設計自動化 44-10
(1988. 10. 28)

チャネル構造列挙の一手法

福井正博 岩崎知恵 羽山繁
松下電器産業株式会社 半導体研究センター

フロアプランの処理において、著者らは既にブロックの配置を実現する全てのスライシングチャネル構造を列挙し、それぞれのチャネル構造に対して最適なブロック形状を求める方法を示した。しかし同方法は列挙されるチャネル構造の個数がブロックの個数に対して指数関数的に増加するといった問題点がある。今回の報告では、同手法を改善した手法を示す。本手法は全てのチャネル構造を列挙するのではなく発見的な方法によって一部の良好な結果を得ると思えるチャネル構造のみを列挙する、スライシングチャネル構造に限らずノンスライス構造も列挙する、階層的なチャネル構造を用いてブロック形状の最適化を行なう等の特徴を持つ。最後に、理論的な評価も行なう。

An Efficient Algorithm to List Channel Structures

Masahiro Fukui, Chie Iwasaki and Shigeru Hayama

Semiconductor Research Center, Matsushita Electric Industrial Co., Ltd.
3-15 Yagumonakamachi, Moriguchi, Osaka, 570 JAPAN

In a floor planning problem, we proposed an algorithm which lists all slicing channel structures constructed by placement of blocks, then optimizes block's boundaries for each channel structure, and finally selects the best channel structure. However, the number of channel structures grows exponentially with the number of blocks. We propose a new improved algorithm. This one lists not only slicing structures but also non-slicing structures, and searches only channel structures from which a fine result seems to be taken. Furthermore, in order to reduce computational complexity, a hierarchical channel structure is used. Theoretical evaluations are also shown.

1。はじめに

近年のＶＬＳＩの大規模化、高集積化に伴いシステムレベルの機能を１チップに搭載することも行なわれるようになってきている。このような大規模なＶＬＳＩを効率的、効果的に設計するためには階層的設計手法がとられるが、その設計段階の中でもとりわけトップダウン設計、すなわちフロアプランの段階が結果の最適性に与える影響が大きい。

フロアプランシステムの目的は、最小のチップ面積や配線長を与えるブロックの配置と各ブロックの形状と外部ピン位置を求めることである。同問題に対して、矩形双対グラフを列挙する手法[1]や、階層的な接続構造を作って階層ごとに矩形双対グラフを列挙し適用を試してゆく方法[2]などが示されている。

また、スライシング構造が与えられた場合にブロックの最適形状を求めるエレガントな方法が[3]で提案され、同手法を部分的に使った方法もいくつか報告されている。[4] では、スライシング構造を2進木で表現し、2進木の組み替えによってスライシング構造を探索する方法が試みられている。これらの従来の多くの手法における問題として、チャネル構造の列挙数がブロックの個数の増加に伴って指数関数的に増大することが指摘されている[1,2]。

[2]では、フロアプラン処理を階層化するためにクラスタリング法によってボトムアップ的に階層構造を構成した後、各階層ごとに適用するフロアプランのテンプレートを探索することによって最適化を行なっている。各階層で扱う矩形領域（ルームという）の個数は5以下と限定している。これは、ルームの個数が5 のときに列挙されるフロアプランのテンプレートの個数は92となり、それ以上ルームの個数を増やすと計算が困難となるためである。

著者らは、フロアプランの一手法として、概略フロアプランと詳細フロアプランの段階に分けて最適化を行なう手法を示した[5]。同手法は概略フロアプランではブロックのおおまかな位置決めと形状の決定を行ない、詳細フロアプランではブロック配置とピン配置を大きく変更せずにスライシング構造を全て列挙し、各チャネル構造に対してブロックの形状を最適化する。同手法は、形状可変ブロックの形状およびブロック間配線領域を精度良く予想できるといった特徴があるが、列挙されるチャネル構造がスライシング構造のみに限定される、列挙されるチャネル構造の個数がブロックの個数に対して指数関数的に増大するといった課題があった。ノンスライシング構造は一般的に配線が困難であるが、同構造を扱うことにより面積効率の良いフロアプランが得られる場合がある。また、同構造を扱う配線システム[6]も提案されていることから同構造も列挙するほうが有利である。

