設 計 自 動 化 80－1
（1996．5．24）

容量制約を満たした配線経路探索方法

濱 利行　　江藤 博明

（株）日本アイ・ビー・エム 東京基礎研究所

本稿ではプリント基板等を対象とした自動配線システムの内部で使用している配線経路探索方法について述べる．本方式は，個々の信号線の位相的配線経路を求める際に，既配線を考慮したうえで，設計規則を検証しながら物理配線可能な経路のみを求めることを特徴としている．従来の方法では，位相配線の段階では設計規則は近似的に満たされているに過ぎなかった．しかし，基板の高密度化に対応するには，位相配線の段階で設計規則を満たしているいることが望ましい．三角形分割を基礎とした経路探索中の設計規則の逐次的検証方式と，検証による負荷を軽減するためのいくつかの高速化の手法についても合わせて述べる．

Routing path search exactly satisfying capacity constraints

Toshiyuki Hama　　Etoh Hiroaki

IBM Research, Tokyo Research Laboratory

This article describes a routing path search algorithm which is embedded in our auto-router designed for printed circuit boards. The algorithm searches for a homotopic path which is verified to be transformed to a physical path satisfying design rules. Existing homotopic path search algorithms verify design rules only locally and approximately. However, it is preferable to exactly satisfy design rules in the global routing phase, especially for routing a dense printed circuit board. We describe a method for incrementally verifying design rules during homotopic path search on a graph based on constrained Delaunay triangulation, and also several improvements on the routing path search algorithm for remeding the overhead of the routability verification.

1 はじめに

本稿ではプリント基板等を対象とした自動配線システムの内部で使用する配線経路探索方法について述べる．我々は現在，一層のプリント基板を対象とした自動配線システムを開発中である．自動配線は位相配線と物理配線の2つ段階に分けて実現されている．本稿で述べる配線経路探索は，位相配線での各信号線の配線経路の探索方法である．

配線経路は，いわゆる最短経路探索アルゴリズム(ダイクストラ法)で求められるのが一般的であるが，設計規則を満たすために配線間の間隙をある程度以上確保しなければならないところに工夫を要する．特に位相配線では既配線の物理的位置が確定していないので，クリティカルカットにより配線経路に十分な余裕が確保されているか検証しなければならない．既存の方法は，位相配線の段階では局所的な検証にとどめておき，後の段階で設計規則違反が発見された場合には修正を加えるという方法である．配線密度が低い場合は，局所的な検証を行なえば設計規則違反が実際に起きることは少ないが，高密度の配線を要求される場合には位相配線の段階で設計規則を厳密に満たした配線経路が得られることが望ましい．

本稿では，まず開発中の一層自動配線システムの概略と内部の基本的なデータ構造について述べる．続いて，このデータ構造上での設計規則の逐次的検証方法とそれを組み込んだ配線経路探索アルゴリズムについて述べる．さらに，設計規則の検証を組み込むことにより経路探索アルゴリズムが必要以上に遅くなってしまうのを避けるための高速化の手法についても述べる．

2 一層自動配線システム

2.1 位相配線と物理配線

現在開発中の一層自動配線システムは，位相配線(Topological Routing: 信号線の位相的な経路だけを決定する)と物理配線(Physical Routing: 設計規則を満たした信号線の物理的位置を決める)の2段階に分けて実現されている．

位相配線では各信号線の物理的位置は確定していないので，その後の物理配線時に設計規則を満たした配線が現実に可能であるかどうかが問題となる．Leiserson ら [3] は，すべてのクリティカルカットが設計規則に違反していなければ物理配線可能であることを示し，位相配線の配線可能性検証のためのアルゴリズム与えている．また，Dai ら [2] により 90 度，45 度配線等の固定角度の配線では，拡張ラバーバンドを構成できれば物理配線可能であることが示され，同様の原理による配線可能性検証アルゴリズムもいくつか報告されている [7, 5]．これらの成果が引きがねとなって位相配線と物理配線を分離する方法が採られ始められるようになった．

