2003－DBS－131　(29)
2003／7／17

超平面アレンジメントに基づく多次元空間幾何アルゴリズムの実装と評価

金子　邦彦†　　牧之内顕文†

† 九州大学 大学院 システム情報科学研究院　〒 812-8581 福岡市東区箱崎 6-10-1
E-mail: †{kaneko,akifumi}@is.kyushu-u.ac.jp

あらまし　本研究では，新しい空間データモデルとして，HA-face complex モデルを提案する．HA-face complex モデルでは，超平面アレンジメントのフェイスの複体として，空間物の位置と形を表現する．超平面アレンジメントのフェイスとは，ある次元の空間を複数の超平面で区切ったときにできる区画のことである．このモデルは，次元に特化しない空間データモデルであって，本質的に，任意の次元の空間データの表現に用いることができる．本研究では，HA-face complex モデル上で動く幾何計算（和や差や積などの集合論演算子や，交わる，離れている，含む，等しい，接するなどの位相関係述語）の効率よいアルゴリズムである localized divide-and-conquer と，2 ステップリファインメントの提案も行う．実験によって，これらアルゴリズムの効果の確認も行っている．

キーワード　空間データモデル，空間データ処理，計算幾何学，超平面アレンジメント

HA-face Complex Spatial Data Model for Uniform Representation of Data and Algorithms

Kunihiko KANEKO† and Akifumi MAKINOUCHI†

† Graduate School of Information Science and Electrical Engineering, Kyushu University　6-10-1 Hakozaki, Higashi-Ku Fukuoka, 812-8581 Japan
E-mail: †{kaneko,akifumi}@is.kyushu-u.ac.jp

Abstract　This paper presents a kernel system for spatial database systems named *Hawk's Eye*. We implemented a new spatial data model named *HA-face complex* on *Hawk's Eye*. The model represents the spatial position and region of spatial objects of various dimensions in 2-, 3- or 4- dimensional space uniformly. A spatial object represented by the model is a finite set of hyperplane arrangement faces, each of which is a region in a hyperplane arrangement induced by hyperplanes in 2-, 3- or 4- dimensional space. We propose a new algorithm named *localized divide-and-conquer* to evaluate geometric operations and spatial relationships of two spatial objects represented by the model. The model is implemented on an object database system. Our experimental tests show that the localized divide-and-conquer is faster than other algorithms using an arrangement construction algorithm incremental or divide-and-conquer.

Key words　spatial data model, spatial query processing, computational geometry, hyperplane arrangement

1. はじめに

行政（上下水道や土地台帳などの管理，都市計画，防災），ビジネス（マーケティング，出店計画，集配送管理，不動産管理など），自然科学（考古学，生物学，環境学，気象学）などのアプリケーションを持つ空間データベースの重要性は高い．我々は，空間データベース構築のための基盤システム *Hawk's Eye* を研究・開発してきた．*Hawk's Eye* は，例えば多角形や立体などの広がりを持った空間物を含む種々の次元の空間データを扱う空間データベースを，容易に構築できるようにするための基盤ソフトウエアシステムであり，空間データの格納・幾何計算・検索に関する種々の基本機能を提供する．

空間データベースでは，空間データの幾何計算 [13] が重要な機能である．ここでいう幾何計算とは，2 つの異なった空間物の間の交差判定や包含判定を行ったり，あ

るいは，交差領域の形を正確に求めたりなどである．例えば，土地利用状況，行政区画，交通網などの地形データが入ったデータベースにおいて,「ある町にある道路を求めよ」,「ある町にある農地を求めよ」など複数のデータを組み合わせたような処理では，図形同士の幾何計算が実行される．

David らの論文 [2] などでは，空間データを超平面アレンジメントという概念を使って取り扱うことの有効性がいわれてきた．超平面アレンジメントは，計算幾何学の分野での有名な概念であり，応用としては，k 次の Voronoi 図，縮体判定，点集合によって張られる体積最小の単体の発見などがある [3] [4]．超平面アレンジメントは，ロボットの行動計画，コンピュータビジョン，コンピュータグラフィックスなどの分野で，実際に利用されている．空間物の幾何計算の方法としては，2 つの空間物の形の情報から，いったん超平面アレンジメントを構成し，それを使って，各種の幾何計算の結果（例えば，2 つの空間物の交差判定や，交差領域の形を求めたりなど）を導き出すというやり方がある．複数の超平面から超平面アレンジメントを構成する効率の良いアルゴリズムもすでにあり，さまざまな文献 [3] [7] [12] で解説されている．超平面アレンジメントを，空間物の幾何計算に利用することのメリットとしては，次元に特化しない（本質的に任意の次元まで扱うことができ，また，同じ空間中にあれば，次元の異なる空間物同士の幾何計算にも使える）こと，空間物の形に制限（凸でなければならないなど）を設ける必要が無いことがある．

従来，超平面アレンジメントの重要性が認識されながらも，その利用が実際にはなされてこなかった理由は性能にあった．超平面アレンジメントを用いて実際の問題を解く場合，実行時間は，超平面アレンジメントの複雑さ（超平面アレンジメントの face の数）に応じて増加する．超平面アレンジメントの複雑さは，その超平面数を n として，$O(n^d)$ と見積もられる [4] [7]．従って，現実的には，超平面アレンジメントを用いての幾何アルゴリズムの実行時間は許容範囲外である [2] とされてきた．現在まで，超平面アレンジメントを使って，実際に空間データベースを実装したとの報告は無されてこなかったのが現状である．