本報告では前回著者らが示した手法[5]に対する改善手法を示し、その評価を行なう。本手法は、効果的なレイアウト結果を得ると考えられるようなチャネル構造を選択して列挙するための発見的手法である。さらに、本手法はノンスライシング構造も扱うことができ、ブロックの個数が多い場合には階層的なチャネル構造の列挙とブロック形状の最適化を行なう。

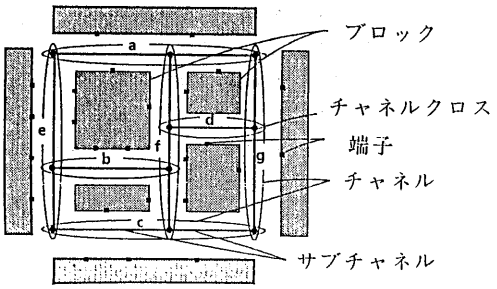

図1 レイアウトモデル

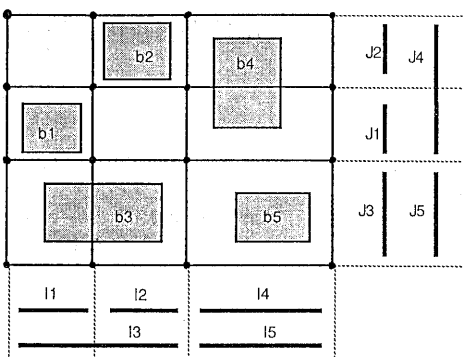

図2 領域表現グラフGz

2。問題定義

2-1。レイアウトモデル

フロアプランにおいて著者等が扱うレイアウトのモデルは図1に示すものを考える。チップの周辺には、4つのＩ／Ｏブロックが存在し、それらに囲まれた矩形の領域に大きさの異なる矩形のブロックが存在する。各ブロックは形状可変と形状固定の2種

類があり、形状可変ブロックはピン位置およびブロック上を通過するフィードスルーを自由に設定できる。チップ上に配置するブロックは、ハンドクラフトまたはスタンダードセル方式またはモジュールゼネレータによって設計されるものを考える。固定形状ブロックはブロック形状、ピン位置、フィードスルー位置等があらかじめ固定されている。さらに階層的な接続情報とクリティカルネットの情報が与えられる。図1に示すように隣接するブロックに挟まれた帯状配線領域をサブチャネル、サブチャネルどうしが交差する点をチャネルクロスといい、サブチャネルを辺で、サブチャネルの両端のサブチャネルクロスをその辺に接続する頂点で表現したものをチャネルグラフをいう。隣接するサブチャネルを併合して得られる帯状の配線領域をチャネルといい、チャネルグラフをチャネルの集合で表現したものをチャネル構造という。与えられたチャネルグラフAにおいて、チャネルグラフの上端から下端まで、あるいは右端から左端まで渡るチャネルが存在するばあいに、そのチャネルによってチャネルグラフAを2つのチャネルグラフA1，A2に分割することをスライシングという。チャネルグラフのスライシングによって新しい2つのチャネルグラフに分割する処理を再帰的に行なうことにより、各チャネルグラフに含まれるブロックの個数が1となるとき、与えられたチャネルグラフはスライシング可能という。このときスライシングが行なわれたチャネルの構造をスライシング構造という。チャネル構造の内、スライシング構造でないものをノンスライシング構造という。

2-2. 領域表現グラフGz

ブロックの配置を表現し、チャネル構造を探索するために、領域表現グラフGzを用いる。同グラフは以下の手順で構成されるグラフである。

ブロックの集合を$B=\{b_1,b_2,\ldots,b_k\}$として、各ブロックのx，yそれぞれの方向で座標の占める区間を$X(b_i)=(xmax(b_i),xmax(b_i))$, $Y(b_i)=(ymax(b_i),ymax(b_i))$、x軸上での値$x_0$を区間の中に含むようなブロックの集合を$S(x_0)$で表現する。x方向のチップの占める区間Iを以下の条件a，b）を満足するように複数の連続した重なりのない区間I1，I2，．．．Inに分ける。

