回路網の計算機援用設計に関する研究

１９７６年１月

坂　本　明　雄

内容梗概

本研究は、著者が大阪大学大学院工学研究科（電子工学専攻）の学生として尾崎研究室において行なった研究のうち、回路網の計算機援用設計に関する研究をまとめたものである。

工学の諸分野における設計の自動化への動きは、時期を同じくして出現した電子計算機の急速な能力の増大と相まって、今日の計算機援用設計（CAD; Computer Aided Design）の研究・開発の著しい発展につながっている。特に回路網の分野でこれまでに計算機援用設計が生かされてきた例としては、伝送用フィルタの設計、論理装置の設計 ── 特に印刷基板あるいは集積回路の配置・配線設計 ── および回路網解析プログラムなどを挙げることができる。

本研究は、回路網の計算機援用設計のうち、特に、疎大行列解析および印刷基板の自動配線システムについて論じている。

第1章緒論においては、本研究の目的ならびにその工学上の意義、およびこの分野での研究の現状について述べ、本研究の新しい諸成果について概説している。

第2章においては、発表論文[1]~[3]を中心に、回路網における疎大行列解析について論じている。

回路網解析の際に現われる連立一次方程式の係数行列は、一般に変数の個数が非常に多くしかも零要素を多量に含んだ、いわゆる疎大行列（sparse matrix）であることが多い。このような疎大行列を係数とする系の解析においてそのスパース性を活用することは、計算機援用設計の一環として回路網解析を行なうとき、演算時間の短縮、メモリの節約あるいは丸め誤差の削減といった観点からきわめて重要である。

このような疎大行列のスパース性を活用する問題について、ここではまずその基本的問題を論じ、このうち特にピボット操作の最適順序づけに関してさらに詳しく考察している。そのために、まずスパース性の保存を考慮した新しいピボット操作を定義し、この操作の最適順序を決定する問題

を定式化している。さらに、疎大行列に対してその構造（非零要素の分布）を表示するp-グラフを導入し、行列のピボット操作に対応するp-グラフの上での操作について考察している。最後に、擬似最適順序を求めるための新しい一つのアルゴリズムを提案している。これは、操作順序を決定するときいくつかの規準を設定し、ある規準に対する評価値が等しいとき別の規準に照らして次の操作順序を決定するもので、特に二段階後の操作で新たに生じる非零要素の個数を規準の一つとして採用している。いくつかの例題に対して、このアルゴリズムによって最適順序の一つが比較的短時間で求められたことを示している。

第3章においては、発表論文[4]～[7]を中心に、印刷基板の自動配線システムについて論じている。

論理装置の設計は、印刷基板（printed wiring board）を用いて実現されることが多く、ここでは特に印刷基板の自動配線問題に着目している。この問題に関しては従来から多くの手法が提案されているが、個々の手法はそれぞれの特徴を有している反面、特殊な情況のもとではその有効性が十分発揮されないことが起こり得る。ここでは、それらのうち代表的な三つの手法を有機的に併用した一つの自動配線システムについて考察している。この三つの手法とは、スペース・チャネル割当法、線分探索法および迷路法である。本システムは、線分探索法を中核にして他の二つがその前後に位置し、線分探索法の欠点を補う役割を果たしている。このような構成によって、全体として印刷基板の配線パターンを設計するための一つの有効なシステムになっていることが示されている。さらに、信号の配線処理の順序の問題およびスルーホールの個数削減についても論じ、最後に具体的な例題の本システムによる設計結果を示している。

第4章結論においては、本研究全般にわたり、その結果の意義と残された問題が述べられている。

関連発表論文

[1] 坂本、白川、尾崎；"回路方程式のpivot操作におけるsparsityの活用"、信学会、回路とシステム理論研究会資料CT71-17(昭46-07).

[2] 坂本、白川、尾崎；"スパースな連立方程式におけるピボット操作の順序づけ"、情報処理、13、3、p.154(昭47-03).

[3] I.SHIRAKAWA, A.SAKAMOTO AND H.OZAKI; "SPARSITY-ORIENTED ORDERING OF PIVOTAL OPERATIONS ON NETWORK EQUATIONS", SIAM J. APPL. MATH., 24, 1, P.121 (JAN. 1973).

[4] 坂本、山村、千葉、山下、河田、白川、尾崎；"最適配置配線問題に関する一手法"、信学会、回路とシステム理論研究会資料CST73-70(昭48-12).

[5] 坂本、千葉、井手、白川、尾崎、山村、杉田、西岡、栗本；"プリント基板自動配線プログラムOSACAについて"、信学会、回路とシステム理論研究会資料CST74-58(昭49-10).

[6] A.SAKAMOTO, T.CHIBA, I.SHIRAKAWA, H.OZAKI, S.SUGITA, T.KURIMOTO AND I.NISHIOKA; "OSACA; A SYSTEM FOR AUTOMATED ROUTING ON TWO-LAYER PRINTED WIRING BOARDS", PROC. USA-JAPAN DESIGN AUTOMATION SYMPOSIUM '75, P.100 (AUGUST 1975).

[7] 坂本、千葉、白川、尾崎、杉田、栗本、西岡；"プリント基板自動配線システムOSACA"、情報処理、掲載予定.

回路網の計算機援用設計に関する研究

目　　次

第1章 緒 論

今世紀後半，工学の諸分野で種々の要因から設計の省力化あるいは自動化の必要性が高まり，そのための研究が盛んになった。時期を同じくして電子計算機が出現し，時代の要望に応じて急速に発展を遂げ，所期の目的であった科学技術計算のみならず事務処理や自動制御などにもその応用範囲を拡げ，次第にその能力を増大していった。さらに大容量化，高速度化，高信頼化などが進んだ結果，計算機を工業生産の合理化のために応用することが強調されはじめ，設計の省力化，自動化の動きと相まって，今日の計算機援用設計（CAD；Computer Aided Design）の研究，開発は著しい発展を遂げている。

回路網の分野でこれまでに計算機援用設計が生かされてきた例としては，伝送用フィルタの設計，論理装置の設計——特に印刷基板あるいは集積回路の配置・配線設計——および回路網解析プログラムなどを挙げることができる。従来，技術者が理論と経験を駆使して設計していたこれらの分野に計算機が導入されたのは，製品需要の量的および質的な向上に伴い，設計対象が大規模化かつ複雑化してきたため，人手による設計では経費および時間などの点で事実上不可能あるいはそれに近い状態になりつつあることがその主たる原因と考えられる。一方計算機を援用すれば，数値計算の精度を高め，大量の変数の取扱いや最適化の際の種々のパラメータの値による繰返し計算を可能にし，さらには単純なミスの除去，製品試験の自動化，製造の迅速化といった設計過程に必要な種々の要求をも満たすことができる。この意味で計算機援用設計は，新しい有力な設計の道具（design tool）としてその重要性を益々高めている。

本論文は，回路網の計算機援用設計のうち特に，疎大行列解析および印刷基板の自動配線システムについて，それぞれ第2章および第3章で論じている。

回路網解析の際にしばしば現われる連立一次方程式は通常，変数の個数が非常に多く，しかも係数行列に零要素を多量に含むという性質がある。このような方程式の係数行列は，疎大行列——あるいはスパース行列（sparse matrix）——とよばれている。また周波数特性や過渡特性を求めるような場合，連立一次方程式をあるパラメータの値を変えて繰返し解くことがあるが，このときの係数行列のスパース構造——どの要素が零でどの要素が非零であるという意味での構造——は一定である。このような疎大行列を係数行列にもつ系の解析においてそのスパース性を活用することは，演算時間の短縮，メモリの節約，あるいは丸め誤差の削減といった観点からきわめて有用であり，近年計算機援用設計の一環としていろいろな立場から考察されてきている[1]~[3]。

2.2節では，疎大行列解析における基本的問題について考察している。ここでは行列をグラフ表現する方法を述べ，疎大行列の分割に関する議論を行なっ

ている。これは，疎大行列の零要素に注目し，行列をある種の形に分解したとき零部分行列 —— すべての要素が零である部分行列 —— の範囲が最大となるように行や列を入れ替える手順に関する問題であり，グラフにおける節点集合の分割の問題に帰着される。さらに2.3節以下で論じられるピボット操作の最適順序づけの問題について考察している。

疎大行列におけるスパース性の活用は，連立方程式の解を求める操作において重要である。すなわち，連立方程式を解く際の Gauss 消去に付随して生じる係数行列の LU分解[4] の過程における，スパース性を保存するようなピボット操作の順序決定問題としてとらえられる[5]~[8]。一方，原始インミタンス行列を係数行列にもつ回路方程式から指定されたいくつか —— 通常二つ —— の端子対に関する縮約方程式の係数行列を求める際に行なわれる端子対縮約操作におけるスパース性保存の問題がある[9][10]。2.3節以下では，スパース性の保存という点でこれらの二つの操作を同時に含むような一つの新しいピボット操作を定義し，この操作の過程におけるスパース性保存の意味で最適なピボット順序を決定する問題を定式化している。さらに新たにp-グラフを導入してピボット操作の擬似最適順序を得るための一つの手法を考察している。これは，操作順序を決定していくときいくつかの規準を設定し，ある規準に対する評価値が等しいとき別の規準に照らして次の操作順序を決定するもので，特に二段階後の操作で新たに生じる非零要素の個数を規準の一つに採用している。最後にいくつかの例題に対して，この手法によって最適順序の一つが比較的短時間で求められることを示している。

論理装置の設計は、印刷基板を用いて実現されることが多く、その過程は論理設計、論理の分割および部品への割付、印刷基板の配置・配線問題などに分けて考えられる。さらには基板の故障検査あるいは基板間の配線などの問題もあるが、第3章では、これらのうち印刷基板の配線問題に着目し、一つの自動配線システムについて論じている。

印刷基板上の配線問題については，従来から多くの手法が提案されているが[19]~[22]，個々の手法はそれぞれの特徴を持つ一方，特殊な情況の下ではその有効性が十分発揮されないことが起こり得る。したがって，情況に即してその特徴を最大限に生かしうる手法を採用しなければならない。ここでは，代表的な三つの手法を用い，それらを有機的に併用した配線システムについて考察している。

配線経路決定問題とは，基板上の指定されたいくつかの点 —— 各点は基板上に搭載されたある回路部品の一つの端子に対応しており，これらの指定された点の集合を信号という —— が電気的に等しくなるように配線を行なうときの基板上の経路を見い出す問題である。ある信号に対する配線経路を決定するとき，基板上に存在する他の回路部品の端子および他の信号のための配線経路な

どは通過不能な場所とみなされる。

一つの印刷基板にはいくつかの回路部品が搭載され，それらの端子の間にはいくつかの信号が定義されている。印刷基板の自動配線システムは，個々の信号についてその配線経路を順次決定していかなければならない。初期の段階では基板上には通過不能な場所は少ないが，各信号のための配線経路が次々に定められていくにつれて通過不能な場所も多くなる。したがって，これらのことを考慮した自動配線システムが必要である。

ここでは中心的役割を果たす配線手法として，線分探索法[25]~[27]を用いる。これは，指定された端子間の配線を線分ごとの探索で行なうもので，特に探索範囲を限定することで無駄な探索を極力避けるように改良されている[25]。しかし，探索範囲を限定したことで効率よく配線経路を決定できる反面，配線が可能であるにもかかわらずその経路を見い出し得ない場合がある。この欠点を補うために迷路法[28]~[30]を用いる。迷路法は，配線の最小単位——これをfメッシュという——ごとに探索していくもので，もし配線が可能ならばそのうちの最短経路の一つを必ず見い出すという特徴を持っている。この迷路法は，すべての信号を線分探索法で配線した後，未完成の部分に適用される。

一方，このシステムの配線ルーチンの第一段階にはスペース・チャネル割当法[23],[24]と呼ばれる配線手法が用いられる。これは，いくつかの端子対間の配線をスペース割当と チャネル割当という二段の操作で同時に定めるもので，前述の二つの手法が個々の信号について順次その配線経路を決定したのとは異なる特徴を持っている。特に，比較的遠距離にある端子対間の配線経路がここで決定され，第二段階で用いられる線分探索法における経路探索を容易にする役割を果たす。

以上のようにこの自動配線システムは，異なる三つの配線手法を用い，互いに他の手法の欠点を補うよう有機的に結合されており，一つの印刷基板の配線パターンを決定する有効なシステムであることが示される。

さらに，信号を取扱う順序の問題およびスルーホール——基板の表と裏にある線分を電気的に結合する孔——の個数削減について論じられ，最後に具体的な例題による設計結果を示し，既存の類似システムとの比較を行なっている。

第2章　回路網の疎大行列解析

2.1 緒　言

零要素を多量に含む大規模行列は疎大行列といわれる。疎大行列を係数行列としてもつ系の解析においてそのスパース性（零要素の存在）を活用することは計算時間の短縮，メモリの節約，あるいは丸め誤差の削減などの観点からきわめて有用である。疎大行列解析に関しては従来いろいろな立場から考察されてきた[1]~[3]。

2.2節では，疎大行列に対する基本的問題として疎大行列のグラフ表現を考察する。さらに，行列の分割ならびにピボット操作の最適順序づけというスパース性保存の立場からの疎大行列解析への二つの接近法を解説する。

2.3節以下では，これらのうち特にピボット操作の最適順序決定問題に関して考察し，その擬似最適順序を得るアルゴリズムについて論じる。2.3節では回路網解析の際にしばしば必要となる係数行列のLU分解ならびに端子対縮約操作に注目し，スパース性の保存という点でこれらの二つの操作を同時に含むような一つの新しいピボット操作を定義し，この操作の過程におけるスパース性保存の点で最適なピボット順序を決定する問題を定式化する。さらに2.4節では、このピボット操作による行列のスパース構造の変化およびスパース性の保存の程度を知るためにp-グラフを定義する。2.5節では、このp-グラフを用いて擬似最適順序を得るアルゴリズムを考察し、例題を用いてその有効性を確かめる。

2.2 疎大行列に対する基本的問題

この節では，n個の未知数 $x_1, x_2, \cdots, x_n$ に関する連立一次方程式，

$$W x = b \,, \qquad (2\cdot1)$$

を解く際の疎大行列 W の取扱いに対する基本的問題について論じる。ただし，式(2·1)において，W は正則なn次正方行列，xおよびbはn次列ベクトルでそれぞれ未知数および定数ベクトルである。

2.2.1 疎大行列のグラフ表現

疎大行列のスパース構造を考察するとき，行列の各要素の数値自体は本質的

なものではなく，その値が零か非零かということのみが意味をもつ。そこで次のような二値化行列を定義する。

［定義２．１］ n次正方行列 $W=[w_{ij}]$ に対して，

$$b_{ij}=\begin{cases}1 & ; \ w_{ij}\neq 0 \ , \\ 0 & ; \ w_{ij}=0 \ , \end{cases} \qquad (2\cdot 2)$$

なる b_{ij} を要素とするn次正方行列 $B(W)=[b_{ij}]$ を，W に対する二値化行列という。（定義終）

行列 W のスパース構造は，それに対する二値化行列 $B(W)$ における $b_{ij}=1$ なる要素の分布を調べればよいが，これを表現するのにしばしばグラフが用いられる。これには大別して次の二通りの表現法がある。なお特に断わらないかぎり以下で定義なしに用いるグラフに関する用語は文献［11］に従うものとする。

［定義２．２］ n次正方行列 $W=[w_{ij}]$ に付随する有向グラフ $G_0(W)=[V, E]$ とは次のような有向グラフである。

(i) 各節点 $v_i \in V$ $(i=1,2,\cdots,n)$ は W の第i列に対応する。

(ii) W の各非零要素 $w_{ij}\neq 0$ には，v_i から v_j に向かう（有向）枝 $(v_i, v_j)\in E$ が対応する。（定義終）

［定義２．３］ n次正方行列 $W=[w_{ij}]$ に付随する２部グラフ（bipartite graph） $G_b(W)=[X, Y ; E]$ とは次のような２部グラフである。

(i) 各節点 $x_i \in X$ $(i=1,2,\cdots,n)$ は W の第i行に対応し，各節点 $y_j \in Y$ $(j=1,2,\cdots,n)$ は W の第j列に対応する。

(ii) W の各非零要素 $w_{ij}\neq 0$ には，x_i と y_j を結ぶ枝 $(x_i, y_j)\in E$ が対応する。（定義終）

［例２．１］ 行列 W に対する二値化行列 $B(W)$ が次式で与えられているとする。ただし式中‘・’は０を表わす。

$$B(W)=\begin{array}{c|ccccc} & 1 & 2 & 3 & 4 & 5 \\ 1 & 1 & \cdot & 1 & \cdot & 1 \\ 2 & \cdot & 1 & \cdot & \cdot & \cdot \\ 3 & 1 & \cdot & \cdot & 1 & \cdot \\ 4 & 1 & 1 & \cdot & \cdot & \cdot \\ 5 & \cdot & 1 & \cdot & \cdot & 1 \end{array} \ , \qquad (2\cdot 3)$$