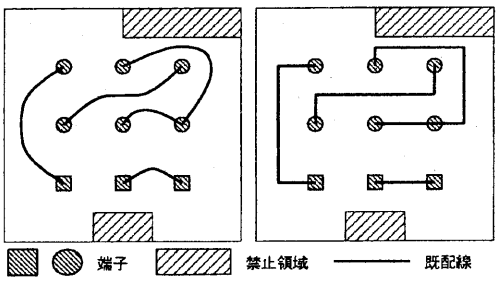

図 1: 位相配線と物理配線

しかしながら，配線可能性検証を位相配線終了後に行なっていたのでは，違反部分の修正のための処理を新たに用意する必要がある．我々は，このような例外処理が起きないように位相配線の結果は常に物理配線可能であることを保証する戦略を採ることにした．

また，基板の高密度化への対応のため，任意角度での配線を採用したので，先述の拡張ラバーバンドのスポーク生成による高速な配線可能性検証アルゴリズムは利用することができない．そこで，Leiserson らの結果に基づいて位相配線時にはすべてのクリティカルカット(ユークリッド距離による)について検証を行なう．

位相配線の結果は配線可能性が保証されているので，物理配線では位相的配線経路(Homotopic Path)をそれと位相的に等価な線分と円弧からなる物理配線(Curvilinear Wire)に変換する．配線可能性はユークリッド距離で検証されているので，スポークの代わりに円盤を用いた一種の拡張ラバーバンドにより最短の配線を求めれば設計規則は満たすことになる．この任意角度による物理配線の方法は別の機会に報告する予定である．

2.2 配線経路表現

位相的配線経路の表現および基礎となるデータ構造は，田中らの報告 [8] を参考とした．配線領域は端子を点とみなした Constrained Delaunay Triangulation[4] により三角形分割されており，確定した配線は三角形分割辺との交点の列により表現されている．無意味な迂回がなければこの表現により位相的に同じ経路は一意に表現される．

可視グラフに基づいた経路表現も考えられるが，設計規則の検証を探索アルゴリズムへ容易に組み込むことができるいう利点はあるものの，可視グラフの枝の数は節の数 n (ほぼ端子の数に等しい) に対して $O(n^2)$ であり，三角形分割による枝の数が $O(n)$ であることと比較してかなり大きい．また配線経路が確定した時のグラフの更新処理の複雑さからいっても三角形分割の方が有利である．

確定した配線は三角形分割辺との交点の列として表現されるので，新たな経路の探索は，既配線の隙間を探索することになる．そこで，図 2に示すように，既配線によって分けられる三角形分割辺の各断片上に対応する節を作り，隣合う節を枝で結んだ**探索グラフ**を作る．探索グラフは，物理的な障害物 (禁止領域，既配線) に邪魔されない限り到達可能な節を連結しているので，設計規則に照らすと実際には連結でない部分も含んでいる．従って，配線経路探索アルゴリズムは設計規則に照らして連結でない部分経路を避けながらこの探索グラフの上で配線経路を見つける．

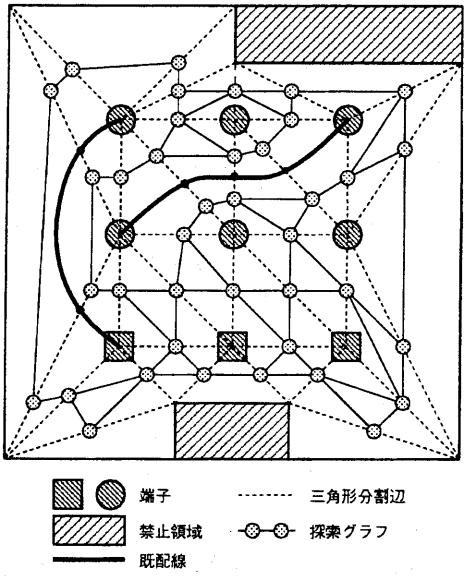

図 2: 探索グラフ

3 設計規則の検証

3.1 容量制約とクリティカルカット

現実のプリント基板の配線では種々の設計規則を満たさなければならないが，本稿では設計規則として配線間，配線と障害物の間に最低限確保すべき間隔だけを考えている．また，配線方法は 90 度，45 度配線ではなく自由角度の配線を前提としている．したがって，本稿の説明の中での距離はユークリッド距離を用いている．しかし，本手法は距離の定義に依存したものではないので，90 度，45 度配線にも適用可能である．

