所探索法：中心点の候補として選ばれたオブジェクトに局所探索をかけると，より良い中心点が選ばれる．まず，中心点候補の座標値を各軸に沿って適度に幅を取り平行移動し，新たな点のスコアを求める．元のスコアより上回るとその点を新たな中心点候補とし，次の軸に同様の操作を繰り返す．すべての軸を調べ終わると平行移動幅を狭めて同様の操作を繰り返す．候補点の隣接点のスコアをすべての軸に対して調べ終わっても候補点のスコアを上回る点が見つからなかった場合，その時点での候補点を中心点とする．

量子化法：前節の局所探索で探索した中心点の座標値を調べると実空間の最大値又は最小値が多く現れることが分かった．そこで，予め，DB 内のオブジェクトの座標値に対し，閾値未満のものを空間の最小値に，閾値以上のものを空間の最大値に量子化し，新たな点を中心点候補とする．素朴法の (1) の直後に `quantize(c);` を挿入する．量子化は次のように行う．ここで，`t` は閾値である．

提案手法 (軸別量子化法)：本研究の目的は，局所探索を行うのと同じ効果の中心点が得られるのと同時に中心点探索時間をより短くするために，データの各次元の基本統計量を調査することによって量子化の閾値を選定することである．量子化の `quantize(c)` を `axis-wise-quantize(c)` に置き換える．ここで，`t[j]` は第 `j` 次元の閾値である．

量子化法のアルゴリズム

```
t : threshold
quantize(c);
for j:= 1 to n do
  if c[j] > t then
    c[j] := 255;
  else
    c[j] := 0;
  end if
end for
```

軸別量子化法のアルゴリズム

```
t[j] : threshold for j-th axis;
axix-wise-quantize(c);
for j := 1 to n do
  if c[j] > t[j] then
    c[j] := 255;
  else
    c[j] := 0;
  end if
end for
```

4. 実験

実験には約 2,900 本の動画から抽出した約 700 万件の画像フレームデータと楽曲約 1,400 曲から切り出した約 700 万件の音フレームデータを用いる．実験には表 1 の性能の計算機を用いる．

表 1. 実験に用いた PC の性能

CPU	Intel(R) Xeon(R) CPU X3230 2.66GHz
メモリ	8GBytes

本論文では画像データを 64 次元から 8 次元へ，音データを 96 次元から 12 次元へ射影する．画像データおよび音データにおいて量子化の有効性を確認する．また，量子化の各閾値選定法を使用した中心点で次元縮小を行い，全データの基本統計量を閾値として量子化を行う従来手法と比較する．閾値選定は以下の (1)〜(9) の方法について比較検証を行った．

(1) 量子化なし+局所探索なし
(2) 量子化なし+局所探索あり
(3) 全データの中央値
(4) 全データの平均値
(5) 各次元の中央値
(6) 各次元の平均値
(7) 各中央値から同じ平行移動の幅
(8) 各平均値から同じ平行移動の幅
(9) 各次元の中央値をグループにまとめ，グループ毎動かす

4.1 画像データに関する実験

画像データの場合は，(1)〜(8) の結果を表 2, (9) を 10, 8, 5, 3, 1 個ずつのグループにまとめて実験を行い，その結果を表 3 に示す．ここで，距離保存率は実距離に対する射影距離の割合を表し，time1 は基本統計量を求める時間，time2 は中心点探索時間を示している．また，表中の太字は S-Map の距離保存率が最大になったところ，距離保存率の横にある括弧内の数字は平行移動して距離保存率が最大となった幅を表す．結果として，64 個の中央値を 8 個ずつのグループにまとめ，グループ毎を動かした場合は最も距離保存率が高いことが分かる．

表 2. 閾値選定法の比較 (画像)

	従来手法				本研究			
閾値	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)
距離保存率 (%)	46.21	56.38	54.02	53.4	56.84	56.33	56.84 (0)	56.93 (+3)
time1(sec)	–	–	1.9	1.9	2.0	2.0	2.0	2.0
time2(sec)	24.9	371	25.0	25.6	25.0	24.9	52.1	51.9

表 3. グループの平行移動の検証 (画像)

	10 個	8 個	5 個	3 個	1 個
距離保存率 (%)	56.86	**57.32**	56.91	57.02	57.04
time2(min)	20.4	23.3	37.8	64.0	184

4.2 音データに関する実験

(1)〜(8) の結果を表 4, (9) を 20, 16, 12, 8, 4, 1 個ずつのグループにまとめて実験を行い，その結果を表 5 に示す．結果として，96 個の中央値を 1 個ずつ動かした場合は最も距離保存率が高いことが分かる．

表 4. 閾値選定法の比較 (音)

	従来手法				本研究			
閾値	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)
距離保存率 (%)	75.68	78.33	78.55	78.55	78.42	78.33	78.61 (+1)	78.59 (+1)
time1(sec)	–	–	2.2	2.2	2.9	2.9	2.9	2.9
time2(sec)	58	1676	58	62	57	57	1200	1200

表 5. グループの平行移動の検証 (音)

	20 個	16 個	12 個	8 個	4 個	1 個
距離保存率 (%)	78.69	78.6	78.67	78.64	78.68	**78.77**
time2(min)	33.3	40.0	53.3	80.0	161	643

5. まとめ

全データの基本統計量を閾値として量子化を行う従来手法は画像データにも音データにも有効性が確認できた．また，全データの中央値を基準として定める閾値を用いる従来手法に対して，本研究で提案する各次元で異なる閾値を用いる手法の優位性を確認することができた．実験結果により，画像データでも音データでも基本統計量から+方向に移動すると良い結果が得られたと分かる．しかし，画像データに対して音データでは各次元の基本統計量を変移しそれを閾値として量子化を行っても，従来手法と比べ距離保存率の差があまり見られなかった．

画像データに関しては各次元の中央値を適度にグループにまとめ動かし，距離保存率が上がる所を調査すると，標準偏差が全体より低い次元の所に現れたことが分かった．また，距離保存率，基本統計量を求める時間および中心点探索時間の三つの要素から，S-Map の中心点探索に用いる量子化の閾値は画像データにも音データにも各次元の中央値という選択がベストだと考える．

今後はデータの各次元のデータの特徴を詳しく調査し，より距離保存率でも計算時間でも効率的な手法を提案する必要がある．

参考文献

[1] A. Guttman: R-trees: A Dynamic Index Structure for Spatial Seaching, in ACM SIGMOD, pp.47–57, (1984).
[2] P. Ciaccia, M. Patella, P. Zezula: M-tree: An efficient access method for similarity search in metric spaces, Proc. 23rd Int. Conf. on Very Large Data Bases, pp. 426–435, 1997.
[3] T. Shinohara, H. Ishizaka: On Dimension Reduction Mapping for Approximate Retrieval of Multi-dimensional Data, Progress Discovery Science, pp. 224–231, (2002).
[4] 中島正八: データベース内オブジェクトの離散化を利用した次元縮小射影 Simple-Map の中心点探索に関する研究, 九州工業大学卒業論文,(2014).
[5] R. Bayer, E. McCreight: Organization and maintenance of large ordered indexes, Acta Informatica Vol. 1, pp. 173–189, 1972.
[6] I. Kamel, C. Faloutsos: Hilbert R-tree: An improved R-tree using fractals, Proc. 20th Int. Conf. on Very Large Data Bases, pp. 500–509, 1994.