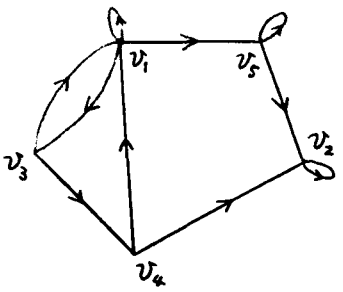
このとき，$G_0(W)$ および $G_b(W)$ はそれぞれ図２・１の(a)および(b)で示される。（例終）

定義２・２から明らかなように，有向グラフ $G_0(W)$ の隣接行列は $B(W)$ で与えられる。一方，$B(W)$ が対称行列であるとき次のような無向グラフが定義される。

[定義2.4]　$B(W)$が対称行列であるようなn次正方行列$W=[w_{ij}]$に付随する無向グラフ$G_n(W)=[V, E]$とは次のような無向グラフである。

(i) 各節点 $v_i \in V$ $(i=1,2,\cdots,n)$ は W の第i列に対応する。

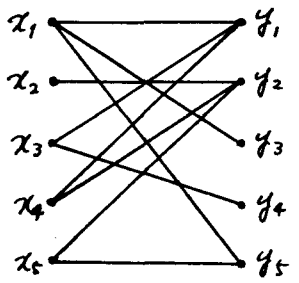
(ii) W の各非零要素 $w_{ij}\neq 0$ $(i \geq j)$ には，v_iとv_jを結ぶ（無向）枝 $(v_i, v_j) \in E$ が対応する。（定義終）



(a) $G_0(W)$

2.2.2　行列の分割

疎大行列をある種の形の分割（例えば図2·2）を目標としてその零部分行列（図2·2における'0'と書かれた部分行列）の範囲が最大となるようにその行や列を入れ替える手順に関する研究がいくつかなされている。



(b) $G_b(W)$

図2·1　行列Wに対するグラフ表現

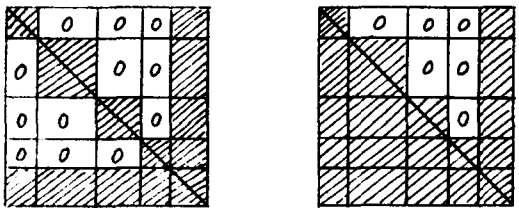

(a)　(b)

図2·2　行列の分割
（斜線部はそこに非零要素を含むことを示す。）

[例2.2]　例2·1の行列$B(W)$について考える。いま二つの置換行列，

$$P=\begin{array}{c|ccccc} & 1 & 2 & 3 & 4 & 5 \\ 1 & \cdot & 1 & \cdot & \cdot & \cdot \\ 2 & \cdot & \cdot & \cdot & 1 & \cdot \\ 3 & \cdot & \cdot & 1 & \cdot & \cdot \\ 4 & \cdot & \cdot & \cdot & \cdot & 1 \\ 5 & 1 & \cdot & \cdot & \cdot & \cdot \end{array}, \quad Q=\begin{array}{c|ccccc} & 1 & 2 & 3 & 4 & 5 \\ 1 & \cdot & 1 & \cdot & \cdot & \cdot \\ 2 & 1 & \cdot & \cdot & \cdot & \cdot \\ 3 & \cdot & \cdot & \cdot & \cdot & 1 \\ 4 & \cdot & \cdot & 1 & \cdot & \cdot \\ 5 & \cdot & \cdot & \cdot & 1 & \cdot \end{array}, \quad (2\cdot4)$$

を用いて $W^*=PWQ$ とするとき $B(W^*)$ は次のような行列になる。

$$B(W^*) = \begin{array}{c|ccccc} & 2 & 1 & 4 & 5 & 3 \\ 2 & 1 & \cdot & \cdot & \cdot & \cdot \\ 4 & 1 & 1 & \cdot & \cdot & \cdot \\ 3 & \cdot & 1 & 1 & \cdot & \cdot \\ 5 & 1 & \cdot & \cdot & 1 & \cdot \\ 1 & \cdot & 1 & \cdot & 1 & 1 \end{array} . \qquad (2\cdot5)$$

与えられた連立方程式 $Wx=b$ に式(2·4)の置換行列を用いてこれと等価な連立方程式 $W^*x^*=b^*$ を得る。ただし,

$$x^* = Q^{-1}x = Q^t x = [x_2\ x_1\ x_4\ x_5\ x_3]^t,$$
$$b^* = Pb = [b_2\ b_4\ b_3\ b_5\ b_1]^t, \qquad (2\cdot6)$$

(注)

である。方程式 $W^*x^*=b^*$ の解は,その係数行列 W^* の構造から容易に求められる。(例終)

この例で得られた行列(2·5)は,図2·2(b)の特別な場合と考えられる。また,置換行列 P および Q は,それぞれ W の行および列を入れ替えるための行列であって, $Q=P^t$ のときは行および列を同時に入れ替えることに相当する。以下で述べる手法はすべて $Q=P^t$ なる置換における行列の分割を議論し,与えられた行列の対角要素はすべて非零であると仮定している。

有吉-白川-尾崎[9]は,与えられた回路網の原始アドミタンス行列 Y に関する端子対縮約(2.3.2参照)を行なう問題に対して,行列 Y を図2·2(a)のように分割する手法を考察している。同図のように行列が分割されたとき,求める縮約行列 Y_0 はブロックごとの縮約操作を順次行なうことで得られ,このとき零部分行列に新たに非零要素を生じることはない。この分割は,与えられた回路網に対応する無向グラフを堅連結な部分グラフに分割する問題に帰着され,[9]ではその具体的な分割アルゴリズムを与えている。

一方,これとほぼ双対な関係にある分割法を戸川-白川-尾崎[10]が考察している。これは原始インピーダンス行列 Z に関する端子対縮約を取扱っており,無向グラフを2つの枝から成るカットセットで分割する問題に帰着している。

図2·2(b)の特別な場合として次の分割を考える。n次正方行列 W を,

$$W = \begin{array}{|c:c|} \hline W_{11} & W_{12} \\ \hdashline W_{21} & W_{22} \\ \hline \end{array} \quad \begin{matrix} \leftarrow n-k \\ \leftarrow k \end{matrix}$$
$$\begin{matrix} \uparrow & \uparrow \\ n-k & k \end{matrix} \qquad , \qquad (2\cdot7)$$

と分割したとき,部分行列 W_{11} が三角行列であるとき W は準三角行列(bor-

(注) ^{t} は転置を示す。

dered triangular matrix）という。ただし，n次正方行列 $T=[t_{ij}]$ において，$t_{ij}=0$ $(1\leq j<i\leq n)$ のとき T は上三角行列であるといい，$t_{ij}=0$ $(1\leq i<j\leq n)$ のとき T は下三角行列であるという。また，両者を総称して三角行列という。

Cheung-Kuh [12] は，与えられた行列の行および列を入れ替えて PWP^t を準三角行列にするとき，kの値（式(2·7)の W_{22} の次数）を最小にする置換行列 P を求める問題を考察している。係数行列が準三角行列であれば，適当な代入操作を繰返すことでn個の未知数に関する方程式をk個の未知数に関する方程式に変換することができる。この問題は，行列 W に付随する有向グラフ $G_0(W)$ から自己閉路を取除いたグラフにおける最小帰還節点集合(注)を求める問題に帰着され，[12]ではその近似解を得る新しいアルゴリズムを提案している。この有向グラフの最小帰還節点（枝）集合を求める問題は，疎大行列の取扱いとは独立に従来から研究されており（例えば[13],[14]），現在のところ問題の規模n（節点あるいは枝の個数）の多項式のオーダの手数で最適解を求めるアルゴリズムは知られていない[15]。

2.2.3 ピボット操作の最適順序づけ

連立一次方程式 $Wx=b$ の解は $x=W^{-1}b$ であるが，実際の数値計算では逆行列 W^{-1} を計算する必要はなく，行列 W を下三角行列 L と上三角行列 U の積の形に分解すればよい。すなわち次の手順で解が求められる。

(i) $W=LU$ なる三角行列 L および U を求める（LU分解；LU decomposition）。

(ii) $Ly=b$ を解く（前進消去；forward elimination）。

(iii) $Ux=y$ を解く（後退代入；backward substitution）。

連立一次方程式の解法として逆行列を求めないのは，必要な演算量，解の精度などの点でLU分解の方が勝っていることによるが，行列 W のスパース性を活用する観点からもLU分解は有効である。すなわち，W が疎大行列であっても W^{-1} は密行列（ほとんどの要素が非零である行列）であることが多いが，LU分解を行なうときはそれほど非零要素の個数は増加しないという性質がある。

LU分解については2.3.1で詳しく述べるが，ここではLU分解の過程におけるピボット操作の最適順序づけの問題に対する二·三の接近法を概説する。

LU分解は，基本的にはGauss消去法であり，ある非零要素 w_{ij} を選び，第j列の第(i,j)要素以外の要素がすべて零になるように各行を操作すること

(注) 有向グラフの有向閉路を消滅させるために，取除くべき最小個の節点（枝）の集合をそのグラフの最小帰還節点（枝）集合という。

を繰返すものである。このとき選ばれた w_{ij} をピボット要素といい，この操作をピボット操作という。また，ピボット操作の過程における新しい非零要素をフィルイン（*fill-in*）という。

解の数値的安定性などの理由から，ピボット要素の候補として行列の対角要素から選ぶことが多く，この場合グラフ表現としては定義2・1(注1)あるいは定義2・3が用いられ，最適順序づけ問題への接近法として，各段階で発生するフィルインの個数を最小にする対角要素を順次選ぶ方法[5],[7],[8]が提案されている。2.3節以下ではこの観点から，二段階後のフィルインの個数までを考慮した新しい手法を考察する。

また，ピボット要素の候補として行列のすべての非零要素を考える立場からの接近法がある。この場合は定義2・2の2部グラフで行列を表現することが多く，そのヒューリスティックな手法がいくつか提案されている[16],[17]。

一方，LU分解の全過程で生じるフィルインの総数を最小にしようとする大局的立場からの研究がある。特に，対称行列 W においてフィルインの総数が0であるような操作順序が存在するための必要十分条件は，$G_n(W)$ が三角化グラフ(注2)（*triangulated graph*）であることが知られている[18]。したがって最適順序づけの問題は，$G_n(W)$ に最小個の枝を加えて三角化グラフにする問題と解釈されるが，真の最適解を能率よく求めるアルゴリズムは未だ知られていない。

2.3 最適順序決定問題

2.3.1 LU分解

以下で考察する正方行列の対角要素はすべて非零であると仮定する。

正方行列 W のLU分解とは，W を下三角行列 L と上三角行列 U の積の形に分解することである。ただし，L は各対角要素がすべて1であるような下三角行列である。ピボット要素として W の各対角要素 $w_{11}, w_{22}, \cdots, w_{nn}$ をこの順序で選ぶものとすれば，LU分解は $W=[w_{ij}^{(0)}]$ を初期値とし，$k=1,2,\cdots,n$ に対して次の操作を順次行なえばよい。

$$u_{kj} = w_{kj}^{(k-1)} \qquad (j=k,k+1,\cdots,n), \tag{2・8a}$$

(注1) 対角要素はすべて非零であるとの仮定から，$G_0(W)$ の自己閉路を取除いた有向グラフを用いることが多い。

(注2) 無向グラフの長さが4以上の閉路において連続していない二つの節点の対を結ぶ枝をこの閉路の弦（chord）という。任意の長さ4以上の閉路が少なくとも一つの弦を含むとき，そのグラフは三角化グラフであるという。

$$l_{ik} = w_{ik}^{(k-1)} / u_{kk} \quad (i = k+1, k+2, \cdots, n), \tag{2·8b}$$

$$w_{ij}^{(k)} = w_{ij}^{(k-1)} - l_{ik} u_{kj} \quad (i, j = k+1, k+2, \cdots, n), \tag{2·8c}$$

この操作から得られた各 l_{ik}, u_{jk} の値に対して，$L = [l_{ik}]$ ($l_{ik} = 0$, $i \leq k$) および $U = [u_{kj}]$ ($u_{kj} = 0$, $k > j$) が求める行列である。式(2·8c)の操作をLU分解における段階 k のピボット操作と呼ぶ。

［例2·3］　次の行列 W のLU分解を考える。

$$W = \begin{bmatrix} 2 & 2 & 1 & 4 \\ 1 & 2 & 0 & 0 \\ -2 & 0 & 1 & 0 \\ 3 & 0 & 0 & 2 \end{bmatrix}, \tag{2·9}$$

この W に対して式(2·8)を用いてLU分解すれば次式を得る。

$$L = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1/2 & 1 & 0 & 0 \\ -1 & 2 & 1 & 0 \\ 3/2 & -3 & -1 & 1 \end{bmatrix}, \quad U = \begin{bmatrix} 2 & 2 & 1 & 4 \\ 0 & 1 & -1/2 & -2 \\ 0 & 0 & 3 & 8 \\ 0 & 0 & 0 & -2 \end{bmatrix}. \tag{2·10}$$

一方，置換行列，

$$P = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}, \tag{2·11}$$

を用いて，

$$W^* = PWP^t = \begin{bmatrix} 2 & 0 & 0 & 3 \\ 0 & 1 & 0 & -2 \\ 0 & 0 & 2 & 1 \\ 4 & 1 & 2 & 2 \end{bmatrix}, \tag{2·12}$$

とし，この W^* に対してLU分解を行えば次式が得られる。

$$L^* = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 2 & 1 & 1 & 1 \end{bmatrix} , \quad U^* = \begin{bmatrix} 2 & 0 & 0 & 3 \\ 0 & 1 & 0 & -2 \\ 0 & 0 & 2 & 1 \\ 0 & 0 & 0 & -3 \end{bmatrix} . \qquad (2\cdot13)$$

行列 W のスパース構造 $B(W)$ と得られた行列のスパース構造 $B(L+U)$ を比較すれば，LU分解の過程におけるスパース性の保存の程度を知ることができる。この例題においては，$B(W^*) = B(L^*+U^*)$ であるが，$B(W) \neq B(L+U)$ となり，W にそのまま式(2・8)を適用すればスパース性は全く保存されない。このことは，ピボット要素の選び方によってスパース性の保存の程度が異なることを意味し，ピボット操作の順序づけの問題が重要であることを示している。（例終）

前節で述べたように，$B(W^*) = B(L^*+U^*)$ となるような置換行列 P が存在するためには $G_n(W)$ が三角化グラフのときのみであって，一般にはこの等式が成立する置換行列が存在するとは限らない。そこで $B(L^*+U^*) - B(W^*)$ の非零要素の個数を最小にする置換行列を求めることが必要になる。

2.3.2 端子対縮約

方程式 $Wx = b$ において $b = [b_1\ b_2 \cdots b_p\ 0\ 0 \cdots 0]^t$ であるとき，原方程式から $\hat{x} = [x_1\ x_2 \cdots x_p]^t$ および $\hat{b} = [b_1\ b_2 \cdots b_p]^t$ に関する縮約方程式 $\hat{W}\hat{x} = \hat{b}$ の係数行列 $\hat{W}$ を求めることがしばしば要求される。

例えば，回路網の指定された p 個の端子対に関するインミタンス行列を求める場合，それらの端子対を含む原始インミタンス行列[11] W を求め，この W から上記の p 次正方行列 $\hat{W}$ を求めればよい。

いま原方程式を次のように分割する。

$$\begin{array}{c} \alpha \\ \beta \end{array} \left[\begin{array}{c|c} W_{11} & W_{12} \\ \hline W_{21} & W_{22} \end{array} \right] \left[\begin{array}{c} x_1 \\ \hline x_2 \end{array} \right] = \left[\begin{array}{c} b_1 \\ \hline b_2 \end{array} \right] \qquad (2\cdot14)$$
$$\qquad \alpha \qquad \beta$$

ここで，$b_2 = 0$ かつ $\det W_{22} \neq 0$ であるとき，上式から次の方程式が導びかれる。

$$W^* x_1 = b_1 , \qquad \text{ただし} \quad W^* = W_{11} - W_{12} W_{22}^{-1} W_{21} . \qquad (2\cdot15)$$

この式において $\beta = 1$ の場合を考えれば $W^* = [w_{ij}^*]$ の各要素は，$W = [w_{ij}]$ として次式で得られる。

$$w_{ij}^{*} = w_{ij} - w_{in} \cdot w_{nj} / w_{nn} \quad (i,j = 1,2,\cdots n-1). \qquad (2\cdot16)$$

