設 計 自 動 化 84－5
（1997．5．23）

配線混雑緩和を目的とした大規模ゲートアレイ用配置改善手法

定兼利行　城田博史　高橋一浩　寺井正幸

三菱電機（株）半導体基盤技術統括部
〒664　兵庫県伊丹市瑞原4-1
E-mail: sadakane@lsi.melco.co.jp

大規模ゲートアレイ用の配置改善手法を発表する．この手法はセルサイズ拡大(padding)のフェーズと高速な反復改善のフェーズからなる．チップ上の局所的な配線混雑を緩和するために，paddingフェーズは配線混雑領域のセルのサイズを仮想的に拡大した後，全てのセルを重なりなく再配置する．我々は，再配置によってセルの相対的な位置関係が変わらないという想定のもとでセルの拡大サイズから再配置後の配線混雑を予測する式を立てた．この式を用いることにより，paddingフェーズはチップ上の配線混雑を均一化するようなセルの拡大サイズをうまく決めることができる．反復改善フェーズは，局所混雑を考慮した良く知られた配置の評価関数を最小化するものであるが，我々はゲインの見積もり値を用いてセルの移動先を決定することにより高速な反復改善処理を実現した．大規模ゲートアレイの回路で実験したところ本手法により配置のroutabilityは大きく改善した．

A Congestion-Driven Placement Improvement Algorithm for Large Scale Sea-of-gates Arrays

Toshiyuki Sadakane,　Hiroshi Shirota,　Kazuhiro Takahashi,　Masayuki Terai
Mitsubishi Electric Corporation
4-1 Mizuhara, Itami, Hyogo 664, Japan

A fast placement improvement algorithm for large scale gate arrays is reported. This algorithm consists of a new cell padding phase and a fast iterative improvement phase. To reduce local routing congestion on a chip, the padding phase virtually expands the size of cells in the congested regions and relocates all the cells to eliminate the cell overlap, preserving the relative cell position. We have developed a formula by which to estimate from the expanded cell sizes the congestion after the relocation in each region on a chip. Using this, the padding phase determines cell sizes that will equalize the congestion throughout a chip, by simulated annealing. The iterative improvement phase minimizes the well known objective function that takes the local congestion into account, but our algorithm is faster because of the use of a new gain estimation method for determining a better position to which to move a cell. The experimental results on large gate array designs indicate that the routability of cell placement is considerably improved by our algorithm.

1 はじめに

大規模なLSIの配置配線設計においては，しばしば局所的な配線の混雑が発生する．例えば論理合成ツールによって生成された回路ブロックは多端子ネットやporosityの低いセルを多用する場合がある．このようなブロックが配置された領域では，その領域内を通るべき配線の量（配線需要）が使用可能な配線トラック長（配線供給）を越えてしまい局所配線混雑となり，配線できないネットが発生してしまう．routabilityの高い配置結果を得るためにはチップ上の各領域において配線の需要量が供給量に見合うようにする必要がある．

一般に配置手法は構成的手法と配置改善法に分けられる．構成的手法では局所的な配線の混雑を考慮することは難しい．なぜなら局所的な配線の需要と供給を見積もるためにはセルの配置位置や配線の経路をある程度詳細に知る必要があるからである．従来の構成的手法[1][2][3]は分割ベースの手法であり，最初にセルの概略の位置を決め次第に詳細な位置を決めていくので，末期段階になるまで詳細な配置位置の情報が得られない．そこで従来の構成的配置手法は総配線長[1]やカット数[2]などのより大局的な配置の目的関数を用いていた．これに対し，配置改善手法では初期配置結果が与えられているので局所的な混雑を検出することが容易である．従って従来，局所混雑を考慮した配置改善法がいくつか発表されている[4][5][6][7].

しかし文献[4][5]の手法はチャネルスタイルの小規模のレイアウトをターゲットにしたものであり，今日の大規模なゲートアレイには適さない．これらの手法は，各セル列内でセルを移動してチャネル密度（チャネル内の水平方向の配線の混雑）を最小化する手法であり垂直方向の混雑を考慮しないからである．文献[6][7]の手法は両方向の配線混雑を考慮した目的関数を用いている．これらの手法の目的関数は，チップを格子状に分割した各分割領域について求めた水平方向及び垂直方向の配線需要量と供給量に基づく．しかしこれらは反復改善法（simulated annealingを含む）を用いており，大規模LSIに適用した場合には非常に多くの処理時間がかかってしまい実用的でないと思われる．

大規模なLSIに対して処理時間がかかることには次の2つの原因がある．

(1)規模の大きな局所混雑： 大規模なLSIでは大規模な（混雑した範囲の広い）局所混雑も起こる．論理合成された回路ブロックなどは比較的規模のおおきい局所混雑を引き起こす．このような混雑を数セルの移動の繰り返しで解消するには非常に多くの反復が必要である．なぜなら，反復改善法は少数のセルを移動する操作の繰り返しによる手法であり，個々のセル移動操作はチップ全体の大局的な混雑の分布を考慮せずに行なわれるからである．このような手法は微細な局所混雑を緩和するのには適しているが大規模な混雑を緩和するためには多くの反復が必要で非常に時間がかかってしまう．

