データベースシステム 119−18
（1999. 7. 21）

多次元空間における類似探索手法の提案

櫻井 保志[†§] 吉川 正俊[§] 植村 俊亮[§] 児島 治彦[†]

本論文では，高次元空間探索を高速化するための新たな索引手法である部分空間符号化法を提案する．部分空間符号化法では，符号によって形作られる仮想包囲領域の概念とそれを用いたアルゴリズムを導入する．仮想包囲領域による木構造は枝の数が大きいため，探索処理において高い利得を生ずる．我々の評価実験において，部分空間符号化法は優れた性能を示しており，中でもSR-treeに適用した場合，40次元でSR-treeと比べ約63.0%，48次元でVA-Fileと比べ約71.1%のディスクアクセスの削減が可能となった．

Similarity Search by Coding for High-Dimensional Data

Yasushi Sakurai[†§] *Masatoshi Yoshikawa*[§] *Shunsuke Uemura*[§] *Haruhiko Kojima*[†]

We present a new indexing method that is useful for spatial search. In our proposed method, the subspace coding method, the notion of virtual bounding region is introduced. Virtual bounding region is a geometrical object that contains and approximates a minimum bounding region. And then, virtual bounding region is compactly represented by short codes within virtual bounding region of parent node using relative position and tree structure is constructed. Our method is an efficient search algorithms and reaches higher performance. In the case applying to the SR-tree, the experimental results with real data set show that our method outperforms the SR-tree with reductions of more than 63.0 % for 40 dimension and the VA-File with reductions of more than 71.1 % for 48 dimension relative to page accesses.

1 まえがき

マルチメディアデータベースを用いた多くのアプリケーションは，多様な次元の空間探索技術を必要としている．高次元の空間探索に関しては，従来手法を大きく2つのタイプに分類することができる．第1はデータ分割法（data partitioning method）であり，R-treeファミリーがこれに属する．R-treeファミリーの中でも，X-tree[4]とSR-tree[7]は高次元空間において優れた性能を示す構造として報告されている．X-treeはスーパーノードの概念を導入し，R*-treeを上回る性能を有する．SR-treeは最小包囲領域（Minimum Bounding Region）として，最小包囲矩形（MBR；Minimum Bounding Rectangle）と最小包囲球（MBS；Minimum Bounding Sphere）の両方をデータ構造に採り入れることにより，MBRのみを扱うR*-tree[2]やMBSのみを扱うSS-tree[10]よりも高い探索性能を示している．

第2は近似ファイルを用いる手法であり，VA-File（Vector Approximation File）[9]がこれに相当する．VA-Fileは非常に単純な手法ではあるが，強力な探索能力を有する索引機構である．この手法ではデータ空間をセルに区分し，各セルにビット列を割り当てる．セルに格納されるデータオブジェクトのベクトルはセルによって近似され，その幾何学的な近似はVA-Fileとして作成される．そして探索時には，そのVA-Fileを走査して探索すべきオブジェクトの候補集合を作り，その後オブジェクトの空間ベクトルが近傍点であることを確かめるために，ベクトルファイルにおける候補集合に関する部位のみを参照する．文献[9]では，6次元以上においてVA-FileがX-tree，R*-treeを上回ると報告されている．つまり，高次元空間の探索に適した空間アクセス法の中では，SR-treeとVA-Fileは他の手法と比べより優れた手法であると言える．

† NTTサイバースペース研究所
NTT Cyber Space Laboratories, Japan
§ 奈良先端科学技術大学院大学 情報科学研究科
Graduate School of Information Science,
Nara Institute of Science and Technology, Japan

本論文では，MBR，MBS，もしくはその両方を扱う索引構造に適用可能な索引手法である部分空間符号化法（SCM；Subspace Coding Method）を提案する．SCMの主たるアイデアは仮想包囲領域（Virtual Bounding Region）の概念にある．これは最小包囲領域を包含し，かつそれを近似する幾何学的オブジェクトである．さらに，仮想包囲領域を低コストで表現する部分空間符号を導入し，索引における枝の数の増大を図る．VA-Fileが特徴ベクトルの絶対的位置を近似しているのに対し，SCMによる仮想包囲領域の表現は親の仮想包囲領域の相対的位置に基づいている．この特徴は実際のアプリケーションで用いられるような実データに特に有効である．SCMでは，仮想包囲領域を保有するノンリーフノードによって構成される木構造を構築する．そして，仮想包囲領域による木構造を用いて探索処理を実施する．また，SCMは他の木構造索引とは種類を異にするものであり，R-tree，R*-tree，X-tree，SS-tree，SR-treeなどの最小包囲領域を用いた空間索引に適用し，探索性能向上を図る手法である．その中でも，SR-treeは各ノンリーフノードにMBRとMBSの両方を格納しているため，部分空間符号の導入の効果が大きく，我々の手法は特にSR-treeに有効であると言える．本論文における実データを用いた実験結果では，高次元空間探索におけるSCMの有効性が明らかとなった．

