配置配線パズルにおける求解アルゴリズム

甘利 拓斗[1,a)]　蓮見 平八郎[1,b)]　藤吉 邦洋[1,c)]

概要：DA シンポジウム 2019 で開催されるアルゴリズムデザインコンテストの題材は配置配線パズルである．この配置配線パズルは集積回路の自動配置配線と親和性が高く，解法に集積回路の自動配置配線アルゴリズムを応用できる．そこで配置しつつ配線を行う階層的配置配線手法を参考にすると共に，既存の平面位相配線手法を拡張して無駄な探索を減らし，解を求める手法を考案した．本稿ではこの手法の中で平面位相配線手法の拡張を中心に述べる．

キーワード：配置配線パズル，階層的配置配線，位相配線

An Algorithm for Placement and Routing Puzzle

Takuto AMARI [1,a)]　Heihachirou HASUMI[1,b)]　Kunihiro FUJIYOSHI[1,c)]

Abstract: The subject of “algorithm design contest” at DA Symposium 2019 is “placement and routing puzzle”. Since automatic placement and routing algorithms of integrated circuits can be applied to slove the puzzle. ,we apply an existing hierarchical placement and routing method and a planar topological routing method. In this paper, we focus on the expansion of planar topological routing.

Keywords: placement and routing puzzle,hieralchical placement and routing,topological routing

1. はじめに

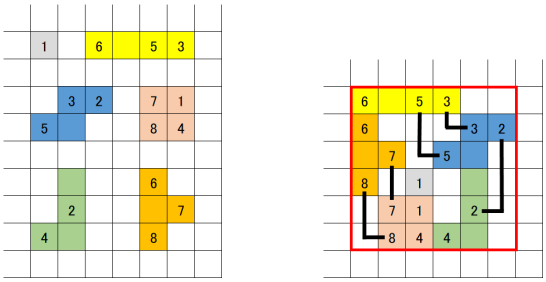
DA シンポジウム 2019 で開催されるアルゴリズムデザインコンテストの題材は配置配線パズルである．配置配線パズルとはナンバーリンクを発展させたパズルであり，盤面にブロックを配置し，ブロック上に記入された同じ数字同士を線で結び，全ての数字が線で結ばれて制約違反を起こしていない状態を解とする．**図 1** に配置配線パズルの問題例とその解答例を示す．

配置配線パズルは，集積回路の自動配置配線と親和性が高く，解法に集積回路の自動配置配線アルゴリズムを応用できる．自動配置配線アルゴリズムでは，配置と配線を分けて行うものが一般的である．しかし配置と配線を分けると，配置と配線の相互依存性により，満足のいく解を得るためには配置と配線の繰り返しが多数回必要になる．

そこで配置しつつ配線を行うことで配置と配線を繰り返す回数を少なくして解く階層的配置配線手法 [1][2] が提案された．階層的配置配線手法では，配置配線領域を分割し，分割して得た領域毎に配置と概略配線を行っていく．

ところで，配置配線パズルで与えられるブロックは，配線を出す方向によっては配線不可能になってしまう．この問題に対しては，平面位相配線の考え [3][4] を用いて平面配線可能性を失わない配線の順序を得ることで，無駄な探索を減らすことができる．

そこで本稿では，配置配線パズルを解くため階層的配置配線手法を参考にし平面位相配線の考えを用いて無駄な探索を省く手法を考案し，平面位相配線手法の拡張を中心に説明する．



(a) 問題例　(b) 解答例

図 1　配置配線パズルの問題例とその解答例

2. 配置配線パズルの詳細 [5]

配置配線パズルとは，ナンバーリンクに配置問題の要素を加えた新たなパズルである．ナンバーリンクとは盤面に

[1] 東京農工大学
Tokyo University of Agriculture and Technology
[a)] amari@fjlab.ei.tuat.ac.jp
[b)] hasumi@fjlab.ei.tuat.ac.jp
[c)] fujiyosi@cc.tuat.ac.jp

ある同じ数字同士を線でつなぐペンシルパズルであり，株式会社ニコリの登録商標である [6].