(2) 1回の反復あたりの改善度が小さい： 反復改善法では一回の反復毎に行なわれる目的関数の計算にかかる時間が全体の時間のほとんどを占める．そこで処理時間を短縮するためには，セルの移動操作において，目的関数値を大きく減少させるようなセルの移動を高い頻度で行なえるようにする必要がある．しかし従来の手法[6]の移動操作は，局所混雑を考慮した目的関数を大きく減少させるような移動先を求めていない．この移動操作は，総配線長の最小化を目的とする昔の反復改善法における移動操作[8]をそのまま用いている．これは総配線長の減少量がなるべく大きくなるようにセルの移動先を決めるもので，混雑を考慮した上記の目的関数値が小さくなるような移動先を必ずしも求めることができない．

本論文では，上記の2つの問題を解決する高速な配置改善手法を発表する．この手法は，文献[7]の目的関数に基づいており，構成的配置手法による配置結果を改善してroutabilityの高い配置を得る．改善処理は"padding"処理と高速な反復改善処理の2段階からなる．padding処理は混雑の分布を大局的に均一化するので，規模の大きな混雑を緩和することができる．また反復改善処理はより微細な混雑を緩和する．反復改善処理は，セルの移動操作を工夫して高速に配置を改善できるようにした．

配線混雑領域内のセルのサイズを仮想的に大きくして("padding"という)から，paddingによって生じたセルの重なりを解消するために全てのセルをもう一度配置し直すというテクニックは一般的である．しかしセルのpadding量を決める手法についての発表は筆者の知る限り無い．再配置後の混雑が緩和されるか否かはpadding量の決め方に大きく左右される．筆者は，与えられた初期配置結果における混雑の度合の分布に基づいてチップ上の各地域のセルのpadding量をうまく決める手法を開発した．この手法は，移動後もセルの相対的な配置位置があまり変わらないという仮定のもとで移動後の混雑度を予測しながら，この混雑予測値がチップ全体でなるべく均一になるようにpadding量を決める．われわれはこのpadding量の決定問題を最適化問題として定式化し，simulated annealingを用いて解くようにした．これにより移動後の混雑をチップ全体でうまく均一化することができ，大規模な局所混雑も容易に緩和することができる．

反復改善処理を高速化するために新しいセル移動操作を開発した．この方法は，セルの移動による目的関数の減少量を見積もり，この見積もり値が大きくなるようにセルの移動先を決定する．あるネットの配線長

の変化が目的関数値に与える影響の大きさ(sensitivity)は，その配線が通る分割領域の混雑度の大きさに依存する．従来手法[6]で用いられているような総配線長を最小化する移動先決定手法は，ネット毎のsensitivityの違いを無視している．われわれは，配線が通る領域の混雑度からネットのsensitivityを求めこれを重みとする配線長の変化量の総和を用いて目的関数の変化量を見積もるので，より正確な見積もりが可能である．

発表する手法を実装して大規模なゲートアレイの回路とMCNCベンチマーク回路に適用して評価し，我々の手法の有効性を検証した．以下では，2章で配置改善問題の定式化を行ない，3章で配置改善手法について述べ，4章で実験結果を述べる．

2 配置改善問題

発表する手法は，min-cut配置法などの構成的配置手法によるセルの配置結果を改善するために用いる手法であり，row-basedのSOGを対象とする．この手法が扱う配置改善問題はマスタスライス上にあらかじめ配置されたセルの移動先を決定する問題である．ただし，セルにはメガセル，埋め込みセル(embedded block)，I/Oセル等の自由な大きさを持つセルと一定の高さを持つ等高セル(uni-height cell)があり，移動の対象となるのは等高セルのみとする．マスタスライス上にはセルベンチ（セル列領域）が設けられており等高セルはセルベンチ内に配置されなければならない（Fig. 1）．水平／垂直の配線トラックがマスタスライス上に設けられており，セル間を接続する配線はトラック上を通らなければならない．マスタスライス上に存在する電源幹線などの既配線パターンは，セル間を接続する配線の障害物である．他に，セル内のトランジスタ間を結ぶ既配線パタンも配線の障害物となる．

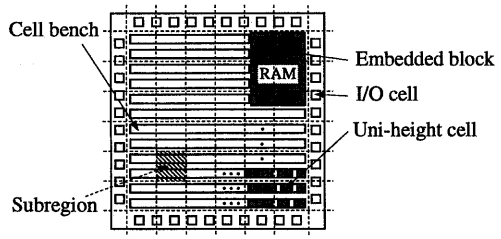

Fig. 1 Cell placement model

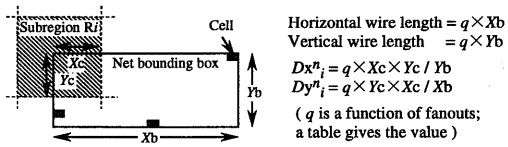

Fig.2 Definition of net routing demands Dx^n_i and Dy^n_i

配線の局所混雑を検出するために，マスタスライスを水平及び垂直の直線で分割する．分割によってできた各部分領域R_i (i=1,2,...,m)を分割領域(subregion)と呼ぶ．各分割領域に対して水平方向及び垂直方向の混雑度Cx_i及びCy_iを以下のように定義する．

$$Cx_i = (Dx_i - Sx_i) / Tx_i, \quad (1)$$
$$Cy_i = (Dy_i - Sy_i) / Ty_i, \quad (2)$$

