遺伝的アルゴリズムによる素子の整列配置

正員 半田 恵一 (東芝)
正員 本位田 真一 (東芝)

Cell Placement by Genetic Algorithm
Keiichi HANDA, Member, Shinichi HONIDEN, Member (Toshiba Corporation)

Cell layout in analog LSI chip designs entails a lot of strict constraints, and its complete automation is known to be a very difficult problem to solve. The layout in analog LSI designs usually has the same relative position of elements as in the circuit schematic. Given this fact, in this paper, we consider the placement problem to minimize the overall net length and the used region area with the constraints of neighbourhood-relation of elements. We propose two ideas based on genetic algorithms; (1) a method to scan the placement tasks of small blocks in overlapping stages, (2) a method to line up several elements along the x-direction gradually by stochastic adjustment of y-coordinates. Computational experiments show the efficiency of these methods.

キーワード: アナログLSI, 配置問題, 遺伝的アルゴリズム, 整列配置

1. まえがき

アナログLSIのレイアウト設計は、対話型CADの導入により設計工数の削減がはかられている(例えば文献⁽¹¹⁾)。しかし、設計工程の中の素子の配置・配線の作業は、デジタルLSIの場合と違い、完全な自動化が困難な分野として知られている。アナログにおいて自動化が難しいのは、

- (1) 様々な形状の素子を出来るだけコンパクトに配置配線したいので、デジタルにおけるスタンダードセルのような単純なレイアウトモデルが使えない、
- (2) 素子の隣接や、配線の離反に関する制約が厳しい、
- (3) 配置処理と配線処理は相互に密接な関係があり、実用上、分離できない

などの理由による。配置・配線アルゴリズムでは、(2)の制約条件を満たし、且つ面積が最小となるような配置・配線結果を得る必要がある。配置と配線を同時に進行させる自動化技術は非常に困難であるため、通常、配置問題と配線問題は分離して扱われ、配置において出来るだけ配線のし易さを考慮するという方法が取られる。また、アナログLSIの場合、回路図上の素子の相対的な位置関係が配置に関する制約をかなり表現しているという事実から、回路図上の信号の流れに沿った配置手法が採用されることが多い^(8,9)。本論文ではこのような背景をもと

に、回路図上の素子の相対的な位置関係をなるべく保存しつつ、且つ素子の隣接性に関する制約(ペア性)を出来るだけ満足し、更に仮想配線長と配置面積を最小化するという問題を扱う。また配線のし易さの1つの尺度と考えられる素子の整列性の問題を扱う。

仮想配線長や配置面積の最小化を目的とする配置問題は、2次割当問題や2次元ビンパッキング問題に抽象化されるNP完全/困難な問題を内蔵している⁽²⁾。この種の配置問題に対する解法として、近年、シミュレーテッド・アニーリング法(SA法)⁽⁵⁾や遺伝的アルゴリズム(GA)^(3,4)などの確率的手法が効率の良い汎用的手法として注目されている。例えば、文献⁽⁹⁾は本論文と同種の問題にSA法を適用したものであり、文献^(1,6)はそれぞれ2次元ビンパッキング問題、2次割当問題にGAを適用したものである。また、文献⁽¹⁰⁾はSA法とGAを統合した手法をブロック配置問題に応用し、従来のSA法よりも良好な結果を与えている。

GAは生物の進化過程に着想を得た最適化手法であり、解候補(個体)の集団に「淘汰・交叉・突然変異」を模倣した操作を繰り返して適用することによって、より良い解候補を探索する手法である。筆者らはGAの確率的操作による集団進化の効率性、及びアルゴリズムとしての汎用性/柔軟性に注目し、先に述べた配置問題に対する基

本手法として GA を採用することにした。

本論文では、まず GA の適用方法とその実験について述べ、特に GA による実験結果が SA 法と比較して良い結果を導いていることを示す。次に GA の集団進化の特徴を生かした手法として、

- 小さな部分回路毎に、一部を重複させ順次、配置を決定する方法、及び
- 確率的な y 座標調整により素子を横方向 (x 軸方向) に整列配置する方法

を提案する。実験結果によれば、一般に、小さな部分回路毎に一部を重複させながら配置していく方法は、重複させない場合あるいは回路全体を配置対象とする場合に比べて、精度の良い解が短い時間で求められる。また確率的な座標調整の導入は、面積の縮小やペー性を高める効果があり、全体的に見た目の良い配置結果を与えることが示される。

