# グローバル・ルータの高速化について

浅野哲夫（大阪電気通信大学）

## 1. はじめに

プリント基板の自動配線手法として，C.Y.Leeの迷路法[1]や，Mikami，Hightowerの線分探索法[2,3]などに代表される多くの技法が提案されているが，これらの手法には，配線順序の決め方によって配線率が大きく左右されるという欠点があった。その最大の原因は，既に見つけられた配線経路が固定されてしまうので，それ以後の経路探索においては既配線を障害物として扱わなければならないところにあると思われる。これに対して，J.Soukupが提案したグローバル・ルータ[4,5]は，すべてのピンの位置から同時に経路探索の波を発生させることにより，すべての配線経路を同時に求めようとするものであり，並列的に実行可能であるように設計されている。またこの方法は，配線経路をピンとピンを結ぶ線としてではなく，一つの配線に属するすべてのピンを含む可動連結領域として捉えるという点において，従来の方法と全く異なる。このように，ネットの連結性を領域の連結性として捉えているので，ある配線の経路を確保するために，既に連結されたネットの領域を移動させることが比較的容易であり，その意味で，従来の配線手法より融通性に富んでいると言えよう。

ところで，既配線の領域の一部を削り取って他の配線の領域に振り替えようとするとき，その操作を行なった後でも領域の連結性を保てるかどうかを判定しなければならない。Soukup[4,5]は，Leeタイプの拡張を行なうことによって連結性を判定しているが，時間がかかり過ぎるのが欠点である。筆者は，従来の線による連結性をSoukupのグローバル・ルータに取り入れることにより，領域の連結性を高速に判定する方法を考案した。具体的には，あるネットが連結されたことが解った時点で，同時に一時的な仮の経路を求めておき，経路上にない部分の置きかえについては，時間のかかる連結性テストを省略しようという考え方である。経路上の部分の置き換えに際しても，迂回路を探索するという方法で計算時間の短縮を図っている。最後に，計算機実験の結果についても述べる。

## 2. Soukupのグローバル・ルータ

Soukupは2種類のグローバル・ルータGR1およびGR2を提案しているが，本報告ではGR1についてのみ考察する。以下，GR1を簡単に説明する。

まず，対象となるプリント基板の面（一層を仮定）を，格子状の網目（セル）に区切る。各セルについて，ネット番号，サブネット番号（最初は，それぞれのピンに対応した番号がつけられている），およびプライオリティが定められる。ただし，障害物上のセルについてはネット番号を－1とし，ピンでも障害セルでもないセルについては，利用可能という意味で，ネット番号とサブネット番号を共に0にしておく。また，チェス盤のように，セルは黒と白に色分けされており，白のセルから黒のセルへの拡張と黒のセルから白のセルへの拡張が交互に繰り返される。

白から黒（黒から白）への拡張においては，それぞれの黒（白）のセルCにつ

いて、その4隣接セルの中で最高のプライオリティをもつセル$C^*$を求め、セル$C^*$の情報（ネット番号，サブネット番号，およびプライオリティ）をセルCに移す。一般的には、セルCのプライオリティがセル$C^*$のプライオリティより高いか、あるいは等しいこともあり得るが、そのような場合は、セルCの書き換えは行なわない。実際にプログラム化する場合は、周囲に自分よりプライオリティの高いセルがあるようなセルの集合を記憶しておくのが望ましい。ただし、セルCがピンや障害物を含んでいたり、セルCをセル$C^*$のネット（またはサブネット）に含ませると、セルCを含んで連結していたネット（またはサブネット）の連結性が損なわれるような場合——このようなセルをボトルネックセルという——には、この書き換えは行なわない。

複数個のピンを含むネット（またはサブネット）の領域内のセルCがボトルネックセルであるかどうかは、2段階に分けて判定される。まず最初は、そのセルを中心とする3×3の局所領域を考え、この中で局所的なボトルネックかどうかを判定する。具体的には、周囲の8個のセルを時計回りに見回したとき、中心セルと同じネット（またはサブネット）に属するセルからそうでないセルへの変わり目が2度以上あれば、その中心セルを局所的なボトルネックセルであると判定する。