a）隣接する2つの区間Ii，Ijに含まれるブロックの集合が異なりしかも一方が他方の部分集合とならない。

b）各部分区間I内の任意の点xに対して、$S(x)\in S(x0)$または、$S(x)\in S(x0)$が成立するようなx0が区間I内に存在する。

y方向についても同様の操作によってy軸方向に占めるチップの区間を連続した重なりのない区間J1，J2，．．．，Jnに分ける。

次に、上で定義したxおよびy方向の区間Ii，Jiによってチップが占める矩形の領域を格子状の領域に分ける。領域表現グラフGz=(Vz；Ez)の頂点は，格子の各交点に対応し、辺e＝（u，v）は、上下または左右に隣接する交点u，vの間の線分に対応して2頂点u，v間に引く。Gzの各辺eに対応した線分を横切るブロックの名前を持つ変数をblock(e)とする。横切るブロックが存在しない場合にはblock(e)＝nilとする。

ブロックの配置と領域表現グラフGzの例を図2に示す。領域グラフGz上で同じx座標値を持つ頂点の集合およびそれらの間の辺よりなる部分グラフを縦の境界、同じy座標を持つ頂点の集合およびそれらの間の辺よりなる部分グラフを横の境界と呼ぶ。block(e)＝nilを与える辺のみからなる境界はチャネルグラフにおけるスライスに対応するものである。

Gzの各頂点vに対して同頂点vに対応した点でのチャネルの方向をD(v)で表現する。チャネルの方向が水平方向の場合はD(v)＝'H'，垂直方向の場合はD(v)＝'V'とする。チャネルの方向が未定義の場合はD(v)＝nilとする。全ての頂点にD(v)が定義されたものは1つのチャネル構造を表現する。ラベルD(v)でチャネル構造を表現した例を図3に示す。1つの領域グラフ上で定義されるチャネル構造の個数は同グラフの頂点数がnのとき2^nである。

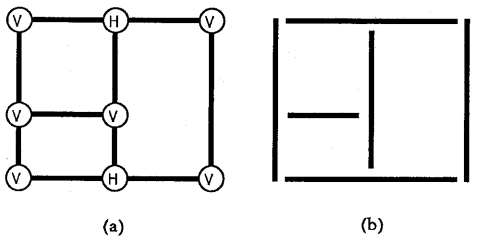

図3 (a) 領域表現グラフの頂点に与えられたラベルD(v)の例、 (b) (a)に対応したチャネル構造

2-3. フロアプランのアルゴリズム概略[5]

著者らが用いるフロアプラン手続きF_PLANを概説する。

<手続き F_PLAN>

（概略フロアプラン）

S1: ブロックの概略配置。

S2: ブロック概略形状、概略ピン位置の決定。

（詳細フロアプラン）

S3: ブロック形状の変化とブロック間配線領域を予想するための情報の作成。

S4: ブロックの拡散配置。

S5: 領域表現グラフの作成、チャネル構造の列挙。

S6: ブロック形状の最適化。

S7: ピン詳細配置、フィードスルー割りあて。

S8: グローバル配線、ブロック形状の最適化。

まずS1，S2の段階では、ブロックをクリティカルネットの配線長と面積を考慮しながら、チップ上に配置し、概略の形状をきめる。ここでは、ブロックは特に矩形である必要はない。スタンダードセル方式ブロックは、一般的にピンの配置を変えたとき、面積も異なるので、この段階であらかじめピンの概略位置を与えておくようにする。ピンの概略位置の決定は、S7と同じ処理によって行なう。

次に、各ブロックを矩形で実現する場合に取ることのできる形状の候補をもとめる。形状の候補の中からS2で求めた形に一番近いものを選択し、チップ上に配置する。次に、チップ上の原点Oを中心として、各ブロックを点Oから離れる方向に移動させる。ここで拡散の距離が大きいほど次の段階で列挙されるチャネル構造の個数が増大する。本システムでブロック配置を広げる距離は、スライシングが可能な最小値か、または外部より与えた値とする。

次に、ブロックの配置を表現する領域表現グラフGzを求め、同グラフ上でスライスをひく手順を探索することによりブロックの配置を実現する全てのスライシング構造を列挙する。

