障害となる既配線の押し退け可能性を考慮した自動配線手法

3C－1

林　中也　奥田　亮輔　中尾　博臣　寺井　正幸
三菱電機（株）　システムＬＳＩ開発研究所

1　はじめに

ゼネラルエリア配線（general area routing）では、各ネットを配線する際、既に引かれている配線が邪魔になって配線できないことが起こる。この時、既配線と短絡を起こす経路をまず見つけ、後に（直後とは限らないが）経路上の既配線を移動させる手法がよく用いられる。これらの手法は大きく２つに分けられる。１つは、配線経路を見つけるときに既配線との短絡を"touch"と"cross"に区別するもの（[1], [2]）、他の１つは区別しないものである（[3], [4]）。

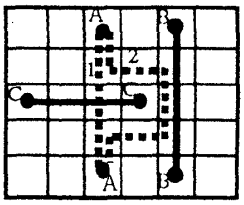


図１　touchを持つ経路とcrossを持つ経路

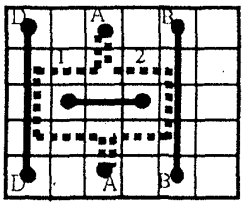


図２　押し退け可のtouchを持つ経路と押し退け不可のtouchを持つ経路

図１は１層の配線問題の例で、細線で区切られた小さなマスが１つの配線グリッドを表す。今、ネットB,Cの既配線が存在し、さらに、ネットAを配線したいとする。破線１の経路はネットCの配線とcrossを起こす経路、２の経路はネットBの配線とtouchを起こす経路である。２の経路を見つければ、ネットBの配線を右側に押し退けるだけで、ネットAを見つけた経路で配線することができる。一般に、touchを起こす経路を見つけた場合、邪魔になっている既配線を押し退けるだけで配線できる可能性が高いが、crossを起こす経路を見つけた場合には、その経路で配線しようとすると、既配線を大幅に移動させなければならず、全ネットを配線できる可能性が低い（既配線を他の層に移動させることも考えられるが、その可能性は層の数が限られていることから低い）。よって、touchとcrossを区別し、crossよりtouchを起こす経路を優先的に見つける方が正しい経路を見つけられる可能性が高いと言える。

しかし、図２の問題の場合、touchとcrossを区別しtouchを優先しただけでは（一度の経路探索では）１の経路か２の経路のどちらを見つけるかは分らない。そこで、著者らは押し退け可のtouch（押し退けられる可能性の高い既配線とのtouch）、押し退け不可のtouch（押し退けられる可能性の低い既配線とのtouch）という概念を導入した。図２の１の経路は押し退け不可のtouch、２の経路は押し退け可のtouchを起こす経路である。押し退け不可のtouchより押し退け可のtouchを起こす経路を優先的に見つけることにより、従来何度も経路探索を（例えば、一度見つけた経路を見つけにくくして）繰り返して初めて正しい経路を見つけることができた問題に対して、経路探索の回数を減らすことができ、計算時間を短縮することができる。また、実用的な時間では正しい経路を見つけられず、解けなかった問題に対しても解くことができると期待できる。

2　配線問題の定義

本論文で扱う配線領域は、グリッドベースであることだけが仮定され、配線層の数、配線領域の形状、端子の位置に制限のない、ゼネラルエリア（general area）である。また、配線領域内に任意の形の障害物があってもよい。ここで、解くべき配線問題は、端子の集合が与えられ各端子にネット番号が対応付けられている時、同一ネット番号に対応付けられているすべての端子を（短絡なしに）繋ぐことである。

3　配線アルゴリズム

既配線とcrossを起こす経路は３次元的にみれば、既配線の上を越えるか下に潜るかで２通りに分かれるにせよ、touchを起こす経路とみることができる。そこで、今までtouchと呼んでいたものをxy-touch、crossをz-touch、これらを共にtouchと呼ぶことにする（xy-touchとz-touchは後述の"経路のコスト"の定義で区別する）。