ここで，$Dx_i(Dy_i)$は領域R_iの水平（垂直）方向の配線リソース需要量（領域R_i内を通ると予想される配線の水平（垂直）成分の長さの和）であり，$Sx_i(Sy_i)$は領域R_iの水平（垂直）方向の配線リソース供給量（領域R_i内の水平（垂直）方向の配線トラックのうちセル内の配線パタンや電源幹線などの配線障害物におおわれていない部分の長さの総和），$Tx_i(Ty_i)$は領域R_i内の水平（垂直）方向の配線トラックの長さの総和である．需要量$Dx_i(Dy_i)$は各ネットnの領域R_iに対する配線需要量$Dx^n_i(Dy^n_i)$の総和として表わされる．

$$Dx_i = \sum_{\text{all nets } n} Dx^n_i, \qquad Dy_i = \sum_{\text{all nets } n} Dy^n_i.$$

ネット毎の需要量$Dx^n_i(Dy^n_i)$の定義は文献[7]に従う．これは，各ネットを実現する配線はそのネットに繋がるセルを囲む最小矩形(bounding box)内の任意の場所を等確率で通るという仮定に基づく．この定義をFig. 2に示す．ここで係数qは，同一形状のbounding boxをもつネットではファンアウト数が多いネット程ほど実際の配線長は長くなる傾向にあるという現象に対応するための補正係数であり，ファンアウト数毎にあらかじめ統計的に求めたqの値を用いる．

配置改善問題の目的関数Eを文献[7]に倣って次のように定義する．

$$E = \sum_{i=1}^{m} Tx_i f(Cx_i) + \sum_{i=1}^{m} Ty_i f(Cy_i). \quad (3)$$

上式の各項にかかる係数Tx_i, Ty_iは，各分割領域(subregion)の大きさに比例した重み付けを行なうために設けられている．関数$f(x)$は下に凸なる単調増加関数である（例えば，$(\max\{x, 0\})^2$）．このため，混雑度の高い分割領域内の配線長の減少は混雑度の低い領域内の配線長の減少に比べて，両者で減少量が同一であっても目的関数値を大きく減らす．このことは混雑した地域を通るネットは混雑していない地域を通るネットに比べて，配線長の変化が目的関数値に与える影響(sensitivity)がより大きいことを意味しており，このことが後で述べる目的関数値の見積もり手法の基になっている．

我々の改善手法の２つの段階（padding処理と反復改善処理）は両方ともこの目的関数Eを用いる．ただし，高速に大局的な改善を行うことを目的とする前者では分割領域を粗くとり（１分割領域が含むセル数が50か

ら100），微細な混雑の解消を目的とする後者ではより細かくとる（1分割領域が含むセルが数個）．

3 配置改善手法

3.1 全体の流れ

我々の配置改善処理の全体フローをFig.3に示す．まず各分割領域(subregion)について混雑度を計算し，これに基づいて各セルのpadding量を決定する．padding量決定手法は3.2節で述べる．次にpaddingによって生じたセルの重なりを解消するために，なるべくセルの相対的な配置位置が保たれるようにセルを再配置する．再配置のアルゴリズムは線形割り当てによる手法[9]をベースにした手法を用いる．この手法は各セルの移動距離の2乗の総和を最小にするような重なりのない配置結果を求める．

次に反復改善処理を実施する．セルの移動操作を実施して目的関数Eが減少したときのみその移動を受け入れるという処理をN_I回繰り返す（N_Iはパラメータ）．移動操作は，移動後の目的関数の減少値を見積もりながらセルの移動先を決める．3.3節で移動操作を，3.4節で見積もり方法を説明する．

