次元縮小射影Simple-Mapを用いた空間索引R-treeの最適化に関する研究

樋口 直哉[†1,a)]　今村 安伸　篠原 武[†1]

概要：大量な多次元データを高速に近似検索するために空間索引 R-tree や M-tree などを用いるのが一般的である．R-tree は高次元データを扱うと検索効率が悪化するため，次元縮小射影 Simple-Map を用いる．本論文では，Simple-Map による射影次元数と R-tree のノードに格納する最大要素数（ノードサイズという）に関する最適化の検証を行う．実験により，従来手法と比べ，Simple-Map の性能によらず画像データでは射影次元数とノードサイズを大きくした場合が高速となり，音データでは射影次元数を小さくし，ノードサイズを大きくした場合に高速となることがわかった．

キーワード：近似検索，R-tree，Simple-Map

A study on optimization of spatial index R-tree using dimension reduction mapping Simple-Map

Naoya Higuchi[†1,a)]　Yasunobu Imamura　Takeshi Shinohara[†1]

Abstract: We usually use spatial index such as R-tree and M-tree for quick approximate retrieval of high-dimensional data. We use dimension reduction mapping Simple-Map to prevent the curse of dimensionality from being deteriorated approximate retrieval. In this paper, we optimize the number of dimensions after mapping by Simple-Map and the maximum number of elements(called node size) to be stored in each node of the R-tree. From experiments, compared with the conventional method, similarity search became faster by increasing the number of dimensions after mapping and node size for the image data, decreasing the number of dimensions after mapping and increasing node size for sound data, regardless of the performance of Simple-Map.

Keywords: approximate retrieval of high-dimensional data，spatial index，R-tree，dimension reduction mapping，Simple-Map

1. はじめに

計算機の演算能力や記憶容量の向上により，大量のマルチメディアデータを用いたシステムが多く作られている．そのため，膨大なデータの中から必要なデータだけを探し出す情報検索技術が重要となる．

マルチメディアデータの検索においては，完全一致検索よりも近似検索の方が重要である．完全一致検索では，データの劣化や加工に対応することができず，また人の感覚で同じであると判断できるような情報であっても計算機では違う情報として取り扱ってしまうからである．そのため，計算機を用いて近似検索を行う場合，厳密に定義された距離などの非近似指標を用いて判断を行う．

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，B-tree [1] を多次元に拡張した R-tree [2] や M-tree [3] などの索引構造が一般的に用いられる．しかし，索引構造を用いて高次元の特徴データを扱う場合に，次元の呪いにより検索効率が悪化することが知られている．そこで本論

[†1] 現在，九州工業大学情報工学部知能情報工学科
Presently with Department of Artificial Intelligence, Kyushu Institute of Technology
[a)] k231065n@mail.kyutech.jp

文では，高次元の特徴データを低次元に射影する次元縮小法 Simple-Map(S-Map) [5] を用いることで検索効率の悪化を防ぐ．S-Map は，実空間での射影距離を座標値として射影を行う手法であり，通常のユークリッド距離だけでなく L_1距離（マンハッタン距離）や文字列編集距離など任意の距離空間に対して適用可能であるという点で優れた射影法の一つである．

計算機や S-Map の性能向上 [7][8][9] により，適切な射影次元数や R-tree のノードに格納する最大要素数（ノードサイズという）が変化していることが考えられるため，本論文ではその最適化を図る．

本論文の構成は以下の通りである．近似検索について 2 章で，次元縮小射影法である S-Map について 3 章で述べる．4 章では，検索の最適化のために射影次元数とノードサイズを変化させる実験を行い，5 章でまとめる．

2. 近似検索と索引構造

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ものは，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．以下の三つの距離計測関数は距離の公理を満たすものである．