2 部分空間符号化法

本論文で提案する部分空間符号化法（SCM；Subspace Coding Method）の主たるアイデアは仮想包囲領域（Virtual Bounding Region）の概念の導入にある．仮想包囲領域は，仮想包囲矩形（VBR；Virtual Bounding Rectangle）と仮想包囲擬球（VBS；Virtual Bounding quasi-Sphere）を総称したものであり，各々MBRとMBSを包含し，かつそれを近似する幾何学的オブジェクトである．仮想包囲領域の蓄積コストは最小包囲領域と比べて非常に小さいため，ノンリーフノードに仮想包囲領域を格納することは，枝の数を増やすことになる．このアイデアはprefix

B-tree[1] における概念と類似している．prefix B-tree と本手法は，双方ともノンリーフノードにおけるエントリのサイズを小さくし，枝の数を大きくする．しかし prefix B-tree と異なり，仮想包囲領域には性能劣化の要因が含まれている．最小包囲領域を包含，近似する仮想包囲領域は，元の最小包囲領域と比べてサイズがわずかに大きく，これは探索処理において枝刈り性能の劣化を招く．この問題に関して3節における実験結果は，仮想包囲領域の正の効果が負の影響を大きく上回ることを示している．

SCM の重要な特徴の一つとして，包囲領域に関する相対的位置の近似を利用していることが挙げられる．R-tree ファミリーにおいてはノード間の最小包囲領域の重なりを許可する手法であるため，多次元空間探索において，下位ノードのデータの近似が探索性能に大きな影響を及ぼす．そこで，位置表現コストを可能な限り低減化しながら，近似誤差の増大を抑制しなければならない．この要求を満たすのが，本論文で提案する SCM である．VA-File ではベクトルデータが絶対的位置に基づいて近似されているのに対し，本手法では親の仮想包囲領域の相対的位置に基づいて仮想包囲領域が近似される．VA-File において用いられているベクトルの絶対的位置の近似はデータ分布に独立であるため，密集したデータは同一の値によって近似される．したがって，実際のアプリケーションで用いられるような，分布が不均一であるデータに VA-File は向かない．対照的に，SCM の相対的位置表現は分布が不均一であるデータに有効に働くものである．

本節では，まず VBR と VBS について説明し，さらに SCM に関するデータ構造および探索，更新アルゴリズムについて述べる．

2.1 準備

n 次元空間における包囲矩形 A は，対角をなす 2 つの端点 a と a' によって表現することができる．すなわち，

$$A = (a, a')$$

ここで，

$$a = [\phi_1, \phi_2, \ldots, \phi_n],\ a' = [\phi'_1, \phi'_2, \ldots, \phi'_n]$$
$$(\phi_i \leq \phi'_i; i \in \{1, 2, \ldots, n\}).$$

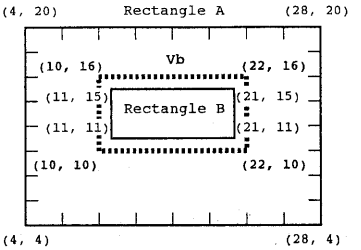
n 次元空間内に幾何学的オブジェクト Bが存在するものとし，ここでは Bが矩形の場合と点の場合を想定する．そして，A，Bに対し，$B \subseteq A$が成り立つものとする．$B \subseteq A$ は，Bが A に包含されていることを意味する．つまり，例えば Bが矩形の場合，$A = (a, a')$ とその座標値 a，a'が与えられており，また同様に $B = (b, b')$ とその座標値 b，b' $(b = [\psi_1, \psi_2, \ldots, \psi_n],\ b' = [\psi'_1, \psi'_2, \ldots, \psi'_n].)$ が与えられているものとすると，下式を満たす場合またその場合にのみ $B \subseteq A$ が成立することを意味する．

$$\phi_i \leq \psi_i \leq \psi'_i \leq \phi'_i \quad (i = 1, 2, \ldots, n). \tag{1}$$

