2005－DBS－137（Ⅰ）（10）
2005／7／13

Extended Cell Splitting Algorithm の効果的な実装

田中　美智子† 　金子　邦彦‡ 　陸　応亮† 　村下　純也† 　牧之内　顕文‡

†九州大学大学院システム情報科学府　〒812-8581 福岡市東区箱崎 6-10-1
‡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: †{tanaka, riku, murasita}@db.is.kyushu-u.ac.jp, ‡{kaneko, akifumi}@is.kyushu-u.ac.jp

あらまし　任意の次元において，有界，非有界 cell を効果的に分割するアルゴリズムを提案する．このアルゴリズムでは，まず始めに１次元 cell を１つずつ分割していく．この時，１次元 cell は，有界，非有界なものを両方含む．その後，１次元以上の cell について分割処理を行っていく．本論文では，このアルゴリズムの性能評価についても示す．

キーワード　cell complex，　cell splitting，　boolean set operation

Efficient Implementation of Extended Cell Splitting Algorithm

Michiko TANAKA† 　Kunihiko KANEKO‡ 　Yingliang LU† 　Junya MURASHITA†
and 　Akifumi MAKINOUCHI‡

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

Abstract　We present an extended cell splitting algorithm which can split both bounded and unbounded cells efficiently in any dimension. The algorithm splits one-dimensional cell, either bounded or unbounded first one by one. Then the algorithm splits cells whose dimensions are more than one. We present an evaluation of the algorithm, also.

Keyword　cell complex，　cell splitting，　boolean set operation

1. はじめに

GIS(geometric information system) や CAD(computer aided design)，線形制約データベースなど，様々な分野で空間物を扱うことが必要とされている．我々は HawksEye という空間データベースシステムを開発してきた．このシステムでは，空間物は cell の集合として表現される．cell とは線形式を利用して定義される凸な領域である(図３)．詳しい定義は３章で述べる．cell に基礎をおく表現により，任意の次元の空間物を一様に表現することが可能になる．

HawksEye において，幾何計算や boolean set operation を実現する為に，cell を超平面で分割することは重要な問題である．Chandrajit らは cell splitting algorithm を提案している[1]．このアルゴリズムは，有界な cell の分割を行うことができる．Nierenstein らは，from-region visibility query において，Chandrajit らのアルゴリズムを利用している．彼らは，二つの cell に空間幾何演算 difference を適用している．

我々は，cell 上に仮想の点を置き，その position を求めることで，非有界 cell の分割判定を行うよう，[1]のアルゴリズムを拡張して Extended Cell Splitting Algorithm を実装した．このアルゴリズムは，有界，非有界 cell をどちらも分割できる．本論文においては，本質的に必要な 1-cell の分割法を詳述する．

また，任意次元の cell について提案アルゴリズムを適用した際，分割の結果を incidence graph として出力するよう実装した．

本論文において，これらのアルゴリズムを示すと共に，性能テストの結果を報告する．

2. 関連研究

Candrajit らは，incidence graph で表現された有界な cell を超平面で二つに分割するアルゴリズムを提案した[1]．このアルゴリズムは任意次元空間中の任意の　次元の cell を効率良く分割することができる．このアルゴリズムは weak cell complex から cell complex への変換を行う為

```
algorithm ExtendedCellSplit(G, φ, d)
input:
G: the incidence graph of a cell
φ: the equation of a hyperplane h to split with
d: the dimension of space
output:
incidence graph split by a hyperplane

1. calculate either ＋, －, or 0 for each
   0-cell in G using φ
2. for each 1-cell σ in G
3.    if σ does not intersect with h
4.       then compute the position of σ
           according to h
5.       else split_node()
6. for each 2-cell σ in G
7.    if σ does not intersect with h
8.       then compute the position of σ
           according to h
9.       else split_node()
10.for j ← 3 to dim(G)
11.   for each j-cell σ in G
12.      if σ does not intersect with h
13.         then compute the position of σ
              according
              to h
14.         else split_node()
```