3.2 セルのpadding手法

まず，セルのpadding量を決める問題を制約付きの最適化問題として定式化する．各分割領域R_i内のセルのサイズを一律の割合s_iでpaddingすることにする．すなわち領域R_i内のセルはpadding後にはもとのサイズのs_i倍となる．本最適化問題において決定すべき変数は各分割領域のpadding率s_iである．

padding率s_iの満たすべき制約は2つある．1つめの制約式は自明のものである．

$$s_i \geqq 1. \tag{4}$$

padding後のセルサイズの総和は，padding後に重なりを解消するアルゴリズムが扱えるようにするためにチップ上のセルベンチのサイズの総和Aより若干小さくなるようにする必要がある．これが2つめの制約式になる．

$$\sum_i s_i K_i = \alpha A, \tag{5}$$

ここでK_iは分割領域R_i内のセルの実サイズの総和，αは定数である（0.96程度に設定した）．

本問題の目的は混雑度が再配置後にチップ全体でなるべく均一となるようにpadding率を決めることである．このため，われわれは各領域のpadding率s_iから再配置後のその領域の混雑度を推測する式を立てた．この推測には，再配置後もセルの相対的位置があまり変わらないことを利用する．相対的位置が保たれているならば，R_i内のセルは再配置後にもある領域内に固まって配置されていると考えられる．この領域をR^*_iとする（Fig. 4）．以下では，padding率s_iを用いて再配置後の分割領域R^*_iの総トラック長T^*x_iとT^*y_i，供給量S^*x_iとS^*y_i，需要量D^*x_iとD^*y_i，そして混雑度C^*x_iとC^*y_iを予測する式を立てる．なおここでは水平方向だけについて述べるが垂直方向も同様に議論できる．

再配置のアルゴリズムはpadding後サイズでのセル密度がだいたい均一になるように配置するので，分割領域R^*_i内のセルベンチのサイズは，$s_i K_i / \alpha$となる．R_i内のセルベンチのサイズをA_iとする．R^*_iとR_iの面積比は，R^*_iとR_iの内部のセルベンチのサイズの比に等しいと考えられるので，padding率s_iを用いて以下のように書ける．

$$\begin{aligned} R^*_i \text{ の面積} / R_i \text{ の面積} &= (s_i K_i / \alpha) / A_i \\ &= B_i s_i, \\ & \quad (B_i = (K_i / \alpha) / A_i). \end{aligned}$$

以下では，この面積比B_i s_iを用いてT^*x_i，S^*x_i，D^*x_i，C^*x_iを見積もる．

配線トラックがR_iとR^*_iの内部で一様に引かれていると考えると，総トラック長T^*x_iを以下のように推測できる．

$$T^*x_i = Tx_i \cdot (B_i s_i). \tag{6}$$

```
procedure PlacementImprovement()
begin
    /* padding処理 */
    各分割領域 Riについて混雑度を計算する ;
    全てのセルのサイズを仮想的に大きくする (padding) ;
    paddingによって生じたセルの重なりを解消するために、相対的配置位置をなる
    べく変えないように全セルを再配置する ;
    /* 反復改善処理 */
    while 反復回数 ≦ NI do
        セルの移動操作を実施する ;
        目的関数値の変化 ΔE を計算する;
        if ΔE > 0 then セルを移動前の位置に戻す ;
    end
end
```

Fig.3 Placement Improvement Algorithm

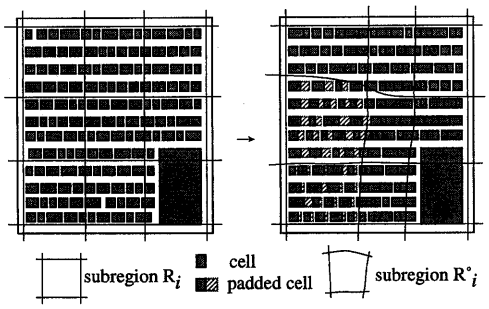

Fig.4 Cell Padding

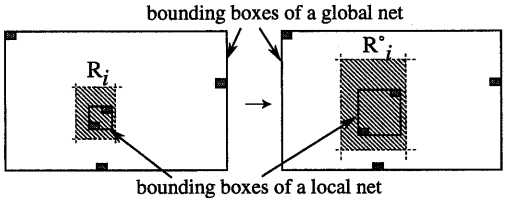

Fig.5 Local net and global net

チップ上に固定された配線障害物の分布がR_iとR^*_iの内部において一様であると考え，供給量S^*x_iを以下のように見積もる．

$$S^*x_i = T^*x_i - F^*x_i - M^*x_i$$
$$= Tx_i \cdot (B_i s_i) - Fx_i \cdot (B_i s_i) - Mx_i \qquad (7)$$

ここで，Mx_iは分割領域R_i内の配線トラックのうち等高セルの内部のパタン（配線障害物）が覆っている部分の長さであり，Bx_iはそれ以外の固定された配線障害物が覆っている部分の長さである．

次に需要量D^*x_iを予測する．分割領域(subregion)R_i内のセル同士を結ぶネットをローカルネットと呼び，bounding boxがその領域R_iをスッポリ含むようなネットをグローバルネットと呼ぶことにする（Fig.5）．再配置後も分割領域内のセルの相対的配置位置が変わらないとすれば，ローカルネットのbounding boxの大きさは分割領域R_iの大きさの変化に比例して変わる（Fig.5）．グローバルネットのbounding boxは領域R_iの大きさの変化にはあまり依存せずに大きくなったり小さくなったりするので，平均的にみて大きさに変化がないものと見なしてよい（Fig.5）．ネット毎の配線需要量の定義（Fig.2）をこれらに適用することにより，ローカルネットとグローバルネットそれぞれの需要量を見積もる以下の式を得る．

$$D^*x^n_i = Dx^n_i \cdot \sqrt{B_i s_i}, \qquad \text{for local net } n,$$
$$D^*x^n_i = Dx^n_i \cdot (B_i s_i), \qquad \text{for global net } n.$$