次に、列挙された各スライシングチャネル構造に対して、[3]の方法をブロック間配線が考慮されるように改善した方法を用いて、ブロック形状の最適化を行なう。さらに、得られた結果の内からチップ面積を最小化するスライシングチャネル構造とブロック形状を数通り求め、自動あるいは会話的手段によりその内の1つを解とする。

最後に、ピン詳細配置、フィードスルー割りあてを行ない、ブロック間のグローバル配線を実行することによってブロック間配線領域の面積を推定し、再度S6で用いたブロック形状最適化を行なう。

2-4. チャネル構造列挙問題

2-3のアルゴリズムにおいてチャネル構造の最適性は、同チャネル構造に対してブロックの形状を最適化した場合のチップ面積によって評価する。

全てのチャネル構造を列挙し、ブロック形状最適化結果を比較すれば最適解が見つかるが、前述したように列挙数はブロックの個数に対して指数関数的に増加する。

本文においてチャネル構造列挙問題は、ブロックの配置より得られる領域表現グラフ上で定義されるチャネル構造のうち、最適または最適に近い結果を与えると思われるチャネル構造を許容可能な計算時間複雑度で列挙することを目的とする。

3. チャネル構造列挙アルゴリズム

3-1. シンプルアルゴリズム[5]

領域表現グラフGzが与えられたときに、スライシング構造を全て列挙する方法<手続きLST_STR>を示す。

スライシング構造は、領域表現グラフGz上のスライスとなりうる境界を見つけ、Gzをその境界によって分割するという操作を繰り返したときに用いたスライスの集合によって得られる。スライスに対応した境界の取り方の場合分けを列挙することにより全てのスライシング構造を列挙する。

本手続きにおいて、領域表現グラフGz={Vz;Ez}の全頂点vの内、領域の上下辺に対応した頂点にD(v)='H'，左右辺と4隅に対応した頂点にD(v)='V'、その他の全頂点vに対してD(v)=nil、が初期値として与えられているものとする。STR_LSTはスライシング構造のリストとして使う。

＜手続きLST_STR(Gz) ＞

1: 手続きLST_STRV(Gz) を呼び出す。

2: 手続きLST_STRH(Gz) を呼び出す。

3: LST_STR(Gz) を終了する。

＜手続き LST_STRV(Gz) ＞

1: 領域グラフGzの縦の境界sを左から順に取り出し、各境界sに対して以下の処理を行なう。

1.1: 境界sがスライスであれば、Gzのsおよびsより右にある頂点よりなる部分グラフをG1、sおよびsより左にある頂点よりなる部分グラフをG2とする。

1.2: sに含まれる頂点vに対して、D(v)=nilならばD(v)='V'とする。

1.3: 全ての頂点vに対してD(v)='V'または'H'が与えられていれば、D(i)（i∈Vz）をチャネル構造のリストSTR_LSTに追加する。

1.4: 手続きLST_STR(G1) を呼び出す。
1.5: 手続きLST_STRH(G2) を呼び出す。
1.6: 境界sの両端を除いた各頂点vに対してD(v) = nilとする。
2: LST_STRV(Gz) を終了する。

＜手続き LST_STRH(Gz) ＞
1: 領域グラフＧｚの横の境界sを下から順に取り出し、以下の処理を行なう。
1.1: 境界sがスライスであれば、Ｇｚのsおよびsより上にある頂点よりなる部分グラフをＧ１、sおよびsより下にある頂点よりなる部分グラフをＧ２とする。
1.2: sに含まれる頂点vに対してD(v) = nilならば、D(v) ='H'とする。
1.3: 全ての頂点vに対してD(v)='V'または'H'が与えられていれば、D(i)（i∈Ｖ）をチャネル構造のリストSTR_LSTに追加する。
1.4: 手続きLST_STR(G1) を呼び出す。
1.5: 手続きLST_STRV(G2) を呼び出す。
1.6: 境界sの両端を除いた各頂点vに対してD(v) = nilとする。
2: LST_STRH(Gz) を終了する。