図１：アルゴリズム extended cell splitting algorithm

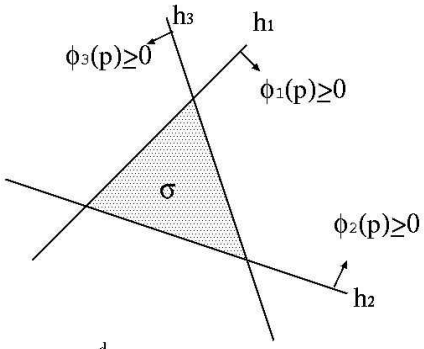
```
algorithm split_node(G', σ, j)
input:
G': the halfway graph of a cell
σ: the cell in G which is split by h
j: the dimension of s
output:
the halfway graph which contain two
split cells of σ

1. make new j-cells σ+ and σ− in G'
2. connect σ+ and σ− with the (j+1)-cell
   whose boundary cell is σ
3. for each boundary (j-1)-cell σ' of σ
4.    if the position of σ' according to h is
        ＋(－)
5.       then connect σ' with σ+(σ−)
           by an arc in G'
6.          make new (j-1)-cell σ0 in G' and
              set the position according to h as 0
7.          connect σ0 with σ+ and σ−
              by an arc
8.          if j>1
9.             for each boundary (j-2)-cell σ2 of
σ
10.               if σ2 has the position 0 to h
11.                  then connect σ0 and σ2
                       by an arc in G'
12.remove σ from G'
```

図２：アルゴリズム split_node

に導入された．これらの complex は，どちらも有界な cell の集合である．この為，このアルゴリズムによって分割されるのは有界 cell のみである．また，このアルゴリズムを３次元以上の cell に対して適用すると，出力されるグラフは incidence graph とは異なる構造になる．

d 次元空間中の BSPtree は，その葉ノードが



$\sigma=\{p\in E^d \mid \phi_1(p)\geq 0 \wedge \phi_2(p)\geq 0 \wedge \phi_3(p)\geq 0\}$

図３：2-cell の線形制約による定義例

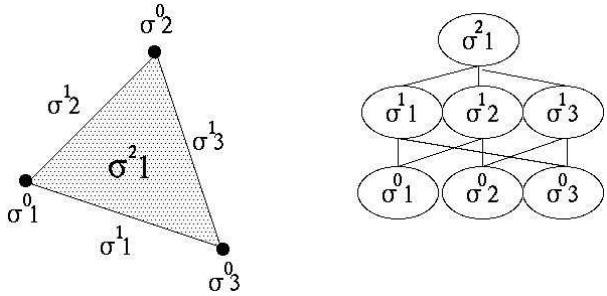


図４：cell とその incidence graph G(σ^2_1)

d-cell を表現する．[7]で提案されているアルゴリズム Partition_Bspt は，BSPtree と超平面を入力とし，超平面によって分割された BSPtree を出力するアルゴリズムである．これにより d-cell の分割が計算できる．[7]においては，3 次元空間でのアルゴリズムが示されている．3 次元空間中の BSPtree において，節は葉ノードが示す 3-cell を隔てる 2 次元多角形である．この多角形を,頂点のリストとして保持している．また，空間を十分大きな box で囲むことで，非有界な cell を含まないようにしている．

