



Title	印刷回路基板の計算機援用設計手法に関する研究
Author(s)	千葉, 徹
Citation	大阪大学, 1977, 博士論文
Version Type	VoR
URL	https://hdl.handle.net/11094/68
rights	
Note	

The University of Osaka Institutional Knowledge Archive : OUKA

<https://ir.library.osaka-u.ac.jp/>

The University of Osaka

印刷回路基板の計算機援用 設計手法に関する研究

1977年1月

千葉 徹

印刷回路基板の計算機援用 設計手法に関する研究

1977年1月

千葉 徹

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、印刷回路基板の計算機援用設計手法に関する研究をまとめたものである。

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、印刷回路基板を自動設計する際の方針を明らかにし、種々の配線手法について比較考察する。さらに、これらの考察に基づいて2層印刷回路基板に対する配線手法を提案し、これを採用した自動配線システムの例を紹介している。

一般に、迷路法と呼ばれている配線手法は、配線可能な場合には必ずその配線経路のうちの一つを見い出すという特徴を持っているが、従来手法ではこの特徴に例外があり、さらに、多端子配線の場合の具体的な方針が明らかにされていない等の欠点があった。ここでは、これらの欠点を補い、かつ、実行の手数の効率化を計った配線手法、迷路法を提案している。

第3章においては、第2章で提案する配線手法を多層印刷回路基板に対して適用可能なように一般化している。

回路が大規模になり、素子数が増すと、これを2層の印刷回路基板で実現するためには、基板面積の増大を招き、始めの目的とは相反する事態が生ずる。従って、これを解決するため、基板の層数を多くした、いわゆる多層印刷回路基板が用いられる場合がある。本章では、基板をグラフで表現することによって、配線問題をグラフ上のある種の木を求める問題に帰着させ、計算機の取り扱い易い形にし、さらにこれを求めるための手法を提案して評価している。

第4章においては、回路を一層印刷回路基板で実現するために必要となるグラフの平面化問題について論じている。

一層印刷回路基板は、コストの面から民生用機器において多く使用されているが、設計自動化要求の強さにもかかわらず、問題の特殊性から放置されてきたものの一つである。ここではこの問題に対する統一的な接近法を示し、それに付随して生ずるグラフの平面化問題に関して効率的な手法を提案している。さらに、その実行の手数を理論的に求め、プログラム実験結果を示して、実用化が可能なことを明らかにしている。

第5章結論においては、本研究で得られた結果と今後残された問題についてまとめてある。

関連発表論文

- (1) 千葉, 井手, 白川, 尾崎, “2層配線に対する迷路法の一手法”, 信学論(A), 59-A, 3. pp. 247 (1976-03).
- (2) 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, CST 73-70 (1973-12).
- (3) 千葉, 山下, 白川, 尾崎, “2層配線における迷路法”, 信学会回路とシステム理論研究会資料, CST 73-83 (1974-02).
- (4) 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, CST 74-58 (1974-10).
- (5) 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, pp.100 (August 1975).
- (6) 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, 17, 6, pp. 486 (1976-06).
- (7) 千葉, 白川, 尾崎, 坂本, “配線問題における迷路法が多層基板適用への拡張について”, 信学会回路とシステム理論研究会資料, CST 76-12 (1976-05).
- (8) 千葉, 上田, 泰, 白川, 尾崎, “グラフの平面化とその配線問題に対する応用”, 信学会回路とシステム理論研究会資料, CST 76-80 (1976-10).
- (9) 千葉, 白川, 尾崎, “配線問題における迷路法が多層基板適用への拡張について”, 信学論(A) 掲載予定.

目 次

第1章 緒論	1
第2章 2層印刷回路基板に対する迷路法	3
2.1 緒言	3
2.2 回路基板に対する配線手法	3
2.3 迷路法	5
2.3.1 配線可能路と信号網	5
2.3.2 ネット配線可能路	8
2.4 アルゴリズム	10
2.4.1 探索アルゴリズム	10
2.4.2 経路決定アルゴリズム	12
2.5 例題	17
2.6 結言	18
第3章 迷路法の一般化	19
3.1 緒言	19
3.2 配線問題の定式化	19
3.2.1 基板のグラフ表現	19
3.2.2 配線問題	21
3.3 アルゴリズム	22
3.3.1 緒定義	22
3.3.2 前進手続き	23
3.3.3 後進手続き	24
3.4 アルゴリズムの評価	26
3.5 例題	29
3.6 結言	33
第4章 一層印刷回路基板におけるグラフの平面化問題	34
4.1 緒言	34
4.2 グラフの平面化問題	35
4.2.1 諸定義	35
4.2.2 予備的考察	39
4.2.3 フロンドリストと生成アルゴリズム	42
4.2.4 平面化アルゴリズム	45
4.2.5 アルゴリズムの評価	46

4.2.6 例題	47
4.3 結言	51
第5章 結論	52
謝辭	53
参考文献	54

第1章 緒 論

近年、電子計算機の急速な発展と社会的な要求とが相まって、各方面において設計、製造の自動化が盛んに進められるようになった。このことは、今日の電子装置の大規模化、高信頼性、実現の小形化等のことから切り離しては考えられない。このように複雑化、多素子化を続ける電子装置に対する多様な設計要求に、急速に対応するためには、電子計算機の助けを借りた自動設計—計算機援用設計 (Computer Aided Design : CAD)—が不可欠のものとなってきた。電子装置の計算機援用設計は、古くは回路解析^[1]に始まり、最近では回路素子 (回路モジュール) の配置や配線を自動化する動きが盛んに行なわれている。

一般に電子装置は、印刷回路基板を用いて実現されることが多く、その設計過程はつぎのように分けられる。

- (i) 回路設計：回路素子を単位として電子回路の電気的特性を解析する。
- (ii) 論理設計：論理素子を単位として論理動作を解析する。
- (iii) 論理を分割し、回路素子へ割付ける。
- (iv) 配置設計：印刷回路基板上へ回路素子を配置する。
- (v) 配線設計：各回路素子の、同電位に結線すべき端子間の配線経路を見い出す。

これらのうち特に (i), (ii) に関しては、BACKIS, PCBII, ARD等の幾つかのプログラムが提案され、一部実用化されているが、結果などの程度、当初の設計要求を満たしているかを判定するのが困難であり、例えば配線率やビアホールの数で評価しなければならないという問題の性質上、決定的なものではない。

ここで視点を換え、電子装置の計算機援用設計、とくに回路素子の配置、配線問題を、そのための道具としてのソフトウェアの面から眺めてみる。この分野においては従来から種々の手法 (アルゴリズム) が多数の著者により提案、考察されている^{[2]~[3]}。これらの各手法はそれぞれ異なる特徴、性質を持っているが、特に配線問題に着目すると、印刷回路基板上で配線が進められていくに従ってその基板の状態は変化するので、単一の手法のみを用いて配線を行なうと、その手法の持つ長所が生かされなくなるということも生じる。この観点に基づき、現在ある自動配線システムは幾つかの性質の異なる手法を、互いに欠点を補な合うように有機的に組み合わせることによってその機能を上げている。

これらの各手法のなかでとくに迷路法は、最初 Lee [2] によって提案され、

後に Geyer [3], Rubin [4] 等により実用に即するように発展されたものであるが、与えられた2端子間の配線が可能ならば必ずそのうちの一つの配線経路を見い出すという特徴を持っており、この極めて実用的な性質のために、上述の幾つかの自動配線プログラムにおいても採用されている。

本論文は、計算機援用設計手法のうち特に迷路法に関して、対象とする印刷回路基板が2層の場合と一般の場合とに分け、アルゴリズムを提案し、考察するものである。さらに、1層の場合についてその設計のための接近法を明らかにし、そのときに必要となるグラフの平面化問題についても論じる。

第2章では、一般に印刷回路基板を自動設計する際の方針を明らかにし、種々の配線手法について比較考察する。さらにこの考察に基づいて、対象とする印刷回路基板が2層の場合、それに適し、かつ従来の手法の欠点を補った効率的な迷路法の一手法を提案し、これを採用して作製した自動配線プログラムを紹介して、提案した迷路方法の有効性を示す。

第3章では、前章で提案した迷路法を多層印刷回路基板に対して適用可能なように一般化することに関して論じる。特に、基板をグラフで表現することにより、配線問題をグラフ上のある種の木を求める問題に帰着させ、電子計算機の取り扱い易い形にし、これを求めるための迷路法の一手法を提案して評価する。

第4章では、印刷回路基板が1層の場合についての統一的な接近法を明らかにし、その際必要となるグラフの平面化問題について論じる。グラフの平面性判定に関しては、効率的なアルゴリズムが J. Hopcroft と R. E. Tarjan [38] によって既に提案されているが、与えられたグラフが平面グラフでない場合、幾つかの枝を開放除去することによって平面グラフにする問題(平面化問題)に関してはあまり論じられていない。本章では、これに対して効率的な手法を提案し、その実行の手数の評価とプログラムの実験結果からその有効性を示す。

第2章 2層印刷回路基板に対する迷路法

2.1 緒言

回路モジュールを印刷回路基板上で系統的に配置、配線する問題は、印刷回路基板の計算機援用設計の立場から極めて重要であり、これに対して従来から多数の著者が種々の観点より接近法を試みている。

特に、印刷回路基板上の回路モジュールの指定された端子間の配線経路を決定する、いわゆる配線問題に注目すれば、その接近法としてスペースチャネル割当て法^[7]、線分探索法^{[8], [9]}、迷路法^{[2]-[6]}が挙げられる。本章では、このうちの迷路法の一手法に関して、2層印刷回路基板を対象として論じる。

2.2節では迷路法のもつ特徴とそれに着目した意義と述べ、2.3節で問題の定義と接近法の概略を示し、2.4節で従来の手法の欠点を補った効率的な手法を提案する。最後に2.5節でこの手法を採用して作製した自動配線システムの例を示し、この提案された手法の有効性を示す。

2.2 回路基板に対する配線手法

印刷回路基板に対する計算機援用設計手法は、従来から多数の著者により種々のものが提案され、一部実用化されてきた。このうちで、回路モジュールを基板上へ配置する手法に関しては、例えば、予想総配線長を評価基準とするもの^{[22], [23]}、回路モジュール間の結線数を評価基準とするもの^{[25]-[27]}等が提案されているが、結果の評価が、配線を実際に行った後でないと明確にならないという点から、効果的な手法は未だ現われていないようである。これに対して、回路モジュールの基板上での配置が与えられたものとして、必要な配線経路を求める問題は、結果が直接配線率（配線が要求されたなかでの、実際に配線が成功したものの割合）によって示されるという点から、実験的な評価もし易く、現在までに線分探索法、スペースチャネル割当て法、迷路法、single rowに対する配線法^[12]等の各種の特徴をもったものが提案されている。このうち線分探索法は、端子間の配線経路を線分ごとのおおまかな探索によって見出し出すものであるが、線分の数（屈折の数）を多くすると配線の可能性が大きいが、反面、実行時間が増大するという性質をもつものである。スペースチャネル割り当て法は、基板を縦、横のいくつかのスペースに分割し、このスペースに関して配線経路を求めるスペース割当てと、その配線経路をスペース内のチャネル（具体的な位置に対応する）に割当てるチャネル割当てとからなる。

迷路法は、セル構造のマトリックスとして表わされたプリント基板上の2つのセルに対し、一方のセルの回りから始めて他方のセルへ到達するまで、順次セルの探索範囲を拡張することにより、セルの系列としての配線経路を見い出すとするものである。この手続きの内容からわかるように、与えられた2つのセル間の最短の配線経路を求めるものである。

これらの配線手法のうち、代表的な線分探索法、スペースチャネル割当て法、迷路法の3つに関して、実行時間、配線率の点から比較すると、表2.1のようになる。ただし、表中で大、中、小は比較的なものであり、絶対的な大きさを示したものではない。

表2.1 配線手法の比較

配線手法	実行時間	配線率
スペースチャネル割当て法	小	小
線分探索法	中	中
迷路法	大	大

表からわかるように、それぞれの手法には一長一短があり、例えば線分探索法を例にとると、その配線手法の性質上、配線率が小さいあいだは実行時間が小さいが、配線率が上昇するに従ってそれが大きくなり、ついには

配線が不可能になるという場合も生じる。さらに、迷路法の場合には、配線率が小さいあいだは実行時間が大きい、配線が進められるに従ってそれが減少するという性質がある。従って、これらのうちのどれか一つだけで自動配線システムを考えるのは効果的ではない。このような状況を考慮して、各種法の特徴を生かしつつ組み合わせ、互いの欠点を補うようにするという観点に基づき、現在までにいくつかの自動配線プログラムが提案されている。

これらの自動配線プログラムを眺めて見ると、ほとんどのものに迷路法が採用されている。これは、迷路法が極めて実用的な特徴を備えているという点からして当然のことであろう。つまり、迷路法は、基板上で与えられた2つのピン(回路モジュールの端子)を結び配線経路を見い出すという問題において、配線が可能な場合には必ずその一つの配線経路を見い出すことができる。

しかるに、従来の迷路法における Lee^[2], Geyer^[3] の両アルゴリズムにおいては、実際に適用される段階で次のような欠点が指摘される。

- (i) これらのいずれの手法においても、同一信号で結線されるべき端子が3個以上ある場合には、それらを総合的にいかにして結線すべきかについてその具体的手順が述べられていない。
- (ii) 文献[3]の手法においては、配線が可能であるにもかかわらず、それを見い出し得ない場合が生じる(2.4節において指摘される)。

これらの2点のうち、特に(i)に関連して次のことがいえる。すなわち、同一信号網に属する端子が3個以上ある場合(それらの端子のうち幾つかのものは既に結線されている場合を含む)、ある端子から出発する配線経路のうちそれと同一の信号網に属する他の端子あるいはそれらを結ぶ既存の配線経路を経由するものは、従来の手法では見い出されない。従って、同一信号網内のすべての必要な配線を完成させるためには適当に選んだ2端子間の配線を繰り返さざるを得ず、そのために同一のセルを何回か重複して探索する場合が生じる。

本章で提案する迷路法は、与えられた信号網の結線が可能な場合には、必ずそれらを結線するための配線経路を見い出すものであり、しかもただ一回のアルゴリズムの適用により同一信号網の必要な配線を行うところに特徴をもつ。

2.3 迷路法

この節では、問題の定義を行ない、さらに配線可能路とネット配線可能路と定義することにより、従来の手法と本手法との差異を明らかにする。

2.3.1 配線可能路と信号網

印刷回路基板(以下簡単のため基板と呼ぶことにする)の使用有効面は2面であるとし、一方をA面、他方をB面と呼ぶことにして、これらの面は同一サイズの矩形であるものとする。

回路モジュールはA面に実装される。いま両面を格子状の網目に区切り、その分割された各矩形領域をセルと呼び、これらの各セルに対して左から又列目、下から又行目にあるA面およびB面上のセルをそれぞれ $(x, y)_A$, $(x, y)_B$ で表わす。更にセル $(x, y)_A$, $(x, y)_B$ の組をメッシュといい (x, y) で表わす。この様子を図2.1に示す。

〔定義2.1〕任意のセル $(x, y)_D$ ($D=A$ 又は B とし以下でもこのように用いる)に対

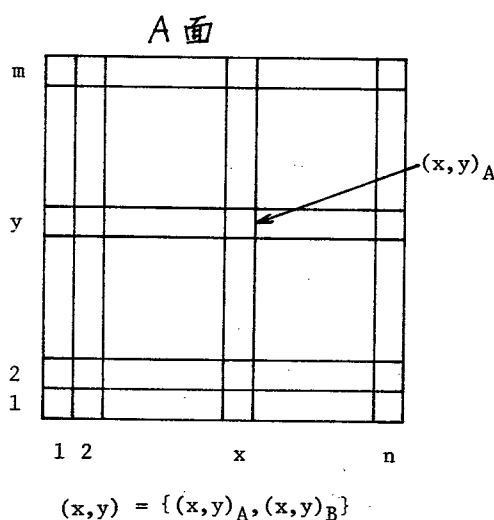


図2.1 基板上のセル

して、それに隣接する4つのセル $(x, y-1)_D$, $(x-1, y)_D$, $(x, y+1)_D$, $(x+1, y)_D$ の各セルを $(x, y)_D$ の隣接セルといい、それぞれ $(x, y)_D^*$ ($k=1, 2, 3, 4$)で表わす。ただし、基板の辺と隅に位置するセルに対しては、それに隣接するそれぞれ3つと2つのセルを上に準じて定義する。更に $(x, y)_D$ のすべての隣接セルの集合を $Adj[(x, y)_D]$ で表わす。(定義終)

[定義2.2] 各セル $(x, y)_D$ の状態を $S(x, y)_D$ で表わし、 $S(x, y)_D$ は状態の組 $\{b, w, \bigcirc, \otimes\}$ のうちのどれかに一つをとるものとし、これらのうち $\bigcirc$, $\otimes$ は以下の制約の下に割当てられるものとする。(注)

$$S(x, y)_A = \bigcirc \Leftrightarrow S(x, y)_B = \bigcirc \quad (2.1)$$

$$S(x, y)_A = \otimes \Leftrightarrow S(x, y)_B = \otimes \quad (2.2)$$

$$S(x, y)_D \in \{\bigcirc, \otimes\} \Rightarrow S(x', y')_D \in \{\bigcirc, \otimes\}, \\ (x', y')_D \in Adj[(x, y)_D] \quad (2.3)$$

(定義終)

具体的には、セルが状態 b にあるとはそのセルが空白であることを意味し、 $\bigcirc$ 又は $\otimes$ にあるとは、そのセルに回路モジュールの端子、またはビアホール(基板上のある位置をA面とB面で同電位に接続するための穴で、配線経路の一部として用いられる。これに対して、回路モジュールと基板に実装するための端子穴をスルーホールということがある)がそれぞれ割当てられていることを意味する(式(2.1), (2.2)参照)。更に、セルが状態 w にあるとはそのセルが配線経路の一部として使用されていることを意味する。

ここで、(2.3)式によって示される制約は、あるセル $(x, y)_D$ にスルーホール又はビアホールが割当てられているとき、その各隣接セルにはスルーホールとビアホールのいずれをも割当てられないことを意味する。すなわち、本章で対象とする配線問題においては、互いに隣接する2つのセルにスルーホールあるいはビアホールを割当ててを禁止する。これは一般の2層印刷回路基板に課せられた物理的制約によるものであるが、この制約がない場合においても以下の議論が成立することに注意しよう。

[定義2.3] 各メッシュ (x, y) に対して、図2.2で示されるような8個のフィールドを設定する。 S_A, S_B フィールドは各セル $(x, y)_A, (x, y)_B$ の各状態を書き込むためのものであり、THフィールドにはそのメッシュがビアホール可能なときは0が、不可能なときには1がそれぞれ書き込まれる。その他のフィールドは後述の配線アルゴリズムを実行する過程において使用される作業領域であり、以下に順次定義する。(定義終)