以上の考察から，$\hat{W}$ を求めるには $W = [w_{ij}^{(n)}]$ を初期値とし，$k = n, n-1, \cdots, p+1$ に対して次の操作を順次行なえばよい。

$$w_{ij}^{(k-1)} = w_{ij}^{(k)} - w_{ik}^{(k)} \cdot w_{kj}^{(k)} / w_{kk}^{(k)} \qquad (2\cdot17)$$
$$(i,j = 1,2,\cdots,k-1).$$

式（2·17）の結果から，求める行列 $\hat{W}$ は，$\hat{W} = [w_{ij}^{(p)}]$ として得られる。この式の操作を変数 x_k に関する端子対縮約操作という。

2.3.3 最適順序決定問題

前述のように，LU分解において適当な置換行列で行および列の入れ替えを行なえば，スパース性の保存に関して有効である。しかも，その置換行列に対応して変数ベクトルおよび定数ベクトルを置換すれば，求める解は同一であることは自明である。このことは，端子対縮約についても同様である。一方，LU分解および端子対縮約において零要素が非零要素に変わる（すなわちフィルインが生じる）のは，それぞれ式（2·8C）および（2·17）のピボット操作および端子対縮約操作においてのみであり，それらはそれぞれ $l_{ik} u_{kj} \neq 0$ および $w_{ik}^{(k)} \cdot w_{kj}^{(k)} \neq 0$ のときである。これらは同じ意味を持つことから，スパース性のみに着目したとき，LU分解と端子対縮約とは全く同様の操作であるということができる。

［定義2·5］ m次正方行列 W に対して，その行 k および列 k に関してなんらかの操作を行なってその行および列を取り去って得られた $(m-1)$ 次正方行列を W^{*k} とする。このとき $B(W) = [b_{ij}]$ と $B(W^{*k}) = [b_{ij}^{*k}]$ との間に，

$$b_{ij}^{*k} = b_{ij} + b_{ik} \cdot b_{kj} \quad (i,j \neq k), \qquad (2\cdot18)$$

なる関係が成立するとき，この操作を k に関する広義のピボット操作という。ただし，ここでの加算および乗算は2を法とする演算である。（定義終）

先に述べた二つの操作のいずれもは，明らかにこの広義のピボット操作の一つであり，以下では特に断わらないかぎりこの広義のピボット操作を単にピボット操作ということにする。

［定義2·6］ k に関するピボット操作によるフィルインに対応する集合 F_k を次のように定義する。

$$F_k = \{(i,j) \mid b_{ij} = 0 \ , \ b_{ij}^{*k} = 1 \}. \qquad (2\cdot19)$$

（定義終）

この集合 F_k を求めるために，行列 W に対して次のような集合を定義する。

[定義2・7] 正方行列 W の列 k および行 k に対して，それぞれ次のような集合 A_k および B_k を対応させる。ただし $B(W)=[b_{ij}]$ とする。

$$A_k=\{\,i \mid b_{ik}=1\ ,\ i\neq k\,\}, \tag{2・20}$$

$$B_k=\{\,j \mid b_{kj}=1\ ,\ j\neq k\,\}. \tag{2・21}$$

(定義終)

[定義2・8] 正方行列 W において，集合 B_k の各元 j に対して次の集合 L_{kj} を対応させる。

$$\begin{aligned} L_{kj}&=\{\,i \mid j\notin B_i\ ,\ i\in A_k\ ,\ i\neq j\,\}\\ &=A_k-[\{j\}\cup A_j]\quad (j\in B_k)\ . \end{aligned} \tag{2・22}$$

(定義終)

$B(W)$ が対称行列であれば，すべての k について $A_k=B_k$ であり，$i\in A_j$ のときかつそのときにかぎり $j\in B_i$ であることは定義より明らかであろう。

[補題2・1] 正方行列 W において次式が成立する。

$$F_k=\{\,(i,j) \mid i\in L_{kj}\,\}. \tag{2・23}$$

(証明) $i\in L_{kj}$ とする。L_{kj} の定義から $b_{ij}=0$，$b_{ik}=b_{kj}=1$ であり，このとき式(2・18)から $b_{ij}^{*k}=1$ である。すなわち，

$$\{\,(i,j) \mid i\in L_{kj}\,\}\subset\{\,(i,j) \mid b_{ij}=0\ ,\ b_{ij}^{*k}=1\,\}, \tag{2・24}$$

が成立する。逆に，$b_{ij}=0$，$b_{ij}^{*k}=1$ ならば式(2・18)から $b_{ik}\cdot b_{kj}=1$ すなわち $b_{ik}=b_{kj}=1$ である。$b_{kj}=1$ であるから L_{kj} が定義されていて $i\in L_{kj}$ である。故に式(2・24)の逆の包含関係が成立し補題が証明される。

(証明終)

[定義2・9] 正方行列 W において，k に関するピボット操作の結果生じるフィルインの個数を $\delta_k(W)$ で表わす。すなわち，

$$\begin{aligned} \delta_k(W)&=|F_k|\\ &=\sum_{j\in B_k}|L_{kj}|, \end{aligned} \tag{2・25}$$

とする。ただし，$|S|$ は集合 S の元の個数を意味する。(定義終)

行列 W の n 個の行および列のうち，ピボット操作の行なわれる行および列の個数を g とする。すなわち，LU分解においては $g=n$ であり，端子対縮約においては $g=n-p$ (ただし p は2・3・2における縮約行列 $\hat{W}$ の次数である)である。このとき，一般性を失うことなくこれらは第1行(列)から第 g 行(列)

にあると仮定する。順次g回のピボット操作を行なう順序は全部で$g!$通りある。いま，g次対称群S_gの一つの元に対応する次のような系列σの集合Σを考える。

$$\Sigma=\{\sigma=(k_1,k_2,\cdots,k_g)\mid \begin{pmatrix}1&2&\cdots&g\\k_1&k_2&\cdots&k_g\end{pmatrix}\in S_g\}. \qquad (2\cdot26)$$

このとき，各$\sigma=(k_1,k_2,\cdots,k_g)\in\Sigma$は一つのピボット操作の順序を定める。この$\sigma$に対して，置換行列$P_\sigma=[p_{ij}]$を次のように定める。

$$p_{ij}=\begin{cases}1 & ; j=k_i,\\ 0 & ; j\neq k_i.\end{cases} \qquad (2\cdot27)$$

操作順序σに対して，LU分解を行なうときは，

$$W'=P_\sigma W P_\sigma^t, \qquad (2\cdot28)$$

なる行列W'に式(2·8)を順次適用すればよく，端子対縮約のときは，

$$\sigma^*=(g+1,g+2,\cdots,n,k_g,k_{g-1},\cdots,k_1), \qquad (2\cdot29)$$

として，

$$W^*=P_{\sigma^*}W P_{\sigma^*}^t, \qquad (2\cdot30)$$

なる行列W^*に式(2·17)を順次適用すればよい。

$\sigma\in\Sigma$なる$\sigma=(k_1,k_2,\cdots,k_g)$に対して$\sigma$の部分系列$\sigma(i)$を，

$$\sigma(i)=(k_1,k_2,\cdots,k_i)\quad(i=1,2,\cdots,g-1), \qquad (2\cdot31)$$

とし，$\sigma(0)=\lambda$ (null sequence)，$\sigma(g)=\sigma$とする。ある$\sigma\in\Sigma$に対応する順序で正方行列Wにピボット操作を順次行なうとき，第i段階までの操作が完了したとき得られる行列を$W^{*\sigma(i)}$と書く。すなわち，

$$\begin{cases}W^{*\sigma(0)}=W,\\ W^{*\sigma(i)}=(W^{*\sigma(i-1)})^{*k_i},\end{cases} \qquad (2\cdot32)$$

とする。

[定義 2·10] 正方行列Wの一つの操作順序$\sigma=(k_1,k_2,\cdots,k_g)\in\Sigma$に対して，その操作順序の全過程でのフィルインの総数を$\Delta_W(\sigma)$と書く。すなわち，

$$\Delta_W(\sigma)=\sum_{i=1}^{g}\delta_{k_i}(W^{*\sigma(i-1)}), \qquad (2\cdot33)$$

である。特に行列Wを明示する必要のないとき単に$\Delta(\sigma)$と書く。(定義終)

以上のことから，最適順序決定問題は次に定義する最適順序σ^*を求める問題とみなすことができる。

［定義 2·11］ 次式を満足する操作順序 σ^* を行列 W のピボット操作の最適順序という。

$$\Delta_W(\sigma^*) = \min_{\sigma \in \Sigma} \Delta_W(\sigma). \qquad (2\cdot34)$$

（定義終）

すなわち，最適順序 σ^* は最もスパース性を保存するようなピボット操作順序である。ただし，このような σ^* はただ一つであるとは限らない。

2.4 p-グラフ

2.4.1 p-グラフ

［定義 2·12］ m 次正方行列 W の p-グラフ（pivoting graph）$G_p(W) = [V, E]$ とは次のような重みをもつ有向グラフである。

(i) 各節点 $v_i \in V$ $(i=1,2,\cdots,m)$ は W の第 i 列に対応する。

(ii) W の各非零要素 $w_{ij} \neq 0$ $(i \neq j)$ には，v_i から v_j に向かう（有向）枝 $(v_i, v_j) \in E$ が対応する。

(iii) 各 $v_i \in V$ および各 $(v_i, v_j) \in E$ に，それぞれ集合 A_i および集合 L_{ij} が重みとして与えられる。（定義終）

この定義による $G_p(W)$ は定義 2·2 の $G_0(W)$ に類似しており，相異点は，$G_p(W)$ は自己閉路を含まないこと，および節点と枝にそれぞれ集合が重みとして与えられていること，の二点である。このうち前者は，W の対角要素はすべて非零であると仮定しているため自己閉路の有無が W に関する情報を何らもたらさないことに起因している。

［例 2·4］ 行列 W の二値化行列 $B(W)$ が次式で与えられているとする。

$$B(W) = \begin{array}{c|ccccc} & 1 & 2 & 3 & 4 & 5 \\ \hline 1 & 1 & 1 & 1 & \cdot & \cdot \\ 2 & 1 & 1 & \cdot & 1 & \cdot \\ 3 & 1 & 1 & 1 & \cdot & 1 \\ 4 & \cdot & 1 & \cdot & 1 & \cdot \\ 5 & \cdot & \cdot & 1 & 1 & 1 \end{array} . \qquad (2\cdot35)$$

この W の p-グラフ $G_p(W)$ は図 2·3 で示される。ただし図中では A_i および L_{ij} を，それぞれ［·］および｛·｝で表わしている。（例終）

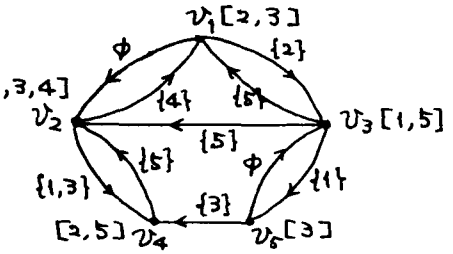


図 2·3 p-グラフ $G_p(W)$

2.4.2 S-ピボット操作

行列 W の p-グラフを $G_p(W)=[V,E]$ とし，W^{*k} の p-グラフを $G_p(W^{*k})=[V^{*k},E^{*k}]$ と表わす。まず $G_p(W)$ から $G_p(W^{*k})$ が次の補題で求められることを示そう。

[補題 2·2] $G_p(W^{*k})=[V^{*k},E^{*k}]$ は $G_p(W)=[V,E]$ に対して次の式で与えられる。

$$V^{*k}=V-\{v_k\}, \tag{2·36}$$

$$E^{*k}=[E-(\{(v_k,v_h)\mid h\in B_k\}\cup\{(v_l,v_k)\mid l\in A_k\})] \cup[\cup_{i\in B_k}H_i], \tag{2·37}$$

ただし，

$$H_i=\begin{cases}\phi & ; L_{ki}=\phi,\\ \{(v_j,v_i)\mid j\in L_{ki}\} & ; L_{ki}\neq\phi,\end{cases} \tag{2·38}$$

である。

(証明) k に関してピボット操作を行なうのであるから，式 (2·36) は明らかである。一方，ピボット操作によって $b_{ij}=1$ であって $b_{ij}^{*k}=0$ (ただし $i,j\neq k$) ということは起こり得ないこと，および W^{*k} では行 k および列 k が取除かれているため $G_p(W^{*k})$ には (v_k,v_h) および (v_l,v_k) などの枝は存在していないことから，

$$E^{*k}\supset E-(\{(v_k,v_h)\mid h\in B_k\}\cup\{(v_l,v_k)\mid l\in A_k\})$$

は明らかに成立する。さらに，E には含まれない枝で E^{*k} に新らたに加えられるべき枝は，W における k に関するピボット操作によるフィルインに対応する枝であり，それらの枝のうち節点 v_i へ向かう枝の集合が式 (2·38) の H_i であるから，結局式 (2·37) が成立する。(証明終)

[補題 2·3] W^{*k} における各集合 A_i^{*k}，B_i^{*k} および L_{ij}^{*k} は，W における各集合 A_i および B_i から次式で求められる。

$$A_i^{*k}=\begin{cases}A_i\cup A_k-\{i,k\} & ; i\in B_k,\\ A_i & ; i\notin B_k,\end{cases} \tag{2·39}$$

$$B_i^{*k}=\begin{cases}B_i\cup B_k-\{i,k\} & ; i\in A_k,\\ B_i & ; i\notin A_k,\end{cases} \tag{2·40}$$

$$L_{ij}^{*k}=A_i^{*k}-[\{j\}\cup A_j^{*k}]. \tag{2·41}$$

（証明）　式(2·39)を証明する。$B(W)=[b_{ij}]$とすれば，$i\notin B_k$のとき定義より$b_{ki}=0$であるからkに関するピボット操作によってW^{*k}の(j,i)要素はすべてのj$(j\neq k)$に対して$b_{ji}^{*k}=b_{ji}+b_{jk}\cdot b_{ki}=b_{ji}$であるから$A_i^{*k}=A_i$である。一方，$i\in B_k$のときは$b_{ki}=1$であるから$b_{ji}^{*k}=b_{ji}+b_{jk}$となる。これは，$A_i^{*k}=A_i\cup A_k-\{i,k\}$と等価であり式(2·39)が成立する。

式(2·40)は，式(2·39)と同様に証明することができ，式(2·41)は定義から明らかである。（証明終）

［定義2·13］　p－グラフ$G_p(W)$から$G_p(W^{*k})$を作ることを，節点v_kに関するs－ピボット操作という。（定義終）

［例2·5］　例2·4のp－グラフ$G_0=G_p(W)$（図2·3）に節点v_1に関するs－ピボット操作を行なえば，図2·4に示されるp－グラフ　$G_0^{*1}=G_p(W^{*1})$が得られる。
（例終）

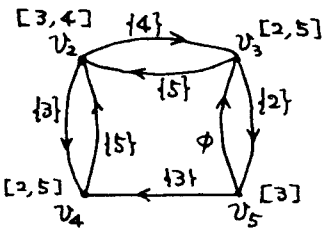


図2·4　p－グラフ$G_p(W^{*1})$

以下では，与えられた行列Wに対して，そのp－グラフ$G_p(W)$を単にG_0あるいはGと書くことがある。また，一つのピボット操作の順序$\sigma=(k_1,k_2,\cdots,k_g)\in\Sigma$の部分系列$\sigma(i)=(k_1,k_2,\cdots,k_i)$に対して$W^{*\sigma(i)}$のp－グラフ$G_p(W^{*\sigma(i)})$を単に$G_0^{*\sigma(i)}$あるいは$G^{*\sigma(i)}$と書くことがある。特に$W^{*k}$のp－グラフを$G^{*k}$と書く。さらに，行列Wに対して定義されたkに関するピボット操作の結果生じるフィルインの個数$\delta_k(W)$――定義2·9――および操作順序$\sigma\in\Sigma$の全過程でのフィルインの総数$\Delta_W(\sigma)$――定義2·10――なる記号は，これらと同じ意味でWのp－グラフGに対してそれぞれ$\delta_k(G)$および$\Delta_G(\sigma)$なる記号で用いる。すなわち，

$$\delta_k(G)=\Sigma_{j\in B_k}|L_{kj}|\,,\tag{2·42}$$

$$\Delta_G(\sigma)=\Sigma_{i=1}^{g}\,\delta_{k_i}(G^{*\sigma(i-1)})\,,\tag{2·43}$$

である。

［補題2·4］　任意の$h\in B_i$に対して次式が成立する。ただしh≠kとする。

(i)　$h\in B_k$のとき：　$L_{ih}^{*k}\subset L_{ih}$，　(2·44)

(ii)　$h\notin B_k$，$i\in B_k$のとき：　$L_{ih}^{*k}-L_{ih}\subset L_{ki}$，　(2·45)

(iii)　$h\notin B_k$，$i\notin B_k$のとき：　$L_{ih}^{*k}=L_{ih}$．　(2·46)

（証明）(i)　$j\in L_{ih}^{*k}$なる任意のjを考える。jは明らかにi，k，hのいずれでもない。$j\in L_{ih}^{*k}=A_i^{*k}-[\{h\}\cup A_h^{*k}]$および$h\in B_k$なる仮定から，$j\notin A_h^{*k}=A_h\cup A_k-\{h,k\}$であり，次の＝式が成立する。

(a) $j \notin A_h$, (b) $j \notin A_k$.