図4にLST_STRによってチャネル構造が列挙される例を示す。スライスとなる境界を縦型探索することによってチャネル構造を列挙してゆくようすがわかる。

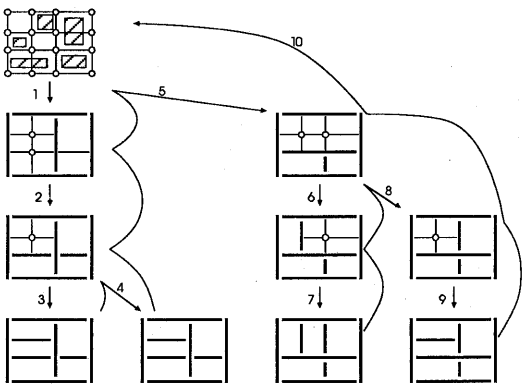

図4 スライスの探索

3-2。改善アルゴリズム

手続きLST_STRはスライシング構造の配置しか扱えなかったが、次にノンスライシング構造も列挙するように改善した方法を示す。ここでは、ノンスライシング構造の配置に対してノンスライシング構造のみを、スライシング構造の配置に対してはスライシング構造のみを探索する手続きLST_STR2を用いる。

概略の処理フローは以下のとおりである。

< 手続き LST_NS&SL_STR >
1: 領域表現グラフGzを作成する。
2: 手続きLST_STR2を呼び出す。
3: ノンスライス構造の配置が与えられた場合、スライシング構造が得られる配置になるまで配置の拡散を行ない[5]、LST_STR2を再度呼び出す。

図5に手続き LST_NS&SL_STRで列挙されるチャネル構造の例を示す。図5(a)は与えられたブロックの配置とその領域表現グラフによる表現、(b)はステップ2:で列挙されるチャネル構造、(c)は配置拡散を行なった後の領域表現グラフ、(d)はステップ3:で列挙されるチャネル構造である。

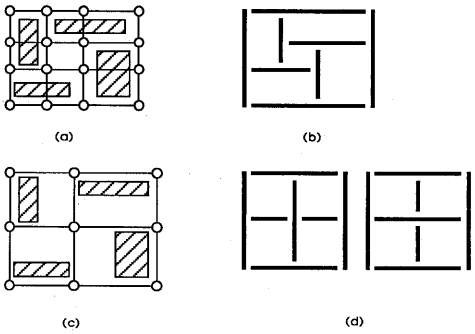

図5 LST_NS&SL_STRで列挙されるチャネル構造

手続きLST_STR2は領域表現グラフにスライスとなる境界が見つからなかった場合に、スライスでない境界の内1本を用いて領域表現グラフを2分割し、2分割された領域表現グラフに対して境界の探索の処理を再帰的に続行する。

また手続きLST_STR2は、スライスの探索を限定するために、スライスを引く場所を評価する。垂直方向のスライスviを評価する関数e(vi)は次式で与える。

$e(vi) = k1*\Delta y(ri) + k2*\Delta w(vi)$

ここで、Δy(ri)は各ブロックに面積に応じた重力を考えた場合のviの左右におけるブロックの重心位置のy方向のずれ、Δw(vi)は左右のブロックの重力の差である。k1，k2は外部から与えるパラメータであり、仮にk1 = 100，k2 = 10に設定する。

＜手続きLST_STR2(Gz) ＞
1: 領域グラフGzのスライスとなる境界sを全て求め、

評価値e(s)を計算する。e(s)の大きな境界からk番目までのスライスの集合をSとする。
2: Sが空でないとき以下を行なう。
2-1: 手続きLST_SL_STRV(Gz,S) を呼び出す。
2-2: 手続きLST_SL_STRH(Gz,S) を呼び出す。
3: Sが空のとき、手続きLST_NS_STR(Gz) を呼び出す。
4: LST_STR(Gz) を終了する。

＜手続き LST_SL_STRV(Gz,S) ＞
1: Sに含まれる縦方向の境界sを左から順に取り出し、境界sによりGzのsおよびsより右にある頂点よりなる部分グラフをG1、sおよびsより左にある頂点よりなる部分グラフをG2とする。
2: sに含まれる頂点vの内D(v) = nilのもの全てに対してD(v)='V'とする。
3: 全ての頂点vに対してD(v)='V'または'H'が与えられていれば、D(i) (i∈Vz) をチャネル構造のリストSTR_LSTに追加する。
4: 手続きLST_STR2(G1) を呼び出す。
5: 領域グラフG2のスライスとなる境界sの内、評価値e(s)の大きなものからk番目までの集合をSとし、手続きLST_SL_STRH(G2,S) を呼び出す。
6: sに含まれる各頂点vに対してD(v) = nilとする。
7: LST_SL_STRV(Gz) を終了する。