(注) 命題 P, Q に対して $P \Rightarrow Q$ は“ P ならば Q である”ことを表わす。

S_A	S_B	MS_A	MS_B	F	T_A	T_B	TH
-------	-------	--------	--------	-----	-------	-------	------

図 2.2 メッシュに割り当てられた
8個のフィールド

〔定義 2.4〕 与えられた2つのセル $(x_a, y_a)_{D_a}, (x_b, y_b)_{D_b}$ に対して、以下の3つの条件を満たす相異なるセルの系列 $P \equiv [(x_1, y_1)_{D_1} = (x_a, y_a)_{D_a}, (x_2, y_2)_{D_2}, \dots, (x_n, y_n)_{D_n} = (x_b, y_b)_{D_b}]$ が存在するとき、これを $(x_a, y_a)_{D_a}, (x_b, y_b)_{D_b}$ 間の配線可能路という。

$$(i) \quad S(x_i, y_i)_{D_i} = b \quad (2 \leq i \leq n-1) \quad (2.4)$$

(ii) 各 i ($1 \leq i \leq n-1$) に関して次のいずれか一方の関係が成立する。^(注)

$$(x_i, y_i) = (x_{i+1}, y_{i+1}) \text{ かつ } TH(x_i, y_i) = 0 \quad (2.5)$$

または、

$$(x_i, y_i)_{D_i} \in Adj[(x_{i+1}, y_{i+1})_{D_{i+1}}] \quad (2.6)$$

$$(iii) \quad (x_i, y_i) = (x_{i+1}, y_{i+1}) \Rightarrow (x_{i+2}, y_{i+2}) \neq (x_{i+3}, y_{i+3}) \quad (1 \leq i \leq n-3) \quad (2.7)$$

(定義終)

なお、以下では $(x_a, y_a)_A$ 又は $(x_a, y_a)_B$ のいずれか一方と $(x, y)_D$ 間の配線可能路を、特にメッシュ (x_a, y_a) からセル $(x, y)_D$ への配線可能路という。

〔定義 2.5〕 メッシュ (x_0, y_0) に対して、それからの配線可能路が存在するようなすべてのセル $(x, y)_D$ の集合を (x_0, y_0) の配線可能域という。
(定義終)

いま、注目する2つのセル $(x_a, y_a)_{D_a}, (x_b, y_b)_{D_b}$ 間の配線可能路をなすセルの系列 $P = [(x_1, y_1)_{D_1} = (x_a, y_a)_{D_a}, (x_2, y_2)_{D_2}, \dots, (x_n, y_n)_{D_n} = (x_b, y_b)_{D_b}]$ が存在したとき、この系列 P 上の各セルに対して次の手続きを施すことを考える。

(注) $TH(x_i, y_i) = 0$ は、メッシュ (x_i, y_i) の TH フィールドが 0 であることを示す。図 2.2 に示される他の各フィールドに対しても同様の記法を用いる。

```

procedure WSTATE(P);begin
  i ← 1;
  while i ≠ 1 do begin
    if (xi, yi) = (xi+1, yi+1) then S(xi, yi)Di ← ⊗, S(xi+1, yi+1)Di ← ⊗;
    else if S(xi, yi) = b then S(xi, yi)Di ← w;
    i ← i+1;
  end;
  if S(xn, yn)Dn = b then S(xn, yn)Dn ← w;
end;

```

[定義 2.6] 与えられた2つのセル $(x_a, y_a)_{D_a}$, $(x_b, y_b)_{D_b}$ 間の1つの配線可能路上の各セルに上記の手続き WSTATE を施こして状態を割当てたとき, このセルの系列を $(x_a, y_a)_{D_a}$, $(x_b, y_b)_{D_b}$ 間の配線経路という。 (定義終)

与えられた印刷回路基板上に実装されるべき回路モジュールの各端子をどのメッシュに配置するかが既に決定されており, それらの各端子の位置するメッシュを構成する2つのセルには状態 '0' が割当てられているものとする。いま, 幾つかの配線経路が既に決定されており, それらの経路上のセルでビアホールであるものには '⊗' が, そうでないものには, 'w' が割当てられているものとして, 以下の定義を与える。

[定義 2.7] 同一信号 ϵ によって結線されるべき端子に対応するメッシュ $\tau_{i1}, \tau_{i2}, \dots, \tau_{in}$ の集合を ϵ に対する信号網といい, $N_\epsilon \triangleq \{\tau_{i1}, \tau_{i2}, \dots, \tau_{in}\}$ で表わす。 (定義終)

このようにして定められた各信号網 N_ϵ に対して, 何らかの方法により $N_\epsilon^\# \triangleq \{\tau_{i1}, \tau_{i2}, \dots, \tau_{in}\} \subset N_\epsilon$ の各メッシュは既に配線経路により, 連結されているものとし, $N_\epsilon^* \triangleq N_\epsilon - N_\epsilon^\#$ の各メッシュは結線されずに孤立しているものとする。このとき, $N_\epsilon^\#$ およびそれらのメッシュ間を結ぶ配線経路上にあるセルの集合を N_ϵ^* とし, この N_ϵ^* を N_ϵ に対する既配線領域という。ただし, $N_\epsilon^\# = \emptyset$ の場合を許し, $N_\epsilon^\# \neq \emptyset$ の場合には明らかに $|N_\epsilon^*| \geq 2$ である。更に $N_\epsilon^c \triangleq N_\epsilon^\# \cup N_\epsilon^*$ と定義する。

2.3.2 ネット配線可能路

本章で論じる配線問題とは, 与えられた信号網 N_ϵ ($\epsilon = 1, 2, \dots, n$) のそれぞれに対して $N_\epsilon^\#$ に属するすべてのメッシュと N_ϵ^* に属するどれかのセルとを連結する配線可能路を見出し, かつ, それら配線可能路上の各セルに手続き WSTATE を施こしつつその配線経路を決定することである。

さて, この問題に対して Lee の手法を $N_\epsilon^\# \neq \emptyset$ の場合に限定して適用することを目指す。以下の操作はその大略を示すものである。

[Lee の手法]

操作1: $N_i^* = \emptyset$ (空) ならば操作終了。

操作2: メッシュ $\gamma_a \in N_i^*$ を任意に選び, $N_i^* \leftarrow N_i^* - \{\gamma_a\}$ とする。

操作3: γ_a に関する配線可能域を γ_a よりセルごとに探索し, それが N_i^* に含まれるかどうか1つのセル γ_0 を含んだ段階で探索を終了して操作4へ行く。 N_i^* のどのセルも配線可能域内に見い出されないとき操作1へ戻る。

操作4: γ_0 より γ_a へ至る配線経路 P_{0a} を決定し, P_{0a} 上のすべてのセルを N_i^* へ含め, $N_i^* \leftarrow N_i^* \cup \{\gamma_a\}$ として操作1へ戻る。

この手続きにおいて次のことが指摘される。まず各メッシュ $\gamma_a \in N_i^*$ よりその配線可能域を探索しなければならず, 従って, 同一のセルを重複して探索する場合が生じる。又, 配線可能域の定義 (Lee の手法において, マーキングシーケンスの付されたセルの集合は, 本質的にこの配線可能域と同等である) より, γ_0 が γ_a の配線可能域に含まれない場合でも, γ_a よりあるセル $\gamma'_0 \in N_i^c$ ($\gamma_0 \neq \gamma'_0$) を経由すれば γ_0 に配線可能であるような場合が存在し得る。このため, 配線アルゴリズムとしては γ_a の配線可能域に含まれなくても, N_i^c に含まれるセルを経由すれば γ_a から結線可能であるようなセルをも γ_a からの探索で見出すものが望ましい。このことを考慮して以下の定義を与える。

[定義2.8] 信号網 N_i に対してセル $(x_a, y_a)_{D_a}, (x_b, y_b)_{D_b} (\in N_i^c)$ 間の以下の3つの条件を満たす相異なるセルの系列 $P_i \triangleq [(x_1, y_1)_{D_1} = (x_a, y_a)_{D_a}, (x_2, y_2)_{D_2}, \dots, (x_n, y_n)_{D_n} = (x_b, y_b)_{D_b}]$ が存在するとき, これを $(x_a, y_a)_{D_a}, (x_b, y_b)_{D_b}$ 間の N_i に関するネット配線可能路という。

$$(i) \quad S(x_j, y_j)_{D_j} = b \text{ 又は, } (x_j, y_j)_{D_j} \in N_i^c \quad (1 \leq j \leq n) \quad (2.8)$$

$$(ii) \quad (x_j, y_j) = (x_{j+1}, y_{j+1}) \text{ かつ } TH(x_j, y_j) = 0 \quad (1 \leq j \leq n-1) \quad (2.9)$$

$$\text{又は, } (x_j, y_j)_{D_j} \in Ad_j[(x_{j+1}, y_{j+1})_{D_{j+1}}] \quad (1 \leq j \leq n-1) \quad (2.10)$$

$$(iii) \quad (x_j, y_j) = (x_{j+1}, y_{j+1}) \Rightarrow$$

$$(x_{j+2}, y_{j+2}) \neq (x_{j+3}, y_{j+3}) \quad (1 \leq j \leq n-3) \quad (2.11)$$

(定義終)

セル $(x_a, y_a)_A$ 又は, $(x_a, y_a)_B$ のいずれか一方とセル $(x, y)_D$ 間の N_i に関するネット配線可能路を, 特にメッシュ (x_a, y_a) からセル $(x, y)_D$ への N_i に関するネット配線可能路という。

[定義2.9] 信号網 N_i に対してメッシュ $(x_0, y_0) \in N_i$ からの N_i に関するネット配線可能路が存在するようなセル (x, y) のすべてからなる集合を (x_0, y_0) の N_i に関するネット配線可能域という。(定義終)

これらの定義より、ある与えられた信号網 N_i の結線を考える場合、任意のメッシュ $\gamma_a \in N_i^*$ の N_i に関するネット配線可能域のみを探索することにより、 N_i の結線をすべて行えるかどうか判定できる。以上の考察に基づき、本章で提案する配線アルゴリズムの概要を以下に記す。

[配線アルゴリズム]

操作1: N_i^* よりメッシュ γ_a を任意に選ぶ。

操作2: γ_a の N_i に関するネット配線可能域を探索し、それが N_i^c のすべてのセルを含めば操作3へ移り、それ以外の場合は操作終了する(このとき、 N_i の結線を完成することは不可能と判定される)。

操作3: 配線経路を決定し、操作を終了する。

2.4 アルゴリズム

ここではより具体的に N_i に関するネット配線可能域を探索する探索アルゴリズムSEARCHとその過程で各セルに付したマーキングシーケンスを逆探索することにより配線経路を決定する経路決定アルゴリズムRDETERMINITIONについて述べる。但し、与えられた各信号網 N_i ($1 \leq i \leq n$)に対応して各セルにはその状態が記され、 MS_A, MS_B, F の各フィールドは0に初期化されているものとする。又、注目する信号網 N_i に対しては N_i^* および $N_i^c - \{\gamma_a\}$ の各セルに対応して、そのメッシュの T_A 又は T_B フィールドに目標セルである印‘ t ’が入れられているものとする。ここで γ_a は N_i^* から適当に選ばれたメッシュとする。

2.4.1 探索アルゴリズム

迷路法の基本となる手続きは、ある指定されたメッシュ $\gamma_a \in N_i^*$ に対してその N_i に関するネット配線可能域をセルごとの探索によって順次拡張しつつ求めることにある。すなわち、まず γ_a をリストFCS (Front Cell Set)へ入れ、この中から1つのメッシュを取り出してそのメッシュから隣接セルを探索し、 N_i に関するネット配線可能域に含まれるセルへマーキングシーケンス n ($n=1, 2, 3$)を付し、対応するメッシュをリストNCS (Next Cell Set)へ入れる。この操作を、FCSにある各メッシュについて繰り返えし、FCSが空になればあらためてNCSをFCSと見なして操作を続ける。ここでマーキン

グシーケンスは、後述の経路決定アルゴリズムにおいて使用されるものであり、セル $(x, y)_D$ に対してはメッシュ (x, y) の MS_D フィールドへその値が書き込まれる。なお、アルゴリズム中、 $I_m(x) \equiv x \pmod{m}$ ($1 \leq I_m(x) \leq m$)とし、 $x = m \cdot n$ (m は整数)のとき $I_m(x) = n$ であるとし、さらに、 $D=A$ ならば $\overline{D}=B$ 、 $D=B$ ならば $\overline{D}=A$ であるものとする。

```

procedure SEARCH( $N_i$ );begin
  FCS  $\leftarrow \phi$ , NCS  $\leftarrow \phi$ , T  $\leftarrow \phi$ , CHECK  $\leftarrow 0$ ,  $i \leftarrow 1$ ;
  FCS  $\leftarrow \{\tau_s=(x_0, y_0)\}$ ;
   $MS_A(x_0, y_0) \leftarrow 1$ ,  $MS_B(x_0, y_0) \leftarrow 1$ ;
  while CHECK=0 do begin
    if FCS $\neq\phi$  then begin
      pick up any mesh  $(x, y)$  out of FCS;
      for  $(x, y)_D$ , where  $MS_D(x, y)=i$ , do begin
        for adjacent cells  $(x, y)_D^k$  of  $(x, y)_D$ , where  $MS_D(x, y)_D^k=0$  do begin
          if  $T_D(x, y)_D^k=t$  or  $S(x, y)_D^k=b$  and  $T_{\overline{D}}(x, y)_D^k=t$  and  $TH(x, y)_D^k=0$  then begin
            if  $T_D(x, y)_D^k=t$  then  $(x_t, y_t)_{D_t} \leftarrow (x, y)_D^k$ ;
            else  $(x_t, y_t)_{D_t} \leftarrow (x, y)_D^k$ ;
             $MS_{D_t}(x_t, y_t) \leftarrow I_3(i+1)$ ;
            T  $\leftarrow (x_t, y_t)_{D_t}$ ;
            if  $TH(x_t, y_t)=0$  then  $MS_{\overline{D}_t}(x_t, y_t) \leftarrow I_3(i+1)$ ;
            if any meshes of  $N_i^{\#}$  or all cells of  $N_i^{\#}$  are not reached then begin
              if  $(x_t, y_t) \notin N_i^{\#}$  then begin
                for  $(x, y)_D \in N_i^{\#}$  do begin
                   $MS_D(x, y) \leftarrow I_3(i+1)$ ;
                  if  $S(x, y)_{\overline{D}}=b$  and  $TH(x, y)=0$  then  $MS_{\overline{D}}(x, y) \leftarrow I_3(i+1)$ ;
                end;
              NCS  $\leftarrow \{N_i^{\#}\}$ ;
            end;
            else NCS  $\leftarrow \{(x_t, y_t)\}$ ;
          end;
          else CHECK  $\leftarrow 1$ ;
        end;
      else if  $S(x, y)_D^k=b$  then begin
         $MS_D(x, y)_D^k \leftarrow I_3(i+1)$ ;
        if  $F(x, y)_D^k \neq 1$  then begin
          NCS  $\leftarrow \{(x, y)_D^k\}$ ;
          if  $S(x, y)_D^k=b$  then begin
            if  $TH(x, y)_D^k=0$  then begin
               $MS_{\overline{D}}(x, y)_D^k \leftarrow I_3(i+1)$ ;
               $F(x, y)_D^k \leftarrow 1$ ;
            end;
          end;
          else  $F(x, y)_D^k \leftarrow 1$ ;
        end;
      end;
    end;
  end;
  end;
  end;
  end;
  end;
  else begin
    if NCS= then CHECK  $\leftarrow 1$ ;
    else FCS  $\leftarrow$  NCS, NCS  $\leftarrow \phi$ ,  $i \leftarrow I_3(i+1)$ ;
  end;
end;
end;

```

SEARCH を適用した結果, MSD フィールドが 0 でなく, $S(x, y)_D = b$ あるいは $(x, y)_D \in T$ であるどのセル $(x, y)_D$ に関しても次のことがいえる。

(I) $MSD(x, y)^k = I_3(MSD(x, y) - 1)$ となるセル $(x, y)^k \in Adj[(x, y)_D]$ が存在する場合: (x_0, y_0) から $(x, y)_D$ への N_i に関するネット配線可能路 (これは明らかに存在する) のうちで状態が b であるセルの個数が最少のものを $P_1, P_2, \dots, P_m$ とすれば, 少なくともどれか一つの P_ℓ ($1 \leq \ell \leq m$) は $(x, y)^k$ をその系列に含む。

(II) I 以外の場合: (x, y) はビアホール可能であり, $MSD(x, y) = MSD(x, y)$ となって, このセル $(x, y)_D$ に関して (I) が成立する。

これら I, II の考察に基づいて, つぎに経路決定アルゴリズムを考える。

2.4.2 経路決定アルゴリズム

SEARCH を適用した結果, 幾つかの目標セルがリスト T へ入れられ, この T に対して以下のアルゴリズムを適用することにより, 信号網 N_i の各孤立部分 (それぞれを T の各セルが代表する) を結ぶ配線経路が決定される。このとき, 得られた配線経路はセルの系列としてリスト P へ貯えられ順次出力される。

procedure RDETERMINATION(N_i, T); begin

while $T \neq \emptyset$ do begin

 CHECK $\leftarrow 0$;

 pick up any cell $(x_t, y_t)_{D_t}$ out of T ;

$(x_c, y_c)_{D_c} \leftarrow (x_t, y_t)_{D_t}$;

$P \leftarrow \emptyset, k \leftarrow 1$;

while CHECK=0 do begin

$P \leftarrow (x_c, y_c)_{D_c}$;

 A: search four adjacent cells $(x_c, y_c)_{D_c}^k$ in the order of $k=k, I_4(k+2), I_4(k+1), I_4(k+3)$;

if, for some $k, MSD_{D_c}(x_c, y_c)^k = I_3(MSD_{D_c}(x_c, y_c) - 1)$ then begin

if $S(x_c, y_c)_{D_c}^k = b$ then begin

if $S(x_c, y_c)_{D_c} = b$ then $S(x_c, y_c)_{D_c} \leftarrow w$;

$(x_c, y_c)_{D_c} \leftarrow (x_c, y_c)_{D_c}^k$;

end;

else begin

if $S(x_c, y_c)_{D_c} = b$ then $S(x_c, y_c)_{D_c} \leftarrow w$;

 output P ;

 CHECK $\leftarrow 1$;

end;

end;

else begin

$S(x_c, y_c)_{D_c} \leftarrow \otimes, S(x_c, y_c)_{\bar{D}} \leftarrow \otimes$;

for $(x_c, y_c)^k$ do $TH(x_c, y_c)^k \leftarrow 1$;

$(x_c, y_c)_{D_c} \leftarrow (x_c, y_c)_{\bar{D}_c}$;

end;

end;

end;

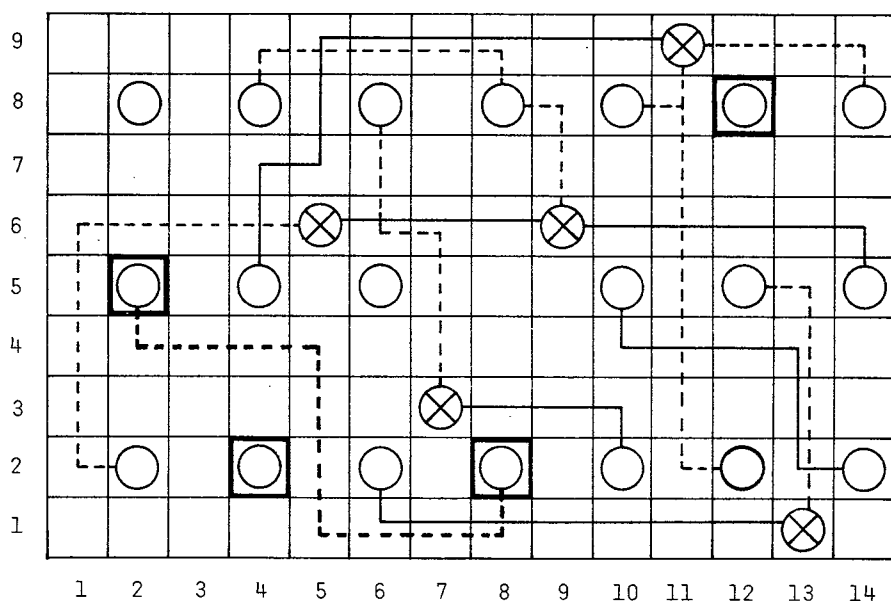
end;