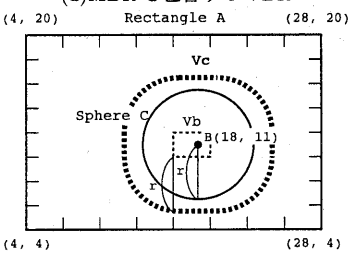
さらに Bが点の場合とは，$b = b'$の条件を満たす場合である．この場合も $B \subseteq A$ が成り立つものとする．これ以降，$b = b'$の条件を満たす場合を矩形の一形態として考える．したがって，以下の論述は Bが点の場合についても成り立つ．

2.2 仮想包囲領域と部分空間符号

従来手法である R-tree ファミリーにおいては，MBR を表現する端点の座標値，もしくは MBS における中心点の座標値を座標系における絶対的位置によって表現している．絶対的位置を用いれば，厳密な位置表現が可能であるが，反面多くの蓄積サイズを必要とし，またそのサイズは次元数に比例する．



(a)MBR を包含する VBR



(b)MBS を包含する VBS

図 1: 仮想包囲領域を用いた空間表現の例

R-tree ファミリーにおいては親ノードの包囲領域が子ノードの包囲領域を包含している．それゆえ親ノードの包囲領域 A の位置に基づいて，子ノードの包囲領域 Bの位置を相対的に表現することが可能である．SCM はこの考えに基づいたものであり，相対的位置の近似を用いることによって位置表現を行うためのビット長を大幅に削減することが可能になる．

定義 1 n 次元空間内において，矩形 A と Bが存在し，$B \subseteq A$ が成り立つものとする．また，qを *1* 以上の整数とする．以下に示される矩形 V_bを，基数 qの A における Bの**仮想包囲矩形**（*VBR ; Virtual Bounding Rectangle*）と呼ぶ．

$$V_b = (v, v')$$

ここで，

$$v = [\tau_1, \tau_2, \ldots, \tau_n],\ v' = [\tau'_1, \tau'_2, \ldots, \tau'_n],$$
$$\tau_i \leq \tau'_i$$

ただし，

$$\tau_i = \phi_i + \left\lfloor \frac{\psi_i - \phi_i}{(\phi'_i - \phi_i)/q} \right\rfloor \cdot (\phi'_i - \phi_i)/q \tag{2}$$

$$\tau'_i = \phi_i + \left\lceil \frac{\psi'_i - \phi_i}{(\phi'_i - \phi_i)/q} \right\rceil \cdot (\phi'_i - \phi_i)/q \tag{3}$$
$$(i = 1, 2, \ldots, n).$$

□

例 1 図 *1(a)* のように，$B \subseteq A$ を満たす矩形 A と Bが与えられているとする．点線で示されている矩形 Vb が，基数 *8* の A における Bの *VBR* である．□

補題 1 A による Bの *VBR* Vに対して，$B \subseteq V$と $V \subseteq A$ が成り立つ．

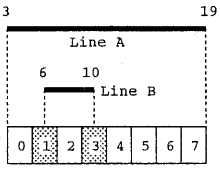


図 2: 部分空間符号の例

証明: $B \subseteq A$ であるため不等式 (1) が成り立つ．それゆえ，式 (2) において

$$\left\lfloor \frac{\psi_i - \phi_i}{(\phi_i' - \phi_i)/q} \right\rfloor \cdot (\phi_i' - \phi_i)/q \ \geq \ 0$$

であるため，$\tau_i \geq \phi_i$.

さらに式 (2) から

$$\begin{aligned} \tau_i &= \phi_i + \left\lfloor \frac{\psi_i - \phi_i}{(\phi_i' - \phi_i)/q} \right\rfloor \cdot (\phi_i' - \phi_i)/q \\ &\leq \phi_i + \frac{\psi_i - \phi_i}{(\phi_i' - \phi_i)/q} \cdot (\phi_i' - \phi_i)/q \\ &= \psi_i \end{aligned}$$

である．それゆえ，$\phi_i \leq \tau_i \leq \psi_i$．同様に，$\psi_i' \leq \tau_i' \leq \phi_i'$．したがって，$B \subseteq V$と $V \subseteq A$ が成立する．□

定義 2 矩形 A，および点 Bを中心とする半径 rの球 Cが与えられており，n 次元空間内において $B \subseteq A$ が成り立つものとする．さらに，V_bを A における Bの *VBR* とする．このとき，Cと V_bの *Minkowski sum[3]* によって表現される幾何学的オブジェクト V_cを A における Cの**仮想包囲擬球** (*VBS ; Virtual Bounding quasiSphere*) と呼ぶ．□