＜手続き LST_SL_STRH(Gz) ＞
1: Sに含まれる縦方向の境界sを下から順に取り出し、境界sによりGzのsおよびsより上にある頂点よりなる部分グラフをG1、sおよびsより下にある頂点よりなる部分グラフをG2とする。
2: sに含まれる頂点v頂点vの内D(v) = nilのもの全てに対してD(v)='H'とする。
3: 全ての頂点vに対してD(v)='V'または'H'が与えられていれば、D(i) (i∈V) をチャネル構造のリストSTR_LSTに追加する。
4: 手続きLST_STR2(G1) を呼び出す。
5: 領域グラフG2のスライスとなる境界sの内、評価値e(s)の大きなものからk番目までの集合をSとし、手続きLST_SL_STRV(G2,S) を呼び出す。
6: sに含まれる各頂点vに対してD(v) = nilとする。
7: LST_SL_STRH(Gz) を終了する。

＜手続き LST_NS_STR(Gz) ＞
1: 領域表現グラフGzの境界の内、同境界の一方の端に位置する辺がブロックと交差しないもので、同境界に交差するブロックの個数が最小であるような境界を1個選択する。(図6)
2: 1:で選択した境界sが縦方向のとき、境界sによりGzのsおよびsより右にある頂点よりなる部分グラフをG1、sおよびsより左にある頂点よりなる部分グラフをG2とする。境界sが横方向の場合も同様に境界の下の部分グラフをG1、上の部分グラフをG2とする。
3: 境界sに含まれる頂点vの内、ブロックと交差する辺の両端点をD(v) = 'H'、その他の頂点vでD(v) = nilのもの全てに対してD(v)='V'とする。
4: 手続きLST_STR2(G1) を呼び出す。
5: 手続きLST_STR2(G2) を呼び出す。
6: LST_NS_STR(Gz) を終了する。

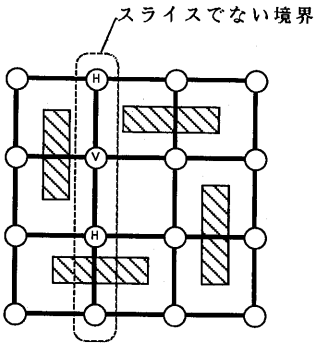

図6 手続きLST_NS_STRの処理例

LST_STR2によって列挙されるチャネル構造の上限は、ブロックの個数をbとした場合に、K^bである。実用上のbの最大数は、6または7程度である。(k = 3, b = 7または8のときそれぞれ列挙数の上限が729または2187である。実際の使用ではブロックを大きく離して置かないため、ほとんどのばあい列挙数は上限の1/2以下である。) ブロック数が多いばあいに列挙されるチャネル構造を効果的なものに限定し、ブロック形状の最適化を効率良く行なうためには、アルゴリズムの階層化が不可欠である。

4. 階層化アルゴリズム

本章では領域表現グラフを階層化したデータ構造を導入し、同グラフを用いてチャネル構造の列挙とブロック形状の最適化を行なう処理について述べる。

4-1. 領域表現グラフの階層化

ブロックが配置された領域を階層的に分割し、それに伴って階層的な領域表現グラフを作る手続きAREA_DIV_SLを示す。

本手法は、複数のブロックを囲む矩形領域を中間階層に設け、さらにそれらの矩形領域のいくつかを囲む矩形領域を上位の階層として設けるといった方

法で、ブロック全体を囲む矩形領域を階層的に表現したものを使う。たとえば、図7の例では、(a)の領域表現グラフは、(b)に示した階層構造に従って、(c)および(d)に示すような階層ごとの領域表現グラフに直す。