本論文で提案する手法は、１ピンペア（端子のペア）ずつ配線する逐次配線手法であり、各ピンペアの配線を終えたときには、短絡は存在しないようにする。どのピンペアから結ぶかは、全ピンペアについて最短経路長を計算し、小さいものから結んでいく。ただし、本論文で言う経路長とは、経路の長さそのものではなく、経路の長さに配線層別、方向（縦横）別、ビアの種類別の重みを掛けた値である。また、最短経路とは配線領域内の障害物を避ける全ての経路のうち、最短のものを表す。

１ピンペアを結ぶためのアルゴリズムを、簡単のため１層配線の場合で説明する（図３）。ここで、

（経路のコスト）＝（経路長）＋Ct×（押しのけ可のxy-touchの回数）＋Cut×（押しのけ不可のxy-touchの回数）＋Cc×（押しのけ可のz-touchの回数）＋Cuc×（押しのけ不可のz-touchの回数）

と定義しておく（回数はtouchを起こすグリッドの数）。Ct, Cut, Cc, Cucは正の定数で、Ct≪Cc, Cut≪Cuc, Ct＜Cut, Cc＜Cucである。

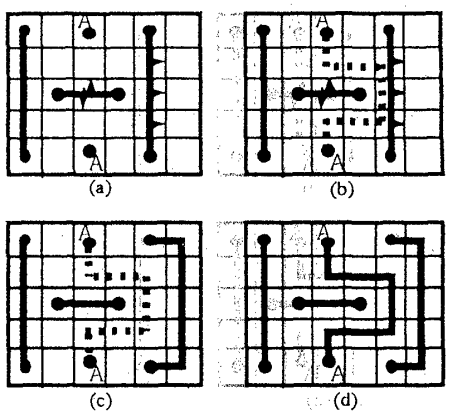


図３　本手法による配線の例

まず、既配線の押しのけ可能性を調べる（図3a、調べ方は後述）。ここでは、既配線を押し退けて空けることのできるグリッドに

A general area router considering the possibility of shoving aside
Chuya Hayashi, Ryosuke Okuda, Hiroomi Nakao, Masayuki Terai
Mitsubishi Electric Corporation
4-1, Mizuhara Itami Hyogo, 664 Japan

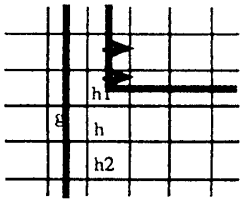
押し退け可能な方向に矢印を付ける。次に迷路法を用いてコスト最小の経路を見つける（図3b）。上で調べた押しのけ可能性を基に、押しのけ不可のtouchより押しのけ可のtouchを持つ経路を優先的に見つけることができる。次に、経路上の既配線を押し退ける（図3c）。ここでは、経路上の、既配線の存在するグリッドについて1グリッドずつ空けていく。最後に実際に線を引く（図3d）。

［1ピンペアを結ぶ手順］
（全ピンペアを結ぶためには、上記の方法で全ピンペアに順序をつけ、1ピンペアずつ結ぶ）
step 1:結線すべきピンペアをスタックに登録。
step 2:既配線の存在する全グリッドに対し、6方向について既配線の押し退け可能性を調べる。
step 3:スタックから最後に登録されたピンペアを取り出し、そのペアに対しコスト最小の経路を見つける。
step 4:見つけた経路上に既配線がなければstep 8へ。
step 5:見つけた経路上の既配線の押し退けを試みる。
step 6:経路上の既配線をすべて押し退けられればstep 8へ。
step 7:経路上の既配線を引き剥す。これにより新たに結線する必要が生じたピンペアをスタックに登録。
step 8:見つけた経路で実際に配線を引く。
step 9:スタックに登録されているピンペアがなければ、成功で終了。
step 10: step 2へ。ただし、step 2～10のループの回数が上限を越えれば、失敗で終了。

本手法では、引き剥し再配線を繰り返すとき、無限ループを避けるために、経路のコストに次のペナルティp(i,j,d)を加算する。p(i,j,d)はネットiの既配線にネットjが方向d（6方向のいずれか）からtouchを起こす時の（touch 1回分の）ペナルティで、最初は0であり、その様なtouchを起こす経路を見つける毎に増加していく。