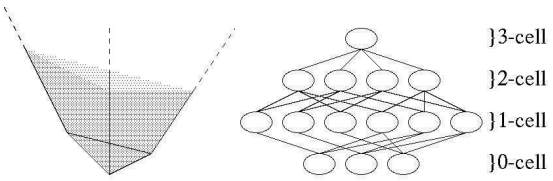
3. cell と incidence graph

この章では，cell と incidence graph についての説明をする．

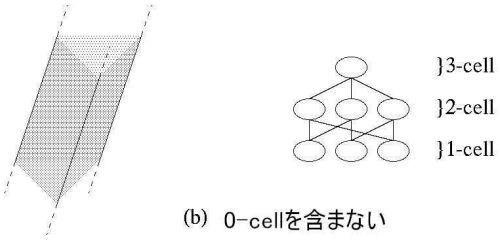
d 次元ユークリッド空間を E^d で表現し，空間物はこの中に存在するものとする．本論文では，cell を σ で表現し，その incidence graph を G(σ) と表記する．E^d 中の任意の点を $p=(x_1, x_2, \ldots, x_d)$ とする．E^d 中の(d-1)次元部分空間である超平面 h_j は d+1 個の係数 $\alpha_{ji}(1\leqq i\leqq d+1)$ により線形式 $\phi_j(p)=\alpha_{j1}x_1+\alpha_{j2}x_2+\ldots+\alpha_{jd}x_d+\alpha_{j(d+1)}$ を定めた時，$\phi_j(p)=0$ で定義される．op_j を比較演算子≦，≧， or =とする時，"$\phi_j(p)$ op_j 0"は，線形制約を表現する．また H(σ)を，σ を構成する超平面集合とする．

```
algorithm is1-CellIntersect(n,
                {σ⁰₁,..,σ⁰ₙ}, hⱼ)
input:
n: the number of 0-cell connect with  σ¹₁
{  σ⁰₁ ,..,σ⁰ₙ}: the 0-cells connect with  σ¹₁
h: hyperplane to split  σ¹₁
output:
true or false
1. if n = 2
2.    then if there aren't both  σ⁰ᵢ which
       position according to h is + and−
3.       then return false
4.       else return true
5.    else if n = 1
6.       then if a vector αⱼ=( αⱼ₁,  αⱼ₂, ...,  αⱼd)
            and a vector v along  σ¹₁ satisfy
              αⱼ・v > 0 and  σ⁰₁ 's position is−or
              αⱼ・v < 0 and  σ⁰₁ 's position is＋or
7.          then return true
8.          else return false
9.       else
10.       if  αⱼ・v = 0
11.         then return true
12.         else return false
```

図５：1cell σ^1_1 が超平面 h_j と交差するかどうかの判定



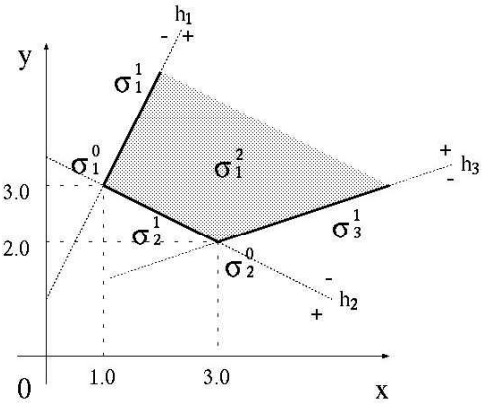
(a) 0−cellの数は3



(b) 0−cellを含まない

図６：非有界 cell の例

cell σ は異なるいくつかの線形制約を満足する点の集合として表現される(図３)．σ を表現する n 個の線形制約を $\phi_1(p)$ op_1 0∧ $\phi_2(p)$ op_2 0 ∧... ∧ $\phi_n(p)$ op_n 0 とし，$\phi_j(p)$=0 ($1 \leqq j \leqq n$) で与えられる超平面を h_j とすると $H(\sigma)=\{h_1, h_2, \ldots, h_n\}$ となる．cell σ を定義する超平面が m 個の異なる方程式 $\phi_j(p)=0$ である場合，σ の次元 dim(σ)は d-m である．本論文では，k 次元 cell を k-cell と表現し σ^k と表記する．



$$\sigma^2_1=\{p\in E^d \mid \phi_1(p)\geq 0 \wedge \phi_2(p)\leq 0 \wedge \phi_3(p)\geq 0\}$$