セルCが局所的なボトルネックセルであると判定された場合には、引き続き、グローバルにも真のボトルネックかどうかが調べられる。これは、セルCを障害物と見なしても、セルCを含んでいたネット（またはサブネット）の領域が非連結にならないかどうかで判断されている。具体的には、セルCを含んでいたネット（またはサブネット）の領域内でLeeタイプの拡張を行ない、領域内の全てのセルに到達可能かどうかで領域の連結性が確かめられる。これは明らかに非常に手間のかかる作業であり、改善が必要である。

拡張を繰り返すうちに、同じネットに属する二つのサブネット領域が接触したことが解ると、一方のサブネット領域内のセルのサブネット番号を他方の番号に書き換えることにより、二つのサブネットを統合する。このようにして、あるネットに対するサブネットがただ一つになったとき、そのネットに対する配線は完了したことになる。

プライオリティは次のルールで定められる[5]。

(i) 未接続のネットは既に接続が完了したネットより高いプライオリティを持つ。

(ii) 既に接続が完了したネットについては、現在そのネットが含んでいるセルの個数に比例したプライオリティをもつ。未接続のネットについては、各サブネットに対して、同じネット内の最も近いサブネットまでの距離に反比例するプライオリティを与える。

(iii) 二つのサブネットが同じプライオリティを持たないように、番号順の優先度も考慮する。

GR1は、すべてのネットの配線が完了したとき、あるいは、変化が全く起こらなくなったとき終了する。プライオリティの定め方から明らかなように、無限ループに陥ることはあり得ない。

## 3. ボトルネック判定の高速化

あるセルがボトルネックセルであるかどうかを判定するのに，Lee タイプの拡張を行なうという Soukup の方法では，必らず領域の面積に比例する時間が必要であり非能率である。並列的に拡張を行なうことを考えてもやはり非能率であろう。そこで，この判定時間，および判定回数を低減させる方法を考案した。基本的なアイデアは，連結されたネットまたはサブネットについては一時的な仮の配線経路を求めておいて，経路上でないセルについては（当然，ボトルネックセルではあり得ないから），ボトルネックセルかどうかの判定を省略しようというものである。ただし，各セルにおいて経路上のセルかどうかを示すフラグを用意しなければならない。この方法の利点を列挙すると次のようになる。

I. 局所的なボトルネックかどうかの判定が不必要になる。

II. 本当のボトルネックかどうかの判定回数も，領域の幅が広い程少なくなると考えられる。ただし，後に述べるように，経路を記憶する場合としない場合とで領域の拡張の様子が若干異なるので，厳密な意味において二つの方法におけるボトルネック判定回数を比較することはできない。

III. 先に得られている経路を利用して迂回路を探すという方法により，ボトルネック判定に要する時間を短縮できる。

IV. 予め，線分探索法などを用いて配線経路を探した後で，未配線の部分を GR1 によって処理しようとするとき，既に得られている配線経路に関する情報を有効に活かすことができる。

筆者の方法では，二つのサブネットが接触したとき，仮の配線経路を求めるという操作が必要になるが，Soukup の方法においても最終的には配線経路を求める必要があるので，この作業のために余分の時間を費すというわけではない。

以下に具体的な手続きを述べる。

### 3.1 サブネットの統合

ネット番号が同じでサブネット番号が異なる二つのセル $c_1$ と $c_2$ が隣接することがわかったとき，図1に示すように，まず $c_1$ を含むサブネットの中で $c_1$ から"経路上"の印がついたセルへ至る経路を探索する。同様に，$c_2$ を含むサブネットの中でも $c_2$ から"経路上"のセルへの経路を探索する。もし両方で経路が見つかれば，それらの経路上のセルに"経路上"なる印をつける。ただし，ピンを含むセルには最初から"経路上"の印をつけておく。