また $j \in A_i^{*k}$ であるが，(b)を考慮すれば $i \in B_k$ および $j \notin B_k$ のいずれの場合でも $j \in A_i$ である。このことと(a)より $j \in A_i - [\{h\} \cup A_h] = L_{ih}$ となり式(2・44)が証明される。

(ii) $j \in L_{ih}^{*k} - L_{ih}$ なる任意の j を考えれば，明らかに j は i，k，h のいずれでもない。$j \in L_{ih}^{*k} = A_i^{*k} - [\{h\} \cup A_h^{*k}]$ であり仮定より，

(a) $j \in A_i^{*k} = A_i \cup A_k - \{i, k\}$, (b) $j \notin A_h^{*k} = A_h$,

が成立する。さらに $j \notin L_{ih} = A_i - [\{h\} \cup A_h]$ であり，(b)を考慮すれば $j \notin A_i$ となる。このことと(a)より $j \in A_k$ であり $j \in A_k - [\{i\} \cup A_i] = L_{ki}$ を得る。

(iii) 仮定より $A_h^{*k} = A_h$ および $A_i^{*k} = A_i$ であるから式(2・46)は明らかである。(証明終)

[補題2・5] $i \in L_{kh}$ であるとき次式が成立する。

$$L_{ih}^{*k} \subset L_{ik} . \qquad (2\cdot47)$$

(証明) $i \in L_{kh} = A_k - [\{h\} \cup A_h]$ であるから $i \in A_k$ すなわち $k \in B_i$ であって L_{ik} が定義されている。さらに，$h \in B_k$ であることから $A_h^{*k} = A_h \cup A_k - \{h, k\}$ である。いま $j \in L_{ih}^{*k}$ なる任意の j を考えれば，明らかに j は i，k，h のいずれでもない。さらに L_{ih}^{*k} の定義より $j \in A_i^{*k}$ かつ $j \notin A_h^{*k}$ である。後者より $j \notin A_k$ であり，前者に $j \notin A_k$ および式(2・39)を用いれば $j \in A_i$ が成立し，$j \in A_i - [\{h\} \cup A_k] = L_{ik}$ が得られる。したがって式(2・47)が証明される。(証明終)

これらの補題を用いて次の定理が証明される。

[定理2・1] p-グラフ G において $\delta_i(G) = 0$ であれば，$k \neq i$ なる任意の k に対して $\delta_i(G^{*k}) = 0$ である。

(証明) この定理を証明するためには，すべての $h \in B_i$ に対して $L_{ih} = \phi$ であるような i については，G^{*k} においてすべての $h \in B_i^{*k}$ に対しても $L_{ih}^{*k} = \phi$ であることを示せばよい。

$h \notin B_i$ かつ $h \in B_i^{*k}$ なる h については $i \in L_{kh}$ であるから補題2・5に $L_{ik} = \phi$ なる仮定を用いれば $L_{ih}^{*k} = \phi$ となる。

一方 $h \in B_i$ のとき，もし $h \notin B_k$ であれば $i \notin B_k$ である。なぜなら，$i \in B_k$ とすれば $k \notin A_h$ かつ $k \in A_i$ であるから $k \in A_i - [\{h\} \cup A_h] = L_{ih}$ となって $L_{ih} = \phi$ の仮定に反する。したがって，補題2・4において(ii)の場合は起こり得ない。さらに，(i)および(iii)の場合については $L_{ih} = \phi$ であることを用いれば $L_{ih}^{*k} = \phi$ が導かれる。(証明終)

p-グラフ G_0 において，s-ピボット操作の行なわれるべき g 個の節点の集合を V_0 とする。$V_0 = \{v_1, v_2, \cdots, v_g\}$ として一般性を失わない。定理

2·1を用いて，次の三つの性質を満足する系列 $\sigma_2=(k_1,k_2,\cdots,k_n)$ を考える。

(i) $v_{k_i}\in V_0$ $(i=1,2,\cdots,n)$, (2·48)

(ii) $\delta_{k_i}(G_0^{*\sigma_2(i-1)})=0$ $(i=1,2,\cdots,n)$, (2·49)

(iii) 任意の $v_j\in V(G_0^{*\sigma_2})\cap V_0$ に対して，

$$\delta_j(G_0^{*\sigma_2})\neq 0. \tag{2·50}$$

ただし，$V(G^{*\sigma_2})$ は p-グラフ $G^{*\sigma_2}$ の節点集合である。

このような系列 σ_2 は次のアルゴリズムで求められる。

［アルゴリズムA］

操作1： $G\leftarrow G_0$, $V\leftarrow V_0$, $\pi\leftarrow\lambda$ (null sequence) とする。

操作2： 集合 $V_1=\{v_k\mid \delta_k(G)=0, v_k\in V\}$ を求める。$V_1=\phi$ のとき操作4へ。$|V_1|=t$ のとき，各 $v_{k_i}\in V_1$ なる t 個の k_i の任意の系列を $\sigma=(k_1,k_2,\cdots,k_t)$ として操作3へ。

操作3： $G\leftarrow G^{*\sigma}$, $V\leftarrow V-V_1$, $\pi\leftarrow(\pi,\sigma)$ として操作2へ。

操作4： 操作完了．現在の系列 π が求める系列 σ_2 である。

（アルゴリズム終）

このアルゴリズムで求められた系列 σ_2 に関して次の定理が成立する。

［定理2·2］ アルゴリズムAで求められた系列を σ_2 とする。また G_0 の最適順序 σ^* の集合を $\Sigma^*=\{\sigma_i^*\}\subset\Sigma$ とする。このとき，σ_2 および $G_0^{*\sigma_2}$ の任意の最適順序 $\sigma_{j2}^{\dagger}$ を用いて得られる操作順序 $\sigma_j^{\dagger}=(\sigma_2,\sigma_{j2}^{\dagger})$ の集合 $\Sigma^{\dagger}=\{\sigma_j^{\dagger}\}$ は Σ^* の部分集合である。

(証明) p-グラフ G の節点集合を $V(G)$ と表わす。$\sigma_2=\lambda$ のとき定理は明らかであるから $\sigma_2\neq\lambda$ とし，$G_0^{*\sigma_2}=G_2$ とおく。p-グラフを有向グラフとみなしたとき，G_2 は節点集合 $V(G_2)$ から生成される G_0 のセクショングラフ(注)である。いま，$V_2=V(G_0)-V(G_2)$ とし，$Z=\{k\mid v_k\in V_2\}$ とする。G_2 の節点および枝の重みをそれぞれ $A_i^{*\sigma_2}$ および $L_{ij}^{*\sigma_2}$ と表わせば，

$$A_i^{*\sigma_2}=A_i-Z \quad (v_i\in V(G_2)),$$

$$L_{ij}^{*\sigma_2}=A_i^{*\sigma_2}-[\{j\}\cup A_j^{*\sigma_2}]$$

$$=L_{ij}-Z,$$

である。このことは，任意の系列 $\sigma=(\sigma_2,k_1,k_2,\cdots,k_l)$ $(k_i\in V_0\cap V(G_2),\ j=1,2,\cdots,l)$ についても成立する。すなわち，

(a) $L_{ij}^{*\sigma}=L_{ij}^{*(k_1,k_2,\cdots,k_l)}-Z$,

(注) 有向グラフ G の節点の部分集合 $V'\subset V$ から生成される G のセクショングラフとは，G から節点 $v_i\in V-V'$ および各 v_i に接続するすべての枝を取除いた部分グラフである。

である。ただし，$L_{ij}^{*\sigma}$ および $L_{ij}^{*(k_1,k_2,\cdots,k_g)}$ はそれぞれ p-グラフ $G_0^{*\sigma} = G_z^{*(k_1,k_2,\cdots,k_g)}$ および $G_0^{*(k_1,k_2,\cdots,k_g)}$ の枝 (v_i,v_j) の重みの集合である。

いま，p-グラフ G_0 の任意の操作順序 $\sigma=(k_1,k_2,\cdots,k_g)\in\Sigma$ に対して，σ の中から $k_i\in Z$ なる元を取り去った系列 $\hat{\sigma}=(k_1',k_2',\cdots,k_r')$ （ただし $r=|V(G_z)|$）を考えればこれは p-グラフ G_z の一つの操作順序である。さらに，

$$\Delta_{G_0}(\sigma) = \delta_{k_1}(G_0)+\delta_{k_2}(G_0^{*\sigma(1)})+\cdots + \delta_{k_j}(G_0^{*\sigma(j-1)})+\cdots+\delta_{k_g}(G_0^{*\sigma(g-1)}),$$

$$\Delta_{G_z}(\hat{\sigma}) = \delta_{k_1'}(G_z)+\delta_{k_2'}(G_z^{*\hat{\sigma}(1)})+\cdots + \delta_{k_i'}(G_z^{*\hat{\sigma}(i-1)})+\cdots+\delta_{k_r'}(G_z^{*\hat{\sigma}(r-1)}),$$

であるが，すべての k_i' について $k_i'=k_j$ なる j が存在し (a) から，

$$\delta_{k_i'}(G_z^{*\hat{\sigma}(i-1)}) = \delta_{k_i'}(G_0^{*(\sigma_z,\hat{\sigma}(i-1))}) \le \delta_{k_j}(G_0^{*\sigma(j-1)}),$$

が成立して，次式が得られる。

(b) $\quad \Delta_{G_z}(\hat{\sigma}) \le \Delta_{G_0}(\sigma)$.

さて，Σ^+ の任意の元を σ_j^+ とし，Σ^* の任意の元を σ_i^* とする。もし，

$$\Delta_{G_0}(\sigma_j^+) > \Delta_{G_0}(\sigma_i^*),$$

であれば，σ_i^* をうえに述べた方法で $\hat{\sigma}_i^*$ に変換したとき (b) から，

$$\Delta_{G_z}(\hat{\sigma}_i^*) \le \Delta_{G_0}(\sigma_i^*) < \Delta_{G_0}(\sigma_j^+) = \min_{\sigma_{jz}\in\Sigma_z}\Delta_{G_z}(\sigma_{jz}),$$

となりこれは矛盾である。ただし，Σ_z は G_z のすべての操作順序の集合である。したがって $\Delta_{G_0}(\sigma_j^+)\le\Delta_{G_0}(\sigma_i^*)$ であるが，$\Delta_{G_0}(\sigma_i^*)=\min_{\sigma\in\Sigma}\Delta_{G_0}(\sigma)$ であるから不等号は成立せず，結局 $\Delta_{G_0}(\sigma_j^+)=\Delta_{G_0}(\sigma_i^*)$ が得られ，$\sigma_j^+\in\Sigma^*$ が示される。（証明終）

この定理から，最適順序の一つ $\sigma^*\in\Sigma^*$ を求める問題は，Σ^+ の元の一つ $\sigma_j^+=(\sigma_z,\sigma_{jz}^+)$ における部分系列 σ_{jz}^+ を求める問題と考えることができる。

2.5 擬似最適順序

2.5.1 擬似最適解

p-グラフ G_0 について前節のアルゴリズムAで求めた系列を σ_2 とする。さらに，$G_2 = G_0^{*\sigma_2}$ とし，G_2 の節点集合 V_2 のなかで s-ピボット操作を行なうべき節点の集合を V_r $(= V_2 \cap V_0)$，$r = |V_r|$ とする。以下では，与えられた x の関数 $f(x)$ において，$x \in S$ なる条件の下で $f(x)$ の値を最小（または最大）にする x の集合を，$\underline{M}(x \in S)[f(x)]$（$\overline{M}(x \in S)[f(x)]$）と書く。たとえば，$R$ を実数の集合として，$\underline{M}(x \in R)[(x-2)^2+5] = \{2\}$，あるいは $\overline{M}(x \in R)[-(x^2-1)^2] = \{1, -1\}$ である。また，p-グラフ G に対してその節点集合を $V(G)$ と書く。

最適順序 $\sigma_j^{\dagger} \in \Sigma^{\dagger}$ の部分系列 $\sigma_{j2}^{\dagger}$ を求めるとき，たとえば分枝限定法（branch and bound method）などを用いても最悪の場合，すべての $r!$ 通りの系列 $\sigma_5 = (k_1, k_2, \cdots, k_r)$ $(v_{k_i} \in V_r)$ について各 $\Delta_{G_0}(\sigma_2, \sigma_5) = \Delta_{G_2}(\sigma_5)$ を求めなければならない。そこで，必らずしも最適である保障はないが，できるだけ少ない手数で，しかも最適に近い系列を求める方法が要求される。このスパース性を保存する s-ピボット操作の擬似最適解を決定する問題に対して次のような手法が提案されている[5],[7]。

（手法1） 次式を満足する系列 $\sigma_1 = (k_1, k_2, \cdots, k_r)$ を求める。

$$|B_{k_1}| \leq |B_{k_2}| \leq \cdots \leq |B_{k_r}| . \tag{2.51}$$

ただし，各 B_i は行列 $W_0^{*\sigma_2}$ の行 i に関する集合である。

（手法2） 次の手順で系列 $\sigma_2 = (k_1, k_2, \cdots, k_r)$ を求める。

各 $i = 1, 2, \cdots, r$ について，k_i を次の条件を満足するように選ぶ。ただし，B_l は行列 $W_0^{*(\sigma_2, \sigma_2(i-1))}$ の行 l に関する集合で，$G^{(i-1)} = G_2^{*\sigma_2(i-1)}$ とする。

$$v_{k_i} \in \underline{M}(v_l \in V_r \cap V(G^{(i-1)}))\,[\,|B_l|\,] . \tag{2.52}$$

（手法3） 次の手順で系列 $\sigma_3 = (k_1, k_2, \cdots, k_r)$ を求める。

各 $i = 1, 2, \cdots, r$ について，k_i を次の条件を満足するように選ぶ。ただし，$G^{(i-1)} = G_2^{*\sigma_3(i-1)}$ とする。

$$v_{k_i} \in \underline{M}(v_l \in V_r \cap V(G^{(i-1)}))\,[\,S_l(G^{(i-1)})\,] . \tag{2.53}$$

これらの三つの手法のうち，手法1は行列 $W_0^{*\sigma_2}$ のみに着目して操作順序を決定するもので，操作過程におけるフィルインの影響は考察しない最も単純な方法である。また，手法1および2では各行の非零要素の個数 $|B_i|$ に着目するが，手法3においてはフィルインの個数 $S_l(G^{(i-1)})$ に着目している。これらの手法の優劣は，与えられた行列のスパース構造あるいはサイズなどにより

異なるが，一般には手法3がより有効であろう。ところが，手法3で k_i を定めるとき $\delta_\ell(G^{(i-1)})$ の最小値を与える ℓ が2個以上存在する場合，それらのうちのどの ℓ を選ぶかは任意であるが，その選び方が最終的に $\Delta_{G_0}(\sigma)$ の値に影響することがある（2.5.3例題参照）。したがって，この点を考慮した手法を考察することは重要である。

2.5.2 アルゴリズム

ここでは，$\sigma_j^\dagger \in \Sigma^\dagger$ に対する擬似最適順序 $\sigma_j^* = (\sigma_2, \sigma_{j2}^*)$ を得るための次のアルゴリズムについて考察する。

［アルゴリズムB］

操作1： 与えられた正方行列 W_0 に対して，$G \leftarrow G_0$，$V \leftarrow V_0$，$\pi \leftarrow \lambda$，$\Delta \leftarrow 0$ として次の操作へ。

操作2： もし $V = \phi$ なら操作完了。$V \neq \phi$ のとき次の操作へ。

操作3： 集合 $V_1 = \{v_k \mid \delta_k(G) = 0, v_k \in V\}$ を求め，$V_1 = \phi$ のとき操作4へ。$|V_1| =$ 大きい0 のときは，各 $v_{k_i} \in V_1$ なる大個の k_i の任意の系列を $\sigma = (k_1, k_2, \cdots, k_x)$ とし，$\pi \leftarrow (\pi, \sigma)$，$G \leftarrow G^{*\sigma}$，$V \leftarrow V - V_1$ として操作2へ。

操作4： 集合 $V_2 = \underline{M}(v_k \in V)[\delta_k(G)]$ を求め，$|V_2| = 1$ ならば $\{v_k\} \leftarrow V_2$ として操作8へ。$|V_2| > 1$ のとき次の操作へ。