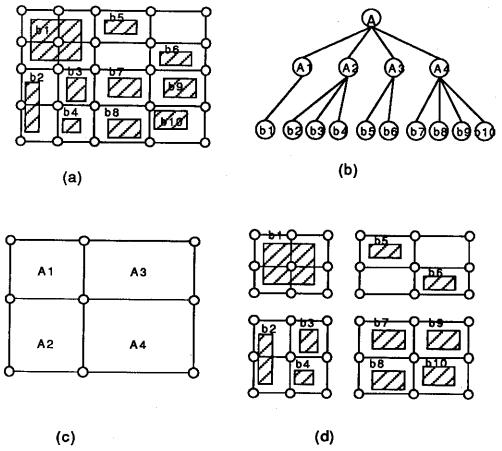

図7 階層的な領域表現

本手続きは、各階層の子の個数をK(K=6とする)以下になるように分割を行なう。本手続きでAはブロックが配置された領域である。

<手続きAREA_DIV_SL(A)>
1: 領域Aに含まれるブロックの個数がK個以下ならば終了する。
2: 領域A内のブロックの配置の基ずいて、領域表現グラフGzを作成する。
3: 領域表現グラフGz上で全てのスライスを列挙する。もしスライスがない場合には、スライスが得られるまでブロックの配置を拡散し、スライスを列挙する。
4: 列挙されたスライスsの内、e(s)の値が大きなものからL本(L=3とする)のスライスを選択し、それらによって領域をA1, A2, ..., Akに分割し、領域A1, A2, ..., Akの配置の基ずいて領域表現グラフGz(A)を作成する、各領域Ai(i=1,2,...,k)に対してAREA_DIV_SL(Ai)を呼び出す。
5: 終了する。

4-2. ブロック形状の最適化

処理アルゴリズムを示すまえに、[3]の方法を概説する。[3]では、ブロックが取りうる形状(Xi, Yi)を図8に示すようなXY座標平面上の点に対応づけ、ブロックを囲む矩形の取りうる形状を座標平面上の斜線で示した部分に対応づけて表現している。2つの垂直方向に隣接したブロックA, Bを囲む矩形の形状は、A, Bそれぞれの不等式をY方向に加えたもので表現できる(図8(c))。図8(c)で点・がA, Bを囲む矩形が取りうる形状の局所最適解である。(c)上で点aを実現するためのブロックA, Bの形状は、不等式(a), (b)上で、点aと同じx座標値を持つ点のy座標を与える点に対応するものである。

水平方向に隣接するブロックについても同様にそれぞれの不等式をX方向に加えたもので表現できる。スライシング構造を囲む矩形を表現する不等式は、同様の操作を繰り返すことによって求めることができる。また逆に、スライシング構造を囲む矩形の各形状から、それを実現する内部の各ブロックの形状が一意的に求まる。

次に、Gzで表現されたブロックの配置を囲む矩形Rの形状は、図9に示すようにGzから得られる各スライシング構造に対して得られる不等式(図9(a), (b))の和集合をとることによって表現できる(図9(c))。ここで、 点は(a)の、 点は(b)のスライシング構造を用いたことを示している。矩形領域Rがとりうる各形状からは、その形状を実現するチャネル構造と下階層の矩形領域の形状が一意的に求まる。

次に処理アルゴリズムを示す。

まず、4-1。で求めた領域の階層表現に従って、各階層ごとに、手続きLST_STR2を使ってチャネル構造を列挙する。次に、下の階層から順に上の階層の矩形領域の形状を示す不等式を組み立ててゆく。次に、組み立てられた不等式上で面積最小等の指標に従って1点を選択し、その形状を実現するための下階層のチャネル構造と形状を順番にたどってゆくことによって求める。

以上の手続きにおいてチャネル構造の列挙を各階層ごとに行なうため、ブロックの個数をbとしたばあいに列挙数は$(b/k)*M^K$である。K, M,はそれぞれ定数で(K=6,M=3)である。このため列挙数のオーダはo(b)である。

4-3. ノンスライシング構造の考慮

4-1,4-2ではスライシング構造に限定して述べたが、ノンスライシング構造も扱うように拡張することもできる。4-1ステップ3では配置を拡散してノンスライシング構造が列挙されることを排除しているが、拡散して矩形領域を定義した後に、各矩形領域を逆に近ずけることによってノンスライシング構造の列挙される場合を含めることができる。