配置配線パズルは，数字が書かれた複数のブロックが用意されており，これらを格子状の 1 つの盤面に配置し，ナンバーリンク同様盤面にある同じ数字の間を線で接続するパズルである．盤面のサイズは問題ごとに与えられ，最大サイズは 72×72 である．格子の 1 区画をマスと呼ぶ．問題例とその解答例を図 1 に示す．配置配線パズルのルールを以下に示す．

(1) 問題で指定された全てのブロックを盤面上に配置する．
  (a) ブロックの形状は 1 マスのモノミノ，および 4 マスのテトロミノ 5 種のうちいずれかである．
  (b) ブロックは回転，反転させてはならない．上下左右の平行移動のみ可能である．
  (c) ブロック同士は重なって配置してはならない．
(2) 盤面上に配置されたブロック上にある同じ数字の間を交差・分岐の無い配線で接続する．（以下ナンバーリンクと同じ）
  (a) 配線の端点が存在するマスは，X, Y 方向に隣接する最大 4 個のマスのうち 1 個と接続される．
  (b) 配線は 1 マスに 1 本のみ引くことができ，交差・分岐してはいけない．
  (c) 配線を構成する端点以外のマスは，X, Y 方向に隣接する最大 4 個のマスのうち 2 個と接続される．
(3) 数字が書かれていないブロック内の領域には，線を引けない．
(4) 解の品質は「全てのブロックと配線を囲む最小の矩形面積」で評価する．

ナンバーリンクは，集積回路の自動配線と親和性が高いパズルである．ナンバーリンクでは数字マスの位置は固定されているが，配置配線パズルでは数字は盤面上を移動可能なブロック上に配置される．従って，数字間の配線とブロックの位置を同時に考慮しつつ，できるだけ小さい面積で配置配線を行うことが目標であり，集積回路の自動配置配線と親和性が高いパズルであるといえる．

3. 関連手法

3.1 階層的配置配線手法 [1][2]

カルフォルニア大学で開発された階層的配置配線手法を用いたレイアウトシステム BEAR は，できるだけ小さい面積内に配置配線をするシステムである．

BEAR で用いられてる階層的配置配線手法では，まずクラスタ成長法という手法を取り入れている．この手法ではいくつかのブロックを選択し，これと接続の強いブロックを次々にボトムアップ的にグループ化していく．この結果全体が 1 個に統合され，クラスタリング木と呼ばれる木が出来上がる．このクラスタリング木の葉はブロックとなる．

次に，できたクラスタリング木の節点を根からトップダウン的に参照して，クラスタ（部分木）の持つブロックの総面積に応じて配置配線領域の分割を繰り返す．

末端のクラスタまで配置配線領域の分割が終わったら，末端の階層のクラスタから分割された領域内でボトムアップ的に配置と概略配線を行う．同じ階層のクラスタの配置と概略配線が全て終わったら，次にもう一段上の階層のクラスタで配置と概略配線を繰り返す．最終的に全体の配置を得て，詳細配線を行う．

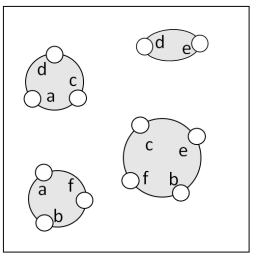


図 2 外部配線問題 (島配線)

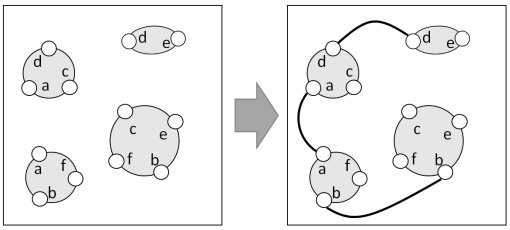


図 3 全ての島を木形状に結んで 1 つにする Tree-connection

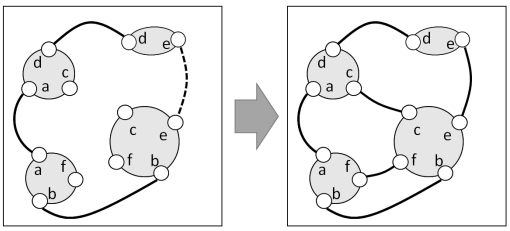


図 4 全ての配線を結ぶ Chord-connection

3.2 CHORD-LAST 法 [3][4]

平面位相配線問題は，各ネットについて全ての端子間を配線同士が交差せず結ぶという平面配線問題において，配線の幅を考慮せずに配線可能性を判定し，可能であれば配線するという問題である．CHORD-LAST 法という配線アルゴリズムを用いることで，与えられた平面位相配線問題を解くことができる．平面位相配線問題は，平面配線問題の中で外部配線問題（**図 2** 参照）の一種である．