アルゴリズム中、Fフィールドは同一セルが不必要に重複してNCSへ入れられるのを防ぐためのフラッグである。また、 $(x_c, y_c)_{D_c}$ からの探索の順番をAのように設定することにより、各段階における配線経路のジグザグの増加が押えられる。更に次のことが明らかである。

〔命題2.1〕 与えられた信号網 N_i とメッシュ $(x_0, y_0) \in N_i^{\#}$ に対して、 (x_0, y_0) の N_i に関するネット配線可能域に存在する幾つかのセルの集合 $T = \{(x_a, y_a)_{D_a}, (x_b, y_b)_{D_b}, \dots, (x_o, y_o)_{D_o}\}$ を考えた場合、 (x_0, y_0) を出発メッシュ、Tの各セルを目標セルとして上記の手続きを適用すれば、これらを同一信号とするような配線経路の組が必ず決定される。

(例 2.1)

配線が一部なされている印刷回路基板が図2.3のように与えられ、且つ信号網 $N_i = \{(2, 5), (4, 2), (8, 2), (12, 8)\}$ に注目したとき、この N_i を前節で提案したアルゴリズムSEARCHとRDETERMINATIONを適用して配線することを考える。但し、図においてそれぞれ実線はA面の、破線はB面の配線経路を表わすものとし、 N_i において端子 $(2, 5)$ と $(8, 5)$ とは既にB面で配線されているものとする。これは図中太い破線で示されている。



太くで囲まれたメッシュが N_i を表わす

図2.3 与えられた信号網と基板

A 面

9					w	w	w	w	w	w	⊗			
8		○		○	w	○		○		○		○		○
7				w	w									
6				w	⊗	w	w	w	⊗	w	w	w	w	w
5		○		○		○				○		○		○
4										w	w	w	w	
3							⊗	w	w	w			w	
2		○		○		○		○		○		○	w	○
1						w	w	w	w	w	w	w	⊗	
	1	2	3	4	5	6	7	8	9	10	11	12	13	14

B 面

9				w	w	w	w	w			⊗	w	w	w
8		○		○		○		○	w	○	w	○		○
7						w			w		w			
6	w	w	w	w	⊗	w	w		⊗		w			
5	w	○		○		○	w			○	w	○	w	○
4	w	w	w	w	w		w				w		w	
3	w				w		⊗				w		w	
2	w	○		○	w	○		○		○	w	○	w	○
1					w	w	w	w					⊗	
	1	2	3	4	5	6	7	8	9	10	11	12	13	14

図2.4 図2.3に対応するセルの状態、

図2.4は、図2.3において表わされた印刷回路基板の各セルにその状態を書き込んだものであり、とくに状態wは空白で示されている。

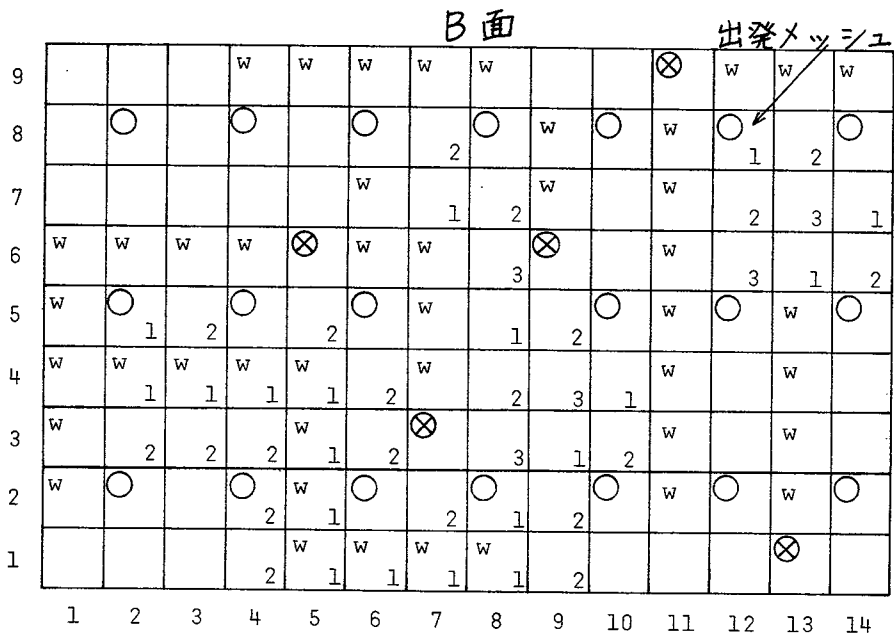
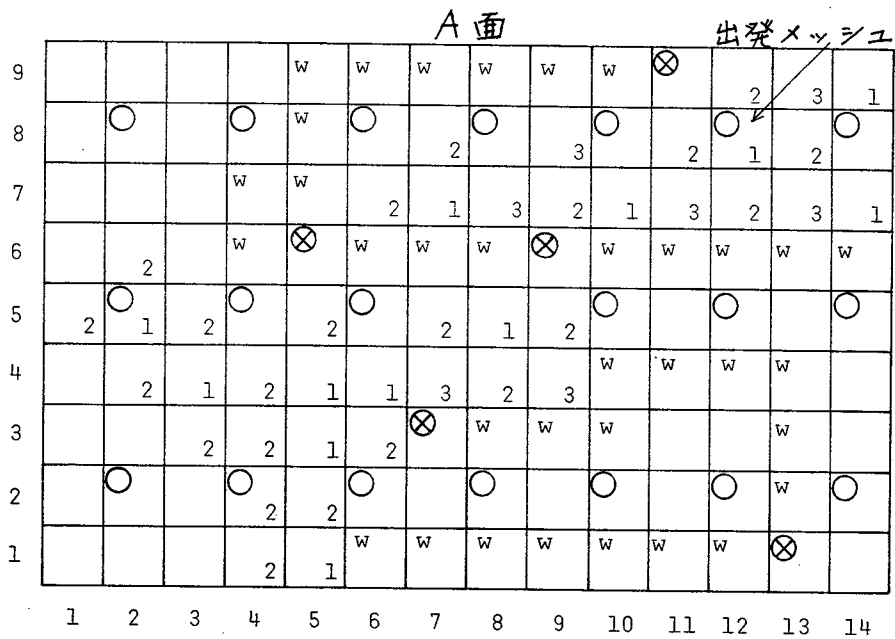


図2.5 図2.4 に SEARCH を適用した結果

N_2 からメッシュ (12, 8) を選択し, 探索アルゴリズム SEARCH を適用して各セルにマーキングシーケンスを書き込むと, 図2.5 のようになる。このと

き、目標セルが入れられたリストTは、 $T = \{(8, 2)_B, (4, 2)_B\}$ となる。

図2.6は、図2.5と目標セルリストTに対して経路決定アルゴリズム RDETERMINATION を施した結果である。決定された配線経路は図中2重線で示されており、出力されたEの内容はそれぞれ次のようになる。

$[(4, 2)_B, (5, 2)_B]$

$[(8, 2)_B, (8, 3)_B, (8, 4)_B, (8, 5)_B, (8, 6)_B,$

$(8, 7)_B, (7, 7)_B, (7, 7)_A, (8, 7)_A, (9, 7)_A,$

$(10, 7)_A, (11, 7)_A, (12, 7)_A, (12, 8)_A)]$

(例終)

図2.6において同じセル(8,7)をA,B面で一度ずつ通る配線経路があるが、このような経路は Geyer の手法では不可能である。しかし、本手法においてはマーキングシーケンスを各セルごとに割当てたために可能になった。更にメッシュ(12,8)から(4,2)へ至るすべての経路は、(8,2)と(2,5)とを結ぶ既配線領域 N_i^* (図2.3の太線で表わされた経路が通るセルの集合)のどれかのセルを必ず経由する。すなわち、(12,8)の配線可能

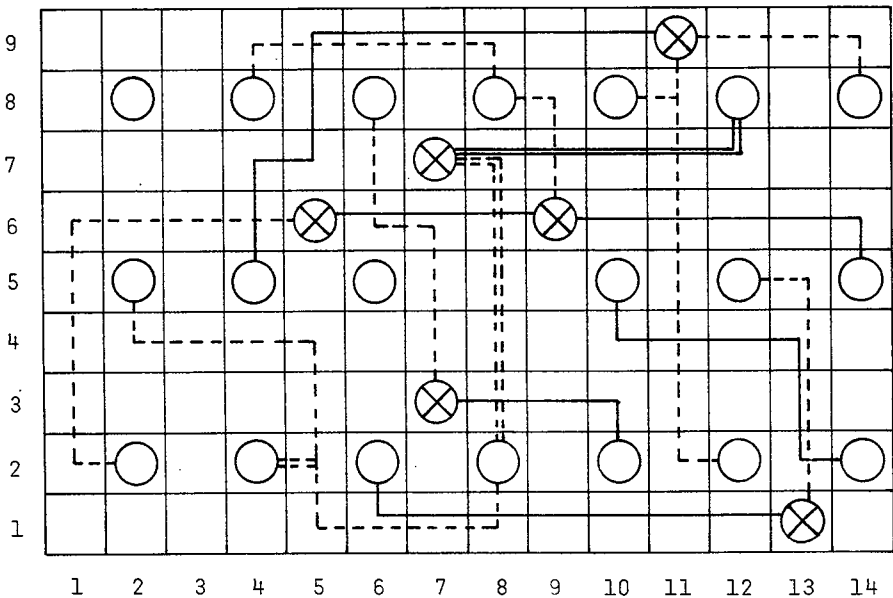


図2.6 図2.5にRDETERMINATION
を適用した結果

域に(2, 5)は含まれないが, (12, 5)の N_2 に関するネット配線可能域には含まれる。この場合, 従来の手法は配線可能域を探索するものであったので, 手続きの一回の適用では必要なすべての配線経路を見出すことは出来ない。従って, 他のセルを出発点として再び同じ手続きを適用しなければならない。しかし, 本手法はネット配線可能路を探索するものであるから, 手続きの一回の適用ですべての必要な配線経路が得られる。

2.5 例題

この節では, 以上提案した迷路法を採用して作製した自動配線プログラム OSACA (Organized System for Automated Connection-Routing Algorithm) の概要を記し, その実験結果を示す。

OSACA は3つの特徴の異なる配線手法, スペースチャネル割当て法, 線分探索法, 迷路法から成り, 図2.7に示すように, 迷路法は最終段階のサブ配線システムとして組み入れられている。すなわち, まず最初に実行時間の短いスペースチャネル割当て法を用いて距離の遠い端子間の配線を行ない, つぎに, 線分探索法を用いて可能な限り配線を実行する。最後に, 残された配線を迷路法を用いて行なう。最後の段階では, 配線密度がかなり高くなっており, 探索するセルの個数が減少しているので, 迷路法をこの段階で用いるのは実行時間の面でも有効である。

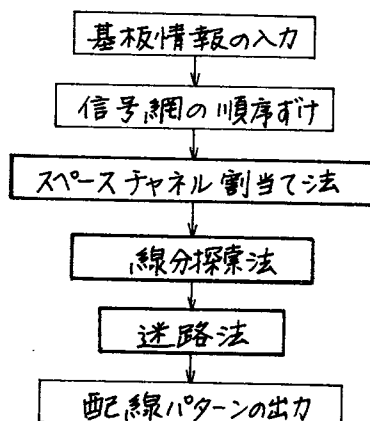


図2.7 OSACAの流れ図

表2.2にOSACAの実験例を示す。

表2.2 OSACAの実験結果(IBM 370-168)

	ICの数 (ピン数)	信号網 の数	配線率/ 実行時間		合 計
			先行する手法	本手法	
1	33 (586)	94	96.2%/14.1sec	3.8%/ 5.3sec	100%/19.4sec
2	64 (793)	145	87.7/ 24.0	12.3 /11.5	100 /35.5
3	42 (616)	154	96.6 /30.6	2.3 / 7.3	98.9/37.9
4	57 (970)	271	93.3 /37.9	4.4 / 5.0	97.7/42.9

入力のパラメータとしては、基板の大きさ、実装する回路モジュールの種類と配置、信号網の数、等のものがあり、各データについてその結果を同一基準で比較するのは困難であるが、配線率の面からは良好な結果であるといえる。とくに、迷路法に関しては、その実行時間が実用的な範囲にあることがわかる。

2.6 結言

本章では、2層印刷回路基板に対する一つの接近法を示し、そのとき必要となる配線手法として迷路法に注目し、従来の迷路法の欠点を克服したより効率的な一手法を提案した。更に、それをサブ配線システムとして組み入れた自動配線プログラムの例を示した。この結果、迷路法が実行時間の上で充分実用に耐えるものであることがわかったが、この手法の持つ極めて実用的な特徴を考えると、今後とも本手法が大いに利用されるであろう。

第3章 迷路法の一般化

3.1 緒言

大規模な電子機器システムに対する高密度パッキング化の問題は極めて重要であるが、これを実現するための有力な一手段として印刷回路基板の多層化がある。ここで、印刷回路基板の設計自動化の一部をなす配線問題に着目すると、これに対して、従来から種々の異なる観点より考察がなされ、いろいろな接近法^{[2]-[3]}が提案されかつ実用化されてきた。しかし、これらの多くは一般の多層印刷回路基板の場合にそのまま適用するには問題がある。そこで本章では特に前章で考察した迷路法に注目し、これを多層基板に拡張して適用することに関して論じる。

3.2 配線問題の定式化

この節では、基板をグラフ表現し、配線問題をそのグラフ上で定義する。このことにより、問題が明確に定式化され、さらに電子計算機の取り扱い易い形になる。

3.2.1 基板のグラフ表現

多層印刷回路基板上に搭載されるべき回路モジュールは、基板の最上層へ実装される。その端子はスルーホールと呼ばれる導通穴へさし込まれ、これはどの層からもアクセスできる。同電位にされるべき端子の各集合（信号網）は、いくつかの層に印刷されたリード線と、それら異なる層上にあるリード線を互いに導通させるためのビアホールと呼ばれる導通穴とからなる配線経路により連結される。本章で論じる配線問題とは、各回路モジュールの端子の位置があらかじめ割り当てられており、ビアホールとなり得る位置は指定されているという条件のもとで、同電位に結ばれるべき端子の集合のそれぞれが連結されるように、基板上の配線経路の配置パターンを決定することである。

議論をより厳密に進めるため、以下では対象とする基板を次のようなものであると仮定する。

- (i) 各層は図3.1のように M 個の垂直チャネル（以下では単にVチャネルという）と m 個の水平チャネル（Hチャネルという）を持ち、リード線を垂直および水平のそれぞれのチャネルのみに印刷することができ

る。

- (ii) スルーホールとビアホール
の位置は、各層のVチャネル
とHチャネルの交差する点の
うち、ある限られた幾つかだ
けに許されている。

基板にさし込まれるすべての端子
の集合の分割(直和分割)をネット
リスト $\mathcal{N} = \{N_1, N_2, \dots, N_k\}$ によ
って表わす。ここで $N_i \in \mathcal{N}$ は同電位
にされるべき端子の集合であり、以
後ネットと呼ぶ。ただし、 $|N_i| =$
 1 ^(注)ならば、その端子は実際には使
用されていない。さらに上記(ii)
の仮定のように、各端子の位置は各
層のVチャネル x_k とHチャネル y_k
の交点 (x_k, y_k) を貫通するように割当てられている。

さて、このような基板の状況に対応して、配線問題をグラフ理論的立場から
定式化するために、基板をグラフで表現することを考える。まず、必要な諸定
義をしておく。

【定義3.1】 グラフ G とは、空でない有限集合 V 、 V と素な有限集合 E 、
および写像 $\alpha: E \rightarrow V \Delta V$ で定義される合成概念 $G = [V(G), E(G), \alpha]$
である。ただし集合 A に対して $A \Delta A$ は、 A におけるすべての非順序対の
集合を表わすものとする。 (定義終)

$V = V(G)$ の各元を節点といい、 $E = E(G)$ の各元を枝という。以下で
は G を単に $G = [V, E]$ と書く。

【定義3.2】 G において同一節点 v_1 を両端点にもつ辺を v_1 に接続する自
己ループという。 (定義終)

【定義3.3】 G において、異なる2つの枝 e_1, e_2 に対して、 $\alpha(e_1) =$
 $\alpha(e_2) = (v_1, v_2)$ ならば、 (v_1, v_2) は並列枝であるという。 (定義終)

【定義3.4】 グラフ $G = [V, E]$ に対して、グラフ $G_0 = [V_0, E_0]$ が
(i) $V_0 \subset V, E_0 \subset E$

(ii) G_0 の写像 α_0 は G の写像 α の E への縮小した縮小写像である。

の2条件を満たすとき、 G_0 を G の部分グラフという。 (定義終)

【定義3.5】 グラフ G の相異なる枝の順序列 $\triangleq [e_1, e_2, \dots, e_l]$

(注) 集合 A に対して、 $|A|$ は A の元の数を表わすものとする。

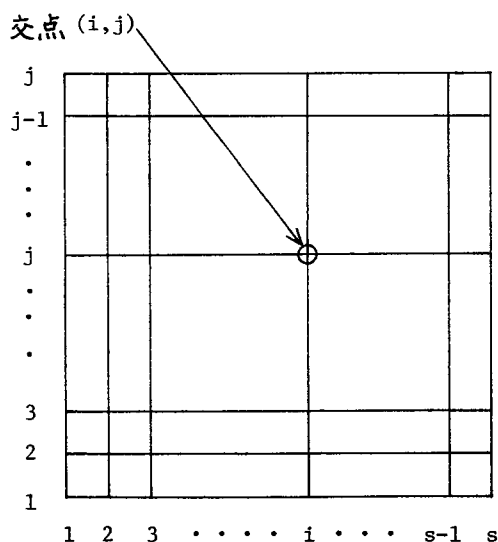


図3.1 基板の第1層

において,

$$\Phi(e_i) = (v_{i-1}, v_i) \quad (i=1, 2, \dots, l) \quad (3.1)$$

とするような相異なる節点 $v_1, v_2, \dots, v_l$ が存在するとき, この Φ が通る枝の集合 E_Φ によって生成される G の部分グラフ P を v_0, v_l 間の道という。とくに, v_0 と v_l に関してだけ $v_0 = v_l$ が成立する場合, P を閉路という。 (定義終)

【定義 3.6】 グラフ G の相異なるどの 2 個の節点間にも道が少なくとも一つ存在するとき, この G を連結グラフという。 (定義終)

【定義 3.7】 連結グラフ G の木とは, $G = [V, E]$ の連結部分グラフ $G' = [V, E']$ ($E' \subset E$) で閉路を含まないものである。 (定義終)

ただし以下では, 閉路を含まない G の連結部分グラフ $G'' = [V'', E'']$ ($V'' \subset V, E'' \subset E$) を, 単に木ということがある。

【定義 3.8】 自己ループを含まないグラフ $G = [V, E]$ ($|E| \geq 2$) のどの 2 つの節点をも同時に含む閉路が存在するとき, G は 2 連結であるという。 (定義終)

【定義 3.9】 グラフ G の 2 節点 v_i, v_j に対して, これらを両端点にもつ枝が存在するとき, v_i, v_j は互いに隣接するといひ, v_i に隣接する節点の集合を $Adj(v_i)$ で表わす。 (定義終)