従来研究における幾何計算の実現は，次元に特化しないアルゴリズム（線形計画法や整数計画法）と，次元に特化したアルゴリズムとがあった．例えば，Dedale [9] では，空間物を線形不等式で表現される制約式を組み合わせて表現し，Half-space intersection 法（これは計算幾何のアルゴリズム）を用いて解く方式をとるが，これは，2 次元に特化したアルゴリズムである．つまり，3 次元の空間物を扱うことは一般にはできない．一方，線形計画法，整数計画法による方法は，空間物の交差判定を行うことができるのみで，その他の幾何計算（交差領域の形を求めたり，包含判定など）はできない．

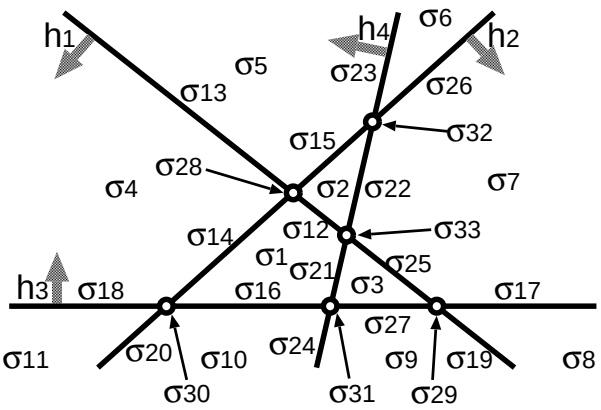

図 1　簡単な超平面アレンジメントの例．4 つの超平面 h_1, h_2, h_3, h_4 が，2 次元の空間を 33 個の HA-face σ_1 から σ_{33} に分割している．図中の矢印は，超平面の表側を示している

本研究では，超平面アレンジメントという概念を使いながら（つまり，超平面アレンジメントを，空間物の幾何計算に利用することのメリットはそのままにして），超平面アレンジメント構成アルゴリズムを利用する従来のアルゴリズムより格段に速いアルゴリズムを考案し，現実的な速さでの幾何計算を達成する．本研究のアルゴリズムは，本質的に任意の次元まで扱うことができる，つまり，3 次元以上の各種の幾何計算が可能であるという点で，線形計画法，整数計画法や，次元に特化したアルゴリズム（Half-space intersection 法など）を凌駕する．

また，超平面アレンジメントに基礎を置くモデルそのものが，本研究の特色である．*Hawk's Eye* では，いかなる種類，次元の空間データをも一様に表現できるようなモデルとして，超平面アレンジメントの区画の複体 (我々は HA-face complex と名付けている) を使ったモデルを実現している．このモデルは，種々の次元の空間物が混在するような空間データベースを可能とする．

本論文の構成は以下の通りである．2 章では，本研究で提案する HA-face complex モデルを説明する．3 章では，*Hawk's Eye* に実装された幾何計算アルゴリズムの説明を行う．4 章では，*Hawk's Eye* の実装において発生した種々の課題と解決策を説明する．5 章は，幾何計算アルゴリズムの性能評価である．

2. HA-face complex データモデル

2.1 超平面アレンジメント

超平面アレンジメントは，空間の分割を表現するために用いられてきた幾何構造である [2] [7] [12]．d 次元空間

$\mathbf{R^d}$ における d-1 次元の超平面の集合 $\mathbf{H}$ は，$\mathbf{R^d}$ を 0 から d 次元の区画に分割する．出来た区画は，互いに重なりあうことは無い．この分割のことを $\mathbf{H}$ のアレンジメント $\mathbf{A(H)}$（the hyperplane arrangement induced by $\mathbf{H}$）という（超平面アレンジメントの例を図 1 に示している）．出来たそれぞれの区画は face と呼ばれる．face という言葉は，これ以外の意味も持つことがあるから，混乱を防ぐために，以下，超平面アレンジメントの区画である face のことを，HA-face と呼ぶことにする．

HA-face の次元は種々ある．図 1 では，2 次元の HA-face が 11 個（3 個が有界で，8 個が非有界な領域），1 次元の HA-face が 16 個（8 個が直線で，8 個が半直線），0 次元の HA-face が 6 個ある．図 1 では，0 次元の HA-face は，超平面の交点であり，1 次元の HA-face は，超平面の部分になっており，2 次元の HA-face は，超平面で囲まれた領域の内部である．一般に，k 次元の HA-face σ は，σ を含むような k 次元の線形部分空間があり，かつ k-1 次元の線形部分空間が存在しないという性質がある．以下，k 次元の HA-face のことを k-HA-face と書く．

超平面アレンジメント $\mathbf{A(H)}$ は，$0 \leqq k \leqq d-1$ なる k に対して，k-HA-face と (k+1)-HA-face との間の全ての接続関係を蓄えた接続グラフ $\mathbf{G(H)}$ によって表現できる．接続グラフのノードは HA-face を表し，エッジは HA-face 間の接続関係を表す．

k 次元 ($k > 0$) の k-HA-face σ の境界 (boundary) は，σ とその外側との境目である凸図形であり，0 から k-1 次元の HA-face の集まりとして表現できる．但し，0-HA-face σ の境界は空であるとする．一方，HA-face σ の閉包（closure）は，σ と σ の境界の和集合であると定義する．