CHORD-LAST 法は Tree-connection と Chord-connection の二つで構成されている．Tree-connection では，異なる島に属する端子対（同ネット）がある限り端子対同士を結び，結ばれた島々を 1 つの島と見なす．例を**図 3** に示す．Chord-connection では，島の周囲で隣接する端子対（同ネット）がある限り，隣接する端子対を選んで結びこれらが周囲上にないものとみなしていく．例を**図 4** に示す．

Tree-connection を終えた後，端子をもたない島を除いて，元の島々は少なくとも 1 本の配線で他の島々と木形状で結ばれる．このとき異なる島に属する同じネットの端子対は存在しない．ここで島それぞれにおいて，端子を 1 つ選んで島の外周に沿って時計回りに 1 周する．自身に帰ってくるまでに通過した端子のネット番号を順に並べて，外周端子列を構成する．得られた外周端子列において，先頭と末尾は連続しているものとする．外周端子列の中で同じネット番号の端子が連続している場合は，これら 2 つの端子は，他の配線と交差することなく配線できるため，これらを配線し外周端子列から取り除く．同じネット番号の端子が連続しなくなるまで，この操作を繰り返す．最後に空の列が得られたなら，与えられた配線問題は平面配線可能

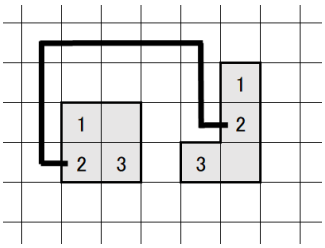


図 5 配線を出す方向によって配線が不可能になってしまったブロックの例

である．

4. 提案手法

配置配線パズルで与えられるブロックの種類は図 1(a) に示したモノミノと 5 種のテトロミノからなる．ここで図 1(a) でモノミノから順に，それぞれのテトロミノを時計回りに見て I 型，O 型，T 型，J 型，S 型と呼ぶこととする．これらが回転したものや鏡像反転したものも同名で呼ぶ．

本稿では 3.1 で紹介した階層的配置配線手法を参考に，接続の強いブロック同士を合体して 1 つのブロックとし，このブロック上にある同一のネット番号の端子を結ぶ．これを繰り返して最終的に 1 つのブロックを得て全ての配線を行うことで配置配線パズルを解く手法を考案した．この手法に加え，無駄な探索を省くために CHORD-LAST 法を拡張する手法を提案する．以下では，CHORD-LAST 法の拡張について述べる．

4.1 CHORD-LAST 法の拡張

CHORD-LAST 法を配置配線パズルに適用するため，各ブロックを 1 つの島と見なし，各島にブロック上の端子を配置することで，配置配線パズルを外部配線（島配線）問題に置き換える．

各島の外周端子列はブロックの外周に沿って時計回りに一周した際に，ブロック上の端子から配線の出る順番に応じて得られる．ここで I 型，J 型のテトロミノは内側の端子が上下もしくは左右方向のみに配線を出すことが可能であるため，外周端子列の順番が一意に定まらない場合がある．そこで I 型，J 型のテトロミノにおいて外周端子列の順番が一意に定まらないものを外周未確定ブロックと呼ぶこととする．

この外周未確定ブロックが配線を出す方向によって，配線が不可能になってしまったブロックの例を**図 5** 示す．この例の場合，J 型のブロック上の 2 の端子は左右どちらからも配線を出すことができるが，左に出してしまったことで，1 か 3 のどちらかのネットだけしか配線できなくなってしまった．右に出していた場合は平面性を失わずに 1 と 3 のネットを両方とも配線できる．従ってこの例の場合平面配線可能性を失わない外周端子列の取り方，つまり，端子から配線の出る方向は一意に決まる．

無駄な探索を減らすために，事前に外周未確定ブロックは配線を出す方向を決定する．ここで平面配線可能性を失わないように配線を出す方向を選択するために，CHORD-LAST 法を拡張し利用する．