【定義 3.10】 グラフ G の各節点 v_i に対して, 順序対の集合 $\{v_i \times Adj(v_i)\}$ のすべての元を任意の順序に並べた系列 $\sigma(v_i)$ を v_i の隣接リストといひ, これらのリストの集合 $\Sigma \equiv \{\sigma(v_i) \mid v_i \in V\}$ を G の隣接構造という。 (定義終)

これらの定義を用いて, 基板の状況に対応した次のようなグラフ $G^* = [V, E]$ を定義する。

(i) 各節点 $v_i = [x_i, y_i, z_i] \in V$ は, 第 z 層交差 (x_i, y_i) に 1 対 1 に対応する。

(ii) $v_i = [x_i, y_i, z_i]$ と $v_j = [x_j, y_j, z_j]$ との間に枝 $(v_i, v_j) \in E$ が存在するのは, 次の 2 つの制約のいずれか一方が成立するとき, かつそのときに限る。

$$1^\circ \quad |x_i - x_j| + |y_i - y_j| = 1, \quad z_i = z_j \quad (3.2)$$

$$2^\circ \quad x_i = x_j, \quad y_i = y_j, \quad |z_i - z_j| = 1 \quad (3.3)$$

かつ, v_i と v_j とにそれぞれ対応する交差は, スルーホールまたはビアホールによって結ばれることが許されている。

3.2.2 配線問題

ここでは前節で述べた配線問題を、先に定義したグラフ G^* を用いて定式化する。

【配線問題】： グラフ G^* において、回路モジュールの各端子（スルーホール） $t_k \in N_k$ ($1 \leq k \leq K$) に対応して、節点 $[x_{ik}, y_{ik}, j]$ ($j=1, 2, \dots, r$) と枝 $([x_{ik}, y_{ik}, j], [x_{ik}, y_{ik}, j+1])$ ($j=1, 2, \dots, r-1$) (ただし r は基板の層数) とからなる道 $P_{ik} = ([x_{ik}, y_{ik}, 1], ([x_{ik}, y_{ik}, 1], [x_{ik}, y_{ik}, j+1]), [x_{ik}, y_{ik}, j+1], \dots, ([x_{ik}, y_{ik}, r-1], [x_{ik}, y_{ik}, r]), [x_{ik}, y_{ik}, r])$ を考える。配線問題とは、この各 $\{P_{ik} | t_k \in N_k\}$ を連結する互いに節点を共有しない木 T_k ($1 \leq k \leq K$) を求めることである。

ただし、各 T_k は信号網 N_k に対応しており、 N_k に含まれるすべての端子（それぞれが道に対応する）と、それらを連結する配線経路（道に対応する）からなる木である。

第2章で述べたように、一般に自動配線システムは異なる特徴をもつ幾つかの手法を有機的に組み合わせるべきであるが、この観点から、本章で提案するアルゴリズムが、他の配線手法が実行されたあとの段階でも適用されるようにより一般性を持たせて上記の問題を以下のように書き直す。

【一般化した配線問題】： グラフ G^* 上で、互いに節点を共有しない木 H_k の集合 $W = \{H_1, H_2, \dots, H_m\}$ と W の分割 $\pi = \{V_1, V_2, \dots, V_k\}$ とが与えられたとき、各 V_k に属するすべての木を部分グラフとして持つような、互いに節点を共有しない木 T_k ($1 \leq k \leq k$) を求めよ。

このとき、各 H_k がすべて端子に対応する道であるとすれば、前出の配線問題と同じになるが、一般には、端子と配線経路に対応すると考えられる。

3.3 アルゴリズム

前節に定義された配線問題に対して、以下では一つの有効なアルゴリズムを提案する。このアルゴリズムは、前進手続き FORWARD と後進手続き BACKWARD とから成る。木の集合 $V_k \in \pi$ ($|V_k| \geq 2$) が与えられたとき、前者は、ある選ばれた木 $H_{k_0} \in V_k$ から、 $V_k - \{H_{k_0}\}$ のすべてのに到達するまで、 $H_k \in W - V_k$ のどの節点をも通らずに到達可能なすべての節点を探索するものであり、後者は、求めようとする木 T_k を作るために、 V_k に付け加えらるべき幾つかの道を決定するものである。

3.3.1 緒定義

各節点 $v_k \in V$ と各枝 $e_l \in E$ とに付随して、ネット番号 $N(v_k)$ と $N(e_l)$ とをそれぞれ次のように与える： $N(v_k) = i$ であるのは、 v_k が V_i のどれかの木に含まれるとき、かつそのときに限るものとし、 $N(v_k) = 0$ であるのは、 v_k が W のどの木にも含まれないとき、かつそのときに限るものとする。 $N(e_l)$ に対しても同様に定義する。

E を以下の式のように、 E_{xy} と E_z とに分割する。ただし、 $v_i = [x_i, y_i, z_i]$, $v_j = [x_j, y_j, z_j] \in V$ とする。

$$E_{xy} \triangleq \{(v_i, v_j) \in E \mid |x_i - x_j| + |y_i - y_j| = 1, z_i = z_j\} \quad (3.4)$$

$$E_z \triangleq \{(v_i, v_j) \in E \mid |x_i - x_j| = |y_i - y_j| = |z_i - z_j| = 1\} \quad (3.5)$$

道 $P = [v_0, [v_0, v_1], v_1, \dots, v_{p-1}, (v_{p-1}, v_p), v_p]$ を単に節点の系列 $[v_0, v_1, \dots, v_p]$ と書く。

【定義 3.11】 木の集合 $V_k \in \pi$ に関して、 $N(v_k) = i$ または 0 である節点 v_k を考えたとき、 $N(v_0) = i$ または 0 であり、かつ v_k から以下の 2 つの条件 (i), (ii) を満たす道 $P = [v_0 = v_k, v_1, \dots, v_p = v_0]$ が存在するならば、この v_0 を V_k に関する v_k の許容節点 (admissible vertex) という。

$$(i) \quad (v_0, v_1) \in E_{xy} \quad \text{かつ} \quad (v_k, v_{k+1}) \in E_z \\ (k = 1, 2, \dots, p-1) \quad (3.6)$$

$$(ii) \quad N(v_k) = 0 \quad (k = 1, 2, \dots, p-1) \quad (3.7)$$

(定義終)

【定義 3.12】 木 $H_k \in W$ が与えられたとき、 H_k^* は H_k に対して次の操作を再帰的に適用することにより得られる G の部分グラフである。ただし、最初に H_k^* を H_k に一致させておく。

操作： $(v_i, v_j) \in E_z$ に対して、 $v_i \in V(H_k^*)$, $v_j \notin V(H_k^*)$, かつ $N(v_j) = 0$ が成立するならば、 v_j と (v_i, v_j) とを H_k^* に加える。

(定義終)

3.3.2 前進アルゴリズム

つぎに前進手続きについて述べる。これは、すべての $H_{ij} \in V_k - \{H_{i_0}\}$ に到達するまで、どの $H_{ij} \in W - V_k$ の節点をも通らずに H_{i_0} から到達可能なすべての節点 v_k に、 H_{i_0} から始めて順次番号 $M(v_k) = 1, 2, 3, \dots$ を割当てて行くものである。後に述べる後進手続きにおいて、この割当てられた番号を用いて求めようとする。

幾つかの道が決定され、これらの道は、 γ_i と共に木 T_i を構成するために付加されるべきものである。

以降では、ある特定の $\gamma_i \in \pi$ ($|\gamma_i| \geq 2$) に対して適用される手続きを述べる。ただし、相異なる $H_{i_k}, H_{i_l} \in \gamma_i$ に対しては、 $V(H_{i_k}^*) \cap V(H_{i_l}^*) = \emptyset$ であるものと仮定する。そうでない場合には、 H_{i_k} と H_{i_l} とは E_Z に属する枝のみからなる道により連結できるので、この道を見出すのは容易である。なお、整数変数 m は、番号 $M(\cdot)$ の値を決定するためのものであり、リスト L は、後進手続きにおいて逆探索の始点となる節点を貯えるためのものである。

```

procedure FORWARD( $N_i$ ); begin
  choose a tree  $H_{i0} \in \psi_i$ ;
  for  $v_k \in H_{i0}^*$  do  $M(v_k) \leftarrow 1$ ;
  for  $v_h \in V(G) - V(H_{i0}^*)$  do  $M(v_h) \leftarrow 0$ ;
   $L \leftarrow \emptyset, \ell \leftarrow 0, m \leftarrow 1$ ;
  while  $\ell \neq |\psi_i| - 1$  do begin
    if there exists a vertex  $v_k$ , where  $v_k$  is not checked and  $M(v_k) = m$ , then begin
E:   choose such a vertex  $v_k$  and check this;
F:   seek a set  $U$  of all the admissible vertices  $v_u$  of  $M(v_u) = 0$  for  $v_k$  associated
      with  $\psi_i$ ;
A:   for  $v_u \in U$  do  $M(v_u) \leftarrow I_3(m+1)$ ;
      if there exists  $v_u \in U$  which is contained in any  $H_{ij} \in \psi_i - \{H_{i0}\}$  then begin
B:   for  $H_{ij} \in \psi_i - \{H_{i0}\}$ , where  $V(H_{ij}) \cap U \neq \emptyset$ , do begin
        choose a vertex  $v_j$  from  $V(H_{ij}) \cap U$  arbitrarily;
         $L \leftarrow \{v_j\}$ ;
         $\ell \leftarrow \ell + 1$ ;
C:   for  $v_h \in H_{ij}^*$  do  $M(v_h) \leftarrow I_3(m+1)$ ;
      end;
    end;
  end;
  else begin
D:   if there exists a vertex  $v_k$ , where  $v_k$  is not checked and  $M(v_k) = I_3(m+1)$ ,
      then  $m \leftarrow I_3(m+1)$ ;
      else  $\ell \leftarrow |\psi_i| - 1$ ;
    end;
  end;
end;

```

3.3.3 後進手続き

前進手続き FORWARD が施こされたグラフ G^* に関して次の定義を与える。

[定義 3.13] $M(v_a)$ キ 0 である節点 v_a が与えられたとき、以下に示す 3 つの条件

$$(I) \quad p \geq 1, (v_{k-1}, v_k) \in E_Z, M(v_k) = M(v_a) \quad (k = 1, 2, \dots, p),$$

$$N(v_k) = O \quad (k = 1, 2, \dots, p-1), N(v_p) = N(v_a) \quad (3.8)$$

$$(II) \quad p = 1, (v_a, v_b) \in E_{xy}, M(v_b) = I_3(M(v_a) - 1) \quad (3.9)$$

(Ⅲ) $p \geq 2$, $(v_{k-1}, v_k) \in E_z$, $M(v_k) = M(v_a)$, $N(v_k) = 0$
 $(k=1, 2, \dots, p-1)$, $(v_{p-1}, v_p) \in E_{xy}$,

$$M(v_b) = I_3(M(v_a) - 1) \quad (3.10)$$

をそれぞれ満す道 $P = [v_0 = v_a, v_1, \dots, v_p = v_b]$ を經由して到達可能な節
 点 v_b ($v_b \neq v_a$, $M(v_b) \neq 0$) が存在するならば, この v_b をそれぞれ v_a の I
 型, II 型, III 型の節点と呼び, その道を v_a から v_b へ至る後進道 (backward
 Path) という。 (定義終)

この定義を用いて, 求めようとする木 T_2 を構成する道を決定するための後進
 手続き BACKWARD を次に記述する。

```

procedure BACKWARD( $N_1, L$ ); begin
  while  $L \neq \emptyset$  do begin
    CHECK  $\leftarrow 0$ ;
    A: take a vertex  $v_a$  out of  $L$  in the fashion of first-in and first-out;
        $v'_a \leftarrow v_a$ ;
        $L \leftarrow L - \{v_a\}$ ;
       while CHECK=0 do begin
        B: if there exists any vertex  $v_b$  of type I for  $v'_a$  then begin
        C:   if there exists any vertex  $v_b$  of type II for  $v'_a$  then begin
        D:     if there exists any vertex  $v_b$  of type III for  $v'_a$  then begin
        E:       for vertex  $v_k (\neq v_b)$  and edge  $e_k$  on backward path from  $v_a$  to  $v_b$  do begin
                  $N(v_k) \leftarrow i$ ;
                  $N(e_k) \leftarrow i$ ;
                 end;
        F:       if  $N(v_b) \neq i$  then  $v'_a \leftarrow v_b$ ;
        G:       else CHECK  $\leftarrow 1$ ;
                 end;
               end;
             end;
           end;
         end;
       end;
    end;
  end;

```

この手続きに関してつぎの2つの命題が成立する。

[命題3.1] B, C, Dにおいて, 節点 v_b ($\neq v_a$) の I, II, または III
 型の節点 v_b が少なくとも一つは存在する。

(略証) $N(\cdot)$ と $M(\cdot)$ の番号付けの方法と, 定義3.13から明らかである。
 (略証終)

[命題3.2] Aにおいて, 節点 v_a がリスト L から取り出されたならば,
 BからEまでの操作を有限回繰り返えした後, 操作Gを必ず実行する。

(証明は補題3.2を参照)

この手続きが終了した後, G 上で $N(\cdot)$ の値が i であるすべての節点, 枝の集
 合 V_i, E_i から, 求めるべき木 $T_i = [V_i, E_i]$ が構成される。

3.4 アルゴリズムの評価

アルゴリズムの正当性と有交性を評価するため、以下に幾つかの補題と定理を与えるが、始めに必要な定義をしておく。

[定義3.14] $N(V_k) = n$ または 0 である節点 V_k とある木 $H_{i_0} \in \mathcal{V}_2$ に対して、 $V_k \in V(H_{i_0})$ かつ $N(V_k) = n$ または 0 ($k = 1, 2, \dots, p$) であるような道 $P = [V_0 = V_k, V_1, \dots, V_p = V_k]$ が存在するならば、この P の長さ $l^{(i)}(P)$ を、

$$l^{(i)}(P) \triangleq \left| \left\{ E(P) - \bigcup_{H_j \in \mathcal{V}_2} E(H_j) \right\} \cap E_{xy} \right| \quad (3.11)$$

と定義し、 V_k と H_{i_0} との距離を、そのようなすべての道の中の最小の長さとして定義して $d^{(i)}(V_k, H_{i_0})$ で表わす。 (定義終)

[補題3.1] FORWARD において、操作 D がすでにも回実行された段階を考える。この段階で、ある節点 V_j の $M(\cdot)$ の値が操作 A または C において $I_3(m+1)$ に置き換えられたとすると、

$$d^{(i)}(V_j, H_{i_0}) = t+1, \quad (3.12)$$

$$I_3(m+1) = I_3(t+2) \quad (3.13)$$

が成立し、逆にこの段階で、 $d^{(i)}(V_j, H_{i_0}) \neq t+1$ である節点 V_j の $M(\cdot)$ の値は変わらない。

(証明) t に関する帰納法により証明する。 $t=0$ の場合には明らかに補題は成立する。 $t=n-1$ の場合に補題が成立すると仮定し、 $t=n$ の場合を考える。 $d^{(i)}(V_j, H_{i_0}) = n+1$ であるような任意の節点 V_j に対しては、定義3.14から長さ $l^{(i)}(P) = n+1$ である道 $P = [V_0, V_1, \dots, V_p = V_j]$ ($V_0 \in V(H_{i_0})$) が少なくとも一つ存在する。この P 上の枝の中で $e_k = (V_k, V_{k+1})$ を、 $e_k \in E_{xy}$ かつ $N(e_k) = 0$ である最後のものとする。

このとき、

$$d^{(i)}(V_k, H_{i_0}) = n \quad (3.14)$$

であるから、仮定より、

$$M(V_k) = I_3(n+1) \quad (3.15)$$

が成立する。

P 上の各節点 V_k ($k+1 \leq l \leq p \leq p$) を V_k の許容節点とすると、操作 A において、 $M(V_k) \leftarrow I_3(n+2)$ とされ、 $p \neq p$ の場合には、各節点 V_k ($p+1 \leq l \leq p$) は V_k の許容節点ではなく、ある $H_{i_j}^* (H_{i_j} \in \mathcal{V}_2)$ に属するので、操作 C において $M(V_k)$ は同じ $I_3(m+1)$ の値に置き換えられる。従って補題の

前半は成立する。後半は、前半の証明の過程および許容節点の定義より明らかである。(証明終)

【補題3.2】 BACKWARD の実行の過程において、操作Aで節点 v_a がリスト L より取り出されたならば、操作BからFまでの操作の有限回数の繰り返しで道 $P = \{v_0 = v_a, v_1, \dots, v_p = v_b\}$ が得られる。ここで、 $N(v_b) = \varepsilon$ であり、すべての v_k ($1 \leq k \leq p-1$) と $e_k = (v_k, v_{k+1})$ ($0 \leq k \leq p-1$) に対して、 $N(v_k) \leftarrow \varepsilon$, $N(e_k) \leftarrow \varepsilon$ にそれぞれ置き換えられる。

(証明) BACKWARD のどの段階においても、 $N(v_a') = 0$ かつ $M(v_a')$ キ0であるような任意の節点 v_a' に対し、 v_a' のI, II, III型の節点の中で少なくともいずれか一つは存在することが命題3.1より導びかれる。そこで、BACKWARD の実行のどの時点においても、操作Aで L から節点 $v_a \in H_{i_0} (\in \mathcal{V}_{i_0})$ が取り出された段階では、この v_a のIIまたはIII型の節点 v_b' が少なくとも一つは存在することを示そう。FORWARD の操作Bで L にこの v_a が入れられた段階においては、 v_a のIIまたはIII型の節点が存在することは明らかである。もし、BACKWARD において v_a が L より取り出された直後に、そのような節点 v_b' がIIまたはIII型の節点の条件を満たさなくなっていると仮定すると、これは次のようなことを意味する。 v_a から v_b' へ至るの後進道^(注)においても、その上の節点の中で少なくとも一つの節点 v_u (キ v_b') は、 v_a が L から取り出される以前の段階で L から取り出された節点 v_c に関する BACKWARD の操作により、 $N(v_u) \leftarrow \varepsilon$ と決定されている。このとき、

$$d^{(i)}(v_u, H_{i_0}) \leq d^{(i)}(v_c, H_{i_0}) \leq v^{(i)}(v_a, H_{i_0}) \quad (3.16)$$

が成立する。一方 v_u は v_a から v_b' へ至る後進道上の中間節点であるから、

$$d^{(i)}(v_u, H_{i_0}) = d^{(i)}(v_a, H_{i_0}) \quad (3.17)$$

が成立する。従って、

$$d^{(i)}(v_u, H_{i_0}) = d^{(i)}(v_c, H_{i_0}) \quad (3.18)$$

であり、このことは、 v_c と v_a とが同じ木 $H_{i_0} \in \mathcal{V}_{i_0}$ に属していることを意味する。FORWARD の操作Bの記述により、同じ木 H_{i_0} に属する2つの異なる節点が L へ入れられることはないから、これは矛盾であり、結局 BACKWARD で v_a が L から取り出された時点では、 v_a のIIまたはIII型の節点が少なくとも一つは存在することがいえる。

$v_b' \in H_{i_0}$ であれば、この証明の最初に述べたことから、 v_b' のI, IIまたはIII型のいずれかの節点が存在するので、BACKWARD の操作B から E までの操作

(注) FORWARD が終了した段階での、 v_a から v_b' へ至る後進道を意味する。

の各繰り返しによって、 H_{i0} へ距離1だけ戻るかまたは同じ距離にとどまるかであるから、これらの操作の有限回数適用の後、 $N(V_0) = i$ である節点 V_0 へ到達することができる。(証明終)