さて，設計規則により基板上の配線領域には通すことのできる配線数に上限ができる．これを**容量制約**と呼んでいる．従って，探索グラフ上で容量制約を満たした最短経路を見つけることが配線経路探索アルゴリズムの目的である．位相配線の場合には配線の物理的位置を決めることはしないので，直接隣合う配線との間隔を測ることはできない．そこで，先述の Leiserson らの結果によりすべてのクリティカルカットを検証することで，設計規則を満たしていることを確認する．

ここで，用語の定義をしておくと，

カット 2 つの対象物 (端子，禁止領域など) 内の互いに可視な 2 点間を結ぶ線分

容量 カットを通過可能な最大の配線幅 (配線および配線間の間隙も含む)

クリティカルカット 2 つの対象物間のカットで容量が最小のカット

以降では，2 点 p, q を結ぶカット pq の容量を $Cap(pq)$ で，カットを通過する配線の総配線幅を $Flow(pq)$ で表現することにする．

配線可能性検証の基礎となる Leiserson らの結果とは，「すべてのクリティカルカット c において，$Cap(c) \geq Flow(c)$ であれば，設計規則を満たした物理配線が可能である」というものである．

3.2 設計規則の検証方法

設計規則の検証にはすべてのクリティカルカットの容量とそれを通過する総配線幅を知ればよいことがわかった．容量はクリティカルカットの長さであるので部品は位置が決まれば一定である．経路表現には2.2節で述べた三角形分割を基礎としたデータ構造を用いているので，このデータ構造から通過する総配線幅を求める方法を説明する．

まず，クリティカルカットと三角形分割辺との関係を考えると，クリティカルカットが，

1. 三角形分割辺と一致する場合

2. 三角形の内部を通過する場合

の2つの場合に分かれる．

(1) の場合には，三角形分割辺上に交点をもつ既配線が交差している既配線のすべてであるので総配線幅は容易に計算できる．

(2) の場合は，図3示すように両端点が三角形の頂点の場合と，片方の端点が障害物への垂線になっている場合の2通りがあるが，いずれにしろそのクリティカルカットが通過している三角形の和領域 (以下では**クリティカルカットの近傍**と呼ぶことにする) を考える．配線は三角形の頂点以外から出てくることはないので，クリティカルカットを通過する配線はこの近傍の境界に必ず現れる．

ただし，境界に現れる配線がすべてクリティカルカットを通過するわけではないので，総配線幅を知るために個々の配線の交差判定をする必要がある．配線とクリティカルカットの交差判定は，クリティカルカットの近傍内での配線の両端が境界のどこに現れるかで判定できる．クリティカルカットに対して同じ側に両端があれば交差せず，半対側に両端があれば交差する．

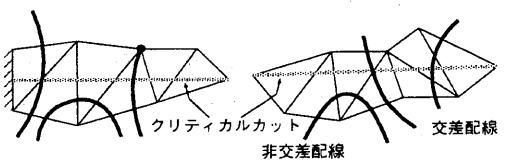

図 3: 三角形分割とクリティカルカット

3.3 逐次的検証のためのデータ構造

位相配線終了後に一括して設計規則の検証をするのではなく，探索中の部分経路が容量違反を起こせば，即座にそれを同定して以降の探索を打ち切る仕組みが配線可能性検証を経路探索アルゴリズムに組み込むためには必要である．経路探索は，探索グラフ上で部分経路を成長させながら行なうので，探索グラフ上の部分経路からそれが交差するクリティカルカットを同定できればよい．

探索グラフ上の部分経路が枝を一つ伸ばすたびにすべてのクリティカルカットとの交差を調べるわけにはいかない．探索グラフ上から直接参照できないクリティカルカットは，前節の (2) のクリティカルカットである．そのために，クリティカルカットの近傍の境界上にある三角形の辺，および頂点に以下のデータを経路探索の前処理として登録する．

- 対象としているクリティカルカット
- 辺，頂点がクリティカルカットに対して右左どちらの境界に位置するか
- 辺，頂点からクリティカルカットの近傍に進入する場合の経路の向き

以上の3つのデータの登録により，探索グラフ上の部分経路の先頭の節に対応する三角形分割辺を参照することで，部分経路がクリティカルカットの近傍へ進入，退出したことを知ることができる．また，クリティカルカットに対して左右どちらから進入してどちらへ退出したかも知ることができるので，近傍からの退出と同時に交差判定を行ない交差するクリティカルカットを同定できる．