4.2 外周端子列の拡張

CHORD-LAST 法では，島の外周端子列は一意に決まっていることが前提であるため，CHORD-LAST 法を用いるためには外周未確定ブロックに仮の外周端子列を決める必要がある．

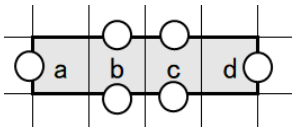


図 6 外周未確定ブロックとなる 4 つの端子を持った I 型ブロック

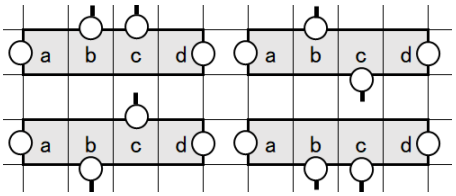
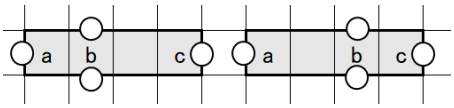
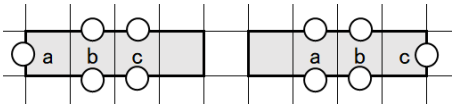


図 7 4 つの端子を持つ I 型ブロックにおける 4 通りの配線の出し方



(a) 他方向に配線を出せる端子を 1 つもつ 2 通りの I 型ブロック



(b) 他方向に配線を出せる端子を 2 つもつ 2 通りの I 型ブロック

図 8 外周未確定ブロックとなる 3 つの端子を持った 4 通りの I 型ブロック

本稿ではこの仮の外周端子列を拡張外周端子列と呼ぶこととし，外周未確定ブロックの配線を出す方向が上下もしくは左右のみの端子のネット番号に，+ もしくは − の重複識別子をつけて，同じネット番号を複数個拡張外周端子列に挿入することにする．− の重複識別子は既に + の重複識別子のついた同じネット番号が存在するときに使用する．

そこで，I 型，J 型のテトロミノがどのようなときに外周未確定ブロックとなり，重複識別子を用いてどのような拡張外周端子列を得るか考える．

まず I 型について考えると，**図 6** に示した 4 つの端子を持った I 型では，b と c の端子から配線を上下に出せるため，**図 7** に示すように $abcd$ と $abdc$ と $acdb$ と $adcb$ の 4 通りの外周端子列のいずれかとなるので外周未確定ブロックであり，外周上の b と c の端子に重複識別子をつけて 2 つずつ端子列に挿入することで，拡張外周端子列 $ab^+c^+dc^+b^+$ を得る．

同様に**図 8** に示す 4 通りの端子位置の I 型ブロックを考えると，他方向に配線を出せる端子の数により 2 種類に分けられる．

図 8(a) では，どちらも b の端子から配線を上下に出せるため，2 通りの外周端子列のどちらかとなるので外周未確定ブロックであり，重複識別子をつけて複数回端子列に挿入することで，拡張外周端子列 ab^+cb^+ を得る．

図 8(b) では，左のブロックは b と c の端子から配線を上下に出せるが，c の端子は配線を上から出しても下から出しても外周端子列に影響を及ぼさない．従って二通りの外周端子列のどちらかとなるので外周未確定ブロックである．よって b のみ重複識別子をつけて複数回端子列に挿入することで，拡張外周端子列 ab^+cb^+ を得る．同様に右のブロックも外周未確定ブロックであり，拡張外周端子列 ab^+cb^+ を得る．

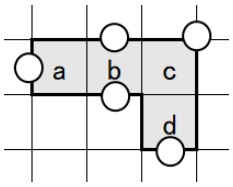
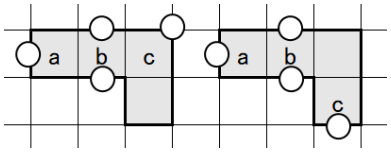
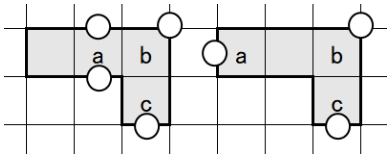