また，VBR と VBS を総称して仮想包囲領域（Virtual Bounding Region）と呼ぶ．

例 2 図 *1(b)* のように，矩形 A と球 Cが与えられているとする．また，rを球 Cの半径，Bを $B \subseteq A$ を満たす球 Cの中心とする．このとき，点線で示されている V_bが A における Bの *VBR*，V_cが A における Cの *VBS* である．□

定理 1 *[5]* のアルゴリズムに関して，最小包囲領域を用いた探索と仮想包囲領域を用いた探索は同じ結果を生ずる．

証明: MBR B_rと点 B_pを中心とする MBS Cが与えられているとする．また，V_r，V_p，V_cをそれぞれ B_rの VBR，B_p の VBR，Cの VBS とする．まず，補題1より $B_r \subseteq V_r$が成り立つ．同様に，$B_p \subseteq V_p$であるため，$C \subseteq V_c$が成立する．以上より任意の問い合わせ点 Q に対して，最小包囲領域 B と，Bの仮想包囲領域 Vの間にメトリック MINDIST *に関する下式が成り立つ．

$$MINDIST(Q,V) \leq MINDIST(Q,B)$$

したがって，最小包囲領域を用いた探索と仮想包囲領域を用いた探索は同じ結果を生ずる．□

*与えられた点 Pと幾何学的領域 Rに関してメトリック MINDIST は，Pが Rに包含されていないときに Pと Rの最小距離を，包含されているときに 0 を示す．

例 3 図 *2*は，部分空間符号を計算する方法について簡単に例示したものである．直線 A が座標値として *(3, 19)* を，そして直線 Bが同様に座標値として *(6, 10)* を占めている．このとき，長さ lの符号を用いて，A の座標範囲を 2^lの部分区間に等分割することにより，Bの始点をそれが存在する部分区間によって近似的に表現することができる．これは，Bの終点についても同様である．例えば，長さ *3* の符号を用いて，A の領域を *8* つの部分区間に分割した場合，Bは A の始点から数えて前から *2* 番目から *4* 番目の部分区間を占めていることになる．それゆえ，Bの領域は *8* 元符号 *(1, 3)*，もしくは *2* 進符号では *(001, 011)* のように近似的に表現することが可能である．この場合，*2* 進符号において必要な符号長は，Bの始点，終点それぞれに *3* であるため，次元あたり合計 *6* となる．また Bが点の場合も同様に計算できる．ただし，点の場合は符号長が半分の長さになる．□

定義 3 $A = (a, a')$ と $B = (b, b')$ を n 次元空間内の矩形とする．また，$V = (v, v')$ を基数 qの A における Bの *VBR* とする．ここで A, B, Vは $B \subseteq V \subseteq A$ の関係を満たす．したがって，下式が成り立つ．

$$\phi_i \leq \tau_i \leq \psi_i \leq \psi_i' \leq \tau_i' \leq \phi_i' \quad (i = 1, 2, \ldots, n).$$

さらに，η_iと η_i'を i 次元座標上の部分区間を表す q元符号とする（$\eta_i \leq \eta_i'$）．すなわち，

$$\begin{aligned} \eta_i &= \frac{(\tau_i - \phi_i) \cdot q}{\phi_i' - \phi_i} \\ \eta_i' &= \frac{(\tau_i' - \phi_i) \cdot q}{\phi_i' - \phi_i} - 1. \end{aligned}$$

また，$F_2(\eta_i, l)$ を符号長 lとするη_iの *2* 進表現とする（$l = \lceil \log_2 q \rceil$）．このとき以下の *2* 進符号を，$A$ における Vの**部分空間符号**と呼ぶ．

$$S = (s, s')$$

ここで，

$$\begin{aligned} s &= [F_2(\eta_1, l), F_2(\eta_2, l), \ldots, F_2(\eta_n, l)], \\ s' &= [F_2(\eta_1', l), F_2(\eta_2', l), \ldots, F_2(\eta_n', l)]. \end{aligned}$$

また Bが点オブジェクトである場合（すなわち $B = (b)$），部分空間符号は $S = (s)$ によって表現される．□

少ない情報量にもかかわらず，部分空間符号は MBR，点オブジェクトおよび MBS を復元する能力を有する．

例 4 図 *1*を用いて，*2* 次元空間における部分空間符号の例を示す．図 *1(a)* では $B \subseteq V_b \subseteq A$ を満足する矩形 A と B，さらに A における Bの *VBR* V_bが与えられている．このとき，基数 *8* の A における Vb の部分空間符号は

$$S = (010, 011, 101, 101)$$

のように表現される．また，図 *1(b)* に示されている球 Cの中心点である Bの量子化も同様に

$$S = (100, 011)$$

のように行う．Cの *VBS* V_cは，部分空間符号 Sと，Cの半径 rによって位置表現することが可能である．□

2.3 部分空間符号の木構造索引への適用

SCM を R-tree ファミリーに基づく木構造索引に適用した場合，以下の点で従来と異なる．

(1) 本手法を適用する木構造とは別に，部分空間符号で構成される木構造を生成する．2 つの木構造を区別するために，絶対的位置表現を用いる元の木構造を実部分（real part），相対的位置表現を用いる部分空間符号により構成される木構造を仮想部分（virtual part）と呼ぶ．