2.2 HA-face complex の提案

超平面アレンジメントの HA-face を貼り合わせることで，いかような図形であろうとも，その位置と形を表現することができる．このアイデアを使って，種々の次元の図形を，同一の形式で表現することができるようになる．本研究では，空間物を表現するモデルを，超平面アレンジメントを基礎に定義する．現実世界にある空間物は，その境界を含むことから，空間物を表現する HA-face 集合も，その境界部分を要素として含む．そこで，空間物を表現するモデルを，超平面アレンジメントの HA-face の複体（complex）として作る．つまり，k 次元の空間物を，0 から k までの次元の HA-face の複体として表現する．複体とは，本来，図形における辺と点の相対性と密接な関係がある数学的概念のことであるが，ここでは，複体という言葉を次の意味で使う．

- HA-face の集合 Γ について，Γ が，Γ の任意の要素 k-HA-face ($k > 0$) と接続関係にある全ての (k-1)-HA-face を含むとき，Γ のことを HA-face 複体（HA-face complex）と呼ぶ．

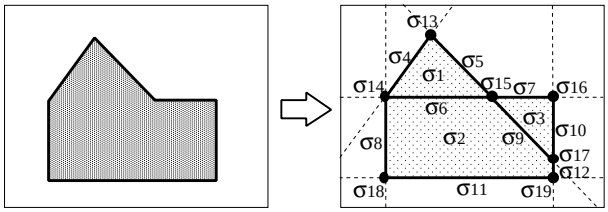

図 2　図形の空間属性（位置と広がり）を，HA-face complex として表現した例

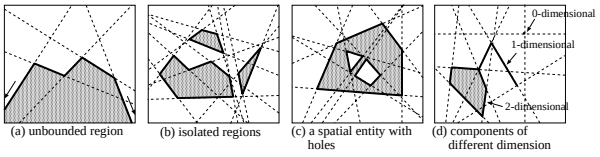

図 3　超平面アレンジメントの HA-face を貼りあわせることで (a) 非有界な領域, (b) 孤立領域, (c) 穴あり, (d) さまざまな次元をもった要素などの種々雑多な図形を同じ形式で表現することができる

図 2 には，HA-face complex の例を示している．図 2 では，$\Gamma = \{\sigma_1, \sigma_2, ..., \sigma_{19}\}$ は，6 つの超平面からなる超平面アレンジメントの HA-face の集合であるが，この集合 Γ は，上記に書いた複体の条件を満たす．例えば，σ_1 については，$\sigma_4, \sigma_5, \sigma_6$ は σ_1 と接続関係にあるが，これら $\sigma_4, \sigma_5, \sigma_6$ は，すべて Γ の要素である．図 3 に示したように，非有界な領域, 孤立領域, 穴，さまざまな次元の要素を含むような空間物も，HA-face complex として表現できる．

3. 幾何計算アルゴリズム

本来，空間データベースシステムは，空間データを格納，操作，検索するための特別な機能をもったデータベースシステムである [11]. 空間データに特有の検索処理には，点検索 (point query)，範囲検索（range queries）, 空間結合（spatial join）などがあり，その処理は，一般にフィルタリング（filtering）とリファインメント（refinement）の 2 段階で行われる．フィルタリングでは，索引上の近似形状のみを使って解候補を求める．リファインメントでは，解候補の本当の形状を使って本当の解かどうか調べる．つまり，フィルタリングは，モデルに独立であり，リファインメントはモデルに依存する．

HA-face complex を使った空間データベースシステムには，空間データの検索のためのリファインメントと，空間データの操作のための幾何計算のアルゴリズムが実装される必要がある．この章では，HA-face complex モデルで動く幾何計算のアルゴリズムの説明を行う．このアルゴリズムのアイデアは，種々の幾何計算で同じく使

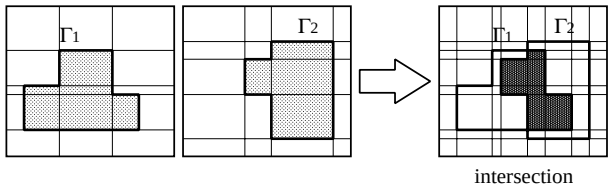

図 4　幾何計算の結果は，元の 2 つの図形に関する超平面集合からなる HA-face の集まりである

えるが，この章での説明は，最も基本的な幾何計算である「交差領域の形を求める処理 intersection」に限ることにする．実装の詳細や，他のオペレーションとの差異については，4 章で説明を行う．

2 つの空間物の交差領域の形を求めるには，2 つの空間物の形の情報から，いったん超平面アレンジメントを構成し，それを使って，交差領域の形を導き出すというやり方がある．以下，2 つの HA-face complex Γ_1, Γ_2 の intersection(Γ_1, Γ_2) について説明する．Γ_1, Γ_2 は HA-face complex であり，それぞれ，2 つの超平面アレンジメント $\mathbf{A}(\mathbf{H_1})$, $\mathbf{A}(\mathbf{H_2})$ の HA-face からなる複体であるとする．すると，intersection(Γ_1, Γ_2) は，超平面集合 $\mathbf{H_1} \cup \mathbf{H_2}$ の超平面アレンジメント $\mathbf{A}(\mathbf{H_1} \cup \mathbf{H_2})$ の HA-face からなる複体として表現できる（このことを図 4 に示している）．以上のことから，最初，$\mathbf{H_1} \cup \mathbf{H_2}$ から超平面アレンジメント $\mathbf{A}(\mathbf{H_1} \cup \mathbf{H_2})$ を構成し（この段階では，既存の超平面アレンジメント構成アルゴリズムを利用），その後，$\mathbf{A}(\mathbf{H_1} \cup \mathbf{H_2})$ の HA-face のそれぞれについて，Γ_1, Γ_2 の両方に含まれているかを判定することで，intersection(Γ_1, Γ_2) を求めることができる．これは，Γ_1, Γ_2 の次元に特化しない方式である．

