有限要素解析のためのバケット法に基づく三角形要素の自動生成*

奥 田 斂^{*1}The Automatic Generation of Triangular Mesh Utilizing Bucket Method
for Finite Element Method

Osamu OKUDA

This paper describes an algorithm to generate the triangular mesh within arbitrary planar domains. When applied finite element method to numerical simulation, the discretization of the domain is the essential part of the procedure. There are many reports of the automatic mesh generation methods, such as coordinate transformation, blending function, quad-tree, delaunay, advancing front method etc. The characteristic of the proposed algorithm is that the position of nodes is generated in original polygonal region as possible as the form of an equilateral triangle and the triangulation of the inner nodes is carried out by bucket method in computational geometry prior to the global construction of the triangular mesh modelling. Then the triangulation of residual region is carried out utilizing advancing front method. The processing scheme is developed and applied to several examples. The numerical results are shown and the effectiveness is discussed.

Key Words: Computational Mechanics, FEM, Triangulation, Computational Geometry, Automatic Mesh Generation, Bucket Method

1. 結 言

有限要素法(FEM)を用いて数値シミュレーションを行うためには、前処理として、解析対象の領域の要素分割が必要となる。有限要素解析のための入力データとなる離散化モデルの作成は煩雑であり、作成ミスも生じる。したがって、FEMの発展とともに、三角形や四面体等の有限要素を自動的に生成、すなわち、自動要素分割の研究が現在まで数多く行われている。二次元問題に対しては、三次元の立体問題に比較して、容易であるために各種の方法があるが、代表的なものとして、

- (1) 座標変換法⁽¹⁾
- (2) ブレンディング関数法⁽²⁾⁽³⁾
- (3) 4分木法⁽⁴⁾
- (4) アドバンシングフロント法^{(5)~(12)}
- (5) デロネイ法^{(13)~(17)}

等がある。他にマルチブロック法も考えられ、上述の(1)、(2)も含めてしまうことも可能である。自動要素分割の各種の方法についてのアルゴリズムの解説は

George⁽¹⁸⁾によりまとめられている。各方法には、それぞれ一長一短があるために、問題に応じて解析対象の物理現象を的確にとらえることのできる方法を選択する必要がある。実用上の汎用性、自動要素生成効率の点から、逐次要素を生成してゆく(4)と計算幾何学の分野においてアルゴリズムの研究が盛んに行われている(5)の研究報告が数多く見受けられる。

本論文では、任意平面領域に対して、領域内部の三角形要素の生成には、一様に分布している点列を高速に探索できる計算幾何学のバケット法の考え⁽¹⁹⁾に基づいて効率良く実行し、残った領域にはアドバンシングフロント法により逐次要素を生成することを主たる特徴とする自動要素分割アルゴリズムについて述べる。さらに、各種の例題に適用して、その有効性を明らかにした。

2. 自動要素分割処理過程

任意平面領域に対するFEM解析のための自動要素分割に必要な基本的な処理機能としては、複雑な形状に対応できるように、

- (1) 元の領域の部分領域への要素分割が可能
- (2) 各部分領域に空領域が存在する、すなわち、多重連結体にも対応が可能

* 原稿受付 1997年11月13日。

^{*1} 正員、物質工学工業技術研究所(☎305-8565 つくば市東1-1)。

- (3) 各部分領域に任意の要素密度の指定ができる
- (4) 生成三角形要素の形状が良好
- (5) バンド幅縮小化を実行できる

を満足するようにした。さらに、有限要素解析の実行のために、反復解法において解析規模を把握できるように非零要素の数も計算できる機能を有している。順次、自動要素分割処理過程について述べる。

2.1 節点列の生成 図1は領域内に一つの空領域を有する分割対象となる元の領域、この領域を包含する長方形および節点列の生成状態を示すものである。元の形状を特性づける端点の番号は外部境界については反時計回りに、内部空領域については時計回りにつける。各節点は、三角形要素として形状が良くなるように、正三角配置とし、各ブロック、すなわち、バケット内に一つの節点が存在する。この i 行 j 列番めの節点列の座標値は、以下

$$\begin{Bmatrix} x \\ y \end{Bmatrix}_{i,j} = \begin{Bmatrix} x \\ y \end{Bmatrix}_0 + \begin{Bmatrix} \alpha d \\ \beta d \end{Bmatrix} + \begin{Bmatrix} d \cos \gamma(i-1) \\ d \sin \gamma(i-1) \end{Bmatrix} + \begin{Bmatrix} d \sin(\pi/3 - \gamma)(j-1) \\ d \cos(\pi/3 - \gamma)(j-1) \end{Bmatrix} \quad (1)$$

$$i=1, \dots, m \quad j=1, \dots, n$$

の式で与えられる。ここに、

m : x 方向のバケットの数

n : y 方向のバケットの数

$$\begin{Bmatrix} x \\ y \end{Bmatrix}_0 : \text{バケットの原点の } x, y \text{ 座標値}$$

d : 要素密度(正三角形の一辺の長さ)