押しのけ可能性の調べ方

右向きの矢印を付ける場合についてのみ説明するが、6方向（“右”、“左”、“上”、“下”、“上の層への方向”、“下の層への方向”）について同様に行う。



1）でhを見る
2）でh1,h2を見る
この場合はgに→が付く

図4　押し退け可能性を調べる例

矢印は最も右のグリッドから付けていく（最も右のグリッドは複数個あるが、その順序はどうでもよい）。既配線の存在する各グリッド（gとする）に右向きの矢印を付けるのは次の2条件を共に満たすときである。ここで、グリッドgの一つ右のグリッドをhとする。

1）hに、配線が存在しないかg上の配線と同じネットに属する配線が存在するかまたは右向きの矢印が付いている。

2）右と垂直な方向（上下、上の層、下の層）でgから配線が出ている全ての方向（例えば、gから上下に配線が出ているときは上と下）について、その方向へhから1グリッド離れたグリッドに、配線が存在しないかg上の配線と同じネットに属する配線が存在するかまたは右向きの矢印が付いている。

4　実験結果

本手法をEWS（Sparc Station 2）上で実行した結果を以下に示す。当社ゲートアレイ用セル（300 Tr以下の回路）の内から配線の難しいと思われるセル5セルを選び、押し退け可能性を調べた場合と調べない場合（Ct＝Cut, Cc＝Cucと設定）で、配線に要するCPU時間を測った（表1）。配線は2層で行った。押し退け可能性を調べることにより、5セルの平均で1.87倍高速に配線できた。配線結果の例を図5に示す。ただし、この図は本手法で通常の配線をした後、経路のコストの定義を変え、（本手法で）第2層配線、迂回配線の削減を行った後の結果である。

表1　押し退け可能性を調べる効果

セル	ネット数	ピン数	配線に要するCPU時間（秒） ①押し退け可能性を調べた場合	②押し退け可能性を調べない場合	比率 ②／①
A	72	388	1871	2275	1.22
B	64	268	1629	2760	1.69
C	72	372	1217	1380	1.13
D	44	216	590	1667	2.83
E	36	192	283	702	2.48

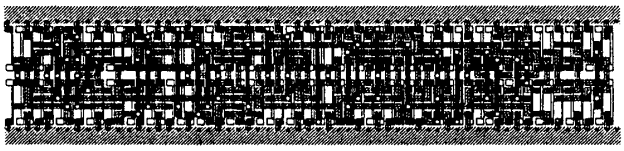


図5　配線結果の例

Burstein's difficult switch box（以後、burstと言う）に適用した結果を表2に示す。comp-burstとはburstを圧縮して、配線領域を小さくした問題である。comp-burst (20x14)（著者の知る限り、今まで解けたという報告はない）を解くのに要したCPU時間は7.2秒であった。

表2　Burstein's difficult switch boxへの本手法及び他の手法の適用結果（○は配線成功、failは配線失敗、？は不明を示す）

	area	touch and cross[1]	Codar[3]	Packer[4]	Mighty[5]	本手法
burst	22×15	○	○	○	○	○
comp-burst	22×14	?	○	○	fail	○
	21×14	?	○	?	fail	○
	20×14	?	?	?	?	○

5　まとめ

既配線の押しのけ可能性を調べ、その結果を基に配線経路を見つける手法を開発した。実験結果から、本手法によって高速に配線できることが分かった。また、今まで実用的な時間では解けなかった問題を解くことも可能であると期待できる。

［参考文献］
[1] K. Kawamura et al., "Touch and Cross Router," Proc. ICCAD, 1990, pp. 56-59.
[2] H. Bollinger, "A Mature DA System for PC Layout," Proceedings International Printed Circuits Conference, 1979, pp. 85-99.
[3] P. S. Tzeng et al., "Codar: A Congestion-Directed General Area Router," Proc. ICCAD, 1988, pp. 30-33.
[4] S. H. Gerez et al., "Switch box routing by stepwise reshaping. IEEE trans. on CAD, Vol. 8, No. 12, 1989, pp. 1350-1361.
[5] H. Shin et al., "A Detailed Router ... : Mighty," IEEE trans. on CAD, Vol. CAD-6, No. 6, 1987, pp. 942-955.