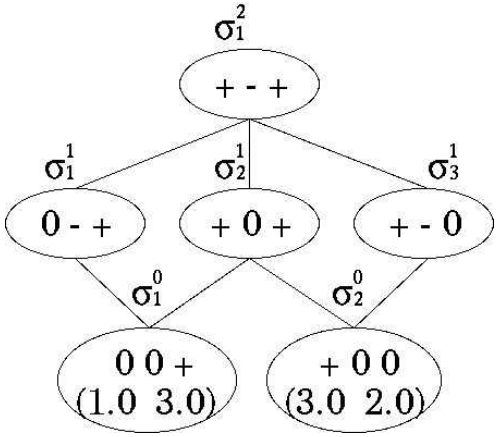


図７：incidence graph の実装

cell には，有界なものと，非有界なものがある．cell σ の中に任意の二つの点をとった時，それらの距離より大きい実数 r が存在するとき，σ は有界である．そうでなければ σ は非有界である(図６)．

k-cell σ^k の incidence graph $G(\sigma^k)$ は，σ^k を構成する k 次元以下（σ^k を含む）の cell の接続関係を表現するものである．cell は節で表現され，s-cell と(s-1)-cell ($1 \leqq s \leqq k$)の接続関係は枝で表現される．図４に例を示す．この三角形は１つの 2-cell σ^2_1，３つの 1-cell σ^1_1 σ^1_2 σ^1_3，３つの 0-cell $\sigma^0_1\sigma^0_2\sigma^0_3$ で構成されている．2-cell σ^2_1 は３つの 1-cell σ^1_1 σ^1_2 σ^1_3 と接続している．その為，それらを結ぶ枝が存在する．

1 つの超平面 $\phi_j(p_i)=0$ と 1 つの点 $p_i=(x_{i1}, x_{i2}, \ldots, x_{id})$ が $\phi_j(p_i)>(<$ or $=)0$ を満たすとき，点 p_i の position は＋(−or0)である．

図７に incidence graph の実装を示す．0-cell σ^0_1 は，h_1, h_2, h_3 それぞれに対して position 0, 0, +を持ち，座標が(1．0，3．0)であるので，これらの情報を節に保存している．k-cell($k \geqq 1$)

については，超平面集合 H(σ)に対する position を格納する．

4. cell の分割

この章においては，まず，分割により新たに G(σ)中に生成された節に対して，枝のつなぎ直しを行うアルゴリズムについて説明する．これは，分割結果を incidence graph として出力する為のものである．次に，cell splitting algorithm [1] が，非有界な cell の分割に適用できない理由を示し，extended cell splitting algorithm の基本となる考えを説明する．

非有界な cell の場合，その incidence graph は 0-cell を含む場合と含まない場合に分かれる．0-cell を含まないのは，その cell が半空間を表すという特別なケースであり，一般的には 0-cell を含むと考えてよい．我々は，その cell が含む 0-cell σ^0 の位置ベクトルの値と，σ^0 に接している 1-cell の方向ベクトル及び h の法線ベクトルとの内積を取ることで，単に，σの線形多項式と h の線形方程式を解くよりも，効率良く交差判定ができることを見出し，実装した．

4. 1 グラフの操作

超平面 h による cell σ の分割は，σ の incidence graph G(σ)を入力とする．分割の結果，新しい節が incidence graph にできる．G(σ)を構成する種々の節のうち，h と交差するような 1 次元以上の cell を表現する節は 2 つに分割され，結果として新しい 3 つの節ができる(元の節は削除される)．例えばある 2 次元 cell に h が交差しているようなときは，分割の結果，2 次元の cell が新しく 2 つできるとともに，この cell と h の積部分を表す 1 次元の cell ができる．この処理は，cell σと超平面 h との交差判定，それらが交差する場合は σ_- に関する新しい節の生成とσの削除，それに伴う枝のつなぎなおしという 3 手順で行われる．この処理は，有界・非有界どちらの cell に対しても同様である．

σが交差する場合のσに関する新しい節の生成とσの削除は，単に dim(σ) 次元の cell を 2 個，dim(σ)-1 次元の cell を 1 個作るに過ぎないので，実装は容易である．拡張 cell splitting アルゴリズムでは，incidence graph 内の cell σを超平面 h で分割するとき，次の手順を踏む．

(1) σを dim(σ) 次元の 2 つの cell σ_+ と σ_- に分割し，σと接続する全ての dim(σ)次元の cell の節と枝でつなぐ．また，σに接続する dim(σ)-1 次元の cell 全てについて，その position により，σ_+ と σ_- のどちらかと枝でつなぐ．