操作5： 集合 $V_3 = \underline{M}(v_k \in V_2)[|A_k| \cdot |B_k|]$ を求め，$|V_3| = 1$ ならば $\{v_k\} \leftarrow V_3$ として操作8へ。$|V_3| > 1$ のとき次の操作へ。

操作6： 集合 $V_4 = \overline{M}(v_k \in V_3)[|\cup_{i \in A_k} L_{ik}|]$ を求め，$|V_4| = 1$ ならば $\{v_k\} \leftarrow V_4$ として操作8へ。$|V_4| > 1$ のとき次の操作へ。

操作7： 集合 $V_4 = \underline{M}(v_k \in V_4)[\sum_{j \in B_k} \sum_{h \in L_{kj}} |L_{hj}^{*k}|]$ とし，V_5 の任意の元を v_k として次の操作へ。

操作8： $\pi \leftarrow (\pi, k)$，$\Delta \leftarrow \Delta + \delta_k(G)$，$G \leftarrow G^{*k}$，$V \leftarrow V - \{v_k\}$ として操作2へ。

（アルゴリズム終）

操作が完了したとき，$\sigma_j^* = \pi$，$\Delta(\sigma_j^*) = \Delta$ となっている。

このアルゴリズムの操作1から3までは，前節で述べたアルゴリズムAと同様であり，系列 σ_2 を求めるためのものである。操作3で一度 $V_1 = \phi$ となれば，それ以後の手続きは $\sigma_j^* = (\sigma_2, \sigma_{j2}^*)$ における部分系列 σ_{j2}^* を求めるためのものになる。

操作4から7までの間に得られる各集合については，$V_i \subset V_{i+1}$ $(i = 2, 3, 4)$ なる関係がある。操作4の集合 V_2 を求めるのは，手法3と同じ考え方であってフィルインの個数が最小となる節点を選び出すことになる。そのような節点

が2個以上あるとき，次の操作5の集合 V_3 を求める。V_3 の元 v_k でS-ピボット操作を行なうとき行列 W の各要素のうち実際の数値を与えるべき要素の数が（V_2 の元のなかでは）最小になっている。操作6では，節点 v_k にはいっている枝の重みの総和が最大のものを選び出す。これは，v_k に関するS-ピボット操作を行なえば，p-グラフ G において v_k に接続するすべての枝は G から取り除かれることから，そのような v_k を選ぶことはその後の操作順序を選択する上で有利になる。また，操作7における集合 L_{hj}^{*k}（$j \in B_k$, $h \in L_{hj}$）は，G^{*k} における新しい枝 (v_h, v_j) の重みであって，V_5 に属す節点を選び出すことは，2段階後のフィルインの個数を考慮していることになる。

以上のように，このアルゴリズムは v_k を決定する際に4種の規準を設定し，ある規準に対する評価値が等しいとき次の規準に照らして選定するというものである。

2.5.3 例題

ここでは，二・三の具体的な例について考察する。

[例2.6] 次の二値化行列をもつ行列 W_0 を考える。$V_0 = V(G_0)$ とする。

$$B(W_0) = \begin{array}{c|ccccccc} & 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ \hline 1 & 1 & \cdot & 1 & \cdot & \cdot & \cdot & \cdot \\ 2 & 1 & 1 & 1 & 1 & \cdot & \cdot & \cdot \\ 3 & 1 & \cdot & 1 & 1 & \cdot & \cdot & \cdot \\ 4 & \cdot & 1 & 1 & 1 & \cdot & 1 & \cdot \\ 5 & \cdot & \cdot & 1 & 1 & 1 & \cdot & \cdot \\ 6 & \cdot & 1 & \cdot & 1 & 1 & 1 & 1 \\ 7 & \cdot & \cdot & \cdot & \cdot & 1 & 1 & 1 \end{array} . \qquad (2.54)$$

この W_0 のp-グラフ G_0 に対してアルゴリズムAで求められる系列 σ_2 は，

$$\sigma_2 = (1, 7, 3, 2, 5, 4, 6),$$

となるが，$\sigma_2 \in \Sigma$ であるからこの σ_2 は明らかに一つの最適解である。(例終)

この例は，$\Delta(\sigma^*) = 0$ であるという特別なものであり，一般に $\Delta(\sigma)$ の最小値が0であるとは限らない。次の例は $\sigma_2 = \lambda$ の場合である。

[例2.7] 例2.4で用いた行列(2.35)について，その操作順序を求めてみる。$V_0 = V(G_0)$ とする。p-グラフ G_0 は図2.3に示されている。同図から明らかなように，$\delta_k(G_0) = 0$ なる v_k は存在しないから $\sigma_2 = \lambda$ である。

アルゴリズムBを用いれば，$\sigma^* = (5, 4, 2, 3, 1)$, $\Delta_{G_0}(\sigma^*) = 1$

を得るが，$\sigma_2 = \lambda$ であることからこの $\sigma^{\#}$ は明らかに最適解である。

一方，手法3で操作順序を求めるとき，式(2.53)の右辺の集合は $\{v_1, v_4, v_5\}$ である。このうち，$k_1 = 5$ とすれば操作順序 $\sigma_3 = \sigma^{\#}$ が得られるが，もし $k_1 = 1$ とすれば例2.5で求めた図2.4の p-グラフが G_0^{*1} となり，$S_i(G_0^{*1}) \geq 1$ $(i = 2,3,4,5)$ であるから，以後どのように系列を選んでも $\Delta(\sigma_3) \geq 2$ となって最適解は得られない。なお，手法3で求められる結果は，$k_1 = 1$ とすれば，$\sigma_3 = (1,4,2,5,3)$，$\Delta(\sigma_3) = 2$ となる。(例終)

$B(W)$ が対称行列である場合について，その擬似最適順序 $\sigma_j^{\#}$ を求めるアルゴリズムのプログラムを作成し，分枝限定法による最適順序 $\sigma_j^{\dagger}$ を求めるプログラムと比較した結果が表2.1である。表中の n は $B(W)$ の次数，g はピボット操作の行なわれるべき行および列の個数 $|V_0|$ を表わす。またCPU timeの'および"はそれぞれ分および秒の意味である。使用機種はNEAC 2200-500である。

表2.1　最適解と擬似最適解の比較

n	g	最適解		擬似最適解	
		CPU time	Δ	CPU time	Δ
8	6	30"	4	30"	4
10	8	3'56"	4	30"	4
11	9	12'55"	20	31"	20
11	9	22'37"	14	31"	14
11	9	37'53"	22	31"	22
12	10	24'30"	28	36"	28
14	12	——	—	32"	20
17	15	——	—	37"	20
28	26	——	—	48"	66

最適解を得るプログラムでは，行列の次数が増加するとき，その演算時間(CPU time)は急激に増加する。これは，分枝限定法が $g!$ のオーダの手数を必要とすることに起因する。一方，擬似最適解を得るプログラムでは，ソースプログラムのコンパイルに必要な時間が演算時間に含まれていることを考慮すれば，次数の増加による演算時間の増加は小さいことが分かる。

2.6 結言

本章では，まず疎大行列の取扱いに関する基本的問題を考察した。この問題は，疎大行列をグラフで表現したときそのグラフの節点集合を何らかの形に分割する問題と，ピボット操作におけるピボット要素の最適順序づけの問題とに大別される。この両者に対するいくつかの接近法について概説した。

さらに，正方行列のLU分解および端子対縮約を，スパース性を保存するという点で同時に含むような広義のピボット操作を定義し，このピボット操作において最適順序決定問題を定式化した。この問題に対して p-グラフなる概念

を導入し，擬似最適順序を得る新しい手法について考察した。

実際に計算機により疎大行列を取扱う場合，本章で考察した手法で得られた擬似最適順序に従って操作を行なえば，計算手数やメモリなどが節約される。特に疎大行列のサイズが大きいほど，非零要素の占める割合は小さくなり，計算機に実行させる際にそれらの非零要素をリスト構造にして記憶させることが多い[8]。このとき，本章で考察した手法で前もって操作順序を定めれば，その操作過程でのフィルインが定められるのでそれらのためのメモリをあらかじめ用意しておくことができる。特に，スパース構造が一定の多数の疎大行列を取扱うとき，その操作順序は $B(W)$ なる＝値化行列のみから定めることができるので，上の手順で計算すれば全体の計算手数をかなり節約することができる。

第3章 印刷基板の自動配線システム

3.1 緒言

回路の小型化，高密度化のため印刷基板（printed wiring board）が広範に用いられるようになり，そのパターン設計の自動化の重要性は，ますます高まりつつある。このパターン設計のうちで特に配線問題に注目すれば，従来から多くの手法が提案されているが[19]~[22]，個々の手法には一長一短があり，単一の手法のみですべての配線を行なうことは配線率（要求された全配線数に対する実現された配線数の割合）の低下，あるいは演算時間の浪費を招く恐れがある。一つの自動配線システムを考える場合，要求された配線のための経路を順次見い出していく方法が一般に採用されているが，各々の経路探索の段階で基板上の情況は変化しているため，その情況に即した配線手法が用いられなければならない。

この章では，配線アルゴリズムとして代表的な三つの手法を採用し，それらの特徴を最大限に生かしうるよう有機的に併用して，一つの印刷基板の自動配線システムを構成することを考察する。

3.2節ではシステムの概要を述べ，特に接続情報の指定，および基板の分割について概説する。3.3節では，三つの配線アルゴリズムの概略を示し，個々の手法の特徴を述べる。3.4節では，自動配線システムとして三つの手法が有機的に結合し，互いに他の手法の欠点を補っていることに言及する。また，信号の順序づけおよびスルーホールの個数削減についても論じ，最後に例題を示す。

3.2 システムの概要

3.2.1 システムの概要

ここで考察する自動配線システムは，二つの配線層を持つ印刷基板を取扱い，部品実装面をA面，他方をB面と呼ぶ。また基板の配線可能領域は矩形であるものとする。

[定義3・1] 基板全体を1.27mm（1/20 inch）単位に区切ってx，yの正整数座標（原点（1，1）を基板の左下隅に置く）を設定し，各領域をfメッシュ（fine mesh）と呼ぶ。各fメッシュは，その座標を用いて（x，y）で表わされ，基板の右上隅のfメッシュを（x_{max}，y_{max}）とする。（定義終）

配線は原則として，A面にはx軸に平行（すなわち水平），B面にはy軸に

平行（すなわち垂直）な線分を引くものとする。ただし，特別なプリアサイン線（3.2.3参照），長さが1メッシュの線分および迷路法（3.3.3参照）で生成される線分についてはこの限りではない。また，両面にある線分を電気的に結合するために導通めっきされた孔（これをスルーホールという）および部品のピン取付孔は隣接するfメッシュに同時に存在してはならないが，対角位置には存在可能とする。

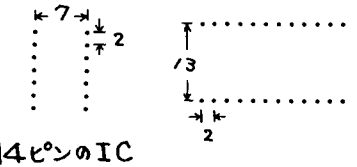
基板上に実装される部品を，次の4種類に分類する。（図3.1参照）

(1) 標準IC（以下では単にICと呼ぶ）：
14，16 あるいは 18 ピンの dual-in-line package IC，またはそれらと同じ大きさとピン数をもつ部品。

(2) LSI ： 上記以外の dual-in-line package IC，または向い合う同一のピン列から成る部品。

(3) 個別部品 ： 1つの端子列から成る部品。

(4) ターミナル ： 基板の端に位置する端子列。



(a) 14ピンのIC　(b) 24ピンのLSI

図3.1 ICおよびLSIの例（数字は f メッシュ数）

一つのコンデンサあるいは抵抗などは，個別部品に分類される。また，ターミナルは基板の外部から電気的にアクセスするための端子の列である。

図3.2に，このシステムのブロック・フローチャートを示す。図中で［　］のルーチンが配線アルゴリズムである。図に示すように，全体を三つのステップ（I，IIおよびIII）に分けて考える。使用言語は，迷路法の一部でアセンブラを用いている他はすべてFORTRANであり，オーバーレイ（overlay）を用いているのでこのシステムの記憶容量は384Kバイトでよい。

ステップIは，印刷基板の接続情報をカードから入力しプログラム内部の表現に編集するルーチン，最初の配線アルゴリズム――スペース・チャネル割当法――を適用して若干の配線を行なうルーチンおよび信号の順序づけを行なうルーチンから成る。ステップIIは，中心的配線アルゴリズム――線分探索法――のみから成り，ここで通常は全体の90%以上の配線が完成される。ステップIIIは，スルーホールを削減するルーチン，最後の配線アルゴリズム――迷路法――で残りの配線を実行するルーチンおよび計

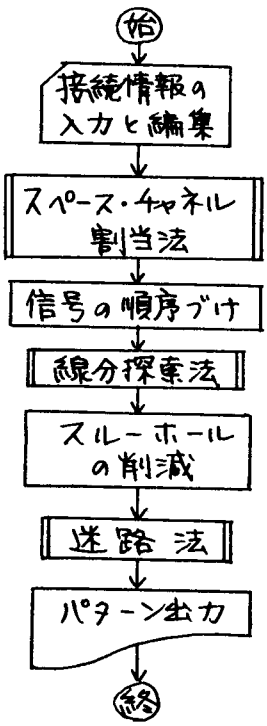


図3.2 ブロック・フローチャート

算結果のパターンを出力するルーチンから成る。ステップⅠ，ⅡおよびⅢの，変数記憶領域を除いたプログラムの大きさは，それぞれ 250 KB（キロバイト），125 KB および 65 KB である。

3.2.2 基板の分割

［定義 3・2］

$$(x,\ ) = \{(x,\alpha) \mid 1 \le \alpha \le y_{max}\} \quad (1 \le x \le x_{max}), \qquad (3\cdot1)$$

$$(\ ,y) = \{(\beta,y) \mid 1 \le \beta \le x_{max}\} \quad (1 \le y \le y_{max}), \qquad (3\cdot2)$$

で定義される f メッシュの集合を チャネル（channel）といい，必要に応じて，式(3・1)を（レベル x の）垂直チャネル，式(3・2)を（レベル y の）水平チャネルと呼んで区別する。（定義終）

［定義 3・3］ 基板全体を m 行 n 列の 2 次元配列状に分割し，それぞれの領域に属す f メッシュの集合をブロック（block）と呼ぶ。ただし，各ブロックの列に含まれる垂直チャネルの本数は x_{max}/m の値が整数のときはすべて等しいものとし，もしその値に端数が生じるときは中央に位置する $x_{max} - m\lceil x_{max}/m \rceil$ (注) 個のブロックの列が他の列より 1 本だけ多くの垂直チャネルを含むように定める。各ブロックの行に含まれる水平チャネルの本数も同様にして定める。左から a 列目で下から b 行目のブロックを $<a,b>$ と書く。（定義終）

［定義 3・4］ 各ブロックを 3 列 4 行の領域に細分割してこれらの各領域に属す f メッシュの集合を s メッシュ（space mesh）と呼び，基板全体の左から i 列目で下から j 行目の s メッシュを $[i,j]$ で表わす。このとき，$1 \le a \le m$，$1 \le b \le n$ なる任意の a，b に対して，左から第 $(3a-1)$ 列目の s メッシュは 7 本の垂直チャネル，下から第 $(4b-2)$ 行目および第 $(4b-1)$ 行目の s メッシュはそれぞれ 6 本および 7 本の水平チャネルをそれぞれ含み，それ以外の s メッシュには残りのチャネルが配分されるものとする。（定義終）

［例 3・1］ $x_{max} = 50$，$y_{max} = 75$ とし，$m = n = 3$ の場合，ブロック $<1,1>$ は図 3・3 のような 12 個の s メッシュに細分割され，次式が成立する。

$$<1,1> = \{(x,y) \mid 1 \le x \le 16,\ 1 \le y \le 25\},$$
$$[1,1] = \{(x,y) \mid 1 \le x \le 4,\ 1 \le y \le 6\},$$
$$[1,2] = \{(x,y) \mid 1 \le x \le 4,\ 7 \le y \le 12\},$$
$$\cdots\cdots$$
$$[3,4] = \{(x,y) \mid 12 \le x \le 16,\ 20 \le y \le 25\}. \qquad (例終)$$

(注) 「・」はガウス記号を表わす。

[定義3·5] 次式で定義される集合，

$$[i,\,] = \bigcup_{\alpha=1}^{4n} [i,\alpha] \quad (1 \le i \le 3m), \tag{3·3}$$

$$[,\,j] = \bigcup_{\beta=1}^{3m} [\beta,j] \quad (1 \le j \le 4n), \tag{3·4}$$

をスペース（space）といい，必要に応じて式(3·3)を（レベルiの）垂直スペース，式(3·4)を（レベルjの）水平スペースと呼んで区別する。（定義終）

図3·4は，fメッシュ，ブロック，Sメッシュおよび水平，垂直スペースを図式的に表わしたものである。ただし，x_{max}，y_{max}，m，nなどの値は例3·1と同一のものとする。