この後，二つのサブネット領域内の各セルのサブネット番号とプライオリティを書き換える。また，吸収された方のサブネット番号を記録しておく。

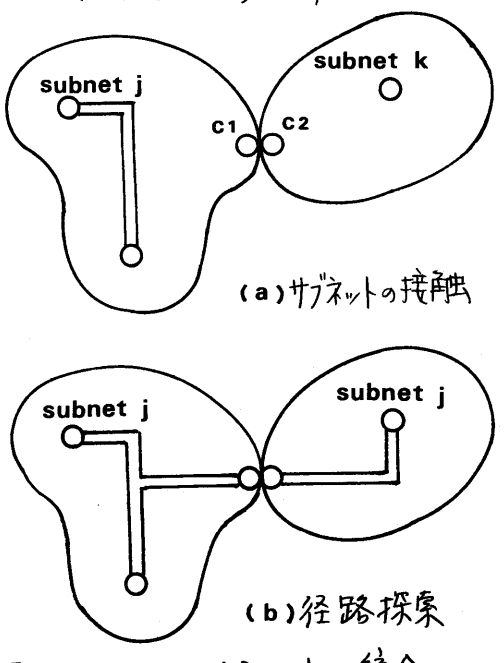


図1. 二つのサブネットの統合

### 3.2 Dead Bayの除去

各サブネット領域がそれぞれ単一の連結領域から成っているなら問題はないが，実際には，内部にピンを全く含まない領域が生ずる場合がある。このような領域をSoukupはdead bayと呼んでいる。Soukupの方法では，連結状態にあるサブネット領域は常に単一の連結領域になるようにボトルネック条件が定められているので，dead bayとなり得るのは孤立したピンに対するサブネットのみである。尤も，そのためには，二つのサブネットが統合されるたびに飛び地を消去しておくという面倒な作業が必要となる。本当のボトルネックかどうかを，Leeタイプの拡張により領域内の全ての点に到達可能かどうかで判定しようとするなら，上記の飛び地消去の作業は必要である。これに対して，筆者の方法では，連結性を重視していないので，頻繁にdead bayが生じる。

上記の理由により，同じネットに属する二つのサブネット領域が$C_1$と$C_2$において接触したとき，$C_1$および$C_2$から"経路上"のセルへ到る経路が必ず見つかるという保証はない。つまり，セル$C_1$あるいは$C_2$を含む領域がdead bayであれば，経路は見つからない。このような場合には，検出されたdead bay領域内のセルのネット番号を0にすることで，利用可能セルにリセットする。

これ以外にもdead bayが検出されることがある。3.1で述べたように，二つのサブネットを統合するとき，一方のサブネット番号は消される。したがって，消去されたほうのサブネット番号をもつセルは以後出現しない筈である。しかしながら，もともとdead bayがあったとすると，この領域内のセルのサブネット番号は書き換えられていないから，消された筈のサブネット番号をもつセルが現われることになる。このように，存在しないサブネット番号をもつセルによってdead bayが検出された場合も，その領域内のセルを利用可能セルにリセットする。

### 3.3 ボトルネック判定

先にも述べたように，サブネットを統合する都度仮の配線経路を求めておけば，GR1における局所的なボトルネックセルかどうかの判定は省略できる。また，経路上にないセルは本当のボトルネックセルではあり得ないわけであるから，経路上のセルを書き換えようとするときにのみ，ボトルネックのどうかの判定を行なえばよいことになる。この場合にも，まずそのセルを中心に3×3あるいは5×5の局所領域を考え，その中で迂回路を探す(図2参照)。したがって，GR1と同様の手間のかかる作業が必要となるのは，そのセルが経路上にあり，しかも局所的な迂回路が見つからない場合のみである。この場合にのみGR1と同様のLeeタイプの拡張を行なったとしても，明らかに高速化が達成されている。筆者の方法では，この場合においても迂回路が見つかるかどうかで連結性を確かめる。その方法を以下具体的に示す。

経路上のセルCがボトルネックセルかどうかを調べるものとする。この要請があるということは，セルCの4隣接セルのうち，少なくとも一つはセルCと異なるネット番号をもつ。したがって，セルCの4隣接セルのうち，セルCと同じサブネットに属し，かつ経路上にあるセルは高々3個である。それらのセルを$C_1 \sim C_m$ ($m = 2$または3)としよう。基本的