これら2つの補題から次の定理が導びかれる。

[定理3.1] G^* において、木の集合 W とその分割 π が与えられたとき、各 $V_i \in \pi$ に対して FORWARD と BACKWARD を適用したとき、 V_i に属するすべての木をその部分グラフとして含む木 T_i ($T_i \cap T_j = \emptyset$, $i \neq j$)が存在するならば、その一つを必ず見出す。

(証明) FORWARDの終了した時点において、リスト L を節点の入れられ順に、 $L = (V_{a_1}, V_{a_2}, \dots, V_{a_m})$ で表わす。このとき補題3.1から、

$$d^{(2)}(V_{a_1}, H_{i0}) \leq d^{(2)}(V_{a_2}, H_{i0}) \leq \dots \leq d^{(2)}(V_{a_m}, H_{i0}) \quad (3.19)$$

が成立する。 $V_{a_j} \in H_{ij} \in V_i$ ($j=1, 2, \dots, m$)とすると、補題3.2から導びかれるように、BACKWARDの操作BからFまでの操作を、 $V_a = V_{a_1}$ へ繰り返し適用することにより、 H_{i0} と H_{i1} とを部分グラフとして持つ木 $T_i^{(1)}$ が得られる。つぎに、 $V_a = V_{a_2}$ へ同じ操作を適用することにより、 $T_i^{(1)}$ と H_{i2} とを部分グラフとして持つ木 $T_i^{(2)}$ が得られる。以下同様にして、 V_i に属するすべての木 H_{ij} を部分グラフとして持つ木 T_i が存在するならば、FORWARDが終了した段階において、リスト L は相異なる H_{ij} からそれぞれ取り出された節点を含み、BACKWARDの適用により、この木 T_i を得ることができる。(証明終)

この定理において、所望の木 T_i が存在しない場合には、FORWARDは操作Eにおいて終了し、BACKWARDの適用により、 H_{i0} とリスト L に属する各節点で代表される木だけをその部分グラフとして含む木 T_i が見い出される。

次に、アルゴリズムの実行の手数を評価するため、以下の定理を与える。

[定理3.2] 定理3.1で述べた木 T_i が存在するならば、FORWARDとBACKWARDの適用により、そのような木 T_i を高々 $O(\lambda^2)$ の手数^(注)で見出すことができる。ただし、 λ は次の式で定義されるものとする。

$$\lambda \triangleq \max_{\substack{V_k \in V(H_{ij}) \\ H_{ij} \in V_i}} [d^{(2)}(V_k, H_{i0})] \quad (3.20)$$

(証明) FORWARDにおいては、 $H_{i0} \in \pi$ からの距離が λ またはそれ以下の各節点に対し、番号 $M(\cdot) = 1, 2$, または3が割当てられるか否かを判定する。これらの節点の数は高々 $O(\lambda^2)$ であり、従ってFORWARDの実行に要する手数は $O(\lambda^2)$ でおさえられる。この手数に比較して、BACKWARDのそ

(注) 許容節点に関して一度判定することを単位とする。

それは無視できる。よって定理が成立する。

(証明終)

3.5 例題

前節において述べたアルゴリズムを, 図3.2に示されるグラフ G^* に適用することを考える。与えられた各木を構成する枝は太線で表わされ, 各節点 v_i に付された番号 $n_i \triangleq N(v_i)$ と $m_i \triangleq M(v_i)$ とは n_i/m_i で示されている。いま,

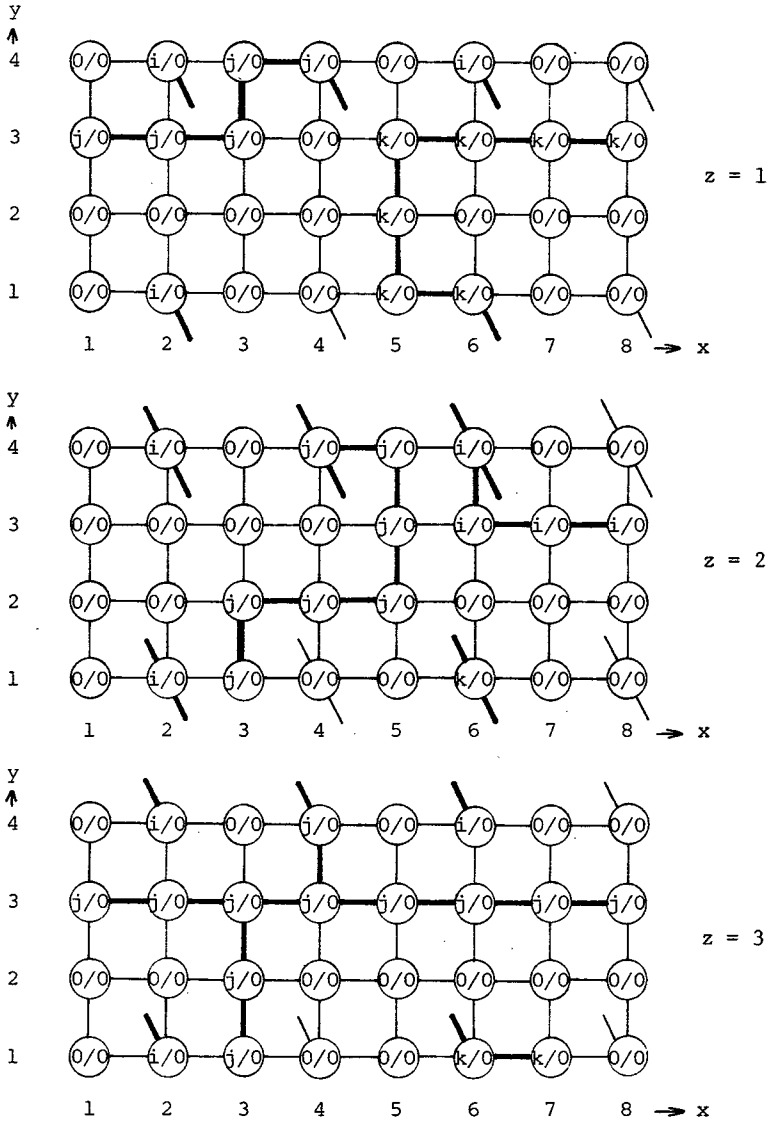


図3.2 与えられたグラフ G^*

木の集合 $\mathcal{W}$ の分割 $\pi = \{ \mathcal{H}_1, \mathcal{H}_2, \mathcal{H}_3 \}$ が図のように与えられており、特に、 $\mathcal{H}_2 = \{ H_1, H_2, H_3 \}$ に注目する。ただし、

$$V(H_1) = \{ [2, 4, 1], [2, 4, 2], [2, 4, 3] \}$$

$$V(H_2) = \{ [2, 1, 1], [2, 1, 2], [2, 1, 3] \}$$

$$V(H_3) = \{ [6, 4, 1], [6, 4, 2], [6, 4, 3], [6, 3, 2], [7, 3, 2], [8, 3, 2] \}$$

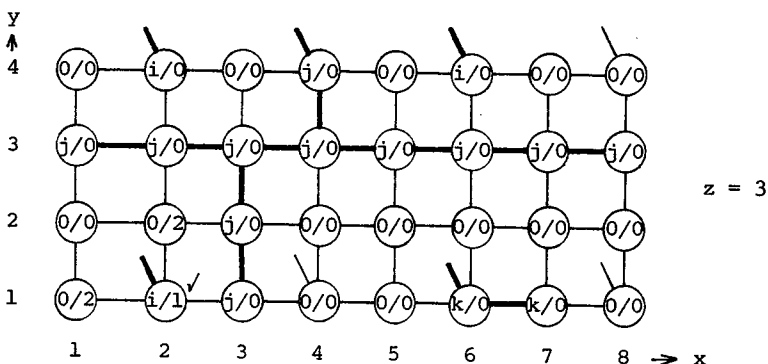
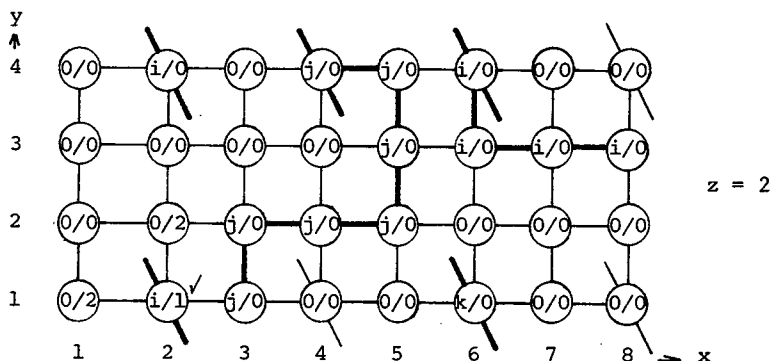
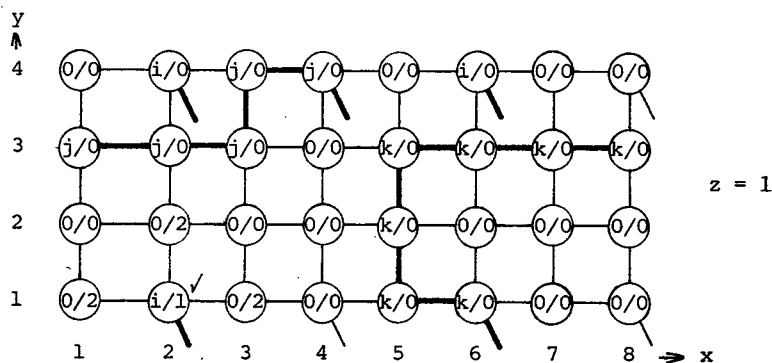


図 3.3 FORWARD の実行のある段階における G^*

である。この H_2 に関して、木 H_1, H_2, H_3 を部分グラフとして含む木 T_2 で、 W に属するその他の木と節点を共有しないような木 T_2 を以下で求める。

H_2 として H_2 を選び、FORWARDを G^* に適用する。手続きがある段階まで進行した時点における m_2 を図3.3に示す。この段階において、節点 $[3.1.1]$ を操作Eにおける v_k として選ぶならば、操作Fにおける集合 Ω は、

$$\Omega = \{[3, 2, 1], [4, 1, 1], [4, 1, 2], [4, 1, 3]\}$$

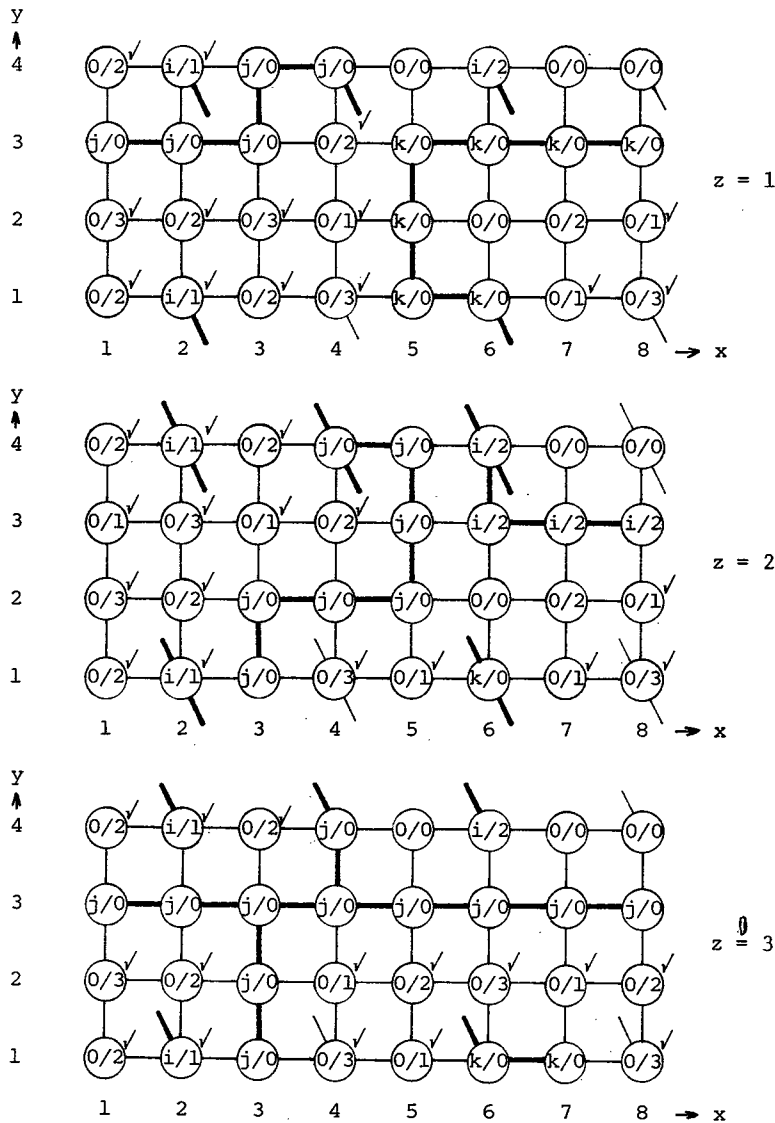


図3.4 FORWARD終了時のグラフ G^*

である。

前進手続きが終了した段階において、各 m_i は図3.4のようになり、リスト L は、節点が入れられた順番に、

$$L = ([2, 4, 2], [8, 3, 2])$$

である。この場合 $|L| = 2$ であり、従って $|L| = |V_i| - 1$ となり、求めようとする木 T_i が存在することがわかる。

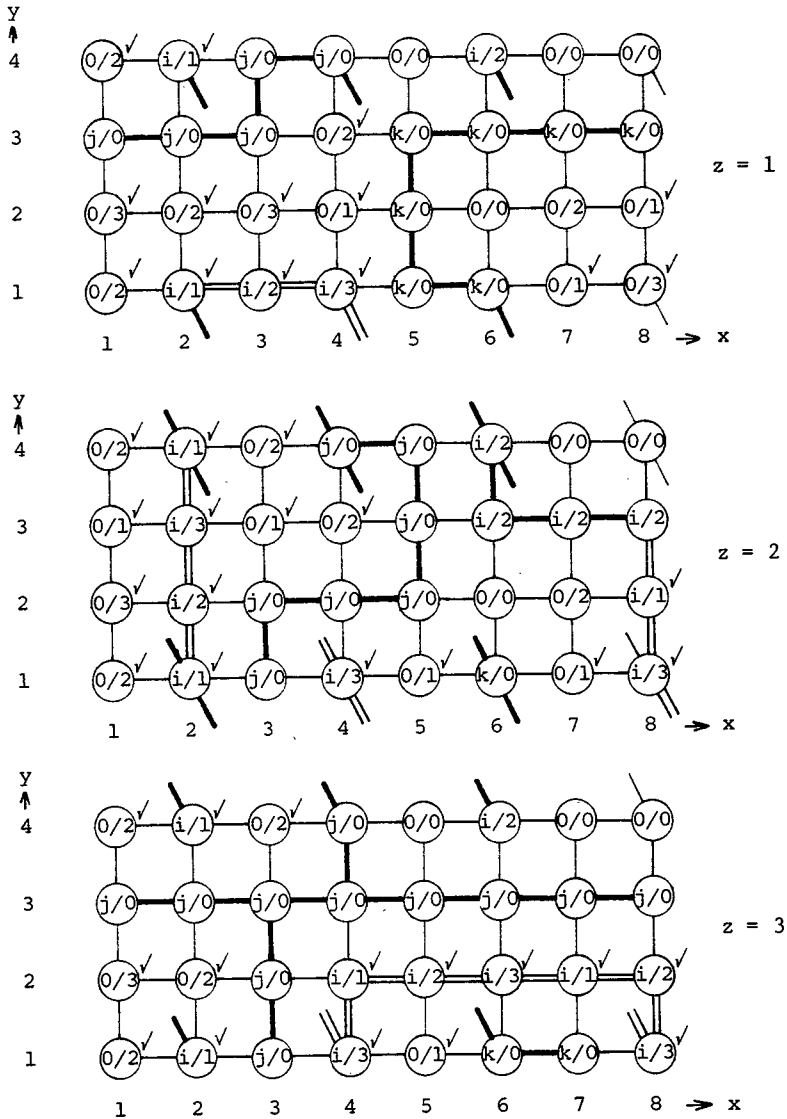


図3.5 BACKWARD が実行された結果

図3.4のグラフ G^* へ BACKWARD を適用すると、 T_2 を構成する節点と枝が得られる。それらは図3.5において2重線で示されている。

3.6 結言

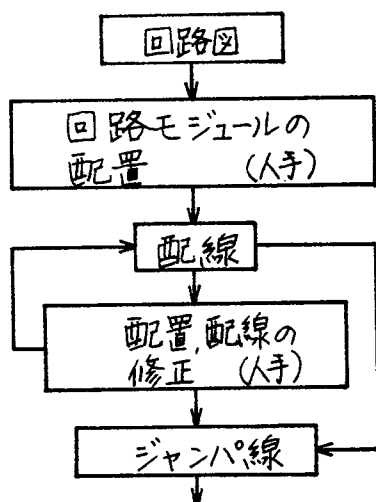
迷路法に関して従来提案されたどのアルゴリズムにおいても、3個あるいはそれ以上のセル(端子)を結線する場合については、具体的な方針が明記されていない。従ってこれらの手法を適用する過程においてこのような場合が生じたときには、適当に選ばれた2つのセルに対してこれを繰り返えし適用せざるを得ない。このとき手続きの実行の手数は、それぞれの選択された2セル間の距離の2乗の総和に比例する。多層基板を対象とする場合にはとくにこの欠点が強調される。これに対し本章で提案したアルゴリズムの実行の手数は $O(N^2)$ であり、より効率的であるといえる。

一般に、配線率を上げるため、各層に単一の方角性を持たせる。(各層のそれぞれに、水平または垂直のリード線のみを許す)ことが行なわれている。この場合においては、それぞれの層に対応する G の部分グラフにおいて、 Y 方向または X 方向の枝を取り除くことにより、本章で提案したアルゴリズムがそのまま適用可能である。

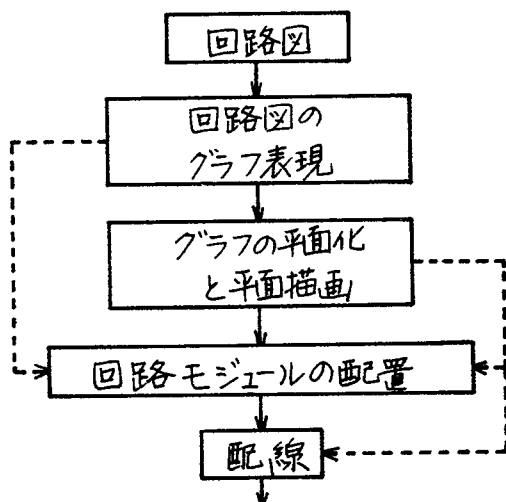
第4章 一層印刷回路基板におけるグラフの平面化問題

4.1 緒言

印刷回路基板の作製において、特に回路モジュールの配置とその間の配線の問題は、相互に密接な関係にある。配線問題の接近法としては、予想総配線長、モジュール間の結線数等を評価基準とすることがしばしば行われているが、現在までに発表されている種々の印刷回路基板自動設計プログラム^{[10], [15]}においては、多くの場合、人手によって回路モジュールの配線を決定し、その後、配線を実行するという方針をとっている。したがって、配線が100%完了されなかった場合、残りの配線をジャンパ線として処理するか、あるいはグラフィックディスプレイ等を用いて、人手で配置、配線を部分的に変更しつつ配線率を上げるのが普通である。図4.1-aは、現在一般に行われている印刷回路基板作製の流れの概略である。