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

$$L_2\text{距離} : D(x,y) = \sqrt{\sum_{i=1}^{n} (x_{(i)} - y_{(i)})^2} \qquad (2)$$

$$L_\infty\text{距離} : D(x,y) = \max_{i=1}^{n} |x_{(i)} - y_{(i)}| \qquad (3)$$

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 r を無限大に設定する．そこから検索をはじめて，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 r 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 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．対象とする空間を MBR(Minimum Bounding Rectangle) と呼ばれる n 次元超直方体で分割する．ここで n はオブジェクトの次元数である．各ノードはそのすべての子ノードを包括する MBR と子ノードへのポインタを持つ．ただし，葉ノードはデータベース中のオブジェクトへのポインタを持つ．ここで，互いに異なるノードを包括する MBR 同士が重なりあう可能性があることに注意する必要がある．高次元空間では，**次元の呪い**と呼ばれる現象により検索の性能が悪化してしまう．これは，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原因であると考える．本論文では，高次元空間での検索性能の悪化を緩和するために，次元縮小法を用いて，高次元の特徴空間を低次元空間へ射影する．この次元縮小法については 3 章で詳しく説明する．

R-tree における検索では，MBR と質問点との距離の順に，MBR と質問範囲が交差しているノードのみを訪問する．次元縮小を用いて R-tree を構築した場合は，葉ノードにおいて，オブジェクトと質問点との距離計算を行う前に，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．図 1 には MBR A，B，C と質問点を示している．

例えば，図 1 のように全ての MBR A，B，C が初期検索範囲と交差している状況を考える．訪問順を考慮しない場合は要素順，この例では MBR A，B，C の順で全て訪問する．これに対し，訪問順を制御する場合は質問点から最も近い MBR B から訪問し，MBR B に含まれるオブジェクトのうち質問点と最も近いオブジェクトとの距離まで検索範囲を収縮する．これにより，訪問順を制御しなかった場合には訪問しなければならなかった MBR A を訪問する必要がなくなる．このように，R-tree は訪問順を制御することで検索範囲の収縮を早め，効率的な検索を実現している．

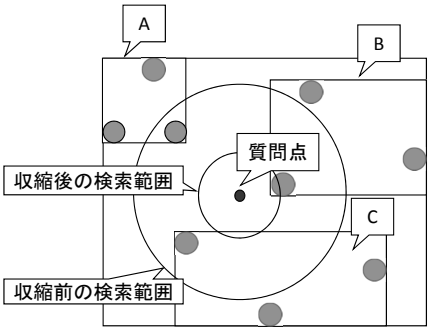

図 1　R-tree による検索
Fig. 1　Similarity search by R-tree

3. 次元縮小

高次元の特徴空間に対して R-tree などの空間索引構造を用いた場合，次元の呪いなどのために，その検索性能が悪化することが知られている．次元の呪いを緩和する手法として，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，Simple-Map(S-Map) [5] を用いる．射影前の特徴空間を元空間，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すべ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可能性がある．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可能性がある．

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

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

S-Map を用いて高次元のデータを低次元空間に射影する際，選択した中心点によって検索効率に大きな差が出てしまう．そのため，中心点の選択方法が高速検索において重要になる．中心点の性能の指標としてスコアを用いる．スコア付けの手法として，射影空間上のオブジェクト対の間の射影距離の総和を用いる．スコアが高いほどより良い中心点であるとしている．

3.3 局所探索

前節の中心点探索法では，データベース内のオブジェクトからランダムに中心点候補を選択していた．しかし，空間全体に対してデータの存在する範囲は非常に小さく，より適した中心点を見逃している可能性が高い．

そこで，局所探索 [6][7] では，前節の手法で中心点の候補として選ばれたオブジェクトを中心に各次元軸の値を少しずつ変化させることで，より良い中心点を探索し検索効率の向上を図っている．中心点候補よりスコアが上回っている点を発見した場合，その点を新たな中心点候補として同様の操作を繰り返す手法である．

3.4 2 値量子化

前節の局所探索で探索した中心点は，座標値として実空間の最大値または最小値が多く現れる．そこで，データベース内オブジェクトに対し，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座標値を量子化した点を中心点候補とし，中心点探索を行う量子化手法がある [8][9]．2 値量子化の閾値として，本論文ではデータベース内オブジェクトの座標値の中央値を用いる．2 値量子化は比較的に短い時間でより良い中心点を見つけることができる．