2. GA の適用

GA を適用するにあたって、まず問題の特性に合った解候補のコード化 (染色体表現) と、染色体間の交叉・突然変異の方法、及び解候補を評価するための適応度関数の設定が必要である。配置問題における解候補とは、全素子の x, y 座標が決定している 1 つの配置状態 (結果) を指す。以下では、染色体で表現された解候補を個体と呼ぶ。

< 2.1 > 解候補のコード化 回路図において、電源からグラウンドに至る電流路をブランチと呼ぶ⁽⁹⁾。例えば、図 6 の回路図において R1, Q1, R2, Q2, R3 は 1 つのブランチを形成している。

回路図におけるブランチ内の素子の上下関係やブランチの左右関係は、レイアウト結果においても保存されることが多い⁽⁹⁾。そこで本稿では、解候補がこのような (保存) 性質を持つことを仮定し、ブランチ毎に素子の基準点 (素子の外枠の左下端の座標) の y 格子座標 (予め y 方向に設定されている格子座標) を並べたものを 1 つの個体と考え、 $-x$ 方向のコンパクション (詳細は < 2.4 > 節で述べる) を導入することによって x 座標を決定する、即ち解候補を対応させる、という方法を用いる。

例えば図 1(a) はコンパクションする前の仮の配置状態であり、3 つのブランチから形成されている。図の左側の文字列は y 格子座標を表しており、素子のラベルは通番を表す。(a) の各素子の y 格子座標を通番に従って列挙すると

5 20 35; 0 20 25 40; 10 25

となるが、これが 1 つの個体であり、コンパクション後の (b) の配置状態 (解候補) に対応している。尚、人手によるアナログ LSI の配置・配線では、通常、配置領域の高さを標準的な値に設定し、数ブランチから成る部分回路毎に配置・配線処理を行う。そこで GA においても同様に、1 つの部分回路を配置処理の単位とみなす。ま

図 1: 解候補の符号化とコンパクション

Fig. 1: Coding of a solution, and the related compaction.

た、配置領域の高さは予め固定された値として設定しておく。ブランチ情報については、前もって人手により与えられているものとする。図 6 の縦の曲線はそのようなブランチ情報を表している。

< 2.2 > 交叉, 突然変異 交叉は指定の確率、交叉率、に従ってブランチ毎に行う。交叉方法には一様交叉または一点交叉を用いる。一様交叉⁽⁷⁾は、各カラム毎に 0.5 の確率で互いの要素を入れ換えるというものであり、換言すれば、ブランチ間の素子の y 格子座標をランダムに入れ換えるという操作に相当する。次の例は 1 番目と 3 番目のブランチに対して一様交叉を施している (0, 1 を 0.5 の確率で発生させて 1 の箇所要素の入れ換えを行っている) :

```
親 1: 0 10 20 30; 0 10 30; 0 20 25
親 2: 5 15 30 35; 0 5 40; 10 15 20
      1 0 0 1          0 1 0
⇒ 子 1: 5 10 20 35; 0 10 30; 0 15 25
   子 2: 0 15 30 30; 0 5 40; 10 20 20
```

一点交叉⁽³⁾は、ブランチを任意の箇所て上下に (染色体上では左右に) 分割し、上部同士を交換するという操作に相当する。

突然変異は交叉同様、指定の確率、突然変異率、に従ってブランチ毎に行う。方法は、ランダムに選んだ 1 つの成分の値を $\pm\alpha$ するというもので、 $+$ か $-$ かの選択もランダムとする。但し、例えば $+\alpha$ によって配置領域をはみ出するような場合は $-\alpha$ に変更する。次の例では子の 2 番目のブランチに対して突然変異を行っている (第 2 成分に -5 を加えている) :

```
子: 5 10 20 35; 0 10 30; 0 20 25
⇒ 子': 5 10 20 35; 0 5 30; 0 20 25
```

突然変異率に関しては、大きめの初期値を与え、世代を経る毎に徐々に減少させていく、或いは全世代にわたって小さな定数を用いる。交叉に関しては、世代を経る毎に互いに似通った親に対して交叉をすることになるので、特に交叉率を下げることはしない。

図 2: コスト関数
Fig. 2: Cost functions.