4 配線経路探索アルゴリズム

3.1節で述べたように経路の配線可能性を正確に検証するには，経路が通過するクリティカルカットの容量を検証すれば十分である．一方，最短経路探索では，ダイクストラ法を使うのが高速であり一般的である．しかし，ダイクストラ法の中にこの検証の処理を埋め込むことは出来ない．というのは，ダイクストラ法ではグラフ上の1つの枝を進む (本方式の場合は，三角形の辺から隣接する次の辺へ進む事) 毎に始点からその点までの最短経路を更新していくので，1つの枝を進む中で容量超過の判定 (超過の場

合は枝の距離を無限大とする) をしなくてはならないからである.

しかし, クリティカルカットとの交差判定はクリティカルカットの近傍へ経路が進入し, 完全に退出して始めて可能であるので, 図3からも明らかなように複数の枝, すなわち部分経路が対象となる. 同様に図4に示すように配線経路が同一のクリティカルカットを複数回通過する時に起こる容量超過も部分経路を対象としなければ容量超過を判定できない.

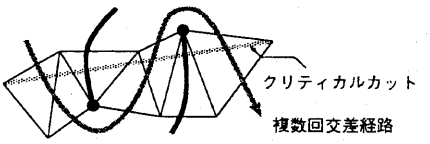

図 4: クリティカルカットとの複数回の交差

このような理由により, 本稿の方式では経路探索アルゴリズムに IDA^*(Iterative Deepning A^*) を用いている. Iterative Deepning とは横型探索を縦型探索でシミュレートする手法であり, 探索の深さを適当に与えてその深さを超えない範囲を縦型で全数探索していく. 深さの増分を適当に制御しながら探索範囲を順次広げることで, 少ないメモリで横型探索を可能にする. IDA^*は, この原理で A^*アルゴリズムを実現したものである. 基本的には縦型探索であるので, 始点から先頭までの経路を保持しており, その経路にかかわる容量制約を検証する事が可能である.

縦型探索で一つ探索を進める毎に 3.3節で述べた方法で, 交差するクリティカルカットがあればそれを同定し容量制約の検証を行なう. また, 同じクリティカルカットに複数回交差する場合のために, 交差したクリティカルカットには配線幅の増加分を加えておく.

IDA^*が解に到達するまでに探索する探索木の範囲は, 最良優先探索の評価値として始点からの距離と終点までの距離の予測値の和を用いる A^*アルゴリズムと同じである. この予測値が実際の残りの距離よりも過小に評価されている時, A^*アルゴリズムでは最短経路が得られる事が保証されている. また過小評価でも実際の距離との差が小さければ小さい程, 探索範囲は少なくて済み, 高速に解が得られる. 経路探索では一般に終点までのユークリッド距離を用いることが多いが, 障害物の配置によっては実際の距離との差が大きくなり, 探索範囲が広がってしまい解に到達するのが遅くなる. したがって, どのような予測値を与えるかが処理速度を左右することになる.

本稿の方式では前処理として, ダイクストラ法を終点から適用することで終点から各点までの最短距離を求め, 残りの距離の予測値としている. ユークリッド距離を用いるよりは予測値としてはるかに実際の距離との誤差が少い. この前処理のための計算コストは, IDA^*の計算コストが予測値の誤差に対して指数的に増えて行くことを考えると問題とならない.

探索の深さの制御も処理速度を左右する重要な要因であるが, 本稿の方式では, 始点から終点までの最短距離を初期値とし, 経路探索の対象であるネットの最大配線長まで順次増やしていく. 残念ながら深さの増分をどのように制御すれば最適であるかに関して一般則はない. 配線経路の探索では, 経験的により長い配線の方が容量違反を起こしやすいということから, 初期値のあたりで深さを細かく増加させ, それ以上では増分を大きくするという制御を行なっている.

5 高速化の手法

4節に示したアルゴリズムは逐次的に設計規則を検証しながら配線経路を探索する基本アルゴリズムであり計算量は $O(\exp(n))$ である. このままでは一般に配線経路探索に使用されるダイクストラの最短経路アルゴリズムの計算量 $O(m+n\log n)$ に比べて非常に遅い場合がある. 最悪の場合を改善することは難しそうであるが, 平均的に実用的な処理速度を得るために基本アルゴリズムに2種類の改善を施して自動配線システムの内部で使用している.