(a) 従来の設計方針



(b) 一層基板を対象とする場合

図4.1 印刷回路基板の設計の流れ

しかし、このように配置と配線を直列に実行し、せいぜいその間にわずかなフィードバックを施すといった方法では、結果と実行時間に関して満足のものを得難いという欠点がある。特に、一層の印刷回路基板を対象とする場合には、配置が配線に及ぼす影響は致命的であり、これらを切り離して別個に

考えることは効率的ではない。これに関連して、とくに比較的小規模なICの設計手法は、既にいくつかのもの^{[21], [28]}が提案され、実用化されているが、一般の民生用一層印刷回路基板に対しては、未だ考察が進められていない。

ここで、配置と配線とを分離して考えることの出来難い典型的な例であり、いまだに広く人手によって設計されている一層印刷回路基板を対象として、その自動設計を図4.1-bに示されるような方針で行なうことを考える。ただし、図において点線は、例えば‘グラフの平面描画’が、配置や配線に影響を及ぼしていることを表わしている。この図に示されるように、配線を実行する段階においては、大まかな配線経路はトポロジカルな意味で決定されているので、あとは具体的な位置を、配線経路間及びそれらと配置された回路モジュール間の関係を考慮しながら決定するだけである。

しかし、この図の流れにおいて、グラフの平面化と平面描画が効率的に行なわれ、実用的でなければ全体の意味が失われる。したがって、この観点から本章では、まず最初に回路図のグラフ表現が与えられたものとして、グラフの平面化問題に関して効率的なアルゴリズムを提案し、それを評価する。次に、このアルゴリズムに基づいて作成したプログラムの実験結果を示し、これが充分実用に耐え得るものであることを示す。

4.2 グラフの平面化問題

グラフの平面性判定と、そのグラフが平面グラフである場合の平面描画の問題は、従来多くの著者によって試みられてきたが^{[32] - [38]}これに対して効率的なアルゴリズムを、J. HopcroftとR. E. Tarjan^[38]が提案した。しかし、与えられたグラフ G が平面グラフでない場合、 G から最小個数の枝を開放除去することによって平面グラフにする(平面化する)効率的な手法は未だ見出し出されてはいない。この節では、[38]の平面性判定アルゴリズムに基づいて、与えられたグラフから最小個数の枝を開放除去し、これを平面化するアルゴリズムを提案するために、必要な定義を行ない、平面化アルゴリズムの基本的な概念と方針について論じる。さらにアルゴリズムを提案して、その実行の手数に関して評価を与える。なお、以下で用いる術語は文献[38]に準じるものとする。

4.2.1 諸定義

以下で取り扱うグラフ $G = [V, E]$ は、並列枝、自己ループを含まない連結なものとし、その隣接構造を AL で表わす。

〔定義4.1〕 幾何グラフ G とは、次の条件を満たす、 n 次元ユークリッド

空間 $R^{(n)}$ の点の集合 V (キョ)と曲線の集合 E とからなる。

- (i) E に属するどの閉曲線も V に属する点を1個かつ、ただ1個含む。
- (ii) E に属するどの閉曲線も V に属する点をちょうど2個含み、かつその2点はその閉曲線の端点である。
- (iii) E に属するどの曲線も自分自身と交わらない。
- (iv) E に属するどの相異なる二つの曲線に対しても、もし共有点があるとしてもそれは V に属する。(定義終)

このとき、 G を $G(R^{(n)})=[V, E]$ で表わし、各 $v_i \in V$ を G の幾何節点といい、各 $e_k \in E$ を G の幾何枝という。

いま、グラフ $G=[V, E, \pi]$ が与えられたとき、各 $v_i \in V$ に $R^{(n)}$ の相異なる幾何節点が、各 e_k に $R^{(n)}$ の v_i, v_j 間の幾何枝がそれぞれ対応する幾何グラフ $G(R^{(n)})$ が存在すれば、 $G(R^{(n)})$ と G の接続構造の間には1対1対応が存在し、したがって $G(R^{(n)})$ は G の $R^{(n)}$ における一つの幾何表現であるということが出来る。

【定義4.2】 2次元ユークリッド空間 $R^{(2)}$ において幾何表現をもつグラフを平面グラフ(planar graph)といい、それ以外のグラフを非平面グラフ(nonplanar graph)という。特に、平面グラフの $R^{(2)}$ における幾何表現の一つをそのグラフの平面描画という。(定義終)

グラフが2連結成分を含まない場合は、それは平面グラフであることが明らかである。一方、与えられたグラフ $G=[V, E]$ を2連結成分に分解する、 $O(|V|+|E|)$ の手数が保証された効率的なアルゴリズム^[39]が知られているので、以下では2連結なグラフのみを取り扱うものとする。

【定義4.3】 グラフ $G=[V, E]$ から枝の集合 D を開放除去したグラフ $G'=[V, E-D]$ が平面グラフであり、かつ D の任意の元 e を G' に付加したグラフ G'' が平面グラフでないとき、この D を極小除去枝集合といい、 $D(G)$ で表わす。

本章は、グラフ G が与えられたとき、 G の極小除去枝集合 $D(G)$ を効率よく求めるという、いわゆる平面化問題について論じるものである。

さて、与えられたグラフ G に対して以下の操作を施す。

【前処理操作】

操作1: G に対して、 $s \in V$ を出発点とし、 AL を用いてDFS(Depth-First Search)を適用し、各節点 $v \in E$ に対して番号 $NUMBER(v)$ を付け、同時に $LOWPT1(v)$, $LOWPT2(v)$ の値を計算する。

操作2: 枝 $e=(v, w) \in E$ に対して以下で定められた関数中、の値の小さい順に対応する AL の節点を並べ換え、その結果を AL' とする。

$$\phi(e) = \begin{cases} 2 \cdot \text{NUMBER}(w) : e \text{ が フロンドのとき} \\ 2 \cdot \text{LOWPT1}(w) : \text{LOWPT2}(w) = \text{NUMBER}(v) \\ 2 \cdot \text{LOWPT1}(w) + 1 : \text{LOWPT2}(w) < \text{NUMBER}(v) \end{cases} \quad (4.1)$$

ただし，操作1で適用するアルゴリズム DFS は以下のものである。

```

begin comment:routine for depth-first search of a graph G represented
  by adjacency lists ALv;
  integer n;
  procedure DFS(v,u);comment:vertex u is the father of vertex v in the
    spanning tree being constructed;
  begin
    n := Number(v) := n+1;
    LOWPT1(v) := LOWPT2(v) := NUMBER(v);
    for w in the adjacency list of v do begin
      if NUMBER(w) = 0 then begin
        comment:w is a new vertex;
        mark (v,w) as a tree edge;
        DFS(v,w);
        if LOWPT1(w) < LOWPT1(v) then begin
          LOWPT2(v) := min{LOWPT1(v),LOWPT2(w)};
          LOWPT1(v) := LOWPT1(w);
        end;
        else if LOWPT1(w) = LOWPT1(v) then
          LOWPT2(v) := min{LOWPT2(v),LOWPT2(w)};
        else LOWPT2(v) := min{LOWPT2(v),LOWPT1(w)};
      end;
      else if NUMBER(w) < NUMBER(v) and w ≠ u then begin
        comment:this test is necessary to avoid exploring an edge in
        both directions;
        mark (v,w) as a frond;
        if NUMBER(w) < LOWPT1(v) then begin
          LOWPT2(v) := LOWPT1(v);
          LOWPT1(v) := NUMBER(w);
        end;
        else if NUMBER(w) > LOWPT(v) then
          LOWPT2(v) := min{LOWPT2(v),NUMBER(w)};
      end;
    end;
  end;
  for i := 1 until V do NUMBER(i) := 0;
  n := 0;
  comment:the search starts at vertex s;
  DFS(s,0);
end;

```

ただし，DFSを適用した結果得られたGの木に属する枝を木枝と呼び，それ以外の枝をフロンドと呼ぶ。

[補題4.1] DFSの実行の手数は $O(|V| + |E|)$ である。

(証明) 文献[38]を参照。

操作2に対しては radix sort を用いることにより, $O(|V| + |E|)$ の手数で実行が可能である。

つぎに, 以上の操作が終了した段階で, 同じ節点を出発点として G に対し, 以下のアルゴリズム PATH FINDER を仮に適用することを考える。

```

begin
  procedure PATHFINDER(v); begin
    for w in the adjacency list  $AL_v$  do
      if (v,w) is a tree edge then begin
        if FLAG=0 then begin
          FLAG := 1;
          start new path;
        end;
        add (v,w) to current path;
        PATHFINDER(w);
      end;
    else begin
      comment:(v,w) is a frond;
      if FLAG=0 then begin
        FLAG := 1;
        start new path;
      end;
      add (v,w) to current path;
      output current path;
      FLAG := 0;
    end;
  end;
  FLAG := 0;
  PATHFINDER(s); comment: this procedure starts at vertex s;
end;

```

このアルゴリズムを適用した結果, 最初に得られる, Λ から Λ へ至る閉路を FC (First Cycle) と呼び, 見い出された各道 $P = \{v_0 = v, v_1, \dots, v_m\}$ (P はフロンド (v_{m-1}, v_m) で終わり, それによって一意的に定義される) を v の FP (First Path) と呼び。

G から FC を取り除いた時に残っている各連結成分をセグメントと呼ぶ。この様子を図4.2 に示す。

各セグメント S_i は, それに属する最小の NUMBER の値を持つ節点 v_i によって一意的に代表される。このとき, S_i を v_i のセグメントと呼ぶ。 v_i の FP (これは必ず存在する) をとくに S_i の FP と呼び。セグメント S からその FP を取り除いたときにできる各連結成分も同様にセグメントと呼び, 以下再帰的にセグメントを定義する。

(例4.1) 図4.2-bのグラフ G のセグメントと FP を図4.3に示す。ただし, 各セグメントの FP は図中太線で示されている。セグメント S_{31} は S_3 に, S_{311} は S_{31} にそれぞれ含まれている。 v_5 の FP は $[v_5, v_7, v_8, v_2]$ であり, v_7 の FP は $[v_7, v_9, v_2]$ である。しかし, v_{10} に関する FP は存在しない。 v_6 の FP は $[v_6, v_3]$ と $[v_6, v_{11}, v_3]$ の2つがある。

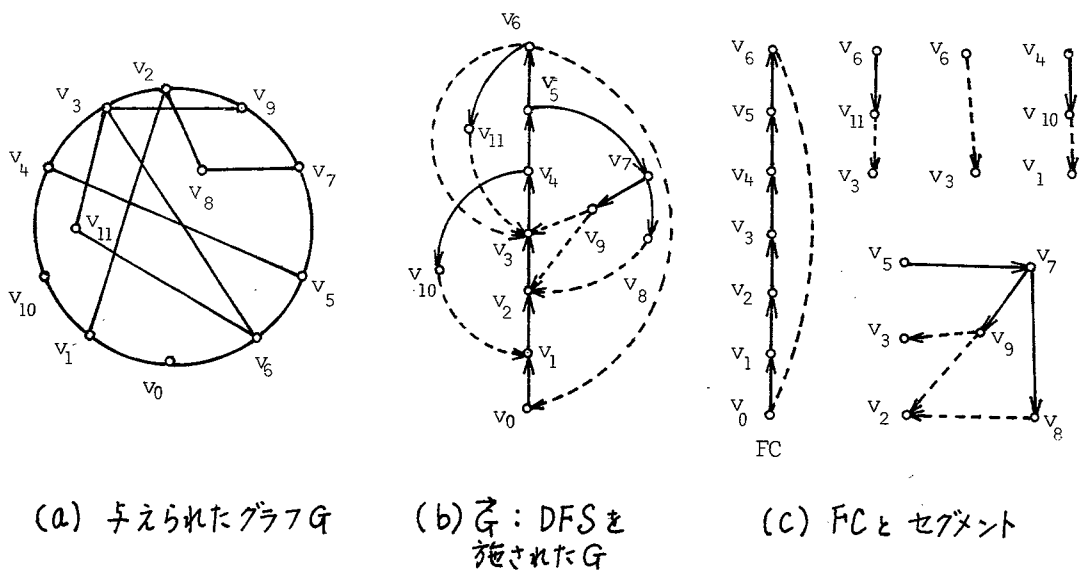


図 4.2 $\vec{G}$ から FC を取り除いたときの各セグメント

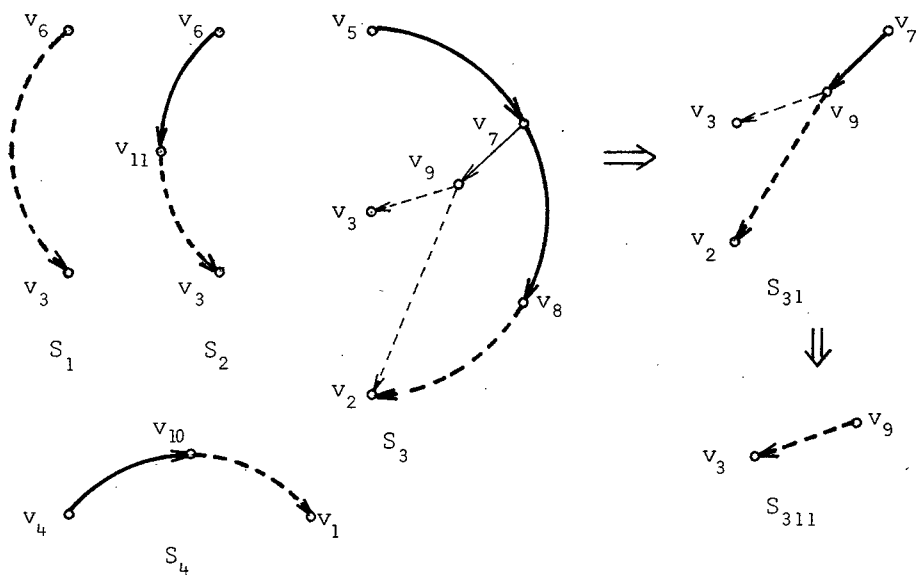


図 4.3 $\vec{G}$ のすべてのセグメントと FP

4.2.2 予備的考察

ここでは、平面化問題に対する接近法の方針を述べる。以下では、[前処理

手続き]の施こされたグラフ G を対象に考察を進める。

まず始めに、文献[38]の平面性判定アルゴリズム $EMBED$ に基づいた平面埋め込み手続き $EMBED'$ を示す。

```
procedure  $EMBED'(G)$ ; begin comment: an outline of embedding a graph in a plane;
  find first cycle;
  while some segment  $S$  is unexplored do begin
    if the FP of  $S$  cannot be embedded on the right side or left side of  $S'$ ,
      where  $S'$  contains  $S$  and is the last embedded one, then begin
        comment:  $G$  proves to be nonplanar;
        comment: there may possibly exist a case when only FP of  $S$  can be embedded
          outside of  $S'$ , and in this case the FP of  $S$  is in parallel with the FP
            of  $S'$ , and they have the same end vertices in common;
        if the FP of  $S$  can be embedded outside of  $S'$  then embed this FP;
      end;
      go to "nonplanar";
    else begin
      embed this FP;
      apply this algorithm recursively to embed the remaining segments which
        are contained in this segment;
    end;
  end;
end;
```

FPが平面に埋め込む($embed$)ことができるか否かの判定と、その埋め込み方の手法は、本論と直接は関連しないので、上記のアルゴリズムにおいては具体的に明示することはしていない。

[補題4.2] $EMBED'$ は、与えられたグラフ G の平面性を $O(|V|)$ の手数で判定する。

(証明) 文献[38]を参照

与えられたグラフ G に対して、その $D(G)$ を求める為の方針の一つとして次の手続きを示す。

[平面化の手続き]

操作1 : G が平面グラフならば操作終了。

操作2 : G へ $EMBED'$ を適用し、このときに埋め込むことができなかったFPを一意的に代表するフロンツを G から開放除去して、残りのグラフを新たに G とする。

操作3 : [前処理手続き]を施し、 AL' を作成して、操作1へ戻る。

この手続きの途中、操作2においてグラフが2連結でなくなった場合には、各2連結成分について、同じ手続きを適用すればよい。

[補題4.3] 上記の手続きにおいて繰り返えしの各段階を考えると、ある段階で埋め込まれたFPのすべての枝は、次の段階においても再び埋め込まれる。FCについても同様のことが成立する。

(証明) 各段階の操作3において「前処理手続き」を実施するとき、隣接構造はALを用いている(ただし、除去されたフロンドに関しては異なる)。したがって、木枝とフロンドの別およびFCはどの段階においても同一である。いま、 i 番目の段階でフロンド e が除去され、 e が代表するFPを $P_e = [v_0 = v_i, v_1, \dots, v_n]$ とする。 e が除去される前と後の v_i のLOWPT1とLOWPT2をそれぞれLOWPT1B(v_i), LOWPT1A(v_i) LOWPT2B(v_i), LOWPT2A(v_i) とすると、次式が成立する。ただし、一般性を失うことなく(v_0, v_i) $\neq e$ であるものとする。

$$\begin{aligned} \text{LOWPT1B}(v_i) &\leq \text{LOWPT1A}(v_i) \\ \text{LOWPT2B}(v_i) &\leq \text{LOWPT2A}(v_i) \end{aligned} \quad (4.2)$$

従って、(4.1)式の関数中の定義より、 $\phi_1((v_0, v_i))$ の値は e の除去によって減少することはない。 i 番目の段階の操作2で埋め込まれた各FP = [$v_1^s, v_2^s, \dots, v_{m_i}^s$] に対して明らかに次式が成立する。

$$\phi_1((v_0, v_i)) \geq \phi_1((v_1^s, v_2^s)) \quad (4.3)$$

上記の議論から、(4.3)式はどの段階においても成立することがわかる。従って、 $i+1$ 番目の段階においても成立し、結局これらFPは、 i 番目と同じ経過を辿って再び埋め込まれる。FCに関しては明らかである。(証明終)

【補題4.4】上記の手続きは、正しく $D(G)$ を算出する。ただし $D(G)$ としては、操作2において除去された枝(フロンド)すべてからなる集合 D である。

(証明) D の任意の元 e を、この手続きが終了した段階のグラフ G^* に付加したグラフ $G^*(e)$ は、平面グラフでないことを証明すれば十分である。

一般に、EMBEDが適用された各段階において、一度埋め込まれたFPは G^* にすべてその枝が含まれている。このことは補題4.3より明らかである。操作2でこの e が除去された段階において、既に埋め込まれていたすべてのFPの枝の集合を P とし、FCの枝の集合を C とすると、 P, C と e によって代表されるFPのすべての枝によって作られるグラフ G_0 は、 $e \in D$ から、非平面グラフであることがわかる。このとき、上述のことから G_0 は $G^*(e)$ の部分グラフであるから、 $G^*(e)$ は非平面グラフとなる。従って、 $D = D(G)$ である。

(証明終)

【平面化の手続き】は、正しく $D(G)$ を算出することがわかったが、一つの枝を除去する度に、再び「前処理手続き」、EMBEDを適用し、これを繰り返すという方法なので極めて効率的でない。従って、本論ではこのような繰り返しを避けるため、あらかじめ、あらゆる状態の情報を含んだ、一種の隣接