超平面アレンジメント構成アルゴリズムには，逐次生成法（incremental）や divide-and-conquer（分割統治法）に分かれるが，いずれも，超平面集合を入力として，超平面アレンジメントを構成する．超平面アレンジメントの複雑さは，その HA-face 数で測るが，ゾーン定理 [5] から，$\mathbf{R^d}$ における n 個の超平面集合 $\mathbf{H}$ の任意の超平面アレンジメント $\mathbf{A(H)}$ の複雑さは $\mathbf{O(n^d)}$ となる [4] [7]. 従って，$\mathbf{A}(\mathbf{H_1} \cup \mathbf{H_2})$ の HA-face 数が大きすぎるために，超平面アレンジメント構成アルゴリズムを用いての intersection(Γ_1, Γ_2) の算出は実用的な性能が得られない．

超平面アレンジメントの HA-face うち，Γ_1, Γ_2 の外部にあるものは，幾何計算の結果に影響しないため，その部分の計算を省略できる．この高速化のアイデアを実現したものが localized divide-and-conquer である．localized divide-and-conquer の概略は次の通りである．

1. Γ_1 の各 HA-face の閉包を表現する cell lattice の構築

intersection(Γ_1, Γ_2) を求めるために，まず，Γ_1 の HA-face σ_i のそれぞれについて，σ_i の閉包（これは HA-face 集合）を表現する lattice 構造を中間データとして作る．このデータ構造を，以下，cell lattice と呼ぶ．最初，σ_i と交差しているか接しているような $\mathbf{H_1}$ の超平面から，法線ベクトルが $\mathbf{R^d}$ 空間を張るように d 個の超平面を取り出す．これら d 個の超平面から作られる接続グラフは，d 次元立方体と同形であり，直接的な方法で作ることができる．残りの超平面があれば，出来た接続グラフに添加する．この結果，σ_i の閉包を表現する cell lattice が構築される．

2. cell lattice の $\mathbf{H_2}$ による分割

前のステップで出来た cell lattice のそれぞれを，超平面集合 $\mathbf{H_2}$ で分割する．つまり，cell lattice の HA-face が，超平面集合 $\mathbf{H_2}$ によって，より小さな部分に分割される．この結果，Γ_1 の各 HA-face の閉包が，$\mathbf{H_2}$ をすべて含むような超平面集合のアレンジメントの HA-face の部分集合として表現されることになる．これは，$\mathbf{H_2}$ の超平面を 1 つずつ逐次的に，それまでに作られている cell lattice に加えていくという方法で作ることができる．現在の cell lattice から，添加される超平面 h と交差する HA-face e を見つける．その後 e から出発して，h と交差する全ての HA-face にマークを付け，適切な更新を行う．こうして，cell lattice に h が挿入できる．

3. 処理結果の併合

Γ_1 の全ての HA-face について cell lattice の分割を終えたら，すべての cell lattice をマージする．これは，元の Γ_1 を，$\mathbf{A}(\mathbf{H_1} \cup \mathbf{H_2})$ の HA-face からなる複体として表現したものになっている．

4. intersection(Γ_1, Γ_2) の導出

この複体の HA-face のそれぞれについて，Γ_2 に含まれているかを判定して，HA-face 集合を得る．こうして，intersection(Γ_1, Γ_2) を求めることができる．

以上のように，localized divide-and-conquer の入力は，2 つの HA-face complex である．上記のステップ 3 の時点で，Γ_1 を $A(\mathbf{H_1} \cup \mathbf{H_2})$ の HA-face からなる複体として表現したものが得られる．ステップ 4 は，処理すべき幾何演算の種類によって，処理の種類が変わってくる．いずれにしても，アレンジメント構成法を使った場合のように $\mathbf{A}(\mathbf{H_1} \cup \mathbf{H_2})$ のすべての HA-face を使うのでない（Γ_1 に含まれる HA-face のみを使う）ので，計算時間の削減ができる．

上記のステップ 1, 2 では，HA-face についての処理を繰り返すが，アルゴリズムの性質から，より次元の高い他の HA-face と接続しているような HA-face は処理の対象から外すことができる．以下，HA-face complex Γ

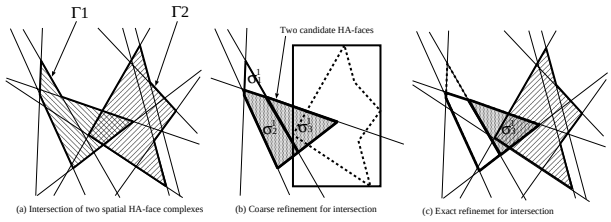

図 5　積 intersection(Γ_1, Γ_2) の two step refinement．Coarse Refinement では，一方の図形の最小包含矩形と，もう 1 方の図形の「輪郭」を用いて処理を行う．このことで，Exact Refinement において処理を行うべき部分を絞り込む