5.1 検証するクリティカルカットの削減

クリティカルカットの数は端子数 n に対して $O(n^2)$ である. 配線可能性検証の処理を軽減するために, 検証するクリティカルカットの数は出来るだけ少なくしたい. あるクリティカルカット c について, 配線経路探索前にすでに確定している既配線の総配線幅を $Flow_{old}(c)$ とし探索中の経路による増加分を

$Flow_{new}(c)$ とすれば，

$$Flow_{old}(c) + Flow_{new}(c) > Cap(c)$$

となる可能性のあるものだけを対象とすればよい．この観点からクリティカルカットを次の3つに分類する．

1. すべての位相配線に対して，常に，
   $Flow(c) \leq Cap(c)$
2. すべての経路に対して，
   $Flow_{old}(c) + Flow_{new}(c) \leq Cap(c)$
3. ある経路に対して，
   $Flow_{old}(c) + Flow_{new}(c) > Cap(c)$

(1) は，位相配線の結果がどのようになっても絶対に容量違反を起こさないクリティカルカットであり，このようなカットは生成時に同定し削除するようにしている．同定の詳しい方法については，[9] で報告している．

(2) と (3) は，その時点での既配線の状態を前提とした分類で，(2) は対象としているネットがどのような経路を採っても容量違反を起こさない場合である．(3) は，経路の採り方によっては，容量違反を起こす場合で，このタイプのクリティカルカットだけを検証すれば十分である．

$Flow_{Old}$既知であるから，$Flow_{new}(c)$ の上限を見積りたい．クリティカルカット c に N 本の既配線が交差しているとすると，これから探索を始める経路は既配線の間を通るので，高々 N+1 回しかクリティカルカット c とは交差しない．すなわち，上限は

$$Flow_{new}(c) \leq (\text{配線 1 本の幅}) \times (\mathrm{N}+1)$$

である．また，分類に変更が起きるクリティカルカットは，交差する既配線の数に変更があったものだけなので，確定した配線の追加と既配線の削除の後に更新する．探索グラフの連結性を考慮すれば，N 回以下の交差に押えられる場合も出てくるので，よりよい上限値が得られるが，経路探索の対象ネット毎に上限が変わって来るので，処理の簡単のためにこの上限値を用いることにした．

5.2 ダイクストラ法との組合せ

4節でダイクストラの最短経路アルゴリズムが使えない理由を述べたが，実際の配線問題ではダイクストラ法で三角形分割の1つの三角形の内部あるいは辺上にあるクリティカルカットだけを検証しながら経路を求めても結果的には設計規則を満たしている場合が多い [6]．特に既配線の密度が高い位相配線の終盤を除けば，ダイクストラ法で求めた経路が実際に容量違反を起こしていることはほとんど無い．また，Constrained Delaynay Triangulation により三角形分割を行なうと検証の効果の高いクリティカルカットが三角形の辺に一致する可能性が高くなる．

そこで，まず高速なダイクストラ法によって，三角形の内部または，辺上のクリティカルカットだけを検証した経路を見つけ，その後見つかった経路全体の設計規則を検証する．ここで容量違反を起こしていた場合にだけ，本稿での配線経路探索アルゴリズムにより経路を求める．容量違反を起こした場合には2度手間になるが，その頻度はかなり低いので，実用的にはこの組合せの方が速い．

6 処理速度に関する議論

いくつかの高速化の手法を採り入れて実用的には満足のいく処理速度を得ることができたが，やはりすべての経路が容量違反を起こしてしまうような場合には，全数探索となり非常に探索時間がかかる．現状では探索時間に上限を設定して探索を打ち切り，経路はなかったものと判断するようにしている．

処理が遅くなっている第一の原因は，容量制約の検証が探索グラフの1つの枝の中で収まらず，部分経路に対して検証を行なわなければならないことである．同じ全数探索でもダイクストラ法と IDA^* では極端に処理速度が違ってくる．IDA^*による探索を観察すると，どのクリティカルカットを通過したかに関わらず到達可能な節へのすべての経路を探索しており非常に無駄が多いことがわかった．