4. 実験

実験には約 2,800 本の動画から抽出した約 700 万件の画像データと約 1,400 曲から抽出した約 700 万件の音データ，質問データにはデータベース内によく似たデータがある近

質問，やや似たデータがある準近質問，似たデータがない遠質問を約 3 万件ずつ用いる．質問データに関しては実験時間短縮のため，質問を 100 飛ばしで計約 900 件で行う．以下の三つ場合の性能別 S-Map に関しての射影次元数と，R-tree のノードサイズを変化させて，画像データでは射影次元 8 ノードサイズ 100，音データでは射影次元 12 ノードサイズ 100 の従来手法と比較を行う．実験に用いた PC は表 1 の通りである．

(1) 2 値量子化なし．
(2) 2 値量子化あり．
(3) 2 値量子化，局所探索あり．

表 1 実験に用いた PC の性能
Table 1 The SPEC of the PC used in Experiment

CPU	Intel(R) Xeon(R) CPU E5-2640 2.5 GHz × 4
メモリ	64GBytes

4.1 実験 1（ノードサイズの影響）

従来手法と同じ射影次元数においてノードサイズを変更して実験を行った．(1)，(2)，(3) に関してそれぞれ画像データの実験結果を表 2，表 3，表 4，音データの実験結果を表 5，表 6，表 7 に示す．訪問ノード数，距離計算回数，検索時間はそれぞれ 1 質問当たりの平均値である．

結果として，検索時間に関しては画像データ，音データともに (1) (2) (3) いずれの場合においても従来のノードサイズ 100 ではないところに最高性能となるノードサイズがあることが分かる．

表 2 ノードサイズの影響（画像（1））
Table 2 The influence of node size (image (1))

ノードサイズ	訪問ノード数	距離計算回数	検索時間 (ms)
50	36616	1286997	143.75
100	18913	1287579	126.81
200	9827	1288436	116.92
400	5140	1290245	112.06
800	2707	1293100	111.24
900	2432	1294032	**110.95**
1000	2206	1294377	111.75

表 3 ノードサイズの影響（画像（2）
Table 3 The influence of node size (image (2))

ノードサイズ	訪問ノード数	距離計算回数	検索時間 (ms)
50	27852	865172	104.18
100	14548	866019	93.65
200	7631	866996	85.94
400	4023	868646	81.78
800	2131	871661	80.97
900	1915	872180	**80.50**
1000	1741	873150	80.63

表 4 ノードサイズの影響（画像（3）
Table 4 The influence of node size (image (3))

ノードサイズ	訪問ノード数	距離計算回数	検索時間 (ms)
50	24851	762045	93.94
100	13050	762648	83.59
200	6885	763755	76.77
400	3653	765714	74.00
700	2203	767854	73.52
800	1951	768743	**73.40**
900	1756	769647	73.51

表 5 ノードサイズの影響（音（1）
Table 5 The influence of node size (sound (1))

ノードサイズ	訪問ノード数	距離計算回数	検索時間 (ms)
50	20497	684141	104.50
100	10676	685026	94.79
200	5579	685966	89.23
400	2939	687515	86.87
500	2391	687928	**86.32**
600	2023	688596	86.70

表 6　ノードサイズの影響（音（2）
Table 6　The influence of node size (sound (2))

ノードサイズ	訪問ノード数	距離計算回数	検索時間 (ms)
50	18918	589240	93.19
100	9919	590284	83.97
200	5233	591478	80.17
400	2779	592919	77.92
500	2267	593634	**77.61**
600	1925	594107	78.00

表 7　ノードサイズの影響（音（3）
Table 7　The influence of node size (sound (3))

ノードサイズ	訪問ノード数	距離計算回数	検索時間 (ms)
50	18699	563099	91.71
100	9870	563661	83.49
200	5222	564901	79.12
400	2782	566372	76.65
500	2274	566798	76.57
600	1930	566890	**76.34**
700	1677	567654	76.36