の k-HA-face ($0 \leqq k \leqq k$) で，他の (k+1)-HA-face に接続していないものを，Γ の独立 HA-face と呼ぶ．例えば，図 2 内の図形は，19 個の HA-face の集合であったが，この独立 HA-face は，$\{ \sigma_1, \sigma_2, \sigma_3 \}$ の 3 個である．上記のステップ 1, 2 では，Γ_1 の独立 HA-face についての処理のみを繰り返す．

4. 実　　装

4.1　2 ステップリファインメント

近似形状をうまく使うことにより，HA-face complex の幾何計算をさらに高速化する手法である 2 ステップリファインメントを提案する．2 ステップリファインメントは，HA-face complex の近似形状を使った Coarse refinement と，HA-face complex の本当の形状を使った Exact refinement に分かれる．Coarse refinement は，2 つの HA-face complex Γ_1, Γ_2 から，Γ_2 の MBR(最小包含矩形; minimum bounding box) と交差するか接している Γ_1 の独立 HA-face を全て求める手続きであり，Exact refinement は，求めた独立 HA-face 集合について localized divide-and-conquer を実行し，実際の幾何計算を行う手続きである．つまり，Coarse refinement は，Γ_1 の独立 HA-face のうち Γ_2 の MBR と交差しないものを取り除く．有界な 2 つの HA-face complex に体する Coarse refinement の手順は次の通りである．

（1）　順序の決定

入力の 2 つの HA-face complex の大きさを調べて，順序を付ける．HA-face complex の大きさは，その独立 HA-face の数で数える．以下，入力の 2 つの HA-face complex のうち，独立 HA-face の数の少ない方を Γ_1，多い方を Γ_2 とする．

（2）　Γ_1 の各独立 HA-face と Γ_2 の MBR との判定の繰り返し

Γ_1 の全ての独立 HA-face について，Γ_2 の MBR と交差するか接しているものを探し出力する．1 次元以上の独立 k-HA-face σ については，その境界を構成する HA-face (次元は 0 から k-1 まで) について，0 次元のものから開始して，Γ_2 の MBR との関係を調べる．1 つでも交差するか接するものがあれば，σ を出力する．つまり，独立 k-HA-face σ の境界を構成する種々の次元の HA-face を段階的に使う．0 次元の k-HA-faec については，その座標値と Γ_2 の MBR で判定を行う．

2 ステップリファインメントで，intersection(Γ_1, Γ_2) を求める例を図 5 に示している．入力となる 2 つの図形の独立 HA-face 集合は，それぞれ，$\{\sigma_1^1, \sigma_2^1, \sigma_3^1\}$ と $\{\sigma_1^2, \sigma_2^2, \sigma_3^2, \sigma_4^2, \sigma_5^2\}$ である．intersection(Γ_1, Γ_2) を求めるために（図 5(a) を参照), coarse refinement は，3 つの独立 HA-face $\{\sigma_1^1, \sigma_2^1, \sigma_3^1\}$ (図 5(b) を参照) のそれぞれについて，Γ_2 を包含するような MBR と交差する（あるいは接する）かどうかを調べる．この例では，σ_1^1 と σ_2^1 が交差する．これらが，coarse refinement で得られる出力である．次の exact refinement では，ペア（σ_1^1 ，Γ_2）と，別のペア （σ_2^1，Γ_2）について，交差するかどうかが精査される．この精査は，積 intersection(Γ_1, Γ_2) を求めるのに必要である．交差するペアは，σ_2^1，Γ_2 の方のみである（図 5(c) 参照）．

以上のように，Coarese refinement では，一方の空間物については近似形状である MBR を使い，もう一方の空間物については正確な形状を使う．これが Coarse refinement の名前の由来である．Coarse refinement は，Exact refinement で調べるべき独立 HA-face の絞り込みを行うが，2 つの空間物の重なり度が小さいほど，Coarse refinement での絞り込みが効果的に働く．

上記の手続きは，基本的には，非有界な空間物についても動くが，非有界な空間物について，MBR に相当する近似形状を導入することは，我々にとっての今後の課題である．現在は，次のような簡便な実装を行っている．

- 演算すべき空間物のうち一方だけが非有界であれば，有界な方を Γ_2 として上記の手続きを実行する．Γ_1 は，非有界な HA-face を含むが，非有界な HA-face にも境界が定義されているので，問題なく動く．
- 2 つとも非有界であれば，Coarse-Refinement は行わない．

4.2　種々の幾何計算

幾何計算の種類としては，図 6, 7 の例に示したように，集合論演算子 (Intersection, Difference, Union)，位相述語 (intersect, disjoint, contain, equal, meet) が考えられる．集合論演算子は，2 つの空間物から新しい空間物を作る．位相述語は，2 つの空間物の空間的な関係を判定し true あるいは false 値を返す述語である．この節では，これら種々の幾何計算について，アルゴリズム上どのような差異が生じるかを説明する．以下，HA-face complex Γ_1, Γ_2 は HA-face complex であり，それぞれ，2 つの超