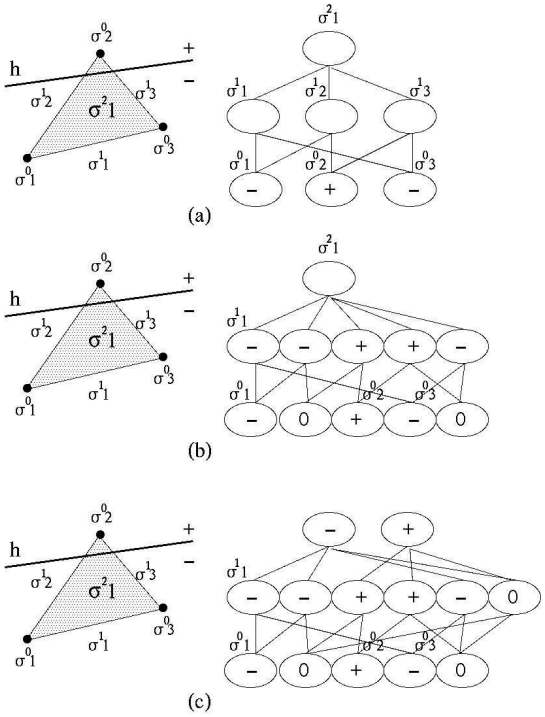


図 8 ：有界 cell の超平面による分割

(2) h とσの交差部分にある dim(σ)-1 次元の cell の節を生成する．この cell を σ_0 とする．σ_0 と(1)で作成した σ_+ と σ_- を枝でつなぐ．

(3) dim(σ) > 1 のときは，σ_0 の境界上にある dim(σ)−2 次元の cell のうち h 上にある cell 全てと σ_0 とを枝でつなぐ処理を行う．こうした cell は，その時点の incidence graph において σから枝を 2 回辿って，まず，σ_0 の境界上にある dim(σ)−2 次元を得て，その後，それらの位置ベクトルを使って h 上にあるかを調べることで得られる．

(4) 最後に，σを削除する

以上の手順で，σが分割され，新しい節ができるとともに，枝のつなぎ直しが行われる．我々は，上記のアルゴリズムを図 2 のように実装した．

但し，(2) の処理は，2 次元の cell の場合には，単純にσ_0とグラフ内の j−2 次元の cell とを全て arc でつなぐという単純な方式でも動く（これは，Chandrajit らが示した実装法）．しかし，3 次元以上では動かない．我々が上記に示したアルゴリズムは，グラフの節に持たせた H(σ)に対する position を使い，グラフ内の j−2 次元の cell でしかも，h 上にあるものを選ぶと]

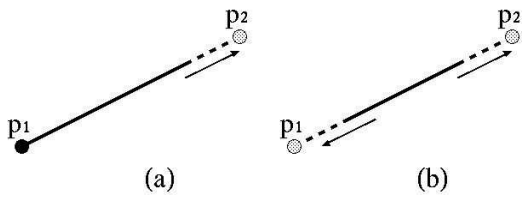


図９：非有界な 1-cell の例
(a) 端点を一つもつ 1-cell
(b) 端点を持たない 1-cell

いう方式をとっており，3 次元以上の任意の次元で正しく動く．

図 8 に例を示す．(a)の σ^1_2，σ^1_3 に対して分割処理を行った結果が(b)である．さらに σ^2_1 の分割処理を行った結果が(c)である．図 8(b)のように，全ての cell に対して分割処理を終えていない時点のグラフは，incidence graph とは異なる構造になっている．

4. 2 有界な 1-cell の分割

Candrajit らの cell splitting algorithm[1] は，0-cell の position を利用して 1-cell（線分）が超平面によって分割されるかどうかを判定する．まず始めに，2 つの 0-cell について超平面に対する position を計算する．そして，２つの 0-cell の position から，1-cell が分割されるかどうかを判定する．もし，２つの position が＋と－であれば，1-cell は分割される．そうでなければ，分割されない．この時，2 つの 0-cell がどちらも 0 であれば，1-cell は 0 である．0-cell がどちらも＋(－), 又は＋と 0(－と 0)であれば，1-cell は＋（－）である．

