I-20　マトリックス構造解析のための節点番号付けアルゴリズム

京都大学大学院　学生員　殿本 卓
京都大学工学部　正 員　白石成人
岡山大学工学部　正 員　谷口健男

1. まえがき

マトリックス構造解析においては連立一次方程式を解く必要があり、この連立一次方程式を解く方法として一般に Band Matrix 法が用いられている。この Band Matrix 法を有効に利用するための帯幅減少問題は、NP-complete 問題[1]であることが知られており有効な帯幅減少アルゴリズムの提案は非常に困難である。このような認識にたって近年、系の形状により連立一次方程式の数値解法を選択することが提案されている。[2] 本報文は、その選択において Band Matrix 法が選ばれたときにそれを有効に利用するための Renumbering Algorithm を提案する。なお一般形状についての帯幅減少問題は、谷口が Filing Field なる特殊な座標系を用いて詳細に研究を行なっている。[3]

2. 帯幅減少法に対する考察

与えられた連立一次方程式を $Ax=C$ とすると、行列Aの帯幅 b は、

$$b=\max_{a_{ij}\neq 0}|i-j| \qquad (1)$$

で与えられる。帯幅減少問題は、節点 i に隣接している節点の節点番号 j をできるだけ小さくする問題であるとも言える。つまり、帯幅減少問題は節点番号付け問題となる。従来より提案されている帯幅減少法は、グラフの直径の一端を出発点とする等距離集合 $\{L_i\}$ を作り、各 L_i の1つのグループとして節点番号付けを行なうものであった。[4]（ここで節点 u, v の距離 $d(u,v)$ とは、u、v を結ぶ path の中で最小の path に含まれる辺数のことである。等距離集合 $\{L_i\}$ とは、出発点を u としたとき $L_1=\{u\}$, $L_2=\{v\,|\,d(u,v)=1\}$, $L_3=\{v\,|\,d(u,v)=2\}$, ……, $L_n=\{v\,|\,d(u,v)=n-1\}$ と作った集合族のことである。）従って、Algorithm の主要な部分は各 L_i 内の節点番号付けをどのように与えるかである。しかし、谷口が述べているように[3]半帯幅（H.B.W.）は、

$$H.B.W.=\max_i|L_i|+\beta+1 \qquad (2)$$

（ここでの L_i は、Filing Field 上の ith column のことであるが、等距離集合 L_i と考えてもよい。β は Filing Field 上の column の上下の shift による補正項である。）で与えられ、β は一般に $\max_i|L_i|$ に比較して小さいことを考慮すると、帯幅減少問題は $\max_i|L_i|$ をいかに小さくするかということにある。L_i 内の節点番号付けは、Filing Field におけると同じように一方の端から他方の端へ規則的に単調に行なっていけば充分である。さて、直径の一端を出発点として等距離集合を作っていくやり方は、$|L_i|$ $(i=1,\cdots,n)$ の相対的な大きさを小さくする方法であって、必ずしも $\max_i|L_i|$ を最小にすることにはなっていない。L_i は直径の一端を利用して作られるのではなくて、直径に直交するように作られるべきである。ここで提案する手法は、$\max_i|L_i|$ に相当する L_c をできるだけ $|L_c|$ が小さくなるようにまず決定し、その L_c を中心として両側に等距離集合を作っていく方法である。これは、帯幅の本質である直径に直交する最大の横断 path を初めに確定させてしまう方法だと言える。なお、節点番号付けもこの L_c を中心にして行なう方法を提案する。

3. 帯幅減少アルゴリズム

提案した Algorithm の中で述べられている節点集合 $\{v_1, v_2, v_3\}$ を結ぶ Shortest Path とは、$\{v_1, v_2, v_3\}$ を結ぶ path のうちその path に含まれる辺の数が最小の path のことである。以下に Algorithm を述べる。

（Algorithm）　Band Width Minimization Algorithm

（Step 1）境界上の節点数を l としたとき、境界上の節点にはあらかじめある任意の点から時計方向に $1, 2, \cdots, l$

と番号付けがなされているとする。境界上の節点の集合をC_1, C_1より距離1にある点の集合をC_2, 以下同様にして$C_3, C_4, \cdots, C_t$を作る。

(Step 2) $u_1 \in C_t$なる任意の点u_1を選ぶ。u_1より等距離集合$\{K_i\}$を作る。最初に境界上の点を含むK_iをK_mとおく。u_1を含むC_tがcycleであるときには、K_{m+1}も求める。