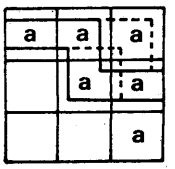

図2．局所的な迂回路

な考え方は，これらのセル$C_j$（$j = 1 \sim m$）を仮想的なピンと考え，既に得られている経路を用いて，これらの仮想ピンを相互に接続する配線経路を見つけようというものである。具体的には，まず各セル$C_j$（$j = 1 \sim m$）から経路を逆に辿り，作業用の配列を用いて，$C_j$からの経路上のセルに番号$j$を書き込んでいく。$C_j$から経路を辿っている途中でセル$C_k$（ただし，$1 \leq j < k \leq 3$）が見つかれば，セル$C_k$にも番号$j$を書き込み，以後$C_k$は考慮の対象からはずす。

次に，セルCを含んでいたサブネット領域に範囲を限定した上で，1～mの番号が書き込まれたセルから同時に波を発生させる（図3(b)参照）。このとき，2ビットで指定される逆戻りポインタも記録していく。このようにして，番号の異なる二つのセルが隣接していることが検出されると，逆戻りポインタを用いてそれぞれのセルから"経路上"のセルまでの経路を求め，経路上のセルの位置をスタックに蓄えると共に，それらのセルに"経路上"の印をつける。以上の操作を，$C_1 \sim C_m$が全て接続されるか，あるいは接続できないことがわかるまで繰り返す。

サブネット領域全体に拡張を行なっても$C_1 \sim C_m$を接続することができなかった場合，セルCはボトルネックセルであるから，セルCをこのサブネットから取り除くことはできない。このとき，$C_1 \sim C_3$のうちの2つを結ぶ経路が見つかっていた場合には，その経路を消しておかなければならない。経路上のセルをスタックに蓄えたのはそのためである。

逆に，$C_1 \sim C_m$を接続する経路が見つかった場合，セルCはボトルネックセルではないから，このサブネットから取り除くことができる。

以上の操作を図的に説明したのが図3(a)～(d)である。ただし，このままでは図3(c)のように余分な経路が含まれているから，整形操作を行なう必要がある。この整形操作は，各$C_j$から経路を辿って最初に出会う分岐点またはピンまでの部分を経路から取り除くことによって達成できる。

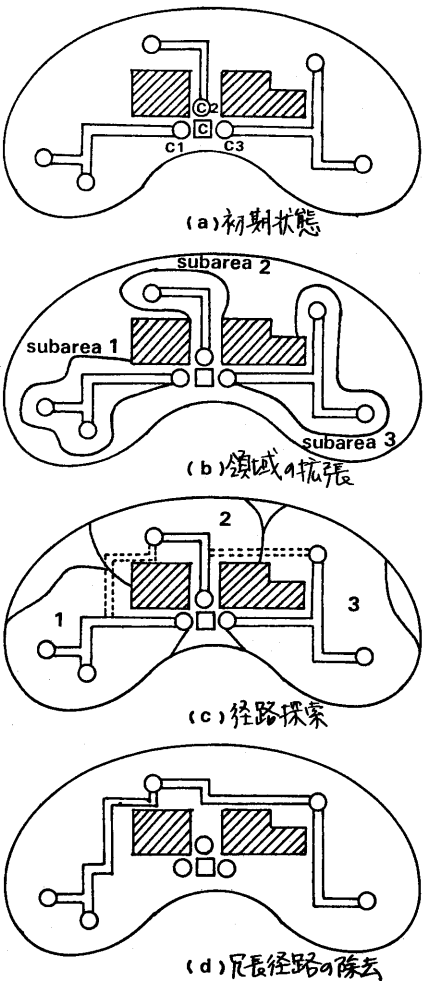

図3. ボトルネックセルかどうかの判定

## 4. 例題

図4(a)に示した例題について考えよう。この例題は，3本のネットおよび7個のピンを含んでいる。図では，丸印○でピンを表わし，a2などの記号によってどのネットの何番ピンかを表わしている。また，#記号は障害物を表わしている。