一つのブロック$<a,b>$に14ピンのICを搭載するとき，そのピンは中央の二つのスペース$[3a-1, 4b-1]$および$[3a-1, 4b-2]$の中に納まるようにピンの位置を定める。逆に，このように納められるように中央の二つのスペースに含むチャネルの本数を定めたと考えてもよい（定義3·4参照）。たとえば，図3·3のブロック<1,1>に対して同図の丸印のfメッシュ上に14個のピンが位置づけられる。

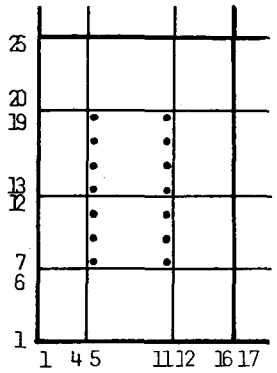


図3·3 ブロック<1,1>および14ピンICの位置

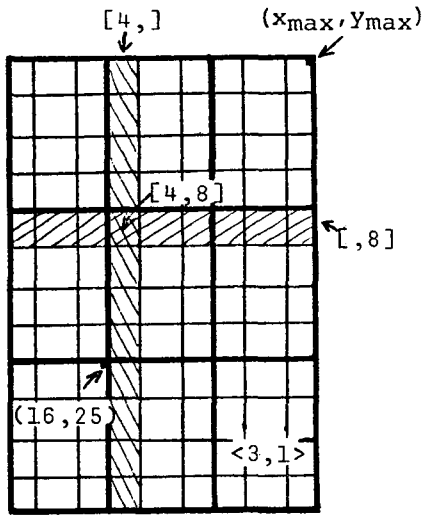


図3·4 基板の分割

3.2.3 接続情報

[定義3·6] 同電位となるように印刷基板上で配線することを要求された端子の集合を信号（signal net）という。（定義終）

印刷基板上に搭載する部品の定義やそれらの間の結線の指定（すなわち信号を定義すること）などは，パンチカードから入力される。

(1) パネル・データ：基板のfメッシュ数（x_{max}およびy_{max}），ブロック数（mおよびn）およびターミナルの位置を指定する。

(2) ターミナル・データ：基板に取付けるターミナルの名称，位置，形状などの指定。

(3) ＩＣデータ ： 搭載する各ＩＣについて，その名称，ピン数，向きおよび位置を示すブロックの座標を指定する。

(4) ＬＳＩデータ ： 搭載する各ＬＳＩを定義する。その位置は１番ピンの座標を，ブロック座標＜X,Y＞と相対座標（$\bar{x}$,$\bar{y}$）を用いて指定する(注1)。

(5) 個別部品データ ： 搭載する個別部品を定義する。位置の指定は前記ＬＳＩの場合に準じて行なう。

(6) 禁止領域データ ： 配線が禁止されているブロックを指定する。

(7) プリアサイン・データ ： プリアサイン線を指定する(注2)。

(8) 第１優先配線データ ： 第１優先の信号を定義する(注3)。

(9) 第２優先配線データ ： 第２優先の信号を定義する(注3)。

(10) 第３優先配線データ ： 第３優先の信号を定義する(注3)。

(注1) 基板上の一つのｆメッシュを指定する方法としてその座標（x,y）を示す他に，ブロック座標＜X,Y＞と相対座標（$\bar{x}$,$\bar{y}$）を用いることがある。このときは，（$x^*+\bar{x}-1$, $y^*+\bar{y}-1$）なるｆメッシュを指定したことになる。ここで，（x^*,y^*）はブロック＜X,Y＞の左下隅のｆメッシュの座標であり，相対座標（$\bar{x}$,$\bar{y}$）は，その座標（x^*,y^*）が原点（1,1）であるとしたときのｆメッシュ座標と考えることができる。

(注2) 電源線やアース線などは，ほとんどの部品に共通な信号であるので，この信号の配線のためにあらかじめいくつかの線分を定義しておくと，その経路探索が容易になる。この線分をプリアサイン線という。また，電気的な理由からプリアサイン線は他の線分より太くすることが要求されることがあり，その場合このプリアサイン・データでその太さも指定する。

(注3) 各信号は，次の三種類に分類される。

Σ_1 ： (7)で指定されたプリアサイン線に結線する端子から成る信号。

Σ_2 ： ターミナルの端子を含む信号でΣ_1に属さないもの。

Σ_3 ： Σ_1にもΣ_2にも属さない信号。

上記のΣ_i（i=1,2,3）に属す信号をそれぞれ第ｉ優先の信号という。基板上で実現されるべきすべての信号の集まりをΣと書けば，ΣはΣ_1,Σ_2およびΣ_3に分割されていることになる。また一つの信号は，それに含まれる各端子を部品名とそのピン番号で指定する。

3.3 配線アルゴリズム

ここでは，このシステムで用いる三つの配線アルゴリズムについて，その概要を述べる。

3.3.1 スペース・チャネル割当法[23],[24] (space and channel assignment)

スペース・チャネル割当法（以下ではSCと略記する）は，二つのステップ——スペース割当とチャネル割当——から成る。前者は経路となる線分を生成して水平または垂直スペースへ登録するものであり，後者は各スペースに登録された線分をその最終的な位置を定めるチャネルに割当てるものである。ただし，SCにおいては Σ_3 に属す信号のみを取扱い，しかも各信号に含まれる端子のなかの適当な2端子のみの配線経路を決定する。このように限定した理由については 3.4.2 で述べる。

［定義 3・7］ 下端が $[i,k]$ で上端が $[i,l]$ なる垂直線分を $[i,]_k^l$ と書き，垂直スペース $[i,]$ に所属するという。同様に，左端が $[k,j]$ で右端が $[l,j]$ なる水平線分を $[,j]_k^l$ と書き，水平スペース $[,j]$ に所属するという。（定義終）

［定義 3・8］ 二つの端子 p_i および p_j の f メッシュ座標をそれぞれ (x_i, y_i) および (x_j, y_j) とするとき，

$$d_f(p_i,p_j) = |x_i - x_j| + |y_i - y_j| , \qquad (3\cdot5)$$

を p_i, p_j 間の f メッシュ上の直角距離（rectilinear distance）という。さらに，それらの s メッシュ座標をそれぞれ $[x_i, y_i]$ および $[x_j, y_j]$ とするとき，

$$d_s(p_i,p_j) = |x_i - x_j| + |y_i - y_j| , \qquad (3\cdot6)$$

を p_i, p_j 間の s メッシュ上の直角距離という。（定義終）

スペース割当の手順の概要は以下の通りである。

［スペース割当］

操作1： $1 \le i \le 3m$，$1 \le j \le 4n$ なるすべての i および j に対して，

$H_{ij} \leftarrow$（水平スペース $[,j]$ に含まれるチャネルの本数），
$V_{ij} \leftarrow$（垂直スペース $[i,]$ に含まれるチャネルの本数），

とおき操作2へ。

操作2： Σ_3 の信号でまだ取扱っていないものがあればその一つを σ として次の操作へ。Σ_3 のすべての信号を取扱ったとき操作完了。

操作3： σ に含まれる端子を $p_1, p_2, \cdots, p_r$ とし，各 p_i は s メッシュ $[x_i, y_i]$ の上にあるとする。次の二つの条件を満たす端子の対 p^1 と p^2 を見い出す。

(i) $d_s(p^1,p^2) = \max_{i,j} d_s(p_i,p_j)$，
(ii) $d_s(p^1,p^2) \ge D$ （D は定数）.

条件(i)および(ii)を満たす p^1, p^2 があるとき次の操作へ。もし見い出されなければ操作2へ。

操作4： Sメッシュ上で p^1 と p^2 を結ぶ経路を探索する。このとき，$H_{ij}=0$ なる $[i,j]$ を水平に通ること，および $V_{ij}=0$ なる $[i,j]$ を垂直に通ることは禁止されている。このような経路が求められたとき次の操作へ。求められないとき操作2へ。

操作5： 求められた経路を構成する線分を，それぞれの所属するスペースに登録し，登録された垂直線分 $[i,]_k^l$ に対しては $V_{it} \leftarrow V_{it}-1$ ($k \leq t \leq l$)，水平線分 $[,j]_k^l$ に対しては $H_{tj} \leftarrow H_{tj}-1$ ($k \leq t \leq l$) として操作2へ。

(アルゴリズム終)

操作3の条件(i)は，σ のなかで最も遠い位置にある2端子を選び出すことを意味している。また，(ii)ではもしそれらの距離がある定数D(Dの値は種Kの実験の結果15を採用している)より小さいならば，信号 σ の配線はSCでは行なわないことを意味している。したがって，このSCでは Σ_3 の各信号のなかの比較的遠距離にある2端子間の配線のみを実行することになる。また，操作4での経路探索は一種のパターン限定法で行なう。これは，2端子 p^1 および p^2 の位置関係あるいは端子の属す部品の種類などによっていくつかの配線パターンを限定して試行を進めるもので，単純な配線パターンの例としては図3·5のようなものがある。例えば，同一レベルのスペース上に2端子がある場合，まずI型のパターンで配線可能か否かを調べ，不可能のときは次にK型で調べる。以下順次複雑なパターンを用いてその経路を探索する。最後の操作5は、求められた経路の線分が通過するSメッシュの容量(H_{ij} および V_{ij})をそれぞれ1ずつ減ずる操作で，これによって各Sメッシュにそれが含む チャネル数以上の線分が登録されることを防ぐことができる。

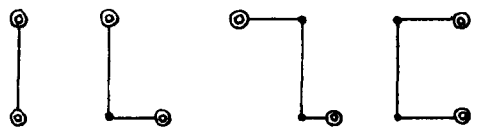


図3·5 配線パターンの例
(◎印は端子，•印はスルーホール)

チャネル割当は，Sメッシュ座標で定義され各スペースに登録されている線分をfメッシュ座標で定義された線分に変換するために，そのスペース内の適当なチャネルに割当てるものであり，このステップは各スペースごとに独立に行なうことができる。

いま，一つの水平スペース $[,j]$ を考える。このスペースに登録されている二つの線分 $l_i=[,j]_a^b$ と $l_k=[,j]_c^d$ に対して，$b<c$ または $d<a$ が成立するとき l_i と l_k は同一のチャネルに割当てることができる。一方，

このいずれの不等式も成立しないとき l_i と l_k を同一のチャネルに割当てることはできない。そこで，これらのことを表現する一つの有向グラフを定義する。

[定義3.9] 水平スペース $[,j]$ に対して次のような有向グラフ $G_j = [V, E]$ を考える。節点集合 V の各元 v_i は，水平スペース $[,j]$ に登録されている線分 $l_i = [,j]_a^b$ に対応し，二つの線分 $l_i = [,j]_a^b$ と $l_k = [,j]_c^d$ に対して，$b<c$（または $d<a$）のときかつそのときに限り有向枝 (v_i, v_j)（または (v_j, v_i)）が枝集合 E に含まれる。（定義終）

ここで定義された有向グラフ G_j の一つの有向道 $(v_{i_1}, v_{i_2}, \cdots, v_{i_p})$ を考えれば，これは水平スペース $[,j]$ 上のある チャネルに左から順に線分 l_{i_1}, $l_{i_2}, \cdots, l_{i_p}$ を割当てることが可能であることに対応している。したがって，登録されたすべての線分を最小数のチャネルに割当てる問題は，有向グラフ G_j において節点を共有しない最小個の有向道で節点集合 V を分割する問題に帰着される。

この問題に対する手法は既に与えられている[24]。しかし，上記の議論においてはスペース内のすべてのチャネルを同一のものとして取扱っているが、基板上には部品の端子あるいはプリアサイン線などが定義されており、それらを含むチャネルと全く含まないチャネルとを同一視できないため、[24]の手法をそのまま用いることはできず若干の変更を必要とする。

垂直スペースについても、チャネル割当は水平スペースの場合と全く同様に行なわれる。

すべてのスペースについてのチャネル割当が完了したとき、各線分の座標が次のようにして定められる。各水平線分 l_i がレベル y_i の水平チャネル $(\ , y_i)$ に割当てられ、各垂直線分 l^j がレベル x_j の垂直チャネル $(x_j, \ )$ に割当てられたとする。一つの水平線分 l_i の両端が、それぞれスルーホールを介して二つの垂直線分 l^j および l^k に連結されている場合（ただし $x_j < x_k$ とする）、l_i は f メッシュ (x_j, y_i) と (x_k, y_i) を結ぶ線分として定められ、A面上に登録される。また、l_i の一端が端子である場合はその端子と l_i の端点を結ぶためにさらに一つの垂直線分を生成する。以上の操作は、垂直線分に対しても同様であり、すべての線分に対してこの操作を行なったとき SC が終了する。

3.3.2 線分探索法[25] (line search method)

線分探索法とは、指定されたいくつかの f メッシュ間を連結する経路を線分ごとの‘おおまか’な探索によって見い出そうとする接近法であり、これに対していくつかの手法が提案されている[25]~[27]。この手法の特徴は、配線経路

が存在してもそれを必ず見い出す保障はないが、比較的短時間で探索を完了しうることにある。

本システムにおける線分探索法（以下ＬＳと略記する）は、前記ＳＣで連結されなかった個々の信号について、その配線経路を見い出すことを目的とし、その配線手法は点対点ルーチン（ＰＰＲ）と、その応用である点対線ルーチン（ＰＬＲ）とから成る。ＰＰＲは、基板上の二つの指定されたｆメッシュ間の配線経路を求めるものであり、ＰＬＲは、基板上の一つのｆメッシュと線分群(注)が与えられたときそのｆメッシュから線分群上のどれかのｆメッシュへの配線経路を求めるものである。これらのルーチンを用いて、ＬＳはおよそ次の手順で一つの信号σに対する経路探索を行なう。

［アルゴリズムＬＳ］

操作１：　σに含まれる２端子が既に連結されているとき操作３へ。そうでないとき次の操作へ。

操作２：　σから適当な２端子を選出し、それらをＰＰＲで連結する。適当な２端子が連結されたら次の操作へ。σのどの２端子間もＰＰＲで連結されないとき操作完了。

操作３：　σの各孤立端子(注)とその線分群をＰＬＲで連結する。すべての孤立端子についての手続きが完了したとき操作完了。

（アルゴリズム終）

操作２で、ある信号σのどの２端子間も連結されないとき、σは全く未完成のまま最後の配線アルゴリズムに引き継がれる。また操作３で、ある孤立端子についてはＰＬＲで線分群に連結されないことが起こり得るが、このときその端子は孤立端子としてやはり最後の配線アルゴリズムに引き継がれる。

以下では、ＬＳの基本となるＰＰＲについてその概略を述べる。

［定義３・１０］　ｆメッシュ（x, y）に対して、Ａ面上でこのｆメッシュを含めて新たに生成することのできる（すなわち基板上に既に登録されている線分および端子を含まない）最大の長さの水平線分を、（x, y）に関する水平可能線分といい $h(x, y)$ で表わす。同様に、Ｂ面上でこのような垂直線分を考え、これを（x, y）に関する垂直可能線分といい $v(x, y)$ で表わす。（定義終）

［定義３・１１］　$h(x-1, y)$、$h(x+1, y)$、$h(x, y-1)$ および $h(x, y+1)$ の各線分を、ｆメッシュ（x, y）に関するレベル０の線分という。さらに、ｆメッシュ（x, y）に関するレベル１の線分とは、（x, y）に関するレベル０の線分上の各ｆメッシュ（x_i, y_i）に対する垂直可能線分 $v(x_i, y_i)$ を指す。（定義終）

これらの定義を用いて、注目する二つのｆメッシュ（x_a, y_a）と（x_b, y_b）

(注)　一つの信号σのいくつかの端子が互いに連結されているとき，その連結のための線分の集まりを線分群といい、線分群に含まれていない端子をσの孤立端子という。

の間の配線経路の探索に関する基本的な操作は次の三つである。
(I) (x_a, y_a) および (x_b, y_b) に関するレベル0の線分をそれぞれ求め、これらのうちで互いに交わる対があるかどうか調べる。
(II) (x_a, y_a) に関するレベル1の線分のなかで、(x_b, y_b) に関するレベル0の線分と交わるものがあるかどうか調べる。
(III) (x_a, y_a) に関するレベル1の線分上の任意のfメッシュ (x_0, y_0) に着目し、(x_0, y_0) に関する水平可能線分 $h(x_0, y_0)$ 上の各fメッシュ (x_i, y_i) に関して、$v(x_i, y_i)$ が (x_b, y_b) に関するレベル0の線分と交わるかどうか調べる。特にこの操作を、探索メッシュ (x_0, y_0) での経路探索という。