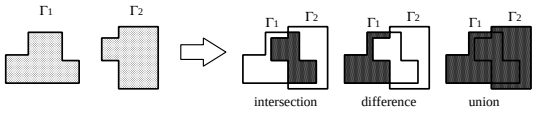

図 6　集合論演算子

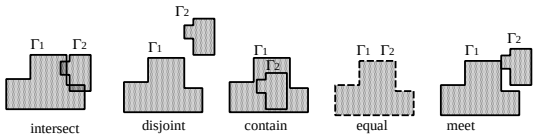

図 7　位相述語

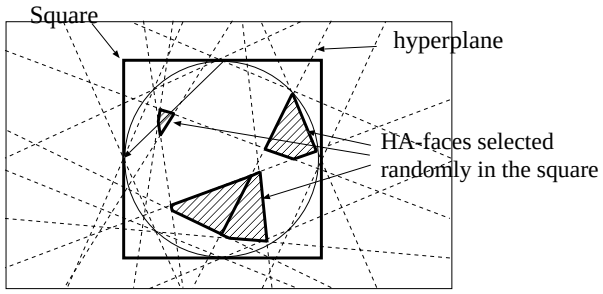

図 8　正方形に重なるような超平面をランダムに生成する

平面アレンジメント $\mathbf{A}(\mathbf{H_1})$, $\mathbf{A}(\mathbf{H_2})$ の HA-face からなる複体であるとする．

集合論演算子

○ **intersection**

intersection(Γ_1, Γ_2) は，超平面集合 $\mathbf{H_1} \cup \mathbf{H_2}$ の超平面アレンジメント $\mathbf{A}(\mathbf{H_1} \cup \mathbf{H_2})$ の HA-face のうち，Γ_1 と Γ_2 の両方に含まれているものの集合である．これは，今まで説明したように，Coarse refinement により，Γ_2 の MBR と交差するか接している Γ_1 の独立 HA-face を全て求めた後に，localized divide-and-conquer により，$\mathbf{H_2}$ を添加し，出来た HA-face 集合から Γ_2 に含まれているものを全て選ぶという手順で行う．

○ **union**

union(Γ_1, Γ_2) は，$\mathbf{A}(\mathbf{H_1} \cup \mathbf{H_2})$ の HA-face のうち，Γ_1 と Γ_2 のどちらか一方に含まれているものの集合である．union では，Coarse refinement は行わない．localized divide-and-conquer を 2 回行って，Γ_1 と Γ_2 を $\mathbf{A}(\mathbf{H_1} \cup \mathbf{H_2})$ の HA-face からなる複体として表現したものを得て，最後にこの 2 つをマージする．

○ **difference**

difference(Γ_1, Γ_2) は，$\mathbf{A}(\mathbf{H_1} \cup \mathbf{H_2})$ の HA-face のうち，Γ_1 に含まれていて，Γ_2 の内部に含まれていないものの集合である．例えば，図 6 では，difference(Γ_1, Γ_2) は交線を含む．これは，Coarse refinement により，Γ_2 の MBR と交差するか接している Γ_1 の独立 HA-face を全て求めた後に，localized divide-and-conquer により，$\mathbf{H_2}$ を添加し，出来た HA-face 集合から Γ_2 の内部に含まれていないものを全て選ぶという手順で行う．

位相関係述語

○ **intersect**

intersect(Γ_1, Γ_2)」は，Γ_1, Γ_2 の積 *intersection*(Γ_1, Γ_2) が空で無いときに限り true 値を返す．

○ **disjoint**

disjoint(Γ_1, Γ_2)」は，Γ_1, Γ_2 の積 *intersection*(Γ_1, Γ_2) が空であるときに限り true 値を返す．

○ **contain**

contain(Γ_1, Γ_2)」は，差 difference(Γ_2, Γ_1) が空であるときに限り true 値を返す．

○ **meet**

meet(Γ_1, Γ_2)」は，他の位相関係述語を使って，次のように定義する．

$$meet(\Gamma_1, \Gamma_2) = contain(boundary(\Gamma_1), intersection(\Gamma_1, \Gamma_2))$$

ここで，boundary(Γ_1) は，Γ_1 の外部との境界を表現する HA-face complex である．

○ **meet**

equal(Γ_1, Γ_2) は，2 つの HA-face complex Γ_1 と Γ_2 を $\mathbf{A}(\mathbf{H_1} \cup \mathbf{H_2})$ の HA-face からなる複体として表現したときに，全く同じ集合になるときに，true 値を返す．これは，localized divide-and-conquer を 2 回実行することで得られる．

5. 性 能 評 価

本章では，性能評価を行い，Localized divide-and-conquer の効果と Coarse refinment の効果を確認する．性能評価実験は SUN Microsystems Ultra 10 ワークステーション (CPU: UltraSPARC-IIi 440MHz, 実メモリ: 1024 Mbytes, OS: Solaris 8) を使って実施した．プログラムは，C and C++ (C/C++ コンパイラ: SUN Forte 6 Developer Update 2）で書き, データベースシステムとして出世魚 [14] を用いた．

テスト用のデータとして T_N^A, T_N^B, T_N^C ($N = 50, 100, 200, 300, 400, 500$) と名付けた 3 種類の 2 次元図形データを，ランダムに生成した．手順は次の通り．

（1） 正方形 S_A, S_B, S_C を次のように決める．

- S_A: [(0, 0), (1, 1)]
- S_B: [(0.2, 0.2), (1.2, 1.2)]
- S_C: [(0.99, 0), (1.99, 1)]

（2） S_A, S_B, S_C に重なるような，50 個の超平面からなる超平面集合 $\mathbf{H_N^A}$, $\mathbf{H_N^B}$, $\mathbf{H_N^C}$ をランダムに生成する．