リストを先に作り、これに従って平面性判定を進めることにより、唯一回の適用により $D(G)$ を求めることを考える。

あるセグメント S の FP, $P = [v_1, v_2, \dots, v_n]$ が埋め込み不可能と判定されたとき, (v_{n-1}, v_n) が除去された新しいセグメント S' において, その FP がどれであるかわかれば, そこから平面性判定を続行することができる。従って, S から (v_{n-1}, v_n) が除去されたときの v_1 の新しい FP (S の FP), それが埋め込み不可能なときの新しい FP という具合に, すべての場合について候補となり得る FP を, あらかじめリストとして求めておけばよい。次の節では, そのリストの定義と性質, および効率的な生成法について論じる。

4. 2. 3 フロンドリストと生成アルゴリズム

フロンドリスト FL_v は, 各節点 $v \in V$ に関する可能なすべての 3 つ組 (v, w, f) を並べたリスト構造を持つものである。ただし, (v, w) は木枝か又は, フロンドであり, f は (v, w) を通り木枝のみを辿って到達可能なフロンドである。 (v, w) がフロンドの場合は, $f = (v, w)$ である。

まず, 各組に対して次式で定められる関数 ϕ_2 を割当てて。

$$\phi_2((v, w, f)) = \begin{cases} 2 \cdot |f| + 1 & : |f| < |g| \text{ を満たす } g \text{ 組} \\ & (v, w, g) \text{ が存在するとき,} \\ 2 \cdot |f| & : \text{存在しないとき.} \end{cases} \quad (4.4) \quad (\text{注})$$

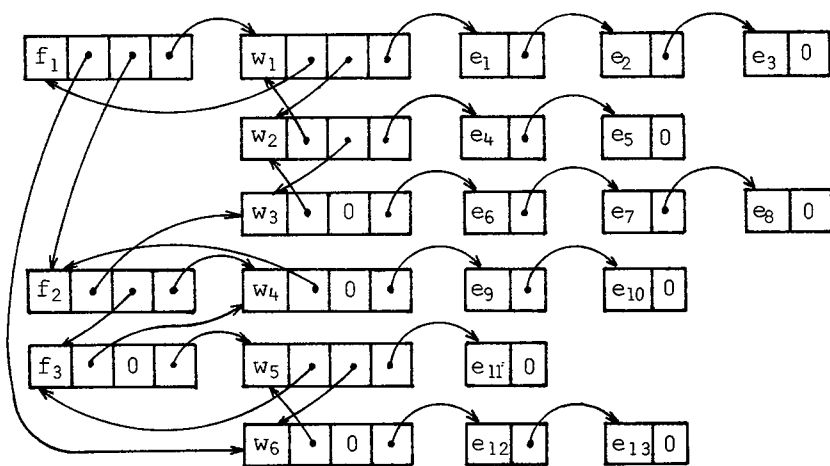


図4.4 FL_v の構造とそのポインタシステム

(注) フロンド $f = (f_a, f_e)$ に対し, f_e へ付された番号を $|f|$ と書く。

フロンドリフト FL_v は、 v に関するすべての3つ組 (triple) を ϕ_2 の値の小さいものから並べたリストであり、それを図4.4に示す。図において、3つ組 (v, w, f) は、 f_e が f_i 、 f_u が e_i 、 w が w_i ($i=1, 2, \dots$) で表わされている。ただし、 $w_i = w_j$ ($i \neq j$) の場合を含む。この例として、図4.2のグラフの節点 v_5 と v_7 のフロンドリフト FL_{v_5} 、 FL_{v_7} を図4.5に示す。 v_5 はFC上にあるので、 (v_5, v_6) の枝は最優先となり、この隣接関係は不用である。

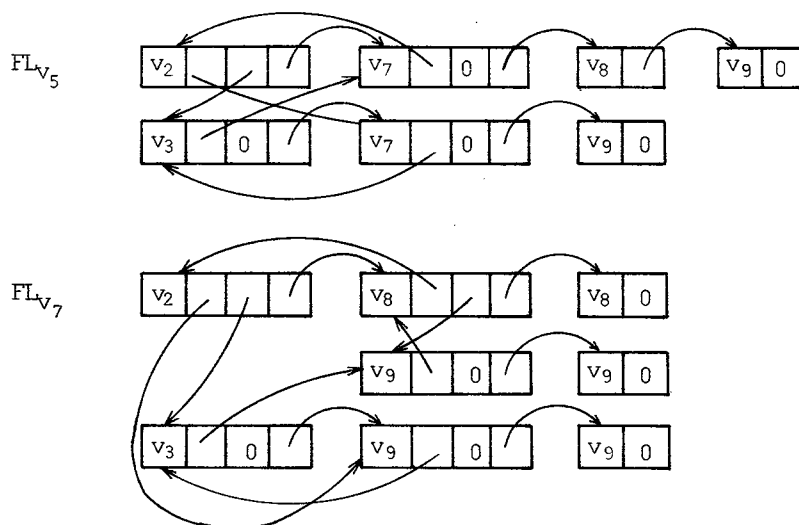


図4.5 図4.2のグラフ G の節点 v_5 と v_7 のフロンドリフト

さて、FL を生成するための手続き FLGENERATION を以下に記す。

```

procedure FLGENERATION(v); comment: this procedure generates FL of
graph G, where FL is the required frond lists, and variable v
is a current vertex of this recursive procedure;
begin
  for w in the adjacency list  $AL_v$  do
    if (v,w) is a tree edge then begin
      FLGENERATION(w);
      merge  $FL_v$  and  $FL_w$ , where all the triples (w,x,f) in  $FL_w$  are
      transformed into (v,w,f), in the order of  $|f|$ ;
      for all the value of  $|f|$  do begin
        if, for a triple (v,w,f), there exists (v,w,f'), where
         $|f'| > |f|$ , then concatenate  $FL_w$  after  $FL_v$  with respect
        to  $|f|$ ;
        else concatenate  $FL_w$  before  $FL_v$  with respect to  $|f|$ ;
      end;
    end;
  comment: (v,w) is a frond and  $f=(v,w)$ ;
  else add (v,w,f) to  $FL_v$  with respect to  $|f|$ ;
end;
end;

```

[補題4.5] 各節点 $v_i \in V$ に対して, 同じ w_i を持つすべての3つ組 (v_i, w_i, f) のなかで, ϕ_2 の値が最小のものを (v_i, w_i, f_i) とすると,

$$\phi_2((v_i, w_i, f_i)) = \phi_1((v_i, w_i)) \quad (4.5)$$

が成立する。

(証明) (v_i, w_i) がフロンドのときは定義より明らかである。従って, (v_i, w_i) が木枝のときのみ考える。仮定により

$$|f_i| = \text{LOWPT1}(w_i) \quad (4.6)$$

が成立する。 $\phi_2((v_i, w_i, f_i)) = 2|f| + 1$ のときは, 定義より,

$$\text{LOWPT2}(w_i) < \text{NUMBER}(v) \quad (4.7)$$

が成立し, 従って,

$$\begin{aligned} \phi_1((v_i, w_i)) &= 2 \cdot \text{LOWPT1}(w_i) + 1 \\ &= 2|f| + 1 \\ &= \phi_2((v_i, w_i, f_i)) \end{aligned} \quad (4.8)$$

が成立する。 $\phi_2((v_i, w_i, f_i)) = 2|f|$ のときは, 同様に

$$\text{LOWPT2}(w_i) = \text{NUMBER}(v) \quad (4.9)$$

が成立し, 従って

$$\begin{aligned} \phi_1((v_i, w_i)) &= 2 \cdot \text{LOWPT1}(w_i) \\ &= 2|f| \\ &= \phi_2((v_i, w_i, f_i)) \end{aligned} \quad (4.10)$$

が成立する。

(証明終)

[補題4.6] FLGENERATION で生成されたFLにおいて, 任意の節点 v を考えると, 各3つ組 (v, w, f) は, w に関して関数 ϕ_2 の値に従って配列されている。

(証明) 手続きの記述より明らかである。

(証明終)

[補題4.7] 任意の節点 v のフロンドリスト FL_v において, 同じフロンドの値 $|f|$ をもつすべての3つ組のなかで, リストの最初にあるものを (v, w, f) とすれば, $|f| = |f'|$ であるすべての3つ組 (v, w', f') に対し, 以下の式が成立する。

$$\phi_2((v, w, f)) \leq \phi_2((v, w', f')) \quad (4.11)$$

(略証) $\phi_2((v, w, f)) = 2|f| + 1$ ならば明らかに成立する。 $2 \cdot |f| + 1$ ならば、手続きの記述より、すべての3つ組 (v, w, f') は同じ $2 \cdot |f| + 1$ の値を持つことになり、補題が成立する。 (略証終)

[補題4.8] FLGENERATIONの実行の手数は高々 $O(|V| \cdot |E|)$ である。

(証明) FL_v と FL_w を合併 (merge) する手数は、それぞれに含まれる3つ組の異なるフロンドの値の数である。更に、同じフロンドの値をもつもののなかでの連結 (Concatenation) の手数は定数であるから、これら合併と連結の手数は高々 $O(|V|)$ である。この2つの操作は各枝について一回行なわれるだけであるから、全体の実行の手数は高々 $O(|V| \cdot |E|)$ である。 (証明終)

4.2.4 平面化アルゴリズム

補題4.5 ~ 4.7 からわかるように、FLには、あるFP, $P_v = [V_0 = v, v_1, \dots, v_n]$ が埋め込み不可能なときの、次にFPとなるべき候補が並べられている。すなわち、 P_v は3つ組 $(v, v_1, (v_{n-1}, v_n))$ によって表わされているが、 $|v_{n-1}, v_n|$ と同じ値をもつ3つ組 (v, w, f) によって表わされているすべてのFPは同時に、埋め込み不可能とわかり、フロンド f が除去される。このとき、 v のセグメント S_v の新しいFPは、 $|v_{n-1}, v_n|$ の値の次にくるべき値をもつ3つ組のなかで初めに並べられているものとなる。

これらの考察に基づき、さらに[平面化の手続き]の方針を考慮に入れ、平面化アルゴリズム PLANARIZATION を以下に記す。

```

procedure PLANARIZATION(G); begin
  a: DFS(s);
  b: FLGENERATION(s);
  find first cycle  $C = [v_1 = s, v_2, \dots, v_n, s]$ ;
   $D \leftarrow \phi$ ,  $v \leftarrow v_n$ ;
  while  $v \neq s$  do begin
    if there exists a segment  $S_v$  of  $v$  then begin
      if FP of  $S_v$  which is represented by a first triple of  $FL_v$  and of the
        form  $P_v = [v_{11} = v, v_{12}, \dots, v_{1m}, w]$ , can be embedded then begin
        embed this  $P_v$ ;
         $v \leftarrow v_{1m}$ ;
      end;
    else begin
      delete the frond  $e = (v_{1m}, w)$  from  $G$ ;
       $D \leftarrow D \cup \{e\}$ ;
      c: for  $v_{1i}$  on  $P_v$  do delete the triple  $(v_{1i}, v_{1i+1}, w)$  from  $FL_{v_{1i}}$ ;
    end;
  end;
  d: delete the triples of the form  $(u, v, \cdot)$  from  $FL_u$ , where  $u$  is a father of  $v$ ;
   $v \leftarrow u$ ;
end;
end;
end;
```

[定理4.1] PLANARIZATION は正しく $D = D(G)$ を算出する。
 (証明) 補題4.1 ~ 4.7 より明らかである。 (証明終)

4.2.5 アルゴリズムの評価

この節では前節で提案した平面化アルゴリズム PLANARIZATION の実行の手数を理論的に求め、これを評価する。

[定理4.2] PLANARIZATION の実行の手数は高々 $O(|V| \cdot |E|)$ である。

(証明) i) c の手数: P_v 上の節点数の数は高々 $|V|$ であり、各節点 v_i について除去すべき3つ組はリスト FL の最初にあるので、これを除去する手数は定数である。埋め込み不可能な、 v の FP をすべて同時に見い出しておき、 C の除去操作を重複なしに行なえば(これらの操作は、手数の増大なしに容易に行なえる)、各節点に関して C を実行する手数は高々 $O(|V|)$ である。 G の節点数の数は $|V|$ だから、結局 C を実行する手数は高々 $O(|V|^2)$ である。

ii) d の手数: 図4.4からわかるように、それぞれの異なるフロンドの値に対して、3つ組 (v, w, f) は w に関してポインタで結ばれている。これらの始まりを示す表を作っておけば(この表を作る操作を、手数の増大なしに、 FL GENERATION に含めることができる)、それぞれを定数の手数で除去できる(実際には、ポインタの付け替えを行えばよい)。異なるフロンドの値は高々 $|V|$ であるから、各木枝についてその手数は高々 $O(|V|)$ であり、木枝の数は $|V| - 1$ であるから、結局 $O(|V|^2)$ である。

iii) a の手数: 補題4.1 より $O(|V| + |E|)$ である。

iv) b の手数: 補題4.8 より $O(|V| \cdot |E|)$ である。

v) その他の手数: $O(|E|)$ である。

i), ii), iii), iv), v) から、PLANARIZATION の実行の手数は高々

$$O(|V|^2 + |V|^2 + (|V| + |E|) + |V| \cdot |E| + |E|) \quad (4.12)$$

すなわち、 $O(|V| \cdot |E|)$ である。

(証明終)

[定理4.3] PLANARIZATION の使用するメモリスペースは高々 $O(|V| \cdot (|E| - |V| + 1))$ である。

(証明) i) FL のメモリスペース: 図4.4よりわかるように、各節点についてそのメモリスペースは、高々フロンドの数 $(|E| - |V| + 1)$ に比例する。従って、 $O(|V| \cdot (|E| - |V| + 1))$ である。

ii) DFS の使用するメモリスペース: 文献[39]より $O(|V| + |E|)$ である。

iii) FP が埋め込み可能か否かを判定するためのメモリスペース：文献 [39] より $O(|V| + |E|)$ である。

i), ii), iii) から, PLANARIZATION の使用するメモリスペースは高々,

$$O(|V| \cdot (|E| - |V| + 1) + (|V| + |E|) + (|V| + |E|)) \quad (4.13)$$

すなわち, $O(|V| \cdot (|E| - |V| + 1))$ である。 (証明終)

【定理 4.4】 [平面化の手続き] の実行の手数は高々 $O(|E| \cdot (|E| - |V| + 1))$ である。

(証明) 操作 2 の一回の繰返し毎に $EMBED'$ の手数, すなわち $O(|V| + |E|)$ が必要である。同様に操作 3 では $O(|V| + |E|)$ である。繰返えしの手数はすべての FP の数, すなわち $(|E| - |V| + 1)$ であるから, 全体としては,

$$O(\{|V| + |E| + (|V| + |E|)\} \cdot (|E| - |V| + 1)) \quad (4.14)$$

すなわち, $O(|E| \cdot (|E| - |V| + 1))$ である (証明終)

定理 4.2, 定理 4.4 からわかるように, 極小除去枝集合を求める際, $EMBED'$ をそのまま適用した場合には $O(|E|) \cdot (|E| - |V| + 1)$ の手数を要したものが, PLANARIZATION を適用することにより, $O(|V| \cdot |E|)$ の手数で行なえる。

4.3 例題

この節では, PLANARIZATION をプログラムして実行させた結果を示し, その評価をする。まず最初に, 操作の過程を理解し易いように, 図 4.6-(a) に示されるグラフ G に対して PLANARIZATION を適用することを考える。ただし, G は 2 連結なグラフであり, その節点と枝の数はそれぞれ 14, 24 である。さらに, 図中の数字は節点を表わすとともに DFS を施した場合の NUMBER の値にも対応している。

図 4.6-(b) は G へ DFS を施した結果のグラフ $\vec{G}$ であり, 実線は木枝を, 破線はフロンドをそれぞれ表わしている。

$\vec{G}$ に対して PLANARIZATION を施した結果, 極小除去枝集合 $D(G)$ は,

$$D(G) = (5, 3) \quad (4.15)$$

であり, これは同時に最小除去枝集合 (すべての極小除去枝集合のなかで元の数が最小のもの) にもなっている。この様子を図 4.6-(c) に示す。一般

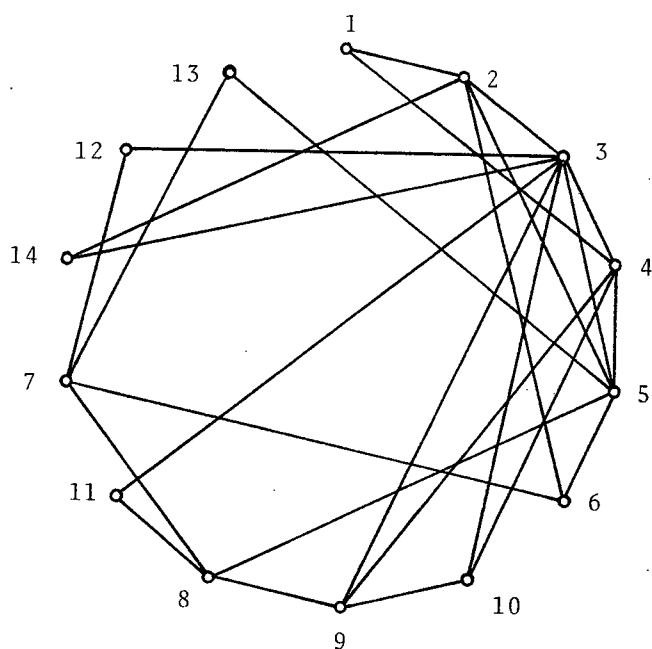


図4.6(a) 与えられたグラフ G

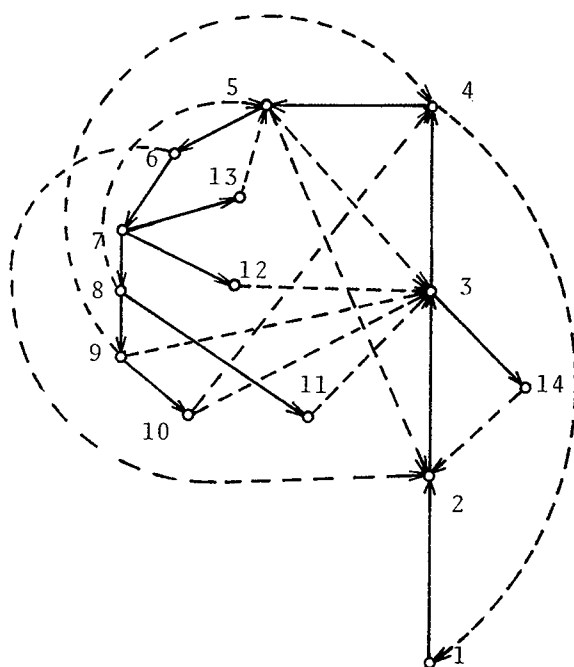


図4.6(b) G へDFSを適用した結果のグラフ $\vec{G}$

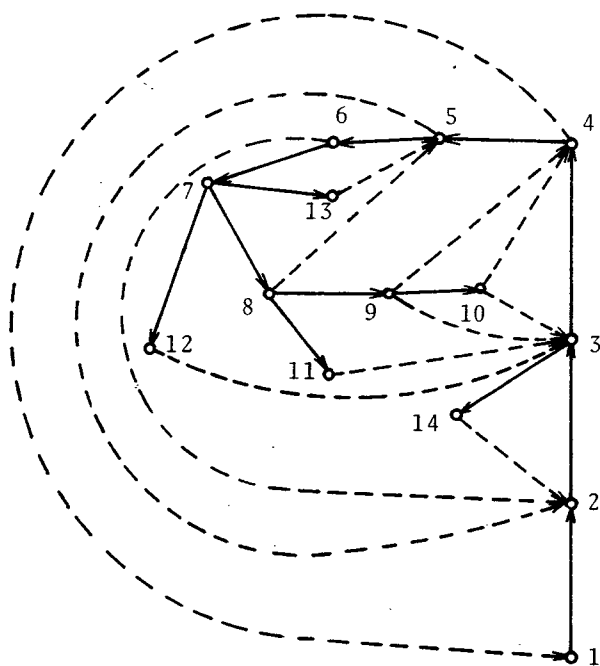


図4.6(C) $\hat{G}$ へPLANARIZATIONを適用して
平面描画したグラフ $\hat{G}$

に，与えられた平面グラフに対してその平面描画は一意的には定まらない。しかし，PLANARIZATIONでは，グラフ G の隣接構造ALとDFSを施す際の出発点 v が決まると， G の平面描画は一意的に決定される。