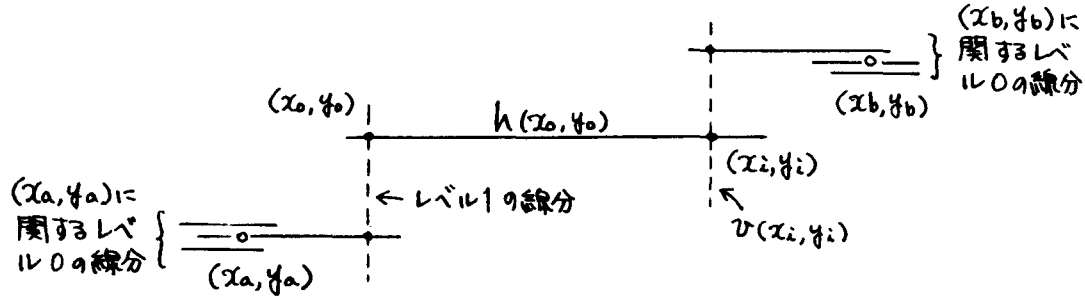


図3·6 探索メッシュ (x_0, y_0) での経路探索

LSでは、上記の操作を重複なく系統的に適用するため、特に操作IIIにおける探索メッシュの選定規準を定め、その規準に従って探索メッシュを順次選び出して経路探索を行なう。さらにその過程において無効域および棄却域なる概念を導入し、それらを用いて配線経路の探索が効率よく実行されるようになっている[25]。

3.3.3 迷路法[28] (maze routing method)

LSでは前述のように効率のよい探索を行なう反面、実際には配線可能であるにもかかわらず、その経路が見い出されないという場合が生じる。したがってそのような配線経路を最終的に探索する配線ルーチンが必要になる。この段階で残っている信号の個数は比較的少なく、基板上に配線可能な領域もかなり制限されていることから、これらの情況に適合する手法が望ましい。

迷路法とは、いくつかの指定されたfメッシュ間を連結する経路をfメッシュごとの探索によって見い出そうとするもので、最初 Lee[29] により提案され、のちに Geyer[30] が多層印刷基板にも適用すべく拡張した。この手法の特徴は、配線が可能な場合にはその最短距離の一つを必ず見い出すことにある。しかしながら、前記のいずれの手法においても二つのfメッシュ間の結線

を目的としたものであり、そのままでは3個以上のfメッシュ間の配線経路決定には不適当である。本システムにおける迷路法（以下ではMZと略記する）では、2層印刷基板上のいくつかの指定されたfメッシュ間を連結する配線経路が可能ならば、その一つを必ず見い出すことを目的とする[28]。

本システムにおいてMZの段階で一つの信号を取扱うとき、すでにSCおよびLSでいくつかの端子間の配線が完了している場合が多いので、MZではいくつかの孤立端子と線分群の間の結線を行なう問題としてとらえる。LSではこの問題に対して、孤立端子を一つづつ取扱い、各端子から線分群への接続経路を順次求めていったが、ここでは複数個の孤立端子がある場合でも一つの信号の結線を同時に行なうところにその特徴がある。また、探索の順序を記憶するための数値（[30]ではこれをmarking sequenceと呼んでいる）をA，B両面にそれぞれ別個に割付けることで、隣接するfメッシュに同時にスルーホールが許されないという条件によって生じる特殊な場合の経路をも見い出し得るようにしている[28]。

一つの信号に対するMZの概要は次の通りである。

［アルゴリズムMZ］

操作1：探索の出発点として注目する信号の孤立端子の中から適当に選び、そのfメッシュをFMS（front mesh set）へ入れ、残りの孤立端子および線分群を目標として記憶する。

操作2：FMSが空のとき操作4へ。そうでないとき、FMSから任意のfメッシュを一つ取り出してそれに隣接するfメッシュで、配線可能かつ未探索のfメッシュをNMS（next mesh set）へ入れる。NMSへ目標の一つが入ったとき操作3へ。その他の場合は操作2を繰返す。

操作3：現在までにすべての目標に達していれば操作6へ。そうでなければ到達したfメッシュを記憶し，特に目標が線分群の場合その線分群のすべてのfメッシュをNMSへ入れて操作2へもどる。

操作4：FMS←NMS、NMS←φ とし、FMSが空のとき操作5へ、そうでなければ操作2へ。

操作5：現在までに到達した目標があれば、それらと出発点とを結ぶ経路を逆探索する。未到達の目標が二つ以上あれば、それらを一つの信号とみなして操作1へ、それ以外のとき操作完了。

操作6：各目標から出発点への経路を逆探索してこの信号に対する操作完了。

（アルゴリズム終）

操作1における出発点は、その信号に含まれる孤立端子の中で中央に位置するものが選ばれる。これは、他の孤立端子への経路が比較的早く発見される可能性が強いためである。また、操作2における配線可能および未探索のfメッシュとは、それぞれそのfメッシュが他の線分あるいは端子に占有されていな

いこと、およびそのfメッシュが今まで一度もFMSまたはNMSに属していないことを意味する。注目する信号が完成されたとき操作6に至るが、一度操作5へ至ったときこの信号は未完成のままとなる。この操作5および6での経路の逆探索は最後に到達した目標から行ない、線分群上のあるfメッシュに達したとき、操作3で記憶したfメッシュからさらに逆探索を実行すれば出発点に至る経路が求められる。

3.4 配線システムの構成

3.4.1 信号の順序づけ

本システムでは信号の処理順序について、グループによる順序づけと、同一グループ内の順序づけを考慮している。前者は、接続情報を入力するときに指定されるもので、三つの信号グループ Σ_1, Σ_2 および Σ_3 を考える（3.2.3参照）。これに対して後者は、それらのうち第2および第3優先の信号グループ、Σ_2 および Σ_3 に含まれる信号間の順序づけであって、SCによる配線が終了した時点でその順序を定める（図3・2参照）。これは、SCでは信号の処理順序が直接その配線結果に影響を及ぼさないが、LSでは配線の可否がその時点の基板の情況、すなわち他の信号のための線分の分布に大きく依存していることによる。そこでSCでは入力された順序で Σ_3 の各信号を処理し、その結果を考慮してLSにおける処理順序を決定する。以下でその手法を考察する。

いま、信号 σ_i の孤立端子を $p_1, p_2, \cdots, p_{k_i}$ とし、各 p_j は部品 $C_{j'}$ に属し（このとき $p_j \in C_{j'}$ と書く）、かつそれはsメッシュ $[x_j, y_j]$ の上にあるものとする。このとき、

$$C^i = \{ C_{j'} \mid p_j \in C_{j'},\ j = 1, 2, \cdots, k_i \}, \qquad (3\cdot7)$$

で定義される部品の集合 C^i を用いて、次の関数 $f_1(i)$ を定義する。

$$f_1(i) = | C^i | + \alpha_i, \qquad (3\cdot8)$$

ここで α_i は、信号 σ_i の2端子がSCですでに結線されているとき1，そうでないとき0の値をとるものとする。この関数 $f_1(i)$ の値は、σ_i の配線を完成するためには、SCで作られた線分群があればそれを一つの部品とみなしたとき、$f_1(i)$ 個の部品間の配線を行なわなければならないことを意味している。さらに、k_i 個のsメッシュ座標 $[x_j, y_j]$ に対して、

$$M_x^i = \max_j [x_j] \ ; \quad m_x^i = \min_j [x_j], \qquad (3\cdot9)$$

$$M_y^i = \max_j [y_j] \ ; \quad m_y^i = \min_j [y_j]. \qquad (3\cdot10)$$

とおく。これを用いて次の関数を定義する。

$$f_2(i) = (M_x^i - m_x^i) + (M_y^i - m_y^i), \qquad (3\cdot11)$$

$$f_3(i) = \left| x_0 - \frac{1}{2}(M_x^i + m_x^i) \right| + \left| y_0 - \frac{1}{2}(M_y^i + m_y^i) \right|. \qquad (3\cdot12)$$

ここで、$[x_0, y_0]$ は基板中央のSメッシュ座標（$x_0 = \lceil \frac{3}{2}m \rceil + 1$, $y_0 = 2n$）である。この $f_2(i)$ は最も効率よく配線されたときの経路を構成する線分の長さの総和をSメッシュの単位で与えるものであり、$f_3(i)$ は孤立端子の拡がりの中心と基板中央とのSメッシュ上での直角距離である。

これらの関数を用いて、信号 σ_i に対し三つの実数の組を値とする次の関数を定める。

$$F(i) = (f_1(i), f_2(i), f_3(i)). \qquad (3\cdot13)$$

[定義3·12] $A = (a_1, a_2, a_3)$ および $B = (b_1, b_2, b_3)$ に対して、次のいずれかの条件が満たされるとき A は B より大きい（または B は A より小さい）といい $A > B$ と書く（辞書式順序づけ；lexicographic ordering）。

(i) $a_1 > b_1$ ；

(ii) $a_1 = b_1$ かつ $a_2 > b_2$ ；

(iii) $a_1 = b_1$, $a_2 = b_2$ かつ $a_3 > b_3$ ：

特に $a_i = b_i$ $(i = 1, 2, 3)$ のとき A と B は等しく、$A = B$ と書く。（定義終）

さきに定義された三つの関数 $f_j(i)$ $(j = 1, 2, 3)$ は、それぞれ信号 σ_i の配線経路の探索の難易性を評価しており、値が小さいほどその経路が限定されることを示している。したがって、その値が小さい信号を優先して処理することが望ましい。そこで、各信号 σ_i に対して与えられる関数 $F(i)$ を定義3·12に従って順序づけし、$F(i)$ の小さい信号から順に配線処理を行なうようにする。

3.4.2 各アルゴリズムの適用範囲

本システムにおいて、前述の三つの配線アルゴリズムが相互にどのような関連をもつかについて、実験結果をもとに概説する。

各アルゴリズムが処理する信号は以下の通りである。

SC：Σ_3 の各信号について、適当な2端子を選びそれらの間の配線を行なう。もし適当な2端子を選べないとき、ここではその信号については何もしない。

LS：(1) Σ_1 の各信号を入力された順序で配線する。

(2) Σ_2 の各信号を3.4.1の順序で配線する。

(3a) Σ_3の信号でSCによって全く配線されなかったものについて、最も遠い直角距離にある2端子を選んで配線を行なう。

(3b) Σ_3の信号で3個以上の端子を含むものについて、3.4.1の順序で配線を行なう。

MZ：SCおよびLSによって完成されていない信号を、LSが処理した順序で経路探索する。

上記から明らかなように、本システムではLSが配線アルゴリズムの中心的な役割を果し、多くの場合全体の90%前後の配線がここで実現される。まずLSにおける配線処理について考察する。

Σ_1の各信号に含まれる端子は、一般に基板全体に分布していて最寄りのプリアサイン線に通常1本の線分で配線される。もし基板上に多くの線分が存在している段階でこの配線を実行すれば、1本の線分のみでは結線が不可能になることが起こり得る。一方、この配線を最優先しても他の信号の経路探索には直接の影響を与えない（上記LS(1)）。

基板上で配線のためのチャネルが不足する箇所は多くの場合、基板中央とターミナル付近である。特に外部端子（ターミナルの端子）からは一方向にしか線分が引けないため、Σ_2の各信号はΣ_3の信号よりも優先される。SCではΣ_2の信号については全く取扱わないので、LSではまず各信号の最も遠い2端子を選んでPPRで配線し、それを目標とみなしてPLRを用いて配線を完成する（LS(2)）。一つの信号についての探索が終了してから次の信号の処理を始める。

Σ_3の経路探索は、最初に各信号の2端子間の配線をSCでその配線が行なわれていない信号について実行する（LS(3b)）。これは、基板上に存在する線分の密度が高くなるにつれてPPRでの経路探索が困難になることを考慮するためで、この段階で2端子間の配線を行なっておけば、他の孤立端子との配線（LS(3b)）にはPLRを用いることができ、この場合は目標が線分群であるため基板が密であってもその経路が見い出される可能性が大きい。

前述のように基板上の線分が密になったとき、信号がまだ孤立端子のみであればその経路探索は困難である。特にΣ_3の信号は、LSにおいて最後に処理されるため、配線が未完成で終ることが多い。そこで、LSの補助的な役割をもつ配線アルゴリズムとしてSCが用いられる。このSCは最初に適用される配線アルゴリズムであるため、基板上にはプリアサイン線のみが存在し、しかもここで生成される線分はチャネル割当によって同時にその位置が決定されるため、比較的長い線分も容易に作ることができる。また、各スペースに登録されている線分を可能なかぎり少ない個数のチャネルに集中して割当てることで、LSのために全く線分が存在していないチャネルを残しておくことができる。後述の例題の結果が示すように、SCでは全体の10%以下の配線しか行なわ

ない。しかし、SCを用いない場合配線率および演算時間の点で悪い結果を招くことが多く、特に基板サイズが大きく信号の個数が多いときその傾向が顕著に表われる。これはΣ_3の信号をLSで処理する場合、ある端子から経路探索するときの目標が線分群であれば、基板がある程度密になっていても効果的に実行されることを示している。

最後の配線アルゴリズムであるMZは、LSで完成しない信号についてその経路探索を行なう。LSでその経路が見い出せないのは、LSでの探索範囲を限定したためであり、より複雑な形状の経路によって配線可能な場合が多く、もし可能ならばMZが必らずそれを見い出す。特に配線の最終段階では、基板上の線分の密度は非常に高いため、このことはMZにおける探索範囲が必要以上に拡がらないという点でも、最後の配線アルゴリズムとしてこのMZは適当である。

3.4.3 スルーホールの削減

本システムでは、MZを用いる直前にそれまでに生成された線分を調べて、スルーホールの個数を減少させる処理を行なう。ここでは、一つの線分がその両端のfメッシュ以外で他の線分と同一のfメッシュを共有しないときに限り、その線分を逆の面上に移すことでスルーホールを削減する。この処理は単純であるが、通常10～20%のスルーホールを削減する効果がある。

[例3.2] 図3.7(a)の配線パターンについて考える。ただし、図中で①および×はそれぞれ部品のピン(端子)およびスルーホールを表わし、実線はA面上の線分、破線はB面上の線分を表わす。前記の条件「一つの線分がその両端のfメッシュ以外で他の線分と同一のfメッシュを共有しない」を満足する線分は5本あり、それぞれ図中に⇒で示されているものである。そこで、これらの線分を逆の面上に移したとき、同図(b)の配線パターンが得られる。この変換によって6個のスルーホールが削減される。(例終)

前記の条件を満足する線分は、この例からも分かるように主に基板の周辺部に多く、基板中央に存在することはまれである。したがって、ここでの処理は基板周辺にある線分のみに適用するだけで十分である。

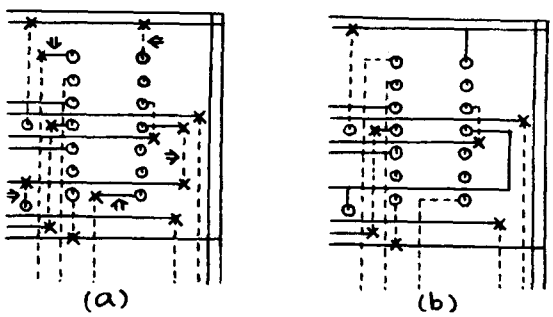


図3.7 スルーホールの削減

3.4.4 例 題

一つの信号 σ_i が n_i 個の端子から成っているとする（このとき $|\sigma_i|=n_i$ と書く）。この σ_i のための配線は（n_i-1）個の端子間の経路探索によって実現される。一方、第1優先の信号の場合すでにいくつかのプリアサイン線が定義されているため、n_i 個の端子をそれらの線分群に接続するために経路探索を n_i 回実行する必要がある。以上のことから、要求されている経路探索の回数の総和として正整数 D をつぎのように定める。

$$D=\sum_{i=1}^{n^*}(|\sigma_i|-1)+n_1^* \quad . \tag{3.14}$$

ここで、n^* は信号の総数（$n^*=|\Sigma|$）、n_1^* は第1優先の信号の個数（$n_1^*=|\Sigma_1|$）である。いま、手法XX（XX＝SC、LSまたはMZ）が、信号 σ_i の端子のうち k_i 個の端子対についての配線を実現したとするとき、その手法の配線率を、

$$R_{XX}=\frac{\sum_{i=1}^{n^*}k_i}{D}\times 100 \quad , \tag{3.15}$$

と定義する（単位は%）。本システムによる最終配線率 R は次式で与えられる。

$$R=R_{SC}+R_{LS}+R_{MZ} \quad . \tag{3.16}$$

この配線率を規準にして、ここでは本システムの適用例を示す。表3・1にここで用いる三つの例題の規模を示す。

例1の配線結果を表3・2に、そのプロッタの出力図を図3・8にそれぞれ示す。表3・2におけるSC、LSおよびMZの所用時間はそれぞれ図3・2のステップⅠ、ⅡおよびⅢのルーチンで要した演算時間を示している。それらの合計時間である19.4 secは必ずしもこのジョブのCPU timeではない。