(Step 3) $K_m (K_{m+1})$に含まれる境界上の点で、連結している点を要素とする集合B_iを作る。一般にこのようにして作られるB_iは2つ以上存在する。$(i \geq 2)$

(Step 4) $b \in B_i, b' \in B_j$でb, b'は節点番号を表わしているとして、集合B_i, B_j間の距離を$\max|b-b'|$で定義する。$\{B_i\}$の中で最大距離をもつpairをB_p, B_qとする。

(Step 5) $B_p (B_q)$の要素の中で真中の節点番号をもつ点を$u_2 (u_3)$と定める。

(Step 6) (u_1, u_2, u_3)を結ぶShortest Pathを見い出す。それをcenter lineと呼ぶ。

(Step 7) center line (L_c)から両側へ等距離集合$\{L_i\}$を作っていく。

(Step 8) center line上にある節点数をr個、center lineより左(u_2を上、u_3を下においた場合)にある節点の個数をs個とする。u_2に番号s+1をつけ、u_3にむかってs+2, s+3, …と順に番号付けを行ない、u_3には番号s+rをつける。

(Step 9) u_2の右隣り(L_{c+1})にある境界点に番号s+r+1をつける。節点番号s+r+1の点に隣接する点が1つのときはその点にs+r+2を、多数あるときは、それらの点に隣接する節点の節点番号が小さいもの順にs+r+2, s+r+3, …と番号をつける。同様にしてL_{c+1}のすべての点に番号をつける。

(Step 10) u_3の左隣り(L_{c-1})にある境界点に番号sをつける。節点番号sの点に隣接する点が1つのときはその点にs-1を、多数あるときは、それらの点に隣接する節点の節点番号が大きいもの順にs-1, s-2, …と番号をつける。同様にしてL_{c-1}のすべての点に番号をつける。

(Step 11) (Step 9), (Step 10) をくり返してすべての節点に番号をうつ。

このAlgorithmの中で述べられているcenter line上の節点数$|L_c|$が、2式の$\max|L_i|$に重大な影響を及ぼすことは明らかであろう。図1, 図2はこのAlgorithmの適用例を、表1は修正C-M法で求めた帯幅と提案した方法で求めた帯幅との比較結果を示している。なおcase 1は、4角形分割の25点系regular Meshを、case 2, case 3は図1, 2のような図形の場合を示している。

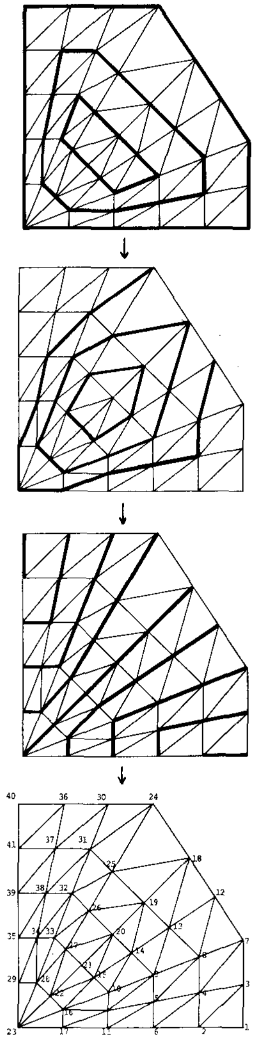

図1 アルゴリズムの適用例

4. あとがき

帯幅の本質である直径に直交する最大の横断pathを初めに確定することで、従来の帯幅減少法より有効なAlgorithmを提案することができた。

適用例	半帯幅	
	修正C-M法	提案した手法
CASE 1	9	6
CASE 2	11	7
CASE 3	14	11

表1 修正C-M法と今回の手法の比較

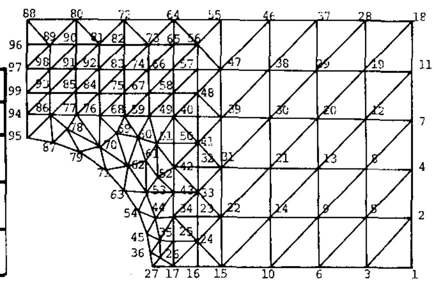

図2 アルゴリズムの適用例

〔参考文献〕

1) Ch.H. Papadimitriou, Computing 16, 1976, pp.263-270

2) 白石, 谷口, 殿本, 第3回電算機利用に関するシンポジウム, 1978, pp.81-84

3) 小西, 白石, 谷口, Journal of Structural Mechanics, Vol 4 No 2, 1976, pp.197-226

4) Cuthill, E. and Mckee, J., Proc. of ACM National Conference, 1969, pp.157-172