（3） こうして出来た超平面集合 $\mathbf{H_N^A}$, $\mathbf{H_N^B}$, $\mathbf{H_N^C}$ について，独立 HA-face をランダムに選んで（図 8），テストデータを作る．選ぶ独立 HA-face の数 N は，

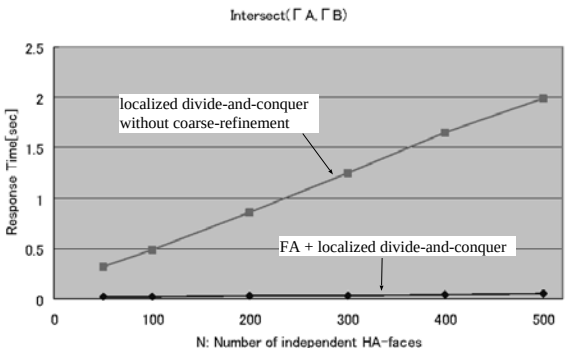

図 9　intersect(Γ_N^A, Γ_N^B): Coarse refinement 有り無し

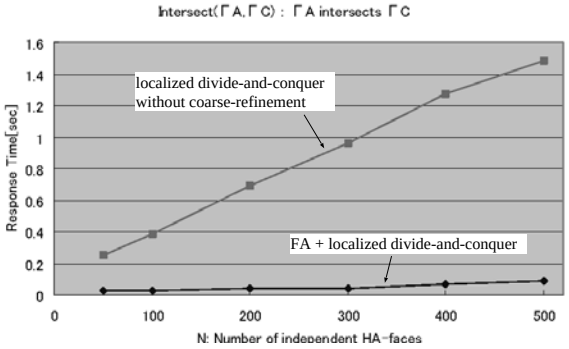

図 10　intersect(Γ_N^A, Γ_N^C), Γ^A intersects Γ^C: Coarse refinement 有り無し

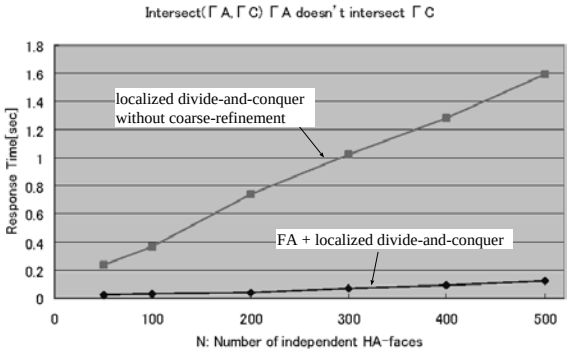

図 11　intersect(Γ_N^A, Γ_N^C), Γ^A does not intersect Γ^C: Coarse refinement 有り無し

$N = 50, 100, 200, 300, 400, 500$ の 6 通りである．

（4） 以上の手順を 5 回繰り返して，同じ性質を持った図形を 5 個得る．

以上の手順で得られたテストデータ T_N^A, T_N^B, T_N^C ($N = 50, 100, 200, 300, 400, 500$) について，intersect($\Gamma_N^A$, Γ_N^B) と，intersect(Γ_N^A, Γ_N^C) を実行し，処理にかかった時間の平均を取って，最終的な実験結果を得た．比較結果がはっきりと現れるように，実験は，テストデータがすべて実メモリに載った状態で行った（つまり ディスクとの I/O コストは，測定結果に現れないような実験を実施した）．実験項目は次の 2 つ．

（1） coarse refinement 有り無しの効果．

coarse refinement を行う場合と，行わない場合で比較を行った図 9 が，積 intersect(Γ_N^A, Γ_N^B) ($N = 50, 100, 200, 300, 400, 500$) の結果，図 10 と図 11 が積 intersect(Γ_N^A, Γ_N^C) ($N = 50, 100, 200, 300, 400, 500$) の結果である．coarse refinement を行う方が，無い場合に比べて 10 から 40 倍速い．

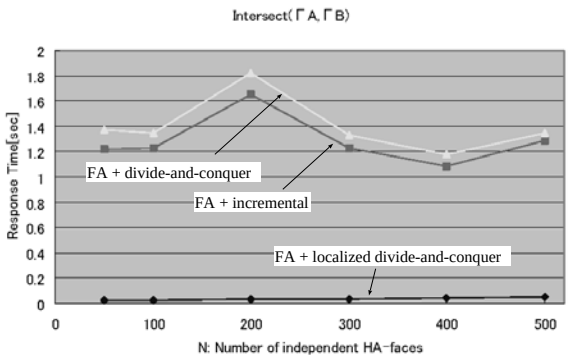

図 12　intersect(Γ_N^A, Γ_N^B): localized divide-and-conquer，逐次添加法，分割統治法の比較

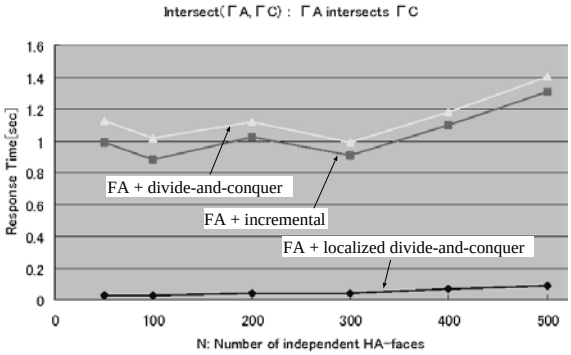

図 13　intersect(Γ_N^A, Γ_N^C), Γ^A intersects Γ^C: localized divide-and-conquer，逐次添加法，分割統治法の比較