最初，各ピンに対応して合計7つのサブ

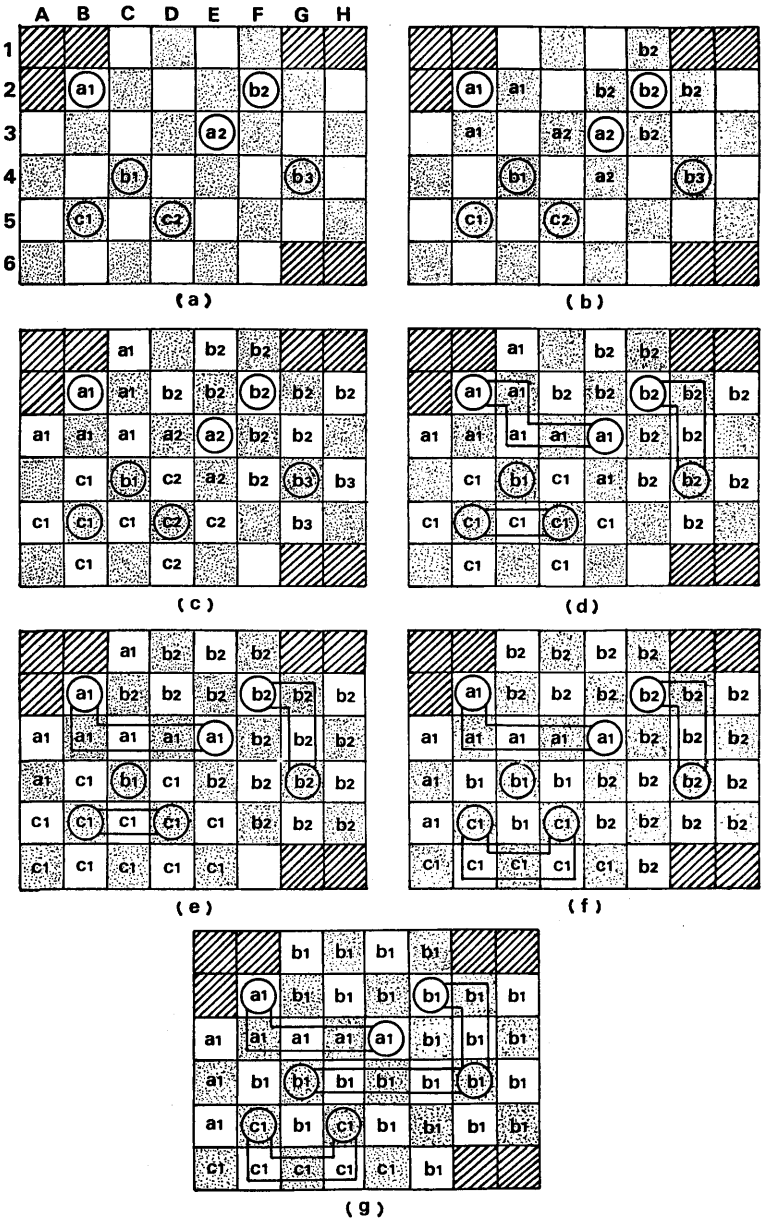


図4．例題
(a)初期状態 ── サブネットのプライオリティの順序は，C1 > C2 > b2 > b3 > a1 > a2 > b1．
(b)黒への拡張．
(c)白への拡張
a1がa2に接触．
b2がb3に接触．
C1がC2に接触．
(d)接触しているサブネットを統合し，経路を求める。ネットaとCは配線が完了したのでプライオリティが下がる。
(e)黒への拡張．
(f)白への拡張
b1がb2に接触．
(g)最終状態．

ネットa1，a2，b1，…，c2が定義される。そのプライオリティは，定義より，c1 > c2 > b2 > b3 > a1 > a2 > b1である。さて，図4(a)の状態から黒のセルへの拡張，引き続いて白のセルへの拡張を行なうと，図4(b)，(c)のようになる。ここで，サブネットa1とa2，b2とb3，およびC1とC2が接触しているので，これらのサブネットを統合し，さらに仮の配線経路を求めると同図(d)を得る。これでネットaとネットCは配線が完了したのでプライオリティが下がり，今度は，b1 > b2 > a1 > C1の順になる。この後さらに黒のセルへの拡張を続けると，1Cの位置にdead bayを生じるが，特別な処理を施さなくてもこの場合には何ら影響はない。この