次に，節点の数と枝の数を変数として，random発生させた2連結な各グラフに対してPLANARIZATIONを適用したときの実行時間を表4.1に示す。

表4.1 PLANARIZATIONの実行結果 (NEAC2200-700)

節点数	枝数	実行時間 (m sec)	節点数	枝数	実行時間 (m sec)
10	16	82	60	116	1675
20	36	256	70	136	2366
30	56	474	80	156	2695
40	76	767	90	176	3266
50	96	1107	100	196	4001

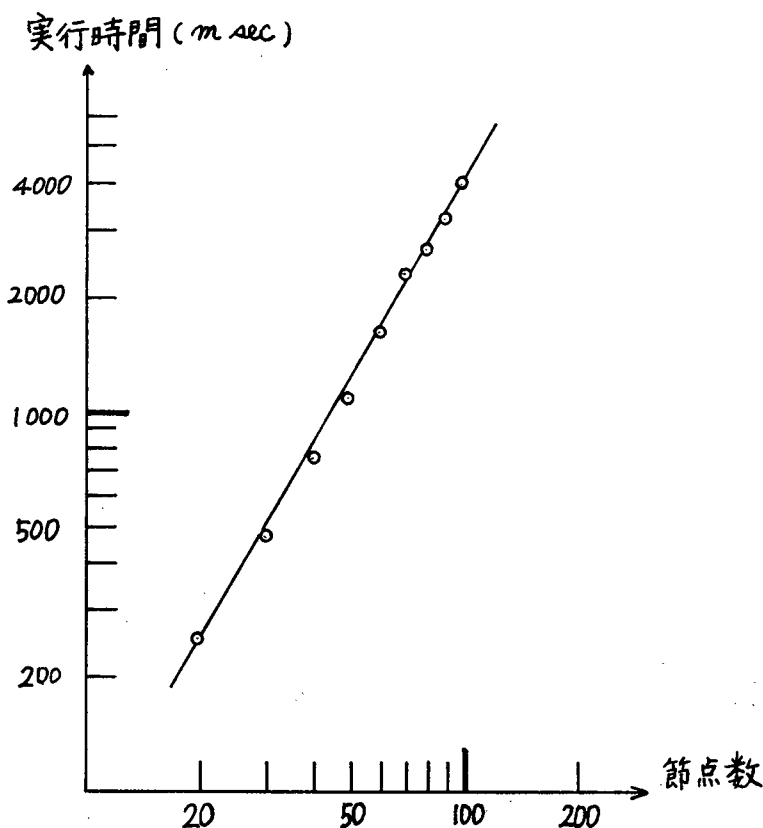


図4.7 表4.1のグラフ

さらに、これを図4.7に表わす。この図から、節点数 $|V|$ と実行時間 T との間には次式で示される関係が成立することがわかる。

$$\log T = 1.73 \cdot \log |V| + G \quad (G \text{ は定数}) \quad (4.16)$$

したがって、

$$T \propto |V|^{1.73} \quad (4.17)$$

が成立する。

つぎに、平均の線度 d をパラメータとして、グラフをランダムに発生させ、それらに対して PLANARIZATION の実行の手数を求めたものが、図4.8である。この図から、 $d = 2, 3, 6, 8$ の各データの実行時間 T はそれぞれ次のようになる。

$$T_{d=2} = O(|V|^{1.88}), \quad T_{d=3} = O(|V|^{1.74}),$$

$$T_{d=6} = O(|V|^{1.65}), \quad T_{d=8} = O(|V|^{1.57}). \quad (4.18)$$

これらの式から、平均線度が高くなるに従ってTの上昇率が低くなることがわ

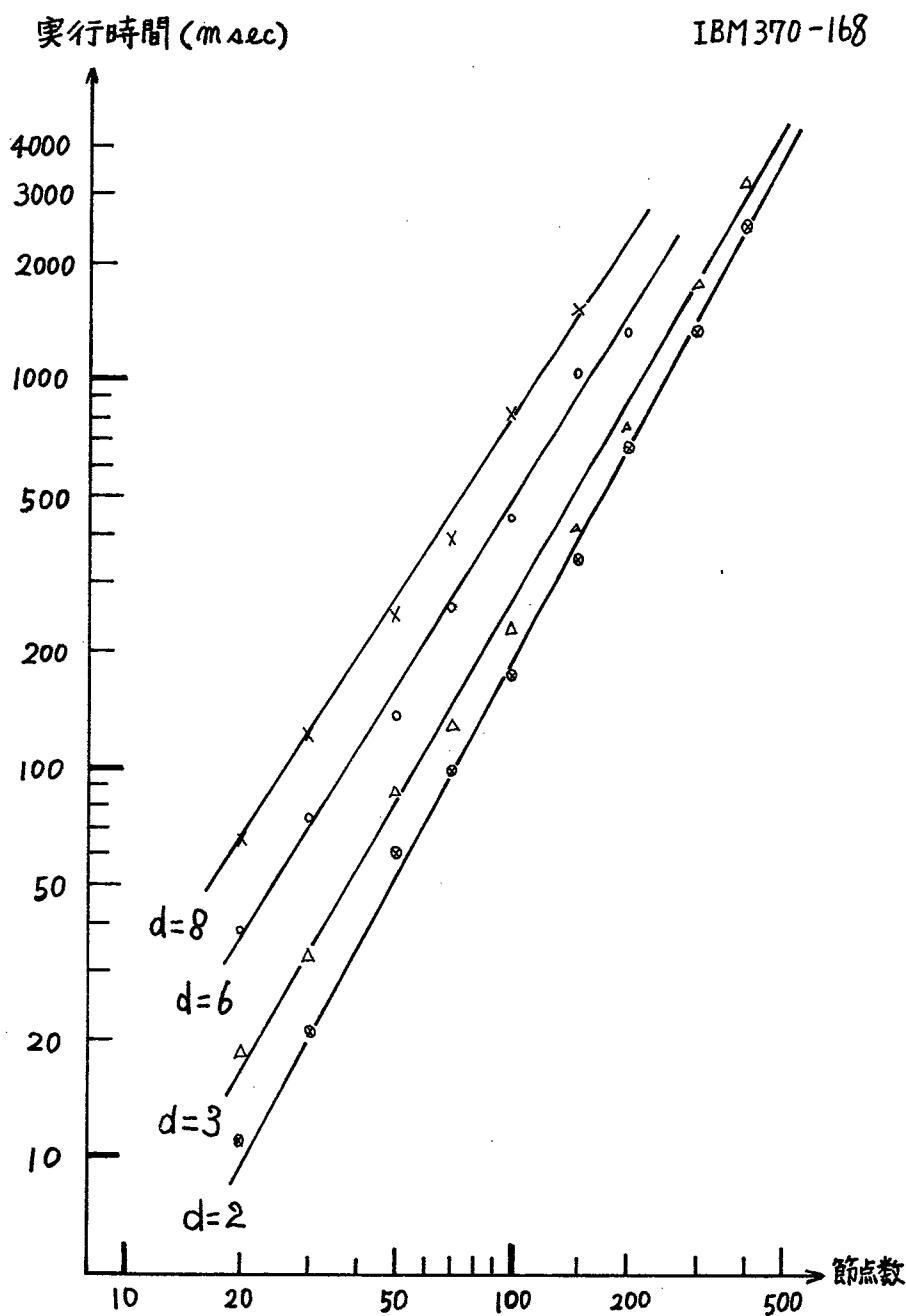


図4.8 PLANARIZATION の実行結果Ⅱ

かる。すなわち、枝数が多くなるに従って、FLの構造が単純化され、一つの節点に対してその可能なつ組の数が枝数の上昇の割には上がらないことが原因であろう。

これらの結果より、PLANARIZATION が実用的な実行時間内にあり、しかもその伸び率が低く、十分実用的なことがわかる。

4.3 結言

一層印刷回路基板の自動設計に関して一接近法を示し、その際必要となるグラフの平面化問題に対し $O(|V| \cdot |E|)$ の効率的なアルゴリズムを示した。このアルゴリズムは、与えられたグラフの極小除去枝集合を求めるものであり、最小除去枝集合を求めるものではないが、図4.1-bに示したようにこの接近法全体がヒューリスティックな手法から成っているので、このグラフの平面化の部分だけを最適化するという必要性はない。

今後残された問題としては次のものが挙げられる。

- (i) 回路の適切なグラフ表現：IC，トランジスタ，抵抗等，各回路素子の持つ特性（端子の数，素子の方向性）を考慮し，さらに，素子の下を通る配線を考慮したグラフ表現。
- (ii) 配線の平面性を保存し，かつ回路素子間の配線容量を考慮した素子の配線。

これらの問題に対して別々に接近法を考えることはできないが，例えば(ii)に関しては，人手を介入させて配置の改良を行なうべきであろう。

なお，本文中グラフの平面描画に関しては言及しなかったが，平面グラフが与えられた場合，文献[38]の手法を用いて $O(|E| + |V|)$ の手数でその平面描画が得られる。作製したプログラムは与えられたグラフの平面化を行なうと同時に，その平面描画も与えるものである。

第5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つぎにまとめる。

第2章では、計算機援用設計手法の一つとして迷路法に着目し、その効率的な手法を提案した。従来の手法においては不完全であった迷路法の特徴——配線が可能ならば、必ず一つの配線経路を見い出す——を、本手法では完全なものとしてすることができた。迷路法の持つこの極めて実用性に富む特徴から考えて、印刷回路基板の計算機援用設計システムにおいて重要な役割を果たすものであると思われる。この手法をいかに他の配線手法と組み合わせるかということと、配線をより一層考慮に入れた配線アルゴリズムの研究、開発が、印刷回路基板の完全な自動設計に近づく道であろう。

第3章では、迷路法と多層印刷回路基板に適用可能なように拡張できることを示し、そのアルゴリズムを提案した。一般に電子装置は増々大規模化される傾向にあるが、基板の多層化もそれに伴って進められている。このような状態において、迷路法が多層基板適用が必要となってくる。今後残された問題としては、与えられた信号網に対してどの層を割当てるか、さらにどのビアホールを割当ててかが、多層印刷回路基板自動設計への重要な問題である。

第4章では、グラフの平面化問題に関して考察し、一つの効率的なアルゴリズムを示した。与えられたグラフから最小個数の枝を開放除去して平面化する問題はNP-Complete ではないかと言われている。これに対し、本アルゴリズムは極小除去枝集合を求めるものであるが、応用として一層印刷回路基板の計算機援用設計を考えた場合、これを求めるだけで十分であるといえる。さらに実験結果から、このアルゴリズムが実行時間の面においても実用に耐えるものであることがわかった。今後は、与えられた回路をいかにグラフ表現するかという問題と、平面化され、平面描画されたグラフを基にして、実際の基板上へいかにして回路モジュールを配置するか、すなわち素子間配線容量を考慮した配線問題が重要なものとして残っている。

一般に、印刷回路基板の計算機援用設計を考える場合、個々の手法は勿論のこと、ハードウェアを含めてそれらをいかに有効に組み合わせて一つのシステムに仕上げるかが大きな問題であるが、ハードウェアとソフトウェアの両面を兼ね備えた機能としての人間をこのシステムに組み入れることも考慮すべきであろう。

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、つねに励まされていた尾崎弘教授、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。

大学院前期、後期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、寺田浩詔教授、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2章について有益な御助言、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、日本電気株式会社可見賢二氏、大附辰夫氏、川西宏氏、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。

第3章について有益な御助言、御討論をいただいたカリフォルニア大学バークレイ分校のE. S. Kuh 教授、B. S. Ting 氏（現在ベル研究所）に深く感謝する。

本研究に関し、福井大学谷口慶治教授、基礎工学部柏原敏伸助手、長崎造船大学川端信賢助手、徳島大学坂本明雄助手、日本電信電話公社武蔵野電気通信研究所松田潤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。

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助教授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、大学院学生世尾勤氏、築山修治氏、トランディンアム氏、久保登氏、上田輝幸氏、秦顕氏、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, G.R.Hogsett, R.L.Lunde and L.E.Kugel, "Ecap II-An electronic circuit analysis program", IEEE Spectrum, vol.8, no.6, pp.14 (June 1971).
- [2] C.Y.Lee, "An algorithm for path connection and its applications", IRE Trns., EC-10, pp.346 (Sept. 1961).
- [3] J.M.Jeyer, "Connection routing algorithm for printed circuit board", IEEE Trans., CT-18, no.1, pp.95 (Jan. 1971).
- [4] F.Rubin, "The Lee path connection algorithm", IEEE Trans., C-23, no.9, pp.907 (Sept. 1974).
- [5] S.B.Akers, Jr., "A modification of Lee's path connection algorithm", IEEE Trans. (Short Notes), EC-16, no.2, pp.97 (Feb. 1967).
- [6] J.H.Hoel, "Some variations of Lee's algorithm", IEEE Trans., C-25, no.1, pp.19 (Jan. 1976).
- [7] A.Hashimoto and J.Stevens, "Wire routing by optimizing channel assignment within large apertures", Proc. 8th Design Automation Workshop, pp.155 (1971).
- [8] D.W.Hightower, "A solution to line-routing problem on the continuous plane", Proc. 6th Design Automation Workshop, pp.1 (1969).
- [9] 山村, 白川, 尾崎, "2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), vol.57-A, no.9, pp.671 (1974)
- [10] K.Mikami and K.Tabuchi, "A computer program for optimal routing of printed circuit conductors", IFIP Congress 68, pp.1475 (1968).
- [11] H.C.So, "Some theoretical results on the routing of multilayer, printed-wiring boards", Proc. IEEE Internat'l Symp. Circuits and Systems, pp.296 (1974).
- [12] E.S.Kuh and B.S.Ting, "The backboards wiring problem: Some results on single-row routing", Proc. IEEE Internat'l Symp. Circuits and Systems, pp.369 (1975).
- [13] H.Nakahara, "Computer-aided interconnection routing: General survey of the state-of-the-art", Networks, vol.2, pp.167 (1972).
- [14] M.A.Breuer (ed.), "Design Automation of Digital Systems, vol.1", Englewood Cliffs, Prentice-Hall, N.J. (1972).
- [15] I.Aramaki, T.Kawabata and K.Arimoto, "Automation of etching-pattern layout", Comm. ACM, vol.14, pp.720 (1971).
- [16] J.R.Allen, "A topological adaptable cellular router", Proc. 13th Design Automation Conf., pp.161 (1976).

- [17] G.L.Patterson and B.H.Phillips,"A proven operational CAD system for p.w.b. design-based on a mini-computer and featuring fully automatic placement and routing", Proc. 13th Design Automation Conf., pp.259 (1976).
- [18] B.R.Rau,"A new philosophy for interconnection on multilayer boards", Proc. 13th Design Automation Conf., pp.225 (1976).
- [19] L.Abel,"On the automated layout of multi-layer planar wiring and a related graph coloring problem", Coodinated Sci. Lab. Rept., no.R-546, Univ. Ill., (1972).
- [20] N.Sugiyama, S.Nemoto, K.Kani, T.Ohtsuki and H.Watanabe,"An integrated circuit Design problem on graph-theoretical approach", ISSCC (Feb. 1970).
- [21] T.Ohtsuki, N.Sugiyama and H.Kawanishi,"An optimization technique for integrated circuit layout design", IECE Internat'l Conf. on Circuit and System Theory (1970).
- [22] L.Steinberg,"The backboard wiring problem:A placement algorithm", SIAM Review, vol.3, no.1, pp.37 (Jan. 1961).
- [23] R.A.Rutman,"An algorithm for placement of interconnected elements based on minimum wire length", Proc. AFIPS Conf. '64 SJCC (1964).
- [24] A.Basden and K.G.Nichols,"New topological method for laying out printed circuits", Proc. IEEE Electronics, vol.120, no.3, pp.325 (March 1973).
- [25] D.G.Schweikert and B.W.Kernighan,"A proper model for the partitioning of electrical circuits", Proc. 9th Design Automation Workshop, pp.57 (1972).
- [26] H.R.Charney and D.L.Plato,"Efficient partitioning of components", Proc. 5th Design Automation Workshop, pp.16-1 (1968).
- [27] D.M.Schuler and E.G.Ulrich,"Clustering and linear placement", Proc. 9th Design Automation Workshop, pp.50 (1972).
- [28] 西川, 岡田, 野下, 苗村, 橋本, "集積回路マスクパターン設計システム MEDIC のアルゴリズム", 信学論 (A), vol. 55-A, no.6, pp. 289 (1972).
- [29] W.M.vanCleemput and J.G.Linders,"An improved graph theoretic model for the circuit layout problem", Proc. 11th Design Automation Workshop, pp.82 (1974).
- [30] D.A.Mlynski,"The graph theoretic approach to the circuit layout problem", Proc. IEEE Internat'l Symp. on Circuit and Systems (1974).
- [31] W.M.vanCleemput,"On the topological aspects of the circuit layout problem", Proc. 13th Design Automation Conf., pp.441 (1976).

- [32] L.Auslander and S.V.Parter,"On the imbedding graphs in the sphere", Journal of Mathematics and Mechanics, vol.10, no.3, pp.517 (May 1961).
- [33] A.J.Goldstein,"An efficient and constructive algorithm for testing whether a graph can be embeded in the plane", Graph and Combinatorics Conf., (Princeton Univ.) (May 1963).
- [34] G.J.Fisher and O.Wing,"Computer recognition and extraction of planar graphs from the incidence matrix", IEEE Trns., CT-13, no.2, pp.154 (June 1966).
- [35] 伊理, "平面グラフを平面上へ描く方法について", 信学会 回路とシステム理論研究会資料, CT 68-28 (1968).
- [36] J.Hopcroft and R.E.Tarjan,"Planarity testing in VlogV steps: Extended Abstract", Computer Science Department, Stanford Univ., STAN-CS-71-201 (1971).
- [37] 伊理, "グラフの理論とその応用", 信学会誌, vol. 54, no. 12 (1971) — vol. 55, no. 5 (1972).
- [38] J.Hopcroft and R.E.Tarjan,"Efficient planarity testing", J. ACM, vol.21, no.4, pp.549 (Oct. 1974).
- [39] R.E.Tarjan,"Depth-first search and linear graph algorithm", SIAM J. Comput., vol.1, no.2, pp.146 (June 1972).
- [40] R.M.Karp,"Reducibility among combinatorial problems", Complexity of Computer Computations, R.E.Miller and J.W.Thatcher, eds., Plenum Press, New York, pp.85 (1972).
- [41] 尾崎, 白川, 翁長, "グラフ理論", コロナ社 (1975).
- [42] 尾崎, 樹下, 白川, "情報回路論(I)", コロナ社 (1970).
- [43] F.Harary,"Graph Theory", Addison-Wesley (1969).

To My Wife

December 8, 1976.