(2) 木構造の実部分における各々のノードに対応して，仮想部分のノードを生成する．ただし，実部分のリーフノードについては，仮想部分において生成されない．実部分の木構造の高さを s，根ノードのレベルを s とするとき，仮想部分の 2 レベルのノードは，子ノードのポインタとして実部分の 1 レベルのノード，つまりリーフノードを指す．

(3) また X-tree[4] もしくは Hilbert R-tree[6] を除き，R-tree ファミリーにおいては，主として 1 ノードが 1 ページを占有するように設計されている．本手法では，仮想部分において 1 ノードが 1 ページを占有するように設計するため，実部分のノードは複数ページを占有することになる．このことにより，木構造における枝の数は大きくなる．

(4) 根ノードの MBR の位置を座標値として保持しておく必要がある．

(5) 仮想部分において，子ノードの仮想包囲領域を表現する部分空間符号は，現ノードの仮想包囲領域と実部分に含まれる子ノードの最小包囲領域を基に計算される．さらに，孫ノードの仮想包囲領域を表現する部分空間符号は子ノードの仮想包囲領域と孫ノードの最小包囲領域を基に計算される．より詳細には，MBR のみを含む木構造（例えば R*-tree や X-tree）においては部分空間符号を現ノードの VBR と子ノードの MBR を基に計算する．そして，MBR と MBS を含む木構造，すなわち SR-tree においては，子ノードの MBS の中心点を近似する部分空間符号を現ノードの VBR と子ノードの MBS の中心点を基に計算する．また VBR の部分空間符号に関しては R*-tree や X-tree などと同様に作成する．さらに，木構造に MBR を含まない SS-tree では，現ノードの VBS を包含する最小矩形をまず計算し，計算によって求めた矩形と子ノードの MBS の中心点を基に部分空間符号を計算する．

(6) 現ノードが根である場合のみ，根の MBR と子ノードの最小包囲領域を基に子ノードの仮想包囲領域を計算する．

(7) 探索処理においては，仮想部分の各ノード毎に，現ノードの仮想包囲領域と子ノードの部分空間符号を基に子ノードの仮想包囲領域を計算する．計算によって求めた仮想包囲領域に対して枝刈り戦略を適用する．

SCM では，実部分とは別に仮想部分を構築する．仮想部分のノードは，ノードの位置座標を符号化していることを除いて実部分のノードと同じ構造を有する．例として，図 3 は実部分と仮想部分の関係を表したものである．図 3(a) において，根ノードの MBR を Rとする．MBR $M1$，$M2$ は Rに包含されており，MBR $M3$，$M4$ は $M1$ に包含されている．この構造において，$V1$，$V2$，$V3$，$V4$ は各々Rにおける $M1$ の VBR，Rにおける $M2$ の VBR，$V1$ における $M3$ の VBR，$V1$ における $M4$ の VBR である．図 3(b) のように，仮想部分と実部分は同じ構造を保つように構築される．また，図 3(c) は SCM を SR-tree に適用した場合の例を示している．この図において，$VS1$ は Rにおける $S1$ の VBS，$V1$ は Rにおける $M1$ の VBR である．根ノードにおいて，$VS1$ と $V1$ は同一のエントリに格納される．

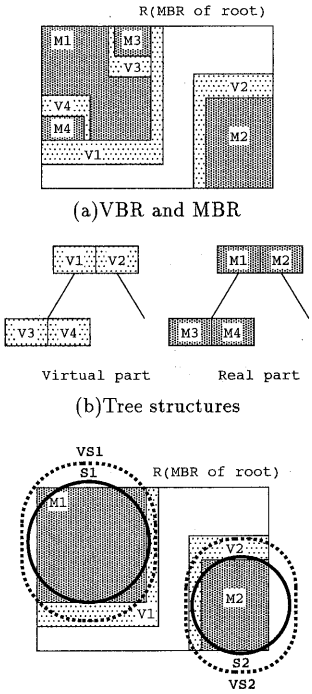


図 3: 木構造における実部分と仮想部分

2.4 探索処理

探索処理においては，オブジェクトを検出し，ノードの枝刈りを実施するために仮想部分のノードと実部分のリーフノードを用いる．定理 1で示したように，仮想包囲領域で示されている枝刈りアルゴリズムに適用することが可能である．