表3・1 例題の規模

例番号		例1	例2	例3
サイズ（インチ）		100×135	230×120	234×125
搭載部品数	ターミナル	5	4	4
	IC	19	34	29
	LSI	2	2	19
	個別部品	7	24	5
	合計	33	64	57
信号総数		94	144	271

表3・2 例1の配線結果

			SC	LS	MZ	合計
スルーホール数	373個	配線率（%）	8.05	88.14	3.81	100 %
削減されたスルーホール	47個	所用時間（sec）	3.1	11.0	5.3	19.4 sec
水平線分数	365本					
垂直線分数	384本					

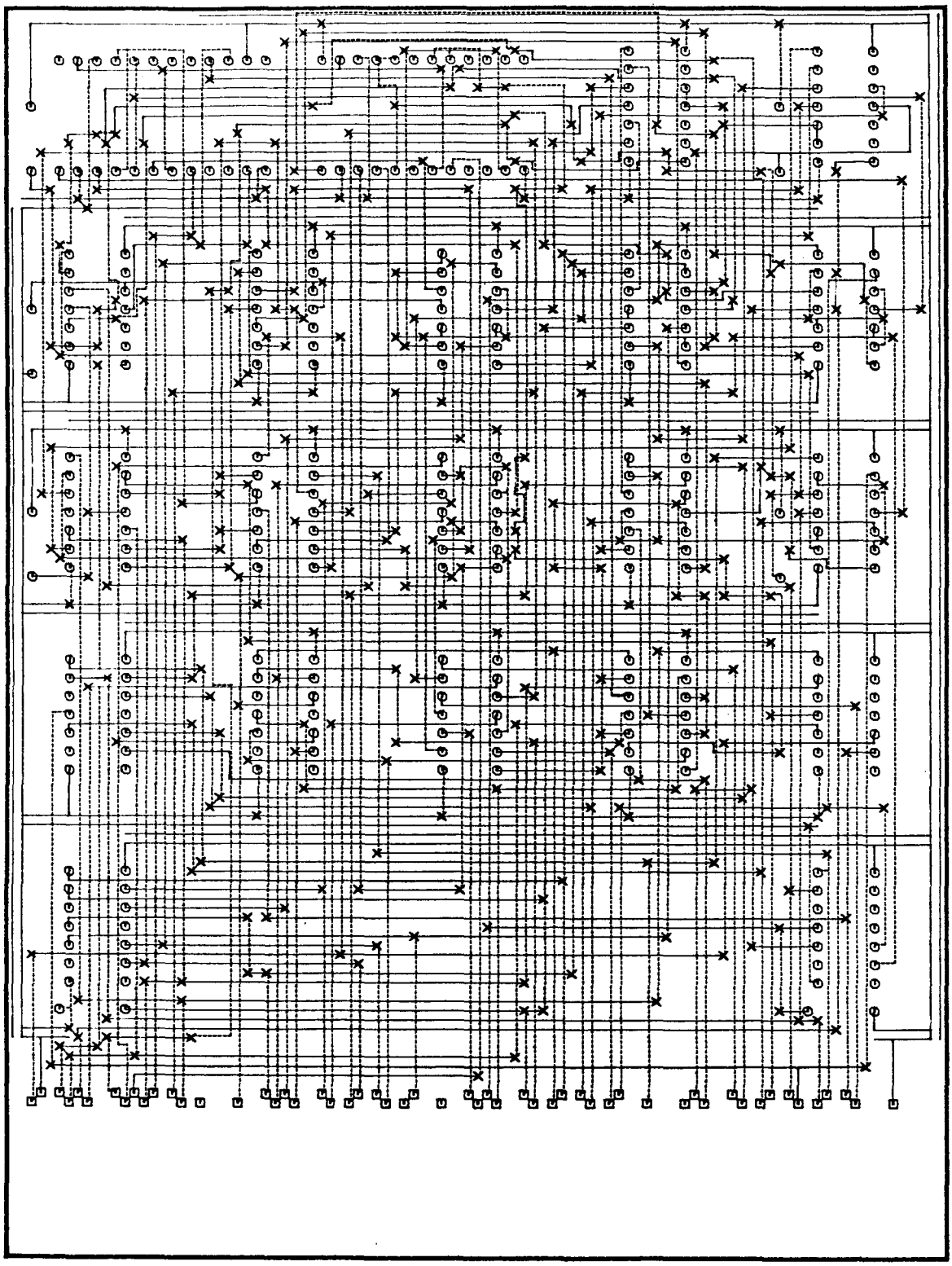


図3·8 例1のプロッタ出力図

表3·3 例2および例3の配線結果

例番号		例 2		例 3	
システム名		APD	本システム	APD	本システム
スルーホール数		329	405	833	794
線分数		961	895	1839	1614
配線率	SC(%)	——	8.20	——	5.18
	LS(%)	97.53	90.98	70.38	88.21
	MZ(%)	2.19	0.82	17.95	4.29
	合計	99.73%	100 %	88.33%	97.68%

表3·1の例2および例3についてその配線結果を表3·3に示す。例題の複雑さを知る目安としてAPD(注)の結果を併記してある。ただし、データ入力形式に差異があるため両者の部品の配置には若干の違いがあり、特に例3については24ピンのLSIの配置を変更した。表3·3の線分数は、水平、垂直線分の合計本数で、APDの配線率はHeuristic methodおよびMaze methodのそれをLSおよびMZの欄にそれぞれ記した。本システムの三つのステップの合計所用時間は、例2が20.11 sec、例3が43.50 secであった(IBM 370/168)。一方APDでは、CPU time(IBM 370/165)が、例2で27.72 sec、例3で72.36 secであった。これらの数値は、使用機種およびデータの取り方が異なるためその比較はできないが、少なくとも基板上の線分の密度がある程度高くても、本システムはその特徴を発揮して効率よく配線すると考えることができる。

3.5 結言

ここでは、2層印刷基板の自動配線システムについてその配線アルゴリズムの概要と、それらの配線処理について述べた。

本システムは、LSを中核の配線アルゴリズムとして持ち、SCおよびMZがそれぞれの特徴を生かしてLSを補助するという構成になっている。SCは、LSがΣ_3の信号の配線を容易に探索できるように前もって2端子間を配線しておくという意味でLSを補助し、一方MZは、LSにおいて配線不可能であった部分を探索するという意味で補助している。このように三つの配線アルゴリズムが互いにその特徴を生かして一つの自動配線システムとしての有効性を発揮する。特に最近のように高密度配線の自動化が益々要請されるに至っている今日、その実用的価値は非常に高いものといえる。

(注) 日本IBM社のプリント基板総合設計プログラム。(Automatic Printed-circuit-board Design)

第4章 結　論

本研究で得られた主な成果と今後に残された問題を簡単にまとめると次のようである。

第2章では、回路網解析の際にしばしば生じる疎大行列を係数とする連立一次方程式の取扱いに関して、行列のスパース性を保存するようなピボット操作の順序づけの問題を考察し、一つの擬似最適順序を得るアルゴリズムを提案した。このアルゴリズムは、比較的短時間で一つの操作順序を与えることから、回路解析プログラムに組み込んで使用するといった場合特に有効である。この場合残された問題として、回路解析プログラムのデータ構造とこのアルゴリズムとの関係がある。データ構造は、解析の目的（線形か非線形か、あるいは定常解析か過渡解析かなど）、あるいはそのために用いる非線形方程式の解法などに深く関係していて一概にはいえないが、基本的には非零要素のみをリスト構造にして記憶する手法がとられる。したがって、一つの回路解析プログラムの一環として疎大行列解析を考え、そのデータ構造との関連において擬似最適順序を与えるアルゴリズムを研究する必要がある。ただしこの場合に、ここで提案したアルゴリズムを基本的に変更する必要はない。

第3章では、2層の印刷基板の自動配線システムについて、そこで用いられる三つの配線アルゴリズムの概要と、それらの配線処理について考察した。これらのアルゴリズムはそれぞれ異なる特徴を有して互いに他の欠点を補っており、全体として有効な一つの自動配線システムを構成している。一方、基板の配線率に大きな影響を与える要因の一つとして部品の配置の問題がある。逆にこの配置問題だけを取り上げれば、良い配置とは配線率を100%にするものであり、それは配線システムすなわち配線手法に依存する。したがって今後に残された問題として、ここで考察した自動配線システムに適合した配置を与えるアルゴリズムの研究が重要であり、これらを統合することで一つの印刷基板自動設計システムが完成する。また、論理装置の設計の自動化の立場からは、論理の分割あるいは割付といった問題に対する有効なアルゴリズムの研究が残された課題である。

一般に設計とは、構成、解析、評価、最適化といった過程を含む一つのフィードバック・システムと考えられる。それぞれの過程において計算機は、設計図表や計算尺など従来からの設計の道具とは比較にならないほどの大きな力を持った新しい道具であり、計算機援用設計の発展は設計に対する姿勢を変えたと言っても過言ではない。しかしながら設計という作業の中に技術者の介入は未だ不可欠であり、今後は工業生産の完全な省力化の目標に向けて、設計自動化あるいは自動設計の研究・開発が重要な課題となるであろうと思われる。

謝辞

本研究の全過程を通じて、直接理解ある御指導を賜わり、つねに励ましていただいた尾崎弘教授、樹下行三助教授ならびに白川功助教授に衷心より感謝の意を表する。

大学院修士、博士両課程において電子工学一般および各専門分野に関し御指導、御教示を賜わった電子工学教室中井順吉教授、小山次郎教授、児玉慎三教授、電子ビーム研究施設裏克己教授、塙輝雄教授、産業科学研究所松尾幸人教授、中村勝吾教授、角所収教授ならびに喜田村善一名誉教授に深謝する。

第3章について御助言、御援助をいただいたシャープ株式会社システム機器部三坂重雄課長、同社中央研究所西岡郁夫係長、栗本卓二氏、東洋情報システム株式会社応用技術部和田英男部長、林泰宏課長、杉田定嗣氏、住友金属株式会社中央研究所山村春夫氏、三菱重工業株式会社高砂研究所井手幹生氏に厚く感謝する。

本研究に関し、福井大学谷口慶治助教授、名古屋工業大学山本勝助教授、基礎工学部細見輝政助手、柏原敏伸助手、富士通株式会社伝送事業部塩浜二郎氏には本学大学院在学中に有益な御助言、御討論をいただき、心から謝意を表する。

また、愛媛大学有吉弘教授、琉球大学喜屋武盛基教授、ならびに佐賀大学高松雄三講師には、いろいろ御助言、御援助をいただき、厚く御礼申し上げる。

筆者の属している尾崎研究室の藤原秀雄助手、河田亨助手、戸松重一技官、大学院学生川端信賢氏、松田潤氏、笹尾勤氏、千葉徹氏、築山修治氏、トラン・ディン・アム氏、また同研究室の藤田基美子氏には種々の面で御協力いただいた。ここに記して感謝する次第である。

参考文献

[1] F.H.Branin,Jr.;"Computer methods of network analysis", Proc. IEEE, vol.55, pp.1787-1801 (Nov. 1967).

[2] R.P.Tewarson;"Computations with sparse matrices", SIAM Review, vol.12, pp.527-543 (Oct. 1970).

[3] J.K.Reid, ed.;"Large Sparse Sets of Linear Equations", Academic Press, N.Y. (1971).

[4] G.E.Forsythe and C.B.Moler;"Computer Solution of Linear Algebraic Systems", Prentice-Hall, N.J. (1967).

[5] W.F.Tinney and J.W.Walker;"Direct solutions of sparse network equations by optimally ordered triangular factorization", Proc. IEEE, vol.55, pp.1801-1809 (Nov. 1967).

[6] G.G.Gustavson, W.M.Liniger and R.A.Willoughby;"Symbolic generation of an optimal Crout algorithm for sparse systems of linear equations", J.ACM, vol.17, pp.87-109 (Jan. 1970).

[7] E.C.Ogbuobiri, W.F.Tinney and J.W.Walker;"Sparsity-directed decomposition for Gaussian elimination on matrices", IEEE Trans., PAS-89, pp.141-150 (Jan. 1970).

[8] R.D.Berry;"An optimal ordering of electronic circuit equations for a sparse matrix solution", IEEE Trans., CT-18, pp.40-50 (Jan. 1971).

[9] 有吉、白川、尾崎;"Sparseアドミタンス行列の計算手順に関するグラフ理論的一手法"、信学論(A)、53-A、pp. 612-619 (昭45-11).

[10] 戸川、白川、尾崎;"区行列におけるブロックごとの端子縮約のための回路網分解"、信学論(A)、54-A、pp. 338-344 (昭46-06).

[11] 尾崎、白川;"グラフとネットワークの理論"、コロナ社(昭48).

[12] L.K.Cheung and E.S.Kuh;"The bordered triangular matrix and minimum essential sets of a digragh", IEEE Trans., CAS-21, pp.633-639 (Sept. 1974).

[13] A.Lempel and I.Cederbaum;"Minimum feedback arc and vertex sets of a directed gragh", IEEE Trans., CT-13, pp.399-403 (Dec. 1966).

[14] G.Guardabassi;"An indirect method for minimal essential sets", IEEE Trans., CAS-20, pp.14-17 (Jan. 1974).

[15] R.M.Karp;"Reducibility among combinatorial problems" in "Complexity of Computer Computations (R.E.Millar and J.W.Thatcher, eds.)", Plenum Press, N.Y., pp.85-103 (1972).

[16] H.Y.Hsieh and M.S.Ghausi;"On optimal-pivoting algorithms in

SPARSE MATRICES", IEEE TRANS., CT-19, PP.93-96 (JAN. 1972).
[17] 喜屋武, 白川, 尾崎; "最適ピボティング順序問題とこれに付随する2部ネットワーク", 信学論(A), 57-A, pp. 864-871 (昭49-12).
[18] D.J.ROSE;"TRIANGULATED GRAGHS AND THE ELIMINATION PROCESS", J.MATH.ANAL.APPL., VOL.32, PP.597-609 (1970).
[19] M.A.BREUER, ED.;"DESIGN AUTOMATION OF DIGITAL SYSTEMS, VOL.1", PRENTICE-HALL, N.J. (1972).
[20] H.NAKAHARA;"COMPUTER-AIDED INTERCONNECTION ROUTING: GENERAL SURVEY OF THE STATE-OF-THE-ART", NETWORKS, VOL.2, PP.167-183 (1972).
[21] D.W.HIGHTOWER;"THE INTERCONNECTION PROBLEM - A TUTORIAL", PROC. DESIGN AUTOMATION WORKSHOP, PP.1-12 (1973).
[22] H.C.SO;"SOME THEORETICAL RESULTS ON THE ROUTING OF MULTI-LAYER PRINTED-WIRING BOARDS", IEEE INTER.SYMP.CIRCUIT AND SYSTEMS, PP.296-303 (1974).
[23] 坂本、山村、千葉、山下、河田、白川、尾崎; "最適配置配線問題に関する一手法"、信学会、回路とシステム研資, CST 73-70 (昭48-12).
[24] A.HASHIMOTO AND J.STEVENS;"WIRE ROUTING BY OPTIMIZING CHANNEL ASSIGNMENT WITHIN LARGE APERTURES", PROC. DESIGN AUTOMATION WORKSHOP, PP.155-169 (1971).
[25] 山村、白川、尾崎; "二層プリント基板上の配線問題に対する線分探索の一手法", 信学論(A), 57-A, pp. 671-678 (昭49-09).
[26] K.MIKAMI AND K.TABUCHI;"A COMPUTER PROGRAM FOR OPTIMAL ROUTING OF PRINTED CIRCUIT CONDUCTORS", IFIP CONGRESS 68, PP.1475-1478 (1968).
[27] D.W.HIGHTOWER;"A SOLUTION TO LINE-ROUTING PROBLEMS ON THE CONTINUOUS PLANE", PROC. DESIGN AUTOMATION WORKSHOP, PP.1-24 (1969).
[28] 千葉, 白川, 尾崎; "2層配線に対する迷路法の一手法", 信学論(A), (掲載予定).
[29] C.Y.LEE;"AN ALGORITHM FOR PATH CONNECTIONS AND ITS APPLICATIONS", IRE TRANS., EC-10, PP.346-365 (SEPT. 1961).
[30] J.M.GEYER;"CONNECTION ROUTING ALGORITHM FOR PRINTED CIRCUIT BOARDS", IEEE TRANS., CT-18, PP.95-100 (JAN. 1971).