われわれは単純に，配線需要量Dx_iのうちβDx_iがローカルネットによる部分で，$(1-\beta)Dx_i$がグローバルネットによるものであるとした（パラメータβは実験的に求めた値$\beta=0.5$と設定した）．

$$D^*x_i = \beta Dx_i \cdot \sqrt{B_i s_i} + (1-\beta) Dx_i \cdot (B_i s_i) \qquad (8)$$

以上の式(6)-(8)を混雑度の定義式に代入することにより以下の混雑度予測式が導出できる．

$$C^*x_i = (D^*x_i - S^*x_i) / T^*x_i$$
$$= \frac{Mx_i}{Tx_i B_i} \frac{1}{s_i} + \frac{\beta Dx_i}{Tx_i \sqrt{B_i}} \frac{1}{\sqrt{s_i}} + \frac{(1-\beta)Dx_i - Tx_i + Fx_i}{Tx_i} \qquad (9)$$

本問題の目的関数E_Cは配置の目的関数E（式3）において混雑度Cx_i, Cy_iと総トラック長Tx_i, Ty_iをそれぞれの予測値C^*x_i, C^*y_iとT^*x_i, T^*y_iでそれぞれ置き換えたものと定義する．

$$E_C = \sum_i T^*x_i f(C^*x_i) + \sum_i T^*y_i f(C^*y_i). \qquad (10)$$

以上により，padding率s_iを決める処理は，制約式(4)-(5)のもとで目的関数(10)を最小化する非線形計画問題として定式化された．われわれの処理はこれをsimulated annealingを用いて解く．

3.3 反復改善処理の移動操作

局所混雑を考慮した配置の目的関数Eの計算には非常に多くの時間ががかかる．これはあるネットにつながるセルが移動する度に，そのセルにつながる各ネットのbounding boxと重なる全ての分割領域について混雑度を計算し直さなければならないからである．我々の経験によると２万セル規模の回路では１ネットのbounding boxと重なる分割領域の数は平均で約50個にもなる．よって反復改善処理を高速化するためには目的関数値の変化ΔEの計算回数（即ち反復回数）が少なくても十分な改善ができるようにする必要がある．このためにはセルの移動操作が，目的関数値を大きく減少させるセルの移動を高い確率で生成する必要がある．筆者は，ある１つのセルがある場所に移動したときの目的関数値の減少量（ゲイン）$-\Delta E$を見積もる手法を開発しこれを用いた移動操作を開発した．この見積もり手法については次の節で説明することにし，ここでは移動操作の全体フローを説明する．

開発した移動操作はセルの位置交換に基づく．基本的な考え方は，まず１つのセルを選びこのセルの交換相手となるセルの候補をたくさん設定し，そのなかから交換によるゲインの見積もり量が最大となるような交換を選んで実施することである．Fig.4に移動操作のフローを示す．移動操作は，まず１セル（順移動セルと呼ぶ）を選んで移動先を決め，次に移動先で順移動セルと重なるセルの内幾つかを選んで（逆移動セルと呼ぶ）順移動セルと逆方向に同じ距離だけ移動するものである．このとき交換後にそれぞれのセルベンチ内のセルサイズの総和がセルベンチサイズを越えないよ

うにする．セルの交換の後，セルの重なりを解消するためにセル列内でセルを最小限の距離だけずらす．

なお候補点数Npは70とした．次の節で述べるように開発したゲインの見積もり手法は，移動したセルに繋がる各ネットについて，そのbounding boxと重なる全ての分割領域について混雑度を計算し直す必要はなく，そのネットの配線長の変化を計算するだけでよいので非常に高速である．よって候補数をこのように多くとっても移動操作の処理時間は目的関数の更新計算にかかる時間に比べて小さい．実験によれば，約3万ネットの回路で移動操作の処理時間は目的関数の変化ΔEの計算時間の23%であった．

3.4 ゲインの見積もり手法

本節では，あるセルcがX軸方向にΔX，Y軸方向にΔYだけ移動したときの目的関数Eの減少量（ゲイン）$-\Delta E$を見積もる手法について述べる．以下では記述の繁雑さを避けるため目的関数E（式3）の水平方向の成分Exだけについて説明する．垂直方向についてもまったく同様に議論できる．Exの定義式を示す．

$$Ex = \sum_i Tx_i f(Cx_i), \quad \text{where } Cx_i = (\sum_n Dx^n{}_i - Sx_i) / Tx_i.$$

セルcが動いたときの配線需要量$Dx^n{}_i$と供給量Sx_iそれぞれの変化量を$\Delta Dx^n{}_i$とΔSx_iと書く．Exの変化量ΔExを$\Delta Dx^n{}_i$とΔSx_iを変数としてテイラー展開した一次の項をとると次のようになる．

$$\Delta Ex \fallingdotseq \sum_n \sum_i f'(Cx_i)\, \Delta Dx^n{}_i - \sum_i f'(Cx_i)\, \Delta Sx_i. \quad (11)$$