図 4は，[5] のアルゴリズムを改良し，VBR と VBS を用いた近傍探索アルゴリズムを記述したものである．ただし，変数の定義は省略している．ステップ 11 と 12 における *reconstruct* は与えられた矩形 A と部分空間符号 Sから仮想包囲領域 Rを計算する関数である．

$$R = reconstruct(A, S).$$

このアルゴリズムにより，問い合わせ点を中心とした近傍オブジェクトを k個収集することができる．

プロシージャ *search* では，まず初期設定として根ノード，距離 0，根ノードの MBR を優先キューに設定する（ステップ 1）．ステップ 3 では問い合わせ点 *query*から接近したノードがキューから取り出される．そして関数 *reconstruct* を用い，訪問したノンリーフノードにおいて，優先キューから取り出した現ノードの包囲矩形と子ノードの部分空間符号に基づいて子ノードの仮想包囲領域を計算する（ステップ 11，12）．仮想包囲領域と問い合わせ点 *query*との距離として，VBR との最小距離と VBS との最小距離を求め，両者の間で最も大きい距離値を選択する（ステップ 13）．もし計算により求めた距離値が，*query*と k番目の最近傍との距離よりも小さければ，子ポインタ，問い合わせ点 *query*と仮想包囲領域との距離，そして仮想包囲領域をキューに挿入する（ステップ 14，15）．また，訪問したノードがリーフ

```
Procedure search(Point query, Integer k)
1.   queueInput(root, 0, rootrectangle);
2.   while emptyQueue() = false do
3.     p = queueOutput();
4.     if p.distance > nnlist[k].distance then break;
5.     if p.node is a leaf node then
6.       for each object ∈ p.node do
7.         if MINDIST(query, object) ≤
                    nnlist[k].distance then
8.           nnlistInput(object,
                    MINDIST(query, object));
9.     else   // p.node is a non-leaf node
10.      for each entry ∈ p.node do
11.        R_VBR := reconstruct(p.rectangle,
                    entry.code_VBR);
12.        R_VBS := reconstruct(p.rectangle,
                    entry.code_VBS);
13.        dist = max(MINDIST(query, R_VBR),
                    MINDIST(query, R_VBS));
14.        if dist ≤ nnlist[k].distance then
15.          queueInput(entry.node, dist, R_VBR);
16.      end;
17.    end;
18.    report(nnlist);   // output answer
   end.
```

図 4: 最近傍探索アルゴリズム

ノードであれば，データオブジェクトと問い合わせ点との距離を算出し，オブジェクトと求めた距離を最近傍の候補として *nnlist* に格納する（ステップ 8）．

ここで，探索アルゴリズムを図 3(a) を用いて説明する．あらかじめ VBR の部分空間符号が作成されているものとする．まず，Rの矩形位置と $V1$ と $V2$ の部分空間符号によって，$V1$ と $V2$ を計算する．そして，もし問い合わせ点と $V1$ との距離が k番目の最近傍との距離よりも小さい値である場合，さらに $V1$ の位置と $V3$ と $V4$ の部分空間符号によって $V3$ と $V4$ を計算する．そして $V3$ と $V4$ に関しても k番目の最近傍と比較する．また，包囲球を用いた探索も同様のアルゴリズムにより実行することが可能である．

3 性能試験

本手法の効果を検証するために，アルゴリズムを実装し，従来手法との比較実験を行った．本手法は SR-tree と R*-tree に対して適用する．また比較対象として VA-File とオリジナルの SR-tree と R*-tree を用いる．

評価実験では，実データを用いており，これは文献 [8] において用いられている画像から抽出した 100,000 件のヒストグラムデータであり，色相の値に基づき次元数に分割して生成している．ページサイズは 4KB である．そして，R-tree ファミリーの索引構造に対する問い合わせには，Hjaltason らによる探索アルゴリズム [5] を用いる．探索コストにおける実験結果は，1,000 回の施行の平均値である．問い合わせ点は，データ集合とは異なるデータを用いている．探索における CPU 時間は，SUN UltraSPARC-II 296MHz によって計測した．

3.1 探索性能

図 5は本手法のディスクアクセス数を VA-File, SR-tree, R*-tree と比較したものである．SCM に関しては，符号長 lに関して，$l=4$, $l=6$, $l=8$, $l=10$, $l=12$ の構造を実験した．最適な構造は 4 次元と 8 次元においては $l=12$,