< 2.3 > 適応度 適応度関数は、仮想配線長、配置面積、素子の隣接性、及びブランチ上の素子の上下関係に関するコストを基に設定する。

ネットの仮想配線長とはそのネットの全端子を含む最小長方形の周囲の長さの 1/2 と定義する [図 2.(a) の左図]。但し、電源やグランドに接続する線に関しては、電源やグランドまでの垂直距離によって定義する [同右図]。配置対象素子に関連するネットの仮想配線長の総和を、以下では単に仮想配線長と呼ぶ。配置面積に関するコストには、配置領域の幅を用いる [図 2.(b)]。ペア素子とは、回路の特性を確保するために並列または近接配置したい特定の素子を指す。各ペア素子には図 2.(c) に示すように予め許容範囲を設けておく。但し、許容範囲は近接配置したい度合い、即ちペア性の強弱によって異なるものとする。図 6、図 9 ではペア性の強い素子同士は二重丸で、弱い素子同士は丸で表示している。ペアコストとはその許容範囲からのズレであると定義する (付録参照)。交叉や突然変異を繰り返していくと、一般に同じブランチ上にある素子の上下関係がくずれていく。それを防ぐために順序コスト⁽⁹⁾を用いる [図 2.(d)]。GA は適応度と呼ばれる非負の評価関数値を最大化するアルゴリズムであるので、上記の各コスト関数値を適応度に変換する。例えば世代 k の個体 $x_1, x_2, \dots, x_n$ (n は個体数) のペアコストの値をそれぞれ $p_1, p_2, \dots, p_n$ とするとき、世代 k における個体 x_i のペア適応度 $f_p(x_i, k)$ は

$$(\max(p_j) - p_i) / (\max(p_j) - \min(p_j))$$

図 3: 概略フロー
Fig. 3: Algorithm outline.

であると定義する。ここで、各コスト関数に対する適応度を平等に扱うために、 $\max(p_j) - \min(p_j)$ で割ることによって $[0, 1]$ 内の値に換算していることに注意する。同様に、仮想配線長、配置領域の幅、順序コストに対する適応度 $f_l(x_i, k), f_w(x_i, k), f_o(x_i, k)$ をそれぞれ定義する。これらを用いて、世代 k における個体 x_i の適応度 $f(x_i, k)$ を次のように定義する：

$$f(x_i, k) = w_1 * f_l(x_i, k) + w_2 * f_w(x_i, k) + w_3 * f_p(x_i, k) + w_4 * f_o(x_i, k)$$

ここで、 $w_1, \dots, w_4$ はコスト関数の重要度に応じた重みであり、予め定数として与えておく。

< 2.4 > アルゴリズム概要 本アルゴリズムは、全ての個体が一斉に子孫を作り次世代を生成する、いわゆる離散世代モデルを採用している [図 3]。図 3 には明記していないが、適応度の線形スケーリング⁽³⁾とエリート戦略⁽³⁾も用いる。スケーリングは初期の段階で局所解に収束することを防ぎ、エリート戦略は各世代の最良解の消滅を防ぐために用いる。以下、図 3 に従いアルゴリズムの概要を述べる。

- (1) 入力 素子やネット等の回路図データ、及びペア素子やブランチに関する情報の入力、GA の制御パラメータ (個体数、世代数、交叉率等) の設定などを行う。
 - (2) 初期世代生成 初期解をランダムに個体数だけ生成する。各初期解に対してコンパクション及び適応度の計算を行う ((4) 参照)。
- 以下の (3)~(5) を世代数だけ繰り返す。但し、途中の世代で個体の適応度が全て同じ値になった場合には、その時点で繰り返しを中止する。
- (3) 新世代生成 まずエリート戦略によって前世代の適応度最大の個体をそのまま新世代の個体とする。残り、つまり個体数 - 1 個の個体については、適応度に応じた確率で 2 個体 (親) を選択し、交叉率に従って

図 4: 垂直方向への微動

Fig. 4: Slight shifting along a vertical line.

図 5: コンパクション制限

Fig. 5: A constraint for compaction.