この式の第1項は配線需要量の変化による部分であり第2項は配線供給量の変化による部分である．第1項は，移動したセルcに繋がる各ネットについてそのネット通る分割領域R_j全てについて和をとる必要がある（それ以外では$\Delta Dx^n{}_i=0$だから計算の必要はない）．第2項は，セルcが移動前あるいは移動後に重なる分割領域についてのみ和をとればよい（それ以外では$\Delta Sx_i=0$だから）．よって式11は以下のように書き直すことができる．

$$\Delta Ex \fallingdotseq \sum_{n\in Nc} \sum_{i\in In} f'(Cx_i)\, \Delta Dx^n{}_i - \sum_{i\in Ic} f'(Cx_i)\, \Delta Sx_i. \quad (12)$$

ここでN_cはセルcがつながる全てのネットの集合，I_nはネットnのbounding boxとセルの移動前あるいは移動後に重なりをもつ全ての分割領域の集合，I_cはセルcと移動前または移動後に重なりをもつ全ての分割領域の集合である．集合I_cのメンバ数は小さいから第2項は高速に計算できる．われわれが高速化すべきは需要量による第1項である．これをΔEx^1と書く．

$$\Delta Ex^1 = \sum_{n\in Nc} \sum_{i\in In} f'(Cx_i)\, \Delta Dx^n{}_i, \quad (13)$$

ΔEx^1は各ネットnに依存する部分の和として表わされており，ネットnに依存する部分は，ネットnの領域R_jに対する配線需要量の変化$\Delta Dx^n{}_i$を，$f'(Cx_i)$で重み付けして足し合わせたものである．重み$f'(Cx_i)$は領域の混雑度Cx_iが大きいほど重い（fが下に凸で単調増加だからf'は正値単調増加である）．ネットによって混雑した地域を通るもの通らないものがある．われわれは，各ネット毎にネットnの通る領域での重み$f'(Cx_i)$の平均値wx_nをとった．このwx_nは，ネットnの配線長変化が目的関数値に与える影響の大きさ(sensitivity)という意味を持つ．式13で$\Delta Dx^n{}_i$の重み$f'(Cx_i)$をwx_nで置き換えた式によりΔEx^1を近似する．

$$\Delta Ex^1 \fallingdotseq \sum_{n\in Nc} \sum_{i\in In} wx_n\, \Delta Dx^n{}_i = \sum_{n\in Nc} wx_n\, \Delta Lx_n, \ (\Delta Lx_n = \sum_{i\in In} \Delta Dx^n{}_i), \quad (14)$$

```
    procedure moveOperation()
    begin
1.      セル c をランダムに1つ選び順移動セルとする；
2.      セルcを中心とする幅W,高さHの矩形領域内に移動先候補点をランダムにNp点とり，
        集合Pcとする（W, Hはパラメータ）；
3.      foreach p ∈ Pc do
            /* 逆移動セルを選択する */
4.          Gc = セルcが候補点pに移動したときの見積もりゲイン；
5.          So = セルcが移動先pにおいて重なる全てのセルの集合；
6.          foreach s ∈ So do Gs = セルsが逆移動したときの見積もりゲイン；
7.          So のメンバsをゲインGs で降順に整列する → So = { s1, s2, s3, ...., sn }；
8.          Soのうち利得Gsが大きいものから順にk個を選び，逆移動セルとする．ここで
            k ( 0 <= k <= n )は，両方のセルベンチでサイズ超過がおこらない範囲で，交換に
            よる推定利得Gp=Gc+Gs0+...+Gsk が最大になるようにとる；
        end
9.      if どの候補点pについてもサイズ超過が起こらない逆移動セルをとれなかった then
            goto line 1;
10.     交換による見積もりゲインGpが最大となる候補点を選び、セルの交換を実施する；
11.     セルをずらして重なりを解消する；
    end
```

Fig.6 移動操作

ここで，Lx_nはネットnの水平方向配線長，即ち$\Delta Lx_n = \Sigma \Delta Dx^n_i$である．これによりbounding boxに重なる分割領域を１つ１つ参照する必要がなくなるので計算を高速化ができる．重みwx_nは具体的にはセル移動前のネットnのbounding box内の$f'(Cx_i)$の平均値とする（後で述べるようにセルはその近傍にしか動かさないから，I_nに属する分割領域はこのbounding boxとその近傍領域内に限られるから，このようにとってもあまり大きな誤差はないと考える）．

$$wx_n = (\Sigma_{i \in In} f'(Cx_i)\, Dx^n_i) / (\Sigma_{i \in In} Dx^n_i) \quad (15)$$

ゲインの見積もり量Gxを(12)の第１項を(14)で置き換えたものをと定義する．

$$Gx = -(\sum_{n \in Nc} wx_n \Delta Lx_n - \sum_{i \in In} f'(Cx_i)\, \Delta Sx_i) \quad (16)$$