α : x 座標の初期位置調整パラメータ

β : y 座標の初期位置調整パラメータ

Fig. 1 Original domain, bucket and generated nodes

γ : 節点列と x 座標のなす角度

$$m = \left\{ \frac{\text{Max}(x_{i,j}) - \text{Min}(x_{i,j})}{d} \right\} + 1$$

$$n = \left\{ \frac{\text{Max}(y_{i,j}) - \text{Min}(y_{i,j})}{\sqrt{3} d/2} \right\} + 1 \quad \dots\dots\dots (2)$$

である。

2.2 三角形要素となる候補節点の選択 候補節点を選ぶには、まず第一に、元の多角形を構成する点列に基づき、三角形化の一辺となる基底 1-2, ..., 24-15 について、各基底の左側にある端点を、

$$S = \begin{vmatrix} x_i & y_i & 1 \\ x_{i+1} & y_{i+1} & 1 \\ x_p & y_p & 1 \end{vmatrix} \quad \dots\dots\dots (3)$$

の符号により判定し、その端点の中から基底の中心点からの距離がある範囲内にある端点を選択する。次に、この端点の中で、迎え角 θ が最大になるものを選ぶ。このとき、選択した端点に基づく二つの基底は他の基底と干渉を起こさないものを採用する。干渉の判定は基底 $P_k - P_{k+1}$ と他の基底 $P_k - P_{k+1}$ の互いの端点位置関係から、式(3)を用いて、

$$S_{P_i} \cdot S_{P_{i+1}} \leq 0 \quad \dots\dots\dots (4)$$

かつ、

$$S_{P_k} \cdot S_{P_{k+1}} \leq 0 \quad \dots\dots\dots (5)$$

であれば二つの基底は交差している。迎え角が最大となる点は、

$$P = P[\text{Max}(\theta_i) | r < R_c] \quad \dots\dots\dots (6)$$

と表現できる。

このようにして、元の多角形を三角形に分割したものが図2である。各三角形に対して、辺が元の多角形の境界を構成しておれば、境界辺から距離 ε だけ離れた直線に基づく三角形内の節点を選ぶ、これは式(3)

Fig. 2 Triangulation of original polygon

を3回使って判定できる。さらに、境界辺を含まない三角形については、半径 ε の円の外側にある節点を選択する。このようにすることにより、極端に凹な多角形に対しても均等な形状の要素の生成が可能となる。

図3の◎印が選択された節点を表している。

2・3 内部節点に関連する要素の生成 多角形内部の節点に基づく三角形要素の生成は、計算幾何学のバケット法により、 i 行 j 列番めの節点に注目すると、隣接する節点が i 行 $j+1$ 列、 $i+1$ 行 $j+1$ 列と $i+1$ 行 j 列のバケット内に存在すれば、二つの要素を生成する。このようにして、隣接するバケットの情報のみで三角形要素化を行うので、全候補節点から探索する必要がなく効率が良い。図4は生成した多角形内部の三角形要素と部分領域内の三角形要素化の節点番号を示している。このとき、残された領域に対する要素生

Fig. 3 Selection of candidate nodes

Fig. 4 Generation of inner triangular elements

成のための基底 64-59, 59-60, ..., 72-71 を生成する。

2・4 残された領域の三角形要素化 外側境界上の節点列は要素密度 d に基づいて、反時計回りに、各多角形辺に対して、

$$dif = \frac{l_{p,p+1}}{d} - \text{INT}\left(\frac{l_{p,p+1}}{d}\right) \quad \dots\dots\dots (7)$$

$$\text{if } dif > 0.5, nd = l_{p,p+1}/d + 1$$

$$\text{if } dif \leq 0.5, nd = l_{p,p+1}/d$$

分割数を決める。ここに、

$$l_{p,p+1} = \sqrt{dx_p^2 + dy_p^2} \quad \dots\dots\dots (8)$$

$$\begin{Bmatrix} dx \\ dy \end{Bmatrix}_p = \begin{Bmatrix} x \\ y \end{Bmatrix}_p - \begin{Bmatrix} x \\ y \end{Bmatrix}_{p+1} \quad \dots\dots\dots (9)$$