交叉を施し、交叉によってできる新しい2個体(子)に対してそれぞれ、突然変異に従って突然変異を施す、という一連の操作を繰り返す。但し、必要に応じて前もって適応度の線形スケリングを行うこともある。

- (4) コンパクション・適応度計算 各個体毎に、コンパクションを行い、各コストを計算する。コンパクションは左側のブランチから順に、ブランチ内の素子を左に(-x方向に)平行移動することによって行う。素子を移動させる順序は、左隣の既に移動済のブランチの素子に接続を持つ素子を優先する。素子を左に平行移動した際、上下に少し(後述の実験例ではy格子1つ分)ずらしてみても更に左に移動できるかをチェックし出来るだけ密度の濃い配置を試みる[図4]。但し、ブランチ間の左右関係をなるべく保持するために、移動中のブランチに属する素子は1つ及び2つ前のブランチに属する素子の最左端座標(x座標の最小値)を越えてはならないものとする[図5]。適応度は全個体の各コストを全て求めてから、 $<2.3>$ 節の方法で計算する。
- (5) 世代交代 現世代のデータ、各個体の適応度や染色体、を前世代のデータに上書きする。
- (6) 出力 最終世代における適応度最大の個体を最良解とし、関連データを出力する。

$<2.5>$ 実験 図6の回路図の例(例1)を用いて、前節で述べたGAを実験する。次章以降も含め本論文の

図 6: 回路図(例1)

Fig. 6: Circuit schematic of sample 1.

実験では、プログラムはC言語で記述され、計算機はSun4-630(OS/AS=V4.1.3)を使用している。

まず各種のパラメータ値の設定について説明する。y方向の格子数は50と設定し、突然変異の際の α の値は1, 2, 又は3とし、突然変異を行うつど{1,2,3}よりランダムに値を選択するものとする。個体数、世代数には、回路の規模を考慮してそれぞれ41, 500を用いる。突然変異率については、全世代にわたって一定とする方法($\rightarrow$ と表示)、及び初期値を与え最終世代で0となるように線形減少させる方法($\searrow$)を用いる。

表1は、種々の交叉率 P_c 、突然変異率 P_m の組合せに対して、それぞれ乱数表を変えて50回実験して得られた結果の平均及び標準偏差(SDと略記)の統計である。コスト関数には、仮想配線長(表ではLと略記)だけを用いている(例1は素子数が少ないので配置領域の幅やベアコストは P_c や P_m の設定による影響は少ない)。尚、仮想配線長の単位は μm であり、時間はCPU時間の平均である。

表1では、

- (a) 一様交叉, $P_c = 0.75$, $P_m = 1.0 \searrow$ の場合、及び
- (b) 一点交叉, $P_c = 0.5$, $P_m = 0.5 \searrow$ の場合

が最も良い結果を与えている。 P_c または P_m が0の場合は、交叉、突然変異の一方だけを施した時の実験である。交叉だけでは局所解に初期収束してしまう。突然変異単独では、交叉と共に用いた場合よりも良い結果は得られていない。

(a), (b)のパラメータの組合せに対して、 $<2.3>$ 節で定義した4種類のコスト関数を用いてそれぞれ50回実験した。仮想配線長(L), 配置領域の幅(W), ベアコスト(P), 順序コスト(O)に対する適応度の重みにはそれぞれ45, 30, 15, 10を用いた。これら4種類のコストの平均は、(a)の場合 $L = 1219.6, W = 136.7, P = 1.7, O = 0.0$, (b)の場合 $L = 1228.4, W = 135.9, P = 3.4, O = 0.0$ であった(単位は全て μm)。図7は、(a)の場合の最良

変更方法	P_c	P_m	L (平均, SD)	時間[秒]
一様	0.5	1.0↘	1195.4 25.0	40.2
		0.5↘	1193.6 31.1	38.5
		0.25→	1204.4 43.4	39.8
	0.75	1.0↘	1186.7 14.2	40.6
		0.5↘	1190.2 29.3	39.1
		0.25→	1211.1 44.0	40.4
	1.0	1.0↘	1189.4 16.2	41.5
		0.5↘	1193.1 36.4	39.9
		0.25→	1199.2 32.2	40.7
一点	0.5	1.0↘	1196.7 20.6	39.6
		0.5↘	1186.8 14.8	37.6
		0.25→	1205.0 35.0	39.0
	0.75	1.0↘	1196.3 32.8	40.2
		0.5↘	1190.9 27.8	38.8
		0.25→	1196.0 20.8	39.3
	1.0	1.0↘	1193.4 31.9	40.5
		0.5↘	1190.4 38.8	39.1
		0.25→	1200.1 28.3	39.6
なし	0.0	1.0↘	1225.0 48.2	36.0
		0.5↘	1247.8 71.0	30.9
		0.3→	1257.5 65.6	38.0
一様	1.0	0.0	1355.3 51.2	4.2
一点	1.0	0.0	1349.6 58.4	3.9