反復改善処理において，重みwx_nの初期計算・更新計算は以下のようにおこなう．なお以下での重みwx_nの計算はすべて式15により計算する．

(a)反復改善処理を始めるまえに全ネットについてwx_nを計算する．

(b)セルの移動操作が受け入れられたとき，移動したセルに繋がる全ネットについてwx_nを再計算する．

(c)N回（Nは回路のネット数×２程度に設定）の反復を行う毎に，全ネットについてwx_nを再計算する．

ここで，(b)においては本当は混雑度Cx_iが変化した領域R_iと，そのbounding boxが重なりをもつようなすべてのネットnについてwx_nを更新する必要がある．しかしこれには処理時間がかかるので(c)の定期的な更新処理で代用している．

重み更新処理(a)-(c)の中では明らかに(b)の処理時間が支配的である．しかし(b)の処理時間は，セルの移動操作が試みられる度に実施する必要のある目的関数の変化の計算に要する時間よりも小さい．なぜなら，(b)の処理と目的関数の変化計算の処理とは両方とも，移動したセルにつながる各ネットについてそのbounding boxと重なりをもつ全ての分割領域(subregion)の配線需要量を参照する計算処理であるが，(b)の処理は移動操作が受け入れられたときのみ（実験によれば受け入れ率は１／５程度）行えばよいからである．

4 実験

開発した手法をゲートアレイ用配置配線システムHGALOP[10]の一部として実装し以下の５つの実験を行った．(a) 発表した配置改善手法の効果を示すための実験，(b)padding処理の効果を示すための実験，(c) 発表したゲイン見積もり手法の効果を示すための実験，(d) 市販の配置ツールとの比較実験，(e) MCNCのベンチマークデータに対する実験．以下でそれぞれについて述べる．(a)-(d)の実験には，ゲートアレイ用の中規模から大規模の５つの回路（Table 1）を用いた．全ての実験において，改善処理を適用する初期配置結果は，min-cut法によるHGALOPの配置プログラム[10]で生成したものを用いた．配置結果のroutabilityを評価するために，われわれは実際より少し小さめのマスタを用いて配置処理を行い，それぞれの配置結果に対して配線処理を実行し結線できなかったネットの数を数えた．実験は全てSPARC Station 20 で行った．なお，表の中でINITは初期配置処理を，PADはpadding処理を，ITは反復改善処理を表す．

(a) 初期配置結果(INIT)とそれに対する配置改善手法の適用結果(INIT+PAD+IT)とを比較した．結果をTable 1に示す．全５回路について改善処理を適用した結果のほうが配線不能ネット数が少なくなっており，routabilityが向上したと言える．

(b) padding処理の有効性を示すため，padding処理の無い改善処理の結果(INIT+IT)とpadding処理のある改善処理の結果(INIT+PAD+IT)を比較した．結果をTable 1に示す．4回路についてpadding処理を行なった場合のほうが配線不能数は少なくなっており，残りの１回路も不能数は同等である．開発したpadding処理が

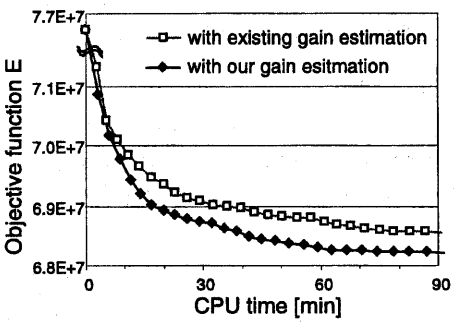

Fig. 7 The results of iterative improvement

Table 1. Results on industrial SOG circuits (CPU time in minutes)

circuits	no. of cells	no. of nets	no. of pins	master size (raw-gates)	INIT no. of unconnects	INIT CPU (INIT)	INIT+IT no. of unconnects	INIT+IT CPU (IT)	INIT+PAD+IT no. of unconnects	INIT+PAD+IT CPU (PAD)	INIT+PAD+IT CPU (IT)
sog1	3,381	4,370	13,635	10K	908	0.8	838	3.2	424	1.3	3.1
sog2	6,033	6,043	15,582	20K	8	2.9	4	5.3	0	1.5	5.0
sog3	8,429	9,626	31,716	20K	43	2.0	16	7.1	3	1.5	7.3
sog4	17,226	32,084	82,699	400K	990	6.4	238	10.5	238	6.4	10.2
sog5	20,836	28,418	84,571	400K	175	6.4	258	18.0	103	9.0	18.0

routabilityを高めるために有効であるといえる．

(c) 開発したゲイン見積もり手法の有効性を示すため，開発したゲイン見積もり手法と従来のゲイン見積もり手法（総配線長の減少量で見積もる手法[6]）とを比較した．両方の手法を3.4節で述べたような方法で移動操作に組み込んで両方の反復改善処理を実行した．ゲートアレイ用の回路sog1の初期配置結果に対して両者を実施し，目的関数値EとCPU時間で比較した(Fig.7)．開発した見積もり手法を用いた方が，同一の目的関数値（例えば6.9E+7）に達するまでにかかるCPU時間が約50%短い．開発したゲイン見積もり手法は反復改善処理の時間を短縮するのに有効であるといえる．