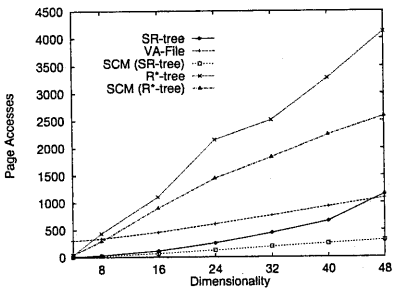


図 5: 次元数に対するディスクアクセスの変動

16 次元から 24 次元までは $l=8$, 32 次元から 48 次元までは $l=6$ と $l=8$ が同等の性能であり，他を上回っている．そこで，4 次元と 8 次元においては $l=12$, 16 次元から 48 次元までは $l=8$ の構造を選択した．また，VA-File に関しては文献 [9] に従い，$l=4$, $l=6$, $l=8$ の構造を実験した．我々の実験では 4 次元から 16 次元までは $l=8$, 20 次元から 48 次元までは $l=6$ が優れていることから，それらを選択した．

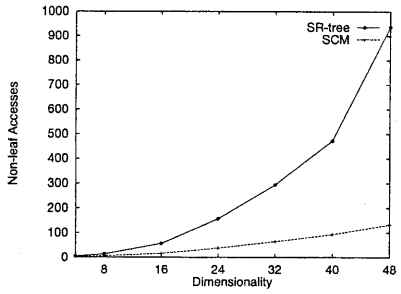
図 5では，全次元における本手法の有効性が確認できる．特に，SR-tree に適用した本手法は全次元数において他の手法を上回っている．高次元になれば SR-tree の性能は悪化するのに対し，本手法の性能は高次元であっても悪化せず，VA-File よりも優れている．加えて，次元が高くなるにつれて本手法の優位性は増す．例えば，SR-tree と比べ 40 次元で約 63.0%，48 次元で約 72.8%，また VA-File と比べ 48 次元で約 71.1%のコスト削減を達成している．さらに，本手法は SR-tree 以外にも有効に働く．我々は実験によって R*-tree に本手法を適用した場合の性能を調査し，8 次元で 32.4%，48 次元で 37.4%の低減化を確認している．

3.2 SCM の優位性

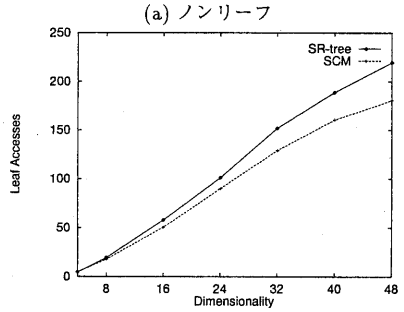
以下では本手法が優位となる理由について説明する．まず，VA-File に対する本手法の優位性について説明する．SCM と VA-File は，座標値の近似というアイデアに関して共通する．しかし，アルゴリズムやデータ構造において大きく異なり，実験結果ではディスクアクセス数には大きな差が生じている．VA-File は探索空間全体から近似値を求めているのに対して，SCM は階層構造をとっており，上位ノードの包囲領域から近似値を求める．つまり，SCM では下位ノードであるほどその近似による誤差は微小なものとなる．このことから，オブジェクトのベクトルデータへのアクセスは非常に少ないものとなり，探索コストの低減化につながる．

次に SR-tree と本手法との比較である．図 6はオリジナルの SR-tree と SR-tree に適用した本手法とのノードアクセス数を示したものである．図 6(a) に示されているように，本手法と SR-tree ではノンリーフノードのアクセス数が大きく異なる．これは次元数が増加するにつれて顕著になる．探索で用いる仮想部分においては，包囲領域の蓄積コストが格段に小さい．これは 2 つの利点がある．まず第 1 にノンリーフノードの容量が小さくなり，その分ディスクアクセスの低減化につながる．第 2 に，枝の数が多くなる．枝の数が大きくなることによって枝刈り能力が大幅に向上し，木構造におけるノンリーフノードのアクセス数が節約される．

次に，図 6(b) に示されているように，本手法は SR-tree と比べてリーフアクセス数で優位に立っている．本手法のノンリーフノードにおける包囲領域は近似されたものであるため，若干の誤差を含んでいる．この誤差は探索性能の