同じ拡張において、サブネット番号a1でかつ"経路上"の印がついたセル 2C が (e) ではサブネット番号b2に変わっている。これは、(e)に示すような（局所的な）迂回路が見つかったためである。これに対して、Soukupの定義によれば、(d)においてセル 2C は局所的なボトルネックであり、しかもこれを取り除くとセル 1C が dead bay になるので本当のボトルネックでもある。したがって、Soukupの方法では、セル 2C のサブネット番号をa1からb2に書き換えることは許されない。

続いて4回目の拡張を行なうと、(f)に示すようにサブネットb1とb2が接触する。この二つのサブネットを統合し、配線経路を求めれば配線完了である。

## 5. 実験結果

本文で述べた方法を実際にプログラム化し、大阪大学大型計算機センターの ACOS 77/900 TSS を用いて、小規模な例題について計算機実験を行なった。どの例題も一層である。使用言語は、電子技術総合研究所の真野芳久氏らによって開発された WESTRAN（プリプロセッサ形式の構造化 FORTRAN）である。比較のために、Soukupのグローバル・ルータ GR1 もプログラム化した。ただし、既に連結されたネットのプライオリティを、Soukupの定義のように、そのネットに含まれるセルの数に比例させようとすると、拡張の都度ネット内のセルのプライオリティを書き換えるという作業が必要となり、処理時間が大幅に増大すると考えられる。そこで、既連結のネットに対しては、ピン間距離の最小値に反比例するプライオリティを与えるようにした。そのため、全処理時間に対してボトルネック判定に要した時間の占める割合が、文献[5]において報告されている数値より高くなっている。

表1に実験結果を示す。実用化には程遠い計算時間を要しているが、2次元セルオートマトンのような並列動作が可能になれば、処理時間は大幅に短縮できるであろう。

表1．実験結果

| データ番号 | サイズ (メッシュ数) | ネット数 | ピン数 | 本手法 | | Soukupの方法 | |
|---|---|---|---|---|---|---|---|
| | | | | 時間(秒) | 配線率% | 時間(秒) | 配線率% |
| 1 | 15×24 | 18 | 48 | 10.15 | 94.44 | 16.76 | 88.89 |
| 2 | 45×75 | 30 | 98 | 34.17 | 100. | 149.04 | 100. |
| 3 | 60×75 | 34 | 109 | 107.04 | 76.47 | 541.58 | 79.41 |

## 6. むすび

本文では、Soukupが提案した配線手法（グローバル・ルータ）の問題点を幾つか指摘し、高速化のための手法を提案したが、その特徴は次の2点に集約される。

I．従来の"線"による連結性と、Soukupの主張する"領域"としての連結性の両方の長所を取り入れた。

II．線分探索法などの従来の手法との結合が容易に行なえる。

実験結果が示すように、速度の面では改善できたが、これはあくまで逐次的に

実行した場合のことである。元来，グローバル・ルータは並列実行可能なアルゴリズムとして記述されているから，並列実行においても本論文で述べた方法により速度が改善されるかどうかを確かめなければならないが，これについては今後の課題として残された。現在，並列計算のモデルを検討中である。

謝辞 最後に，種々の面で御協力頂いた本学学生の三浦起史氏と小山義弘氏に深く感謝する。特に，三浦起史氏にはプログラム作成において幾つか重要な示唆を頂いた。

参考文献

[1] C.Y. Lee : " An Algorithm for Path Connections and its Applications," IRE Trans., EC-10, p.346 (1961).
[2] K. Mikami and K. Tabuchi : " A Computer Program for Optimal Routing of Printed Circuit Conductors," IFIP Congress, 68, p.1475 (1968).
[3] D.W. Hightower : "A Solution to Line-routing Problems on the Continuous Plane," Proc. Design Automation Workshop, p.1 (1969).
[4] J. Soukup : " Global Router," Proc. of 16th Design Automation Conference, p.481 (1979).
[5] J. Soukup : " Global Router," Journal of Digital Systems, vol.4, p.59 (1980).