図 9 外周未確定ブロックとなる 4 つの端子を持つ J 型ブロック



(a) 外周未確定ブロックとなる 2 通りの J 型ブロック



(b) 外周未確定ブロックとならない 2 通りの J 型ブロック

図 10 3 つの端子を持つ 4 通りの I 型ブロック

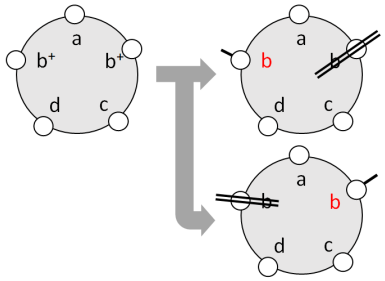


図 11 重複識別子のついた端子の削除に伴った拡張外周端子列

次に J 型について考えると，**図 9** に示す 4 つの端子を持つ J 型ブロックは，b の端子から配線を上下に出せるため，2 通りの外周端子列のどちらかとなるので外周未確定ブロックであり，拡張外周端子列 ab^+cdb^+ を得る．

同様に**図 10** に示す 4 通りの端子位置の J 型ブロックを考える．図 10(a) では，どちらも b の端子から配線を出す方向が上下にあるため，2 通りの外周端子列のどちらかとなるので外周未確定ブロックであり，拡張外周端子列 ab^+cb^+ を得る．

図 10(b) では，左のブロックは a の端子から配線を上下に出せるが，a の端子は配線を上から出しても下から出しても外周端子列が同じなので外周未確定ブロックではない．右のブロックは配線を出す方向が上下もしくは左右のみの端子がないため外周未確定ブロックではない．

重複識別子を持ったネット番号を含む拡張外周端子列は，配線を出す方向を決めることで重複識別子のない拡張外周端子列にすることができる．**図 11** に一例を示す．a の後ろの b^+ を消すと $abcd$ の順の拡張外周端子列が，a の前の b^+ を消すと $acdb$ の順の拡張外周端子列が得られる．重複識別子のついた片方のネット番号を消すことで配線を出す方向を決定できる．

4.3 Tree-connection の工夫

重複識別子がついている端子は自由度があるため Tree-connection は，重複識別子のついてない端子同士のみで行う．この制約により、異なる島に属する同ネット端子対が

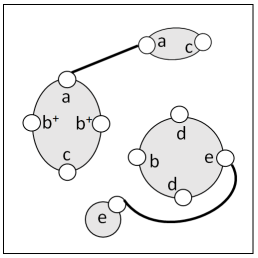


図 12 異なる島に属する同ネット端子対があるにもかかわらず片方の端子に重複識別子がついているため結ぶことができないネットが残る例

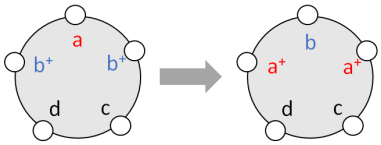


図 13 拡張外周端子列上で同じ重複識別子のついた同じネット番号の端子に挟まれた端子と同じ重複識別子のついた同じネット番号の端子の入れ替え

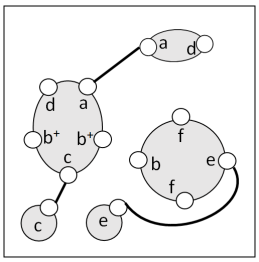


図 14 異なる島に属する同ネット端子対があるにもかかわらず入れ替えが行えないため結ぶことができないネットが残る例

あるにもかかわらず片方もしくは両方の端子に重複識別子がついているため結ぶことができないネットが残る可能性がある．このようになる一例を**図 12** に示す．この例では異なる島に属する同ネット端子対 b があるが，片方の島の b には重複識別子がついているため，結ぶことができないネットが残る．

そこで重複識別子のついた端子から重複識別子のついてない端子に重複識別子を入れ替える方法を示す．**図 13** に示すように ab^+cdb^+ という端子列上の b^+ab^+ は，a と b の順はどちらでもよいという意味なので，これらを a^+ba^+ に入れ替えて a^+ba^+cd という文字列に変換が可能である．このように重複識別子を入れ替えてから Tree-connection を行い異なる島に属する同ネット端子対がなくなるようにする．

しかしそれでも重複識別子のついた端子と重複識別子のついてない端子を入れ替えることができないために，異なる島に属する同ネット端子対があるにも関わらず，結ぶことができないネットが残る可能性がある．このようになる一例を**図 14** に示す．この例では異なる島に属する同ネット端子対 b があるが，片方の島の b には重複識別子がついており，なおかつ既に Tree-connection の影響で重複識別子の入れ替えもできないため，結ぶことができないネットが残る．