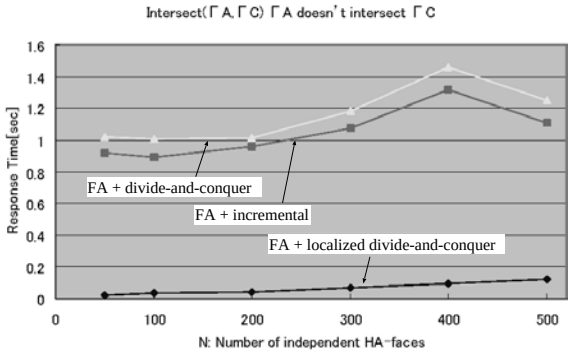

図 14　intersect(Γ_N^A, Γ_N^C), Γ^A does not intersect Γ^C: localized divide-and-conquer，逐次添加法，分割統治法の比較

（2） localized divide-and-conquer の評価

localized divide-and-conquer と，アレンジメント構築アルゴリズムである逐次添加法と分割統治法で比較を行ったつまり，localized divide-and-conquer，逐次添加法，分割統治法の 3 つの比較である．図 12 は積 intersect(Γ_N^A, Γ_N^B) の結果であり，図 13 と図 14 は積 intersect(Γ_N^A, Γ_N^C) の結果である．(図中，localized divide-and-conquer は localized divide-and-conquer，逐次添加法は incremental，分割統治法は divide-and-conquer と書いている.)．local-

ized divide-and-conquer は，他の方法と比べて 3 から 27 倍速い．

以上の実験から，2 次元データの場合について，localized divide-and-conquer, Coarse refinement の効果を確認することができた．

6. おわりに

我々は，空間データベース構築のための基盤システム *Hawk's Eye* を研究・開発してきた．Hawk's Eye には，各種の多次元幾何計算アルゴリズムが実装されており，それを使って，0 から 4 次元までのさまざまな空間データが混在するような「可変次元時空間データベース」が高速に処理できる．いかなる種類，次元の空間データをも一様に表現できるようなモデルとして超平面アレンジメントの区画の複体 (我々は HA-face complex と名付けている) を使ったモデルを実現していることが本研究の特色である．このモデルは，種々の次元の空間物が混在するような空間データベースを可能とする．本研究では，超平面アレンジメントという概念を使いながら，超平面アレンジメント構成アルゴリズムを利用する従来のアルゴリズムより格段に速く幾何計算を行うことができる次元に独立なアルゴリズム localized divide-and-conquer を考案した．このアルゴリズムは，本質的に任意の次元まで扱うことができる，今回提案の localized divide-and-conquer は，HA-face complex モデルに限らずに，原理的には，1 つの図形を，複数の凸な図形の集まりとして扱っているようなケースに広く適用できると考えており，このことは今後の研究課題である．

文　献

[1] M. de Berg, M. van Kreveld, M. Overmars, and O. Schwarzkopf, Computational Geometry: Algorithms and Applications second edition, Springer-Verlag, Berlin, 2000.

[2] David P. Dobkin, and Ayellet Tal, Efficient and Small Representation of Line Arrangements with Applications, ACM Symposium on Computational Geometry, pp.293–301, 2001

[3] H. Edelsbrunner and J.O'Rourke and R. Seidel, Constructing arrangements of lines and hyperplanes with applications, SIAM J. Comput. vol.15, pp.341-363. 1986.

[4] Herbert Edelsbrunner, Algorithms in Combinatorial Geometry", Springer-Verlag, 1987.

[5] Edelsbrunner, H., R. Seidel and M. Sharir, On the zone theorem for hyperplane arrangements, SIAM J. Comput. Vol.22, pp.418-429, 1993.

[6] Luca Forlizzi, Ralf Hartmut Guting, Enrico Nardelli, and Markus Schneider, A Data Model and Data Structures for Moving Objects Database, ACM SIGMOD 2000.

[7] J. E. Goodman and J. O'Rourke, editors. Handbook of Discrete and Computational Geometry, CRC Press LLC, Boca Raton, FL, 1997.

[8] S.Grumbach, P.Rigaux, M.Scholl, L.Segoufin, DEDALE, A Spatial Constraint Database. DBPL 1997, pp.38-59, 1997.

[9] S.Grumback, P.Rigaux, L.Segoufin, The DEDALE System for Complex Spatial Queries, Proc. ACM SIGMOD Intl. Conf. On.Management of Data, pp. 213-224, 1998.

[10] Grünmaum, B., Convex Polytopes, Wiley-Interscience, New York, 1967.

[11] R.H.Güting, An Introduction to Spatial Database Systems, The VLDB Journal, vol. 3, no. 4, pp. 357–399, 1994

[12] Handbook of Computational Geometry, Elsevier Science Publishers B.V. North-Holland, Amsterdam, 2000

[13] Agnes Voisard, Benoit David, A Database Perspective on Geospatial Data Modeling, IEEE Transactions on Knowledge and Data Engineering, vol.14, no.2, pp.226-243, 2002

[14] G.Yu, K.Kaneko, G.Bai, A.Makinouchi, Transaction Management for a Distributed Object Storage System WAKASHI – Design, Implementation and Performance, 12th Int'l Conf. on Data Engineering, pp.460-468, 1996.