例えば，図８において，３つの 0-cell σ^0_1，σ^0_2，σ^0_3 の position は，それぞれ＋，－，＋である．1-cell σ^1_1 は２つの 0-cell σ^0_1，σ^0_3 に接続しているが，両方とも position は－である．よって，σ^1_1 も－と判定される．また，1-cell σ^1_2 と σ^1_3 は＋と－の 0-cell に接続している為，超平面 h により分割されると判定される．

1-cell が超平面により分割されると判定されると，G(σ)に新たな節を作成し，枝のつなぎ直す処理が行われる．例えば，1-cell が分割される場合，２つの 1-cell と，その間にある 0-cell が新たに生成され，もとの 1-cell は削除される．また，これらの節に対して枝のつなぎ直しが行われる．このようにして，全ての 1-cell について，処理が終了した後，k-cell(k≧2)について，処理が繰り返される．k-cell が分割されるか否かの判定は，k-cell に接続する(k-1)-cell の position から判定する．(k-1)-cell の position に＋と－が両方含まれている場合は，k-cell は分割されると判定される．この場合も，1-cell の場合と同様に節の作成，枝のつなぎ直しの処理が行われる．

[1]に示されているアルゴリズムでは，非有界な 1-cell の分割判定を正しく行うことができない．このアルゴリズムを適用する為には，全ての 1-cell が二つの 0-cell に接続している必要がある為である．しかし非有界な cell は，この条件を満足しない．例えば，半直線は１つの点にのみ接続し，直線は点に接続しない．

4. 3 extended cell splitting algorithm

extended cell splitting algorithm は，２つの点の position を利用して，1-cell が超平面により分割されるかどうかを判定する．1-cell が有界である場合は，２つの端点の position を利用する．1-cell が非有界で端点を一つだけ持つ場合は，その端点と，1-cell 上にあり端点から十分離れた所にある点の position を利用する(図９(a))．1-cell が端点を持たない場合は，1-cell 上にあり，互いに十分離れた２つの点の position を利用する(図９(b))．

これらの点の position は，次のようにして求めることができる．l を任意の直線とする．図 10 に３次元空間での例を示す．図 10 に示すように，$\vec{v}$ は l に沿ったベクトル，$\vec{u}$ は，原点から l への垂線の足の点 q を表すベクトルとする．$\vec{p} = (x_1, x_2, ..., x_d)$ を l 上の点 p を表す d 次元ベクトルとすると，$\vec{p}$ は $\vec{v}$，$\vec{u}$ と２点 q，p の距離を示す t を用いて次のように表現される．

$$\vec{p} = t\vec{v} + \vec{u}$$

ここで，超平面 h_i の垂直ベクトル

$\vec{\alpha} = (\alpha_{j1}, \alpha_{j2}, ..., \alpha_{jd})$ を用いて $\phi'_j(\vec{p})$ を次のように定める．

$$\phi'_j(\vec{p}) = \vec{\alpha}_j \bullet \vec{p} + \alpha_{j(d+1)}$$

この時，$\phi_j(p) = \phi'_j(\vec{p})$ であり，ベクトル $\vec{v}$ と

$\vec{\alpha}_j$ は次のような性質を持つ．

$$\vec{v} \bullet \vec{\alpha}_j > 0 \Leftrightarrow \lim_{t\to\infty} \phi'_j(\vec{p}) \to +\infty$$

$$\vec{v} \bullet \vec{\alpha}_j < 0 \Leftrightarrow \lim_{t\to\infty} \phi'_j(\vec{p}) \to -\infty$$

$$\vec{v} \bullet \vec{\alpha}_j = 0 \Leftrightarrow \lim_{t\to\infty} \phi'_j(\vec{p}) = \vec{\alpha}_j \bullet \vec{u} + \alpha_{j(d+1)}$$