(d) 5回路の内規模の大きい2回路について市販の配置ツールとの比較を行った．結果をTable 2に示す．初期配置結果に配置改善手法を適用した結果は，市販ツールによる配置結果に比べて両方の回路で配線不能数が少ない．初期配置を含めた我々の手法の処理時間は平均で市販ツールの処理時間の3割増以内である．

(e) 開発した手法をMCNCのベンチマーク回路[11]の内の2回路に適用した．このベンチマークデータはセルベース用配置配線手法を評価するためのデータである．開発した手法はゲートアレイ用であるので，様々なサイズのマスタスライスで配置配線を実行し配線不能が起こらなかった最も小さいマスタスライスの結果を採った．結果をTable 3に示す．この回路に対する様々な配置手法の適用結果が学会で発表されている．そのうち比較的最近発表された配置手法(GORDIANL&DOMINO)の結果を文献[12]から引き写してTable 3に示す．両者は異なる配線ツールを用いているのでこの比較はあまり厳密とはいえないが，配線後のレイアウト面積で見て，我々の手法はGORDIANL&DOMINOと同等の結果を生成している．

5 まとめ

配置のroutabilityの向上を目的とするゲートアレイ用の配置改善手法を開発した．本手法はpaddingフェーズと高速な反復改善フェーズの2段階からなる．padding処理は，チップ上の各領域のpadding量から再配置後のその領域の混雑度を見積もりながらうまくpadding量を決めるので，再配置後の混雑度をチップ上でより平坦化できる．また反復改善処理は，セルが移動したときの目的関数値の減少量（ゲイン）を高速に見積もりながらセルの移動先を決めるのでより短い時間で配置を改善することができる．開発した手法を約3,000～20,000セル規模の回路に適用した結果，配置のroutabilityを大きく改善することができた．

参考文献

[1] R.-S.Tsay, et.al., "PROUD: A sea-of-gates placement algorithm," IEEE Design & Test of computers, 1988, pp.44-56.

[2] M.A.Breuer, "Min-cut placement", J. Design and Fault Tolerant comput., vol.1, no.4, pp.343-362, 1977.

[3] S.Mayrhofer, et.al., "Congestion-driven placement using a new multi-partitioning heuristic," ICCAD, pp.332-335, 1990.

[4] G.Persky, "PRO - an automatic string placement program for polycell layout," DAC, 1976, pp.417-424.

[5] H.Shiraishi, et.al., "Efficient placement and routing techniques for master slice LSI," 17th DAC, 1980, pp.458-464.

[6] J.Jung, S.Goto and H.Hirayama, "A new approach to the two-dimeinsional placement problem with wire-congestion in master-slice LSI layout design," Trans. IECE vol.J64-A, No.1, pp.55-62, 1981.

[7] C.-L.E.Cheng, "RISA: Accurate and efficient placement routability modeling," ICCAD, 1994, pp.690-695.

[8] S.Goto, "An efficient algorithm for two-dimensional placement problem in electrical circuit layout," Proc. 1979 ISCAS, pp.850-853.

[9] N.R.Quinn, Jr., et.al., "A force directed component placement procedure for printed circuit boards," IEEE Trans. Circuits Syst.,Vol.CAS-26, pp.377-388, 1979.

[10] K.Takahasi, et.al., "Adaptive cut line selection in min-cut placement for large scale sea-of-gates arrays," ICCAD, 1994, pp.428-431.

[11] K.Kozminski, "Benchmarks for layout synthesis," 28 th DAC, pp.265-270, 1991.

[12] K.Doll, et.al., "Iterative placement improvement by network flow methods," IEEE Trans. CAD, Vol.13, no.10, pp.1189-1200, 1994.

Table 2. Comparison with tool A

circuits	INIT		INIT+PAD+IT		tool A	
	no. of unconnects	CPU [min]	no. of unconnects	CPU [min]	no. of unconnects	CPU [min]
sog4*	137	6.4	0	38.8	74	35.4
sog5*	26	6.4	0	49.7	8,413	37.9

* Using different size of masterslice from the one in Table 1.

Table 3. Results on MCNC benchmark data

circuits	no. of cells	no. of nets	no. of pins	INIT+PAD+IT				GORDIANL&DOMINO II (from [12])		
				no. of unconnects	CPU* [min]	area [mm^2]	wire length [m]	no. of unconnects	CPU** [min]	area [mm^2]
industry2	12,142	13,419	48,404	0	45.9	214.36	14.96	0	159.8	214.25
industry3	15,032	21,940	68,290	0	36.6	573.39	28.44	0	172.5	575.54

* on SPARC Station 20 (204.7MIPS) ** on DEC station 5000/200