表 1: 例 1 の実験結果
Table 1: Results of sample 1.

図 7: 例 1 の配置結果例
Fig. 7: A placement for sample 1.

解を図示したものであり、各コストは $L = 1198.5, W = 119.0, P = 0.0, O = 0.0$ である。抵抗はトラジスタの内側の枠まで重なることは許容されている。

従来法との比較のため、SA 法による実験を行った。ここでの SA 法は GA と同様に、解候補を染色体で表現し、突然変異を施すことによって次の解候補（状態）を生成する。突然変異では 1 度に複数の素子の座標が変化しうするため、解候補の全素子をコンパクションした後で、コストを計算する。改悪となる状態遷移は

$$\exp((\text{状態遷移前のコスト} - \text{遷移後のコスト}) / \text{温度})$$

初期温度	ループ回数	P_m	L (平均, SD)	時間[秒]
50	100	1.0↘	1242.0 80.9	38.3
		0.5↘	1256.5 99.8	37.6
		0.3→	1265.6 93.3	36.9
	150	1.0↘	1212.1 61.7	36.5
		0.5↘	1270.1 110.2	35.8
		0.3→	1234.0 86.1	36.1
	200	1.0↘	1211.2 82.7	37.3
		0.5↘	1210.5 61.1	36.4
		0.3→	1244.1 92.2	36.8
100	100	1.0↘	1223.9 79.5	37.3
		0.5↘	1285.9 110.8	36.5
		0.3→	1274.9 96.0	36.9
	150	1.0↘	1202.3 47.0	36.4
		0.5↘	1219.9 65.4	35.5
		0.3→	1272.1 114.8	36.2
	200	1.0↘	1191.8 30.6	37.3
		0.5↘	1248.4 95.6	36.5
		0.3→	1235.9 88.9	37.0

表 2: SA 法による実験結果
Table 2: Results by an SA method.

図 8: SA 法における改悪状態の受容確率
Fig. 8: Acceptance ratio of bad moves in an SA method.

という確率で受容することとし、温度の下げ方には前回の温度を $\beta\%$ ずつ減少させていく方法を用いた。従って、初期を 0 期とみなすと一般に x 期目の温度 $T(x)$ は

$$T(x) = T(x-1) * \beta = \text{初期温度} * \beta^x \quad (x \geq 1)$$

によって定まる。

各種パラメータ値は予備実験の結果、初期温度は 50 または 100、同じ温度でのループ回数は 100、150 または 200、 β の値は 0.95 と設定した。アルゴリズムの終了条件には、GA との比較を見るために、チェックする解候補の総数を用いる（突然変異で状態が変化しない場合はカウントしない）。表 1 では (個体数-1) * 世代数 = 20000 であるから、この総数には 20000 を用いる。ここで -1 の 1 はエリート戦略で保存される個体に相当する。 P_m を線形減少させる場合は、同一温度では同一の確率を用いる。但し、 P_m の値が小さくなり、突然変異を 50 回施しても新しい解候補が生成されない場合はその時点で終了する。

図 9: 回路図 (例 2)

Fig. 9: Circuit schematic of sample 2.