部分経路に対する容量制約が存在する時でもダイクストラ法が使えればかなりの高速化が期待できる．同じような問題をダイクストラ法の拡張で対処しているものに最短の K 個の経路をみつける K shortest paths problem の1つの解法がある [1]．解法の概略は，まず最短経路を見つけ，2番目に短い経路を見つける時にはグラフを拡張して最短経路と全く同じ経路がグラフ上のどの経路に現れないようにする．この拡張したグラフ上でダイクストラ法により最短経路を見つければ，それが2番目に短い経路となる．以下，3番目，4番目と同様のグラフの拡張を行な

いながら毎回ダイクストラ法で最短経路を見つけていく．

配線経路探索の場合には，探索中の経路が探索グラフのある節に到達した時，交差したクリティカルカットの集合が同じで，より短い経路がすでにみつかっていれば，具体的な経路に違いに関わらず，ダイクストラ法と同様に長い方の経路の探索はここで打ち切っても構わない．また，交差したクリティカルカットの集合が異なる場合には仮想的に探索グラフを拡張し異なる節として処理する．現在このような原理に基づいた配線経路探索アルゴリズムの改訂版を開発中である．

しかし，このアルゴリズムの処理速度は交差したクリティカルカットの集合のバリエーションの数に依存する．現実の基板データに適用した時，交差したクリティカルカットの集合にどれくらいのバリエーションが出てくるのか予測できていない．最悪の場合にはバリエーションの種類はクリティカルカットの数に対して指数的に増加するので，現在の IDA^* を凌ぐことができるかどうかは実験を待たなければならない．

7 まとめ

本稿では，自動配線システムの位相配線で使用する配線経路探索アルゴリズムについて述べた．経路探索の中に配線可能性検証の処理を組み込んだ割には実用的に高速な経路探索アルゴリズムが得られた．ただ，問題として，最悪の場合の処理速度の改善が残った．この点は，6節でふれた改訂版の結果に期待したい．

また，現在は一層基板のみを対象としたシステムであることから，比較的単純な構造の探索グラフを対象として配線経路探索アルゴリズムを構成できた．しかし，今後，位相配線の利点を活かしたままで両面，多層基板へと展開するにあたって，同様の考え方で配線可能性検証を組み込んだ配線経路探索が可能であるかも検討を要する課題である．

謝辞

一層自動配線システムを開発するにあたり，サンプルデータの提供および開発中システムに対する貴重な助言をいただいた(株)図研の菊地課長，(株)横河ディジタルコンピュータの豊田部長，井上課長に感謝致します．

参考文献

[1] J. A. Azevedo, M. E. O. S. Costa, J. J. ao E. R. Slivestre Madeira, and E. Q. V. Martins. An algorithm for the ranking of shortest paths. *European Jounal of Operational Research*, 69:97–106, 1993.

[2] W. W.-M. Dai, R. Kong, and M. Sato. Routability of a rubber-band sketch. In *Proceedings of the 28th ACM/IEEE Design Automation Conference*, pp. 45–48, 1991.

[3] C. E. Leiserson and F. M. Maley. Algorithms for routing and testing routability of planar VLSI layouts. In *Proceedings of the 17th Annual ACM Symposium on Theory of Computing*, pp. 69–78. ACM, 1985.

[4] Y. Lu and W. Dai. A numerical stable algorithm for constructing constrained delaunay triangulation and application to multichip module layout. In *Proceedings of 1991 International Conference on Circuits and Systems*, pp. 644–647, June 1991.

[5] F. M. Maley. Testing homotopic routability under polygonal wiring rules. *Algorithmica*, 15:1–16, 1996.

[6] D. Staepelaere, J. Jue, T. Dayan, and W. W.-M. Dai. Surf: A rubber-band routing system for multichip modules. *IEEE Design & Test of Computers*, 10(4):18–26, 1993.

[7] 田中, 佐藤, 大附. スケッチレイアウトシステムにおける配線可能性検証. 信学技報 VLD 94-101, pp. 33–40, 1995.

[8] 田中, 金沢, 田中, 佐藤, 大附. スケッチ表現に基づく多層配線システム. 情報処理学会研究報告 DA 70-9, pp. 63–70, 1994.

[9] 濱, 江藤. 経路探索高速化のためのクリティカルカット削減方法. 第52回全国大会講演論文集(分冊6), pp. 37–38. 情報処理学会, 1996.