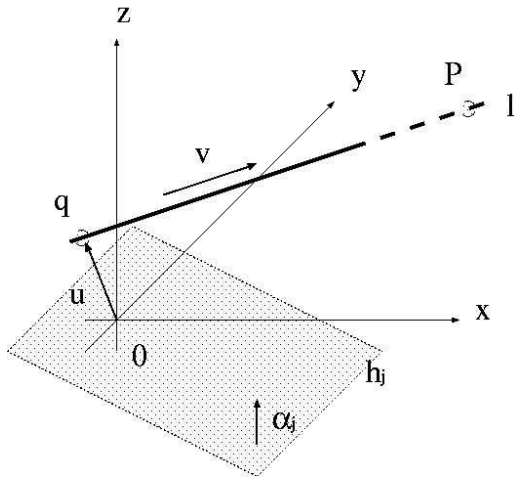


図１０：３次元空間の直線 l

4. 4 半直線の超平面による分割

半直線の場合は，端点の position は，接続 0-cell の座標と超平面により求められる．そして，もう１つは，半直線上にあり，0-cell から十分離れた所にある点の position を利用する．この点の position は 4. 2 章で示した方法で求めることができる．そして，求まった２つの position から，1-cell が超平面により分割されるかどうかを判定する．

図 11 は簡単のため，２次元空間での例を示す．図 11(a)において，0-cell σ^0_1 の h_1 に対する position は＋である．次に，

$$\vec{\alpha}_1 \bullet \vec{p} = (1,3) \bullet (0.2, 0.15) = 0.45 > 0$$

より，p の position は＋と判定される．よって２点の position がどちらも＋であるので，σ^1_1 は＋である．図１１(b)においても，σ^0_1 は h_2 に対して＋である．次に

$$\vec{\alpha}_2 \bullet \vec{p} = (-3,4) \bullet (0.2, 0.15) = 0$$

$$\vec{\alpha}_2 \bullet \vec{u} + \alpha_{23} = (-3,4) \bullet (-0.12, 0.16) + 2 = 3 > 0$$

より，p の position は＋と判定される．よってこの場合も，２点の position がどちらも＋であるので，σ^1_1 は＋である．図１１(c)においては，σ^0_1 は h_3 に対して－である．そして

$$\vec{\alpha}_1 \bullet \vec{p} = (1,3) \bullet (0.2, 0.15) = 0.45 > 0$$

より，p の position は＋である．２点の position が＋と－であるので，σ^1_1 は分割される．

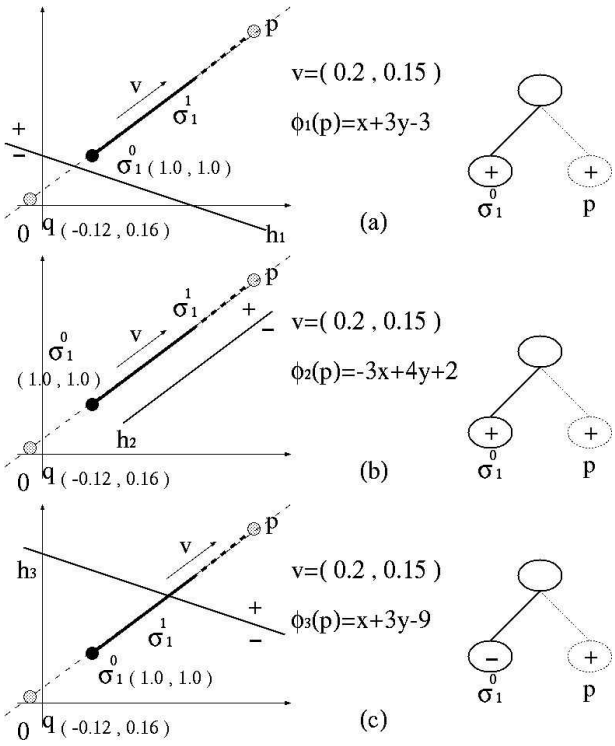


図１１：端点を一つ持つ 1-cell σ^0_1 の超平面による分割

5. 性能テスト

我々は，extended cell splitting algorithm を実装し，性能テストを行った(図１２)．このテストでは，入力として空間次元 d と cell の分割に用いる超平面数 N_h が与えられる．まず，d 個の超平面で構成される d 次元非有界 cell を作成する．そして，この cell をランダムに作成した N_h 個の超平面で繰り返し分割し，それに要した時間を測定した．cell の集合を一つの超平面で分割する処理は，次のように行う．まず，全ての 0-cell の position を計算する．そして，4 章で示した方法で，全ての 1-cell について，超平面により分割されるかどうかを判定していく．k-cell ($2 \leqq k \leqq d$) については，[1]と同様の方法で分割されるかどうかを判定する．incidence graph の操作は図１，図２に示した通りである．