その場合は一度合体後の島ごとに既存の Chord-connection と**図 15** に示す同じ重複識別子を持った同じネット番号の端子の統合を行う．この端子の統合では，拡張外周端子列に同じ重複識別子のついた同じネット番号の

(a) 拡張外周端子列の一部　(b)(a) の統合後

図 15　同じ重複識別子を持った同じネット番号の端子の統合

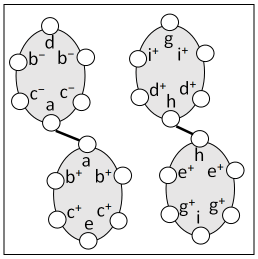


図 16　異なる島に属する同ネット端子対があるにもかかわらず入れ替えも既存の Chord-connection も統合も行えないため結ぶことができないネットが残る例

端子が連続していたら，両方の端子を合併し重複識別子を除く．既存の Chord-connection とネット番号の統合を経て重複識別子を外してから Tree-connection を行い異なる島に属する同ネット端子対がなくなるようにする．

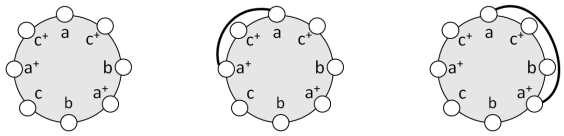
以上を繰り返し行ってもなお，異なる島に属する同ネット端子対のどちらか一方もしくは両方に重複識別子がついてるため，結ぶことができないネットが残る可能性がある．このようになる一例を**図 16** に示す．この例では異なる島に属する同ネット端子対 d と e があるが，片方の島の d と e にはどちらも重複識別子がついており，なおかつ既に Tree-connection の影響で重複識別子の入れ替えもできず，既存の Chord-connection と統合を行っても拡張外周端子列は変わらないため，Tree-connection が行えない．

その場合は，異なる島に属するつなげられない同ネット端子対のうち重複識別子がついている片方の端子を取り除き，もう一方の端子から重複識別子を外し，配線を出す方向をどちらかに決定する．この決定によっては平面配線可能が失われる可能性がある。そのため、最終的に拡張 Chord-connection が上手くいかなかった場合はバックトラックを行い、決定した向きと逆向きに決定しなおす。

4.4　Chord-connection の拡張

以上の工夫を経て，異なる島に属する同ネット端子対がなくなるまで Tree-connection を終えいくつかの合体した島々になった後は，拡張した Chord-connection を行う．拡張した Chord-connection では、拡張外周端子列上で連続している端子だけでなく、離れている端子も平面位相配線可能性を失わないように結ぶ。平面位相配線可能性を失わないで端子を結ぶには、拡張外周端子列を配線を結ぶ端子の内側と外側に分け、いずれのネットの端子も内側か外側どちらかの領域に、結ぶことのできる同ネットの端子対が属するように結ぶ。**図 17** に例を示す。図 17(b) では、配線を結ぶ端子の内側が c^+、外側が c^+, b, a^+, b, c となりネット b, c を結ぶ端子対が外側に残っているため平面位相配線可能性を失わないが、図 17(c) では、配線を結ぶ端子の内側が c^+, b、外側が b, c, a^+, c^+ となりネット b の端子対が外側と内側で分断されてしまったため平面位相配線可能性が失われる。

拡張した Chord-connection は以下の 4 つのステップからなる．ステップ 1 を繰り返し，連続している同ネット端



(a) 拡張外周端子列　(b)(a) の平面位相配線可能性を失わない端子の結び方　(c)(a) の平面位相配線可能性を失う端子の結び方

図 17　平面位相配線可能性を失わない端子の結び方

(a) 拡張外周端子列の一部　(b)(a) に拡張 Chord-connection のステップ 1 を適用後

図 18　拡張 Chord-connection のステップ 1

(a) 拡張外周端子列の一部　(b)(a) に拡張 Chord-connection のステップ 2 を適用後

図 19　拡張 Chord-connection のステップ 2

子がなくなったらステップ 2 を行い，再度連続した同ネット端子が得られたらステップ 1 を繰り返す．ステップ 2 で結べる端子対がなくなった場合ステップ 3 を行い，再度ステップ 1 を繰り返す．ステップ 3 で結べる端子対がなくなった場合ステップ 4 を行い，再度ステップ 1 を繰り返す．