また、ノンスライシング構造に対するブロック形状の最適化に関しては、[1]に示された方法などが適用可能である。

5。評価とまとめ

フロアプランの処理において、チャネル構造列挙に関する問題点を明確にし、階層化によるアプローチを示した。本手法は全てのチャネル構造を列挙するのではなく、良好な結果を得ると考えられるチャネル構造のみを選択し列挙するので計算時間を減らし、扱えるブロックの個数を大幅に増やすことができる。本手法は処理を階層化することによって列挙されるチャネル構造の個数をo(n)(nはブロックの個数)に限定できる。また、本手法によればスライシングチャネル構造のみでなく、ノンスライシングチャネル構造も列挙できる。

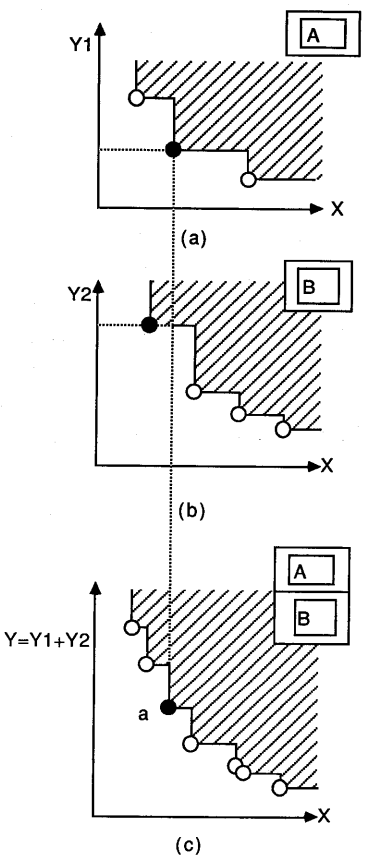

図8 ブロック形状の不等式による表現

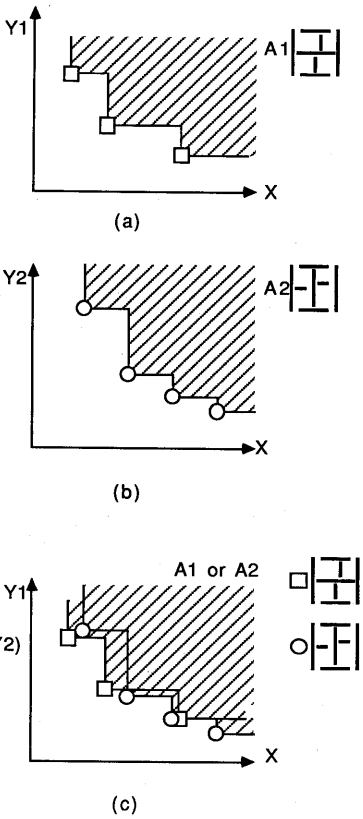

図9 複数のチャネル構造をとりうる領域の不等式による表現

参考文献

[1] 小池恵一、柳文鉱、谷勝則、築山修治、白川功：矩形双対グラフを用いたチップフロアプランの一手法、信学会技報、回路とシステム研究会、CAS85-144, pp.93-100, 1985.

[2] W. M. Dai and E. S. Kuh, " Simultaneous floor-planning and global routing for hierarchical building block layout",Proc. 1986 Int. Conf. on CAD, Santa Clara, Nov. 1986.

[3] L. Stockmayer, "Optimal Orientations of Cells in Slicing Floorplan Designs," Information and Control, Vol.59, pp.91-101, 1983.

[4] D. F. Wong and C. L. Liu, "A new algorithm for floorplan design," Proc. on Design Automation Conf., pp.101-107, 1986.

[5] 福井正博、山本敦志、岩崎知恵、羽山繁：ＶＬＳＩレイアウト設計におけるブロック形状ピン配置の最適化の一手法、信学会技報、ＶＬＳＩ設計技術研究会、 VLD88-4, pp.25-32, Apr. 1988.

[6] M. Fukui, A. Yamamoto, R. Yamaguchi, S. Hayama and Y. Mano," A block interconnection algorithm for hierachical layout system," IEEE Trans. on Computer-Aided Design, Vol.CAD-6, No.3, pp.383-391, May 1987.