であり、 p は多角形を形成する端点の x, y 座標である。基底となる節点の座標は、

$$\begin{Bmatrix} x \\ y \end{Bmatrix}_q = \begin{Bmatrix} x \\ y \end{Bmatrix}_p + \begin{Bmatrix} dx/nd \\ dy/nd \end{Bmatrix}_p (k-1) \quad \dots\dots\dots (10)$$

$$k=1, \dots, nd+1$$

により求めることができる。内部境界の多角形に対しても、時計回りに節点を連続して付番する。内部節点の番号付けは、すべての内部境界上の番号付けの処理の後、連続してつける。

残された領域に対する三角形要素化は、基底 1-2, 2-3, ..., 39-1, 40-41, ..., 57-40, 59-60, ..., 72-71 から順次開始する。例えば、1-2 の場合、1 → 2 の左側にある節点列の中から、基底の中点 p から距離 R_c よりも小さい節点を数個選択する。次に、選択した節点の中から、式(6)のように、迎え角 θ が最大となるものを選び、要素 1-2-39 を生成する。この時点で、基底 1-2, 39-1 を元の基底のあるファイルから削除し、新たに 2-39 を追加する。この処理過程を順次繰返して要

Fig. 5 Final triangular mesh

素を生成していき、最終的に基底がなくなれば一つの部分領域内の三角形要素の生成が完了となる。

図5は一つの部分領域内の三角形要素化の結果を示すものである。

2・5 全体領域の節点の番号付け 全体領域を数個の部分領域に分割した場合、各部分領域に対して、外側境界上に生成した節点の番号付けは反時計回りに、空領域を構成する内側境界上の節点については時計回りにつけ、それに続けて内部領域の節点に連続してつける。最初の部分領域からそれに続く部分領域への全体系における節点の番号付けは部分領域の共通境界上の節点かどうかを判定して、重複しないように、連続して最終の部分領域までこの処理を実行する。

領域内部の節点については、ラプラシアンスムージ

ングの機能⁽²⁰⁾をもたせてあり、

$$C_i^k(x, y) = \sum_{j=1}^m C_{N_j}^{k-1}(x, y) / m \quad \dots\dots\dots (11)$$

で求めることができる。ここに、 N_j は節点 i に結合している節点番号、 m は結合している節点の数、 k は処理回数である。このスムージングの機能は各部分領域の中だけではなく、全体の領域における内部節点に対しても有している。図5は一つの部分領域に対するラプラシアンスムージングを実行した三角形要素化の結果である。

2・6 バンド幅縮小化 節点番号は有限要素解析における連立方程式の解法の効率を左右するので、直接法あるいは反復法のどちらの解法を使用するにしても、バンド幅縮小化の機能をもたせてある。ここでは、Collinsの方法⁽²¹⁾とreverse Cuthill-McKee法⁽²²⁾に基づいて、バンド幅縮小化を行うとともに、全体系の特性行列の非零成分の数およびスカイライン内の非零成分の数を求め、解析対象となる問題の規模を把握できる機能を有している。

3. 数値計算例

上述のアルゴリズムに基づいて開発した三角形自動要素分割プログラムが目標とした機能を満たしていることを確認するために、いくつかの例題に適用した。

3・1 琵琶湖 図6は琵琶湖を三つの部分領域に分割したものを示している。この三つの部分領域に対して、それぞれ要素密度を、3.0, 4.5, 5.0と指定して三角形要素分割した結果が図7であり、要素数603、節点数367である。図8は自由度を1とした場合の元の節点番号つけの非零成分の分布状態を示すものであ

Fig. 6 Sub-domain of Lake Biwa

Fig. 7 Triangulation of Lake Biwa

Fig. 8 Non-zero elements of original numbering

る。半バンド幅、非零成分の数、スカイライン内の数はそれぞれ 157, 970, 10 797 である。図 9 は Reverse Cuthill-McKee 法に基づくバンド幅縮小化を実行した場合の結果であり、半バンド幅、スカイライン内の成分の数はそれぞれ 18, 3 875 となっている。したがって、FEM を用いて数値シミュレーションを実行する場合には、その解法に対応して、上述の自動要素分割のデータを使用する。

3・2 C 形フレーム 図 10 は C 形フレームを部分領域に分割せずに要素密度 5 で自動要素分割した結果であり、要素数 273, 節点数 164 であった。この例の場合、形状から多角形内部の領域に三角形要素がバケット法に基づいて効率良く生成されていることが図

11 よりわかる。

3・3 大腿骨 図 12 は大腿骨の断面を四つの部分領域に分割して、各領域に対して、要素密度をそれぞれ 6, 6, 4, 4 と指定して分割結果を示すものである。この場合内部共通境界上にある節点にはスムージング処理を行わないで、各部分領域の内部にある節点だけに処理を実行した。

3・4 タービン翼 図 13 は一つの領域内に五つの空領域を有するタービン翼の断面を自動要素分割した結果で、要素数, 節点数はそれぞれ 266, 195 であった。