4.2　実験 2（射影次元数とノードサイズ）

射影次元数とノードサイズを変更し，各次元数において検索時間が最も短いものの従来手法に対する高速化率に関して，(1) (2) (3) についてそれぞれ画像データの実験結果を表 8，表 9，表 10，音データの実験結果を表 11，表 12，表 13，に示す．

結果として，画像データ，音データともに（1）(2) (3) のいずれの S-Map においても最も高速となる射影次元数，ノードサイズは従来手法とは異なるということがわかる．

表 8　最適ノードサイズ（画像（1））
Table 8　The optimum node sizes (image (1))

射影次元数	8	16	20	24	32	64
ノードサイズ	900	900	600	800	600	400
高速化率 (%)	**12**	5.4	5.5	−0.34	−4.3	−37

表 9　最適ノードサイズ（画像（2））
Table 9　The optimum node sizes (image (2))

射影次元数	8	16	20	24	32	64
ノードサイズ	900	800	600	600	600	400
高速化率 (%)	14	19	**21**	18	19	1.6

表 10　最適ノードサイズ（画像（3））
Table 10　The optimum node sizes (image (3))

射影次元数	8	16	20	24	32	64
ノードサイズ	800	400	500	600	500	400
高速化率 (%)	12	11	**13**	6.6	0.87	−15

表 11　最適ノードサイズ（音（1））
Table 11　The optimum node sizes (sound (1))

射影次元数	4	6	8	12	20	32
ノードサイズ	800	500	600	500	600	500
高速化率 (%)	4.1	8.4	**9.9**	8.9	6.9	−1.7

表 12　最適ノードサイズ（音（2））
Table 12　The optimum node sizes (sound (2))

射影次元数	4	6	8	12	20	32
ノードサイズ	800	600	600	500	500	400
高速化率 (%)	8.0	**12**	10	7.6	5.1	−5.1

表 13　最適ノードサイズ（音（3））
Table 13　The optimum node sizes (sound (3))

射影次元数	4	6	8	12	20	32
ノードサイズ	900	400	600	600	500	400
高速化率 (%)	7.7	**10**	**10**	8.6	4.5	−5.5

5.　まとめと今後の課題

画像データに関しては（1）では 8 次元射影，ノードサイズ 900 で約 12%，（2）では 20 次元射影，ノードサイズ 600 で約 21%，（3）では 20 次元射影，ノードサイズ 500 で約 13%，音データに関しては（1）では 8 次元射影，ノードサイズ 600 で約 9.9%，（2）では 6 次元射影，ノードサイズ 600 で約 12%，（3）では 6 次元射影，ノードサイズ 400 と 8 次元射影，ノードサイズ 600 で約 10%の高速化を実現した．

今後の課題として，ノードサイズによって同じ高さとなる木であっても性能が違うことから，ノードのまとめ方を工夫することでさらなる高速化を期待できる．また，多くの実験を行わずとも S-Map や計算機の性能によらない最適なチューニングを行えるようにすることが挙げられる．

参考文献

[1] R. Bayer, E. McCreight: Organization and maintenance of large ordered indexes, Acta Informatica Vol. 1, pp. 173–189, (1972).

[2] A.Guttman: R-trees: A dynamic index structure for spatial searching, Proc. ACM SIGMOD, International Conference on Management of Data, pp. 47–57, (1984).

[3] 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).

[4] I. Kamel, C. Faloutsos: Hilbert R-tree: An improved R-tree using fractals, In Proceedings of the Twentieth International Conference on Very Large Data Bases, pp. 500–509, (1994).

[5] T. Shinohara, H. Ishizaka: On dimension reduction mappings for approximate retrieval of multi-dimensional data, Progress in Discovery Science, pp. 224–231, (2002).

[6] 中西義和：高次元データの高速近似検索システムのための次元縮小射影法 S-Map の中心点選択法に関する研究，九州工業大学 修士論文，(2006).

[7] 小川史宏：次元縮小射影 Simple-Map の中心点探索における焼きなましと局所探索の有効性に関する研究，九州工業大学 卒業論文，(2014).

[8] 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).

[9] NGUYEN THI BICH HAU：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，九州工業大学 卒業論文，(2015).