上記のテストを d=3，$1 \leqq N_h \leqq 20$ について行った．結果を図１２に示す．横軸は，k-cell ($0 \leqq k \leqq 3$)の総数であり，縦軸は分割に要した時間である．環境は Sun Blade 100，512MB メモリ，OS は SunOS 5．10 である．

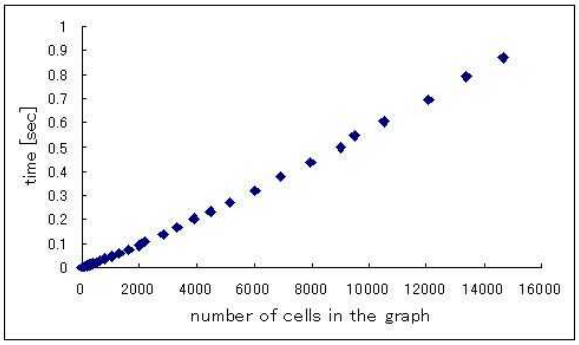


図１２：性能テストの結果

6. まとめ

本論文では，非有界な 1-cell が超平面により分割されるかどうかの判定法を示した．1-cell は，その上にある２点の position を求め，それらを利用することで超平面により分割されるかどうかを判定することができた．次に 3 次元以上の cell に対しても,分割結果を incidence graph として出力するアルゴリズムを示した．また，上記を実装し,その性能テストの結果を示した．

7. 謝辞

本研究の一部は，日本学術振興会科学研究費補助金課題番号 17700117，(A)(2)16200005 による．

文　献

[1] Chandrajit L. Bajaj and Valerio Pascucci, “Splitting a Complex of Convex Polytopes In Any Dimension, ” ACM Press, proceedings of the twelfth annual symposium on computational geometry, pp88-97, 1996.

[2] S. Nirenstein, E. Blake, J. Gain, “Exact from-region visibility culling, ” Proceedings of the 13th Eurographics workshop on Rendering, June 26-28, 2002, Pisa, Italy

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

[4] H. Edelsbrunner, R. Seidel, M. Sharir, “On the Zone Theorem for Hyperplane Arrangements” , SIAM jurnal on Computing, Volume 22Issue 2, pages 418 - 429, 1993

[5] M. de Berg, M. van Kreveld, M. Overmars, O. Schwarzkopf, Computational Geometry Algorithms and Applications, Springer, 1998

[6] Jacob E. Goodman, Joseph O’Rourke, Handbook of discrete and computational geometry, CRC Press , 2004.

[7] Bruce Naylor, John Amanatides, William Thibault , " Merging BSP trees yields polyhedral set operations" , ACM SIGGRAPH Computer Graphics , Proceedings of the 17th annual conference on Computer graphics and interactive techniques, Volume 24 Issue 4, September 1990

[8] Michiko Tanaka , Kunihiko Kaneko , Yingliang Lu, Akifumi Makinouchi, “An Extended Cell Splitting Algorithm for Spatial Databases, ” IEEE TENCON 2004, p. 371-374, November 2004

[9] Yingliang Lu, Kunihiko Kaneko, Akifumi Makinouchi, Michiko Tanaka, “Reducing the Data Size of Spatial Databases Using Sign Vectors, ” IEEE TENCON 2004, p. 367-370, November 2004

[10] Yves Nievergelt, "Analysis and applications of Priest's distillation", ACM Transactions on Mathematical Software (TOMS), Volume 30 Issue 4, December 2004