3・5 フック 図 14 は幅の狭いフックに対して、要素密度 0.5 と指定して、自動要素分割した結果で、要素数 184, 節点数 134 であった。この例の場合のよ

Fig. 9 Non-zero elements of reverse Cuthill-McKee numbering

Fig. 11 Inner triangular mesh of C-type frame

Fig. 10 Triangulation of C-type frame

Fig. 12 Triangulation of human femur

Fig. 13 Triangulation of hook with narrow width

Fig. 14 Triangulation of turbine blade

うに幅の狭い多角形には内部に三角形要素の生成数が少ないために、バケット法による効率は上がらない。

4. 結 論

任意平面領域に対して、有限要素法を用いて数値シミュレーションを実行する場合にバケット法を用いることにより効率良く、かつ形状の良い三角形要素を自動生成できるアルゴリズムを提案した。それに基づいて開発したプログラムを用いて、いくつかの代表的例題に適用した結果、最初に目標設定した有限要素解析のための三角形自動要素分割として必要となる機能を満足していることを確認できた。

文 献

- (1) Zienkiewicz, O. C., ほか 1 名, An Automatic Mesh Generation Scheme for Plane and Curved Surfaces by Isoparametric Co-ordinates, *Int. J. Numer. Methods Eng.*, 3(1971), 519-528.
- (2) Gordon, W. J., ほか 1 名, Construction of Curvilinear Co-ordinates Systems and Applications to Mesh Generation, 5(1973), 461-477.
- (3) Haber, R., ほか 4 名, A General Two-dimensional Graphical Finite Element Preprocessor utilizing Discrete Transfinite Mapping, 17(1981), 1015-1044.
- (4) Yerry, M. A., ほか 1 名, A Modified Quadtree Approach to Finite Element Mesh Generation, *IEEE, CG & A*, (1983), 39-46.
- (5) Cavendish, J. C., Automatic Triangulation of Arbitrary Planar Domains for the Finite Element Method, 8(1974), 679-696.
- (6) Suhara, J., ほか 1 名, Automatic Mesh Generation for Finite Element Analysis, *Advances in Computational Methods in Structural and Design*, (1972), 607-624, UAH Press, Huntsville, Alabama.
- (7) Shaw, R. D., ほか 1 名, Modifications to the Suhara-Fukuda Method of Network Generation, *Int. J. Numer. Methods Eng.*, 12(1978), 93-99.
- (8) Sadek, E. A., ほか 1 名, A Scheme for the Automatic Generation of Triangular Finite Elements, 15(1980), 1813-1822.
- (9) Bykat, A., Automatic Generation of Triangular Grid : I-Subdivision of a General Polygon into Convex Sub Regions II-Triangulation of Convex Polygons, 10(1976), 1329-1342.
- (10) 城道介・ほか 1 名, 有限要素法のための可変メッシュ生成プログラム, *精密機械*, 47-4(1981), 399-404.
- (11) Lo, S. H., A New Mesh Generation Scheme for Arbitrary Planar Domains, *Int. J. Numer. Methods Eng.*, 21(1985), 1403-1426.
- (12) 奥田敏, 有限要素解析のためのパターン発生による任意平面領域の自動要素分割, *精密機械*, 51-4(1985), 802-808.
- (13) Lo, S. H., Delaunay Triangulation of Non-convex planar domains, *Int. J. Numer. Methods Eng.*, 28(1989), 2695-2707.
- (14) Sloan, S. W., A Fast Algorithm for Constructing Delaunay Triangulation in the Plane, *Adv. Eng. Software*, 9-1(1987), 34-55.
- (15) 谷口建男・ほか 1 名, 直線辺で構成される任意 2 次元領域へのデラウニー三角形分割の適用, *土木学会論文集*, 432/I-16(1991), 69-77.
- (16) 谷口建男, FEM のための要素自動分割, (1992), 9-27, 森北出版.
- (17) Watson, D. F., Computing the n -dimensional Delaunay Tessellation with Application to Voronoi Polytopes, *Comput. J.*, 24-2(1981), 167-172.
- (18) George, P. L., *Automatic Mesh Generation*, (1991), 1-262, John Wiley & Sons.
- (19) 今井浩・ほか 2 名, 計算幾何学 2, bit, 16-13(1984), 1647-1657.
- (20) Buell, W. R., Mesh Generation-A Survey, *Trans. ASME, J. Eng. Indust.*, (1973), 332-338.
- (21) Collins, R. J., Bandwidth Reduction by Automatic Renumbering, *Int. Numer. Methods Eng.*, 6(1973), 167-172.
- (22) 村田健朗・ほか 2 名, スーパーコンピュータ, (1985), 116-118.