ステップ 1　拡張外周端子列の中で重複識別子が異なるもしくはどちらか一方でも重複識別子がついていない同じネット番号の端子が連続している場合，これら 2 つの端子を選択する。

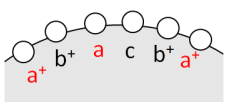
ステップ 2　拡張外周端子列の中でどちらも重複識別子がついていない同じネット番号の端子の間に、重複識別子のついた端子しか存在しない場合，これら 2 つの端子を選択する。

ステップ 3　拡張外周端子列の中で、重複識別子により配線を結ぶことができる同ネット端子の選び方が複数通りあるネット番号において、1 通りの選び方のみいずれのネットの平面位相配線可能性をなくさずに配線でき、これらの端子の間に重複識別子のついた端子しか存在しない場合、これら 2 つの端子を選択する。

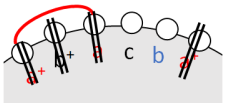
ステップ 4　拡張外周端子列の中で、配線を結ぶことができる同ネット端子をいずれのネットの平面位相配線可能性をなくさずに配線でき、これらの端子の間に重複識別子のついた端子しか存在しない場合、これらの端子対を選択する。

各ステップ後　選択した 2 つの端子を結び列から取り除く。これら 2 つの端子を結ぶ配線と交差する配線を出さないように，間にある重複識別子のついた端子を取り除く。このとき取り除いた端子の中に重複識別子が異なる同じネット番号の端子があった場合は、それらの端子を結び列から取り除く。配線を結んだ端子と同じネット番号の端子も取り除く．配線を結ばずに取り除いた端子と同じネット番号で同じ重複識別子のついた端子から重複識別子を消す。

ステップ 1 の動作例を**図 18** に示す．図 18(a) では a^+

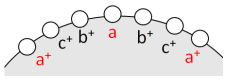


(a) 拡張外周端子列の一部

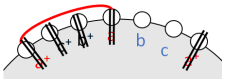


(b)(a) に拡張 Chord-connection のステップ 3 を適用後

図 20 拡張 Chord-connection のステップ 3



(a) 拡張外周端子列の一部



(b)(a) に拡張 Chord-connection のステップ 4 を適用後

図 21 拡張 Chord-connection のステップ 4

と a^- の端子が連続しているので，それらの端子を結び，結んだ端子と同じ a のネット番号の端子も取り除き (b) となる．

ステップ 2 の動作例を**図 19** に示す．図 19(a) では a のネット番号の端子には重複識別子がついてないうえに、端子間に重複識別子のついた端子しか存在しないため，a の端子間を結び列から除き。間にある b^+，c^+ の端子を取り除く。配線を結んだ端子と同じ a のネット番号の端子を取り除き、配線を結ばずに取り除いた b^+，c^+ の端子と同じネット番号で同じ重複識別子のついた端子から重複識別子を消すと (b) となる．

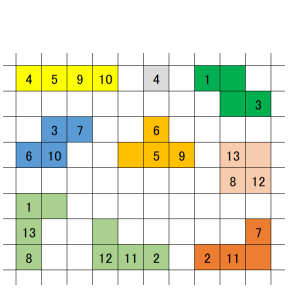
ステップ 3 の動作例を**図 20** に示す．図 20(a) では a のネット番号の 2 通りの端子対のうちの、端子間に c, b^+ を含む端子対を結ぶと平面位相配線可能性を失ってしまうが，端子間に b^+ のみを含む端子対を結んでも平面位相配線可能性を失わない上に、端子間には重複識別子のついた端子しか存在しないため，端子を結び列から取り除き、b^+ を挟む a の端子を結び列から取り除き、間にある b^+ の端子を取り除く。配線を結んだ端子と同じ a のネット番号の端子を取り除き、配線を結ばずに取り除いた b^+ の端子と同じネット番号で同じ重複識別子のついた端子から重複識別子を消すと (b) となる．