(a) ノンリーフ



(b) リーフ

図 6: 次元数に対するノードアクセス数

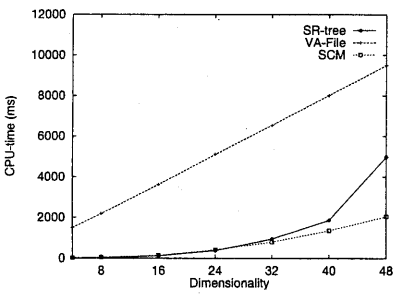


図 7: 探索における CPU 時間

低下につながる．一方で，ノンリーフノードに関する枝の数が多くなるために，枝刈り性能が向上する．枝の数が多くなった効果は誤差の影響よりも大きく，結果としてノンリーフノードのアクセスのみならずリーフアクセスについても節約が可能となる．

さらに我々は VA-File，SR-tree，そして SR-tree に適用した本手法の CPU 時間を計測した．これを図 7に示す．本手法と VA-File における符号長に関しては，図 5における実験と同一にしている．VA-File は全オブジェクトの近似値を求めるために，非常に多くの時間を要する．一方，本手法も包囲領域の位置を部分空間符号から計算するための時間を要するが，VA-File より低い値となっている．これは，我々の手法の枝刈り性能が高いためである．

4 むすび

本論文では，多次元空間において探索性能を高めるために有効な部分空間符号化法（SCM ; Subspace Coding Method）を提案した．SCM における部分空間符号は，親ノードの位置に基づく相対的な位置を表現したものである．さらに仮想包囲領域（Virtual Bounding Region）は，その部分空間符号を用いて構成するものであり，仮想包囲領域を用いた探索アルゴリズムは従来手法である VA-File，SR-tree を大きく上回る探索性能を実現した．

また空間探索の分野において SCM は R-tree ファミリーのような最小包囲領域を用いた空間索引の種類とは直交する．すなわち，R-tree，R*-tree，X-tree，SR-tree などの最小包囲領域を用いた空間索引に SCM を適用し，探索性能を向上させることが可能となる．

部分空間符号は最小包囲領域の座標値を近似するものであるが，最終的に得られる探索結果は近似解とはならない．すなわち，探索処理において近傍オブジェクトを失策することは無い．したがって，大幅な探索性能向上が可能である本手法は実用的な使用状況において適した手法であると言える．

参考文献

[1] Rudolf Bayer and Karl Unterauer. Prefix B-Trees. *ACM Trans. on Database Systems*, Vol. 2, No. 1, pp. 11–26, March 1977.

[2] N. Beckmann, H. P. Kriegel, R. Schneider, and B. Seeger. The R*-tree: An Efficient and Robust Access Method for Points and Rectangles. In *Proc. ACM SIGMOD Conf.*, pp. 322–331, Atlantic City, NJ, May 1990.

[3] S. Berchtold, C. Bohm, D. A. Keim, and H.-P. Kriegel. A Cost Model For Nearest Neighbor Search in High-Dimensional Data Space. In *Proc. ACM Symp. on Principles of Database Systems*, pp. 1–12, March 1997.

[4] Stefan Berchtold, Daniel A. Keim, and Hans-Peter Kriegel. The X-tree: An Index Structure for High-Dimensional Data. In *Proc. of the 22nd International Conference on Very Large Data Bases (VLDB)*, pp. 28–39, Bombay, September 1996.

[5] G. R. Hjaltason and H. Samet. Ranking in Spatial Database. In *Proceedings of the 4th Symposium on Spatial Databases*, pp. 83–95, Portland, Maine, Aug. 1995.

[6] Ibrahim Kamel and Christos Faloutsos. Hilbert R-tree: An Improved R-tree using Fractals. In *Proceedings of the Twentieth International Conference on Very Large Databases*, pp. 500–509, Santiago, Chile, 1994.

[7] Norio Katayama and Shin'ichi Satoh. The SR-tree: An Index Structure for High-Dimensional Nearest Neighbor Queries. In *Proc. ACM SIGMOD International Conference on Management of Data*, pp. 369–380, May 1997.

[8] T. Satou, A. Akutsu, and Y. Tonomura. Video Corpus Construction and Analysis. In *Proc. of IEEE International Conference on Multimedia Computing and Systems (ICMCS)*, pp. II-479–485, Florence, June 1999.

[9] Roger Weber, Hans-J. Schek, and Stephen Blott. A Quantitative Analysis and Performance Study for Similarity-Search Methods in High-Dimensional Spaces. In *Proc. of the 24th International Conference on Very Large Data Bases (VLDB)*, pp. 194–205, New York City, NY, August 1998.

[10] David A. White and Ramesh Jain. Similarity Indexing with the SS-tree. In *Proc. of IEEE 12th International Conference on Data Engineering*, pp. 516–523, 1996.