ステップ 4 の動作例を**図 21** に示す．図 21(a) では a のネット番号の 2 通りの端子対では、どちらの端子対を結んでも平面位相配線可能性を失わない上に、どちらの端子間も，b^+，c^+ を含み，重複識別子のついた端子しか存在しない。そのうちの一方の端子対を選択し端子を結び列から取り除き，間にある b^+，c^+ の端子を取り除く。配線を結んだ端子と同じ a のネット番号の端子を取り除き、配線を結ばずに取り除いた b^+，c^+ の端子と同じネット番号で同じ重複識別子のついた端子から重複識別子を消すと (b) となる．

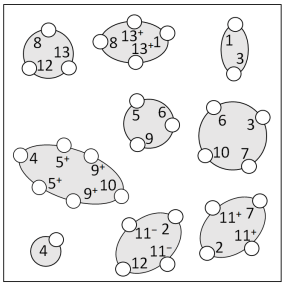
最後に交差なく配線できた場合，問題は平面配線可能である．ステップ 4 を行っても結べる端子対がなかった場合は，Tree-connection の工夫で配線を出す方向を決め打ちした所までバックトラックし，配線を出す方向を逆にして再度 Tree-connection から行う．

以上により外周未確定ブロックの配線を出す方向を拡張した CHORD-LAST 法を用いることで必ず解を持つ方向に決めることができる．

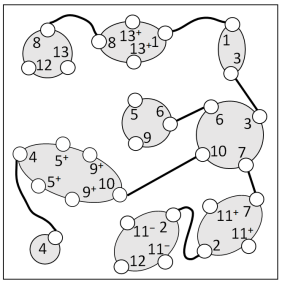
図 22 に配置配線パズルの例と拡張した CHORD-LAST 法の適用例を示す．



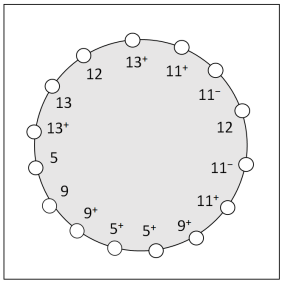
(a) 問題例



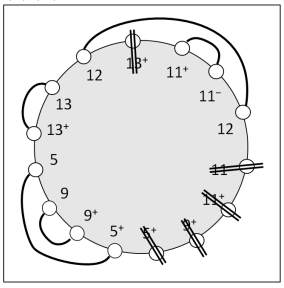
(b)(a) を外部配線問題化



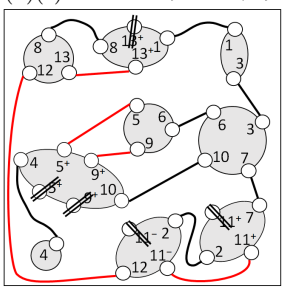
(c)(b) の Tree-connection 後



(d)(c) を 1 つの島とした表現



(e) 拡張 Chord-connection 後の (c)



(e) 拡張 Chord-connection 後の (d)

図 22 配置配線パズルの例と拡張 CHORD-LAST 法の適用例

5. まとめ

CHORD-LAST 法を拡張して用いることで配置配線を行う前の段階で，外周未確定ブロックの配線を出す方向を決定することができ無駄な探索を減らすことができた．

参考文献

[1] W. M. Dai and E. S. Kuh, ”Simultaneous floor planning and global routing for hierarchical building block layout“, IEEE Trans. CAD, vol. 6,No.5, pp. 828-837,1987.

[2] W. M. Dai, B. Eschermann, E. S. Kuh, M. Pedram, ”Hierarchical placement and floorplanning for BEAR“, IEEE Trans. CAD, vol. 8, pp. 1335-1349, 1989.

[3] C.-Y. Chin, C.-Y. Kuan, T.-Y. Tsai,H.-M. Chen, and Y. Kajitani, ”Escaped Boundary Pins Routing for High-Speed Boards“ IEEE Trans. CAD,Vol.32,No.3,pp.381-391,2013.

[4] 田中 雄一朗, 高橋 篤司,”領域分割を用いた CHORD-LAST 法に基づくナンバーリンク解法 “,DA シンポジウム 2014 論文集,pp.221-226,2014

[5] ”アルゴリズムデザインコンテスト 2019（ADC2019）ルール説明 “,2019/7,https://dasadc.github.io/adc2019/,（参照 2019/7/17）

[6] ”ナンバーリンクの遊び方, ルール, 解き方 — WEB ニコリ “,2019/7,https://www.nikoli.co.jp/ja/puzzles/numberlink/,（参照 2019/7/17）