表 2 は P_m の各値に対してそれぞれ 50 回ずつ実験して得られた結果の統計である。表 1 と比較すると、GA の方が標準偏差が低く平均も低いことが分かる。

図 8 は初期温度 = 100, ループ回数 = 150 の場合、各期における改悪状態の受容確率を改悪の程度に応じて示したものである。例えば図の 1 番上の曲線は

$$y = \exp(-1/T(x))$$

即ち、仮想配線長が $1\mu\text{m}$ 長くなるような状態の受容確率の温度依存性を表している。この図より、終盤では殆んど改悪への状態遷移が起らないことが分かる。

3. ブロック配置の重複走査

< 3.1 > 概要 規模の大きな回路に GA を適用する場合の時間削減の方法として、少数のブランチから成る部分回路 (以下ではブロックと呼ぶ) 毎に一部を重複させ順次、配置を決定するという方法を提案する。回路の規模は、本稿では数十～百素子程度の機能セルを想定している。

2 章で説明した GA はブランチ毎に交叉や突然変異を行っているのが 1 つの特徴である。ここで導入するブロック毎の配置処理の重複走査はその特徴を用いたもので、 q ブランチから成るブロック配置処理を r ブランチ ($r < q$) ずつスキップさせながら最適な配置座標を順次、決定していくというものである。以下に概略を述べる。

- (1) 入力データとして、ブロックを構成するブランチ数 q とスキップ数 r を設定する。また、交叉率 P_c 及び (初期) 突然変異率 P_m をブランチ毎に設定する。 P_c , P_m は左側のブランチ程小さい値とする。
- (2) 回路図の一番左側の q 個のブランチから成るブロック (第 1 ブロック) に GA を適用する。 $i = 1$ とする。
- (3) 第 i ブロックに対する GA が終了したら、その出力結果 (配置結果) のうちの左側の r 個のブランチに属す素子の配置座標を正式に決定し、代わりに右側か

ら未配置のブランチを r 個取り入れ第 $i+1$ ブロックを作る。そして第 $i+1$ ブロックに GA を適用する。

以下、 i に 1 ずつ可算し (3) を繰り返す。

GA における個体数は各ブロックに対して同じ値を用いるが、世代数については第 1 及び最終ブロックについては他のブロックにおける世代数 k よりも少し多めにとる (後述の実験では $1.5 \times k$ としている)。また、第 1 ブロックの各ブランチに対する P_c , P_m は、ブロックの最右のブランチに対する率に合わせる。例えば ($q=4$) ブランチに対する P_c がそれぞれ 0.5, 0.5, 0.7, 0.7 ならば、第 1 ブロックに対する P_c は一律 0.7 とする。

初期解の作成方法は、第 1 ブロックについてはランダムに、第 2 以降のブロックについては、左側の $q-r$ 個のブランチについては直前のブロックの最終世代における個体群の遺伝子 (y 格子座標) をそのまま継承し、新しく追加した r 個のブランチについてはランダムに作成する。

第 i ブロックの仮想配線長とペアコストの計算においては、第 i ブロックの左側の素子群、即ち、既に座標が決定している素子群との接続関係によるコストも計算に含める。第 i ブロックの配置面積に関するコストには、第 i ブロックのコンパクション後における (第 i ブロックまでの) 配置済み領域の最右端座標を用いる。

< 3.2 > 実験

図 9 の回路図 (例 2) を用いて、ブロック配置の重複走査の有効性を検証する。

例 2 は 55 素子から成るが、実際には分割抵抗を数に含めて 83 素子から成る回路と見なす。分割抵抗は、抵抗値の相対精度を上げるためのものである。本論文ではそのような抵抗は予め人手により分割された状態で回路図データに入っているものとする。例えば図 9 の抵抗 RR7 は予め RR7I, RR7J, RR7K, RR7L に分割された状態で回路図データに入っている。

表 3 は、コスト関数として仮想配線長 (L)、配置領域

q	個体数 r	世代数	P_c (ブランチ毎、一様) P_m (ブランチ毎)	L (平均) $L(SD)$ 時間[秒]	W (平均) $W(SD)$ 時間[秒]	P (平均) $P(SD)$ 時間[秒]
(a)	4	41	0.4, 0.4, 0.8, 0.8	5016.8	349.2	512.2
	2	300	0.4, 0.4, 0.8, 0.8 $\searrow$	145.0 317.4	7.3 288.8	78.9 308.3
(b)	4	41	0.7, 0.7,..., 0.7	5093.6	349.2	622.6
	4	600	0.9, 0.9,..., 0.9 $\searrow$	161.3 324.1	7.6 296.9	116.7 313.3
(c)	15	41	0.8, 0.8,..., 0.8	6072.1	361.8	858.7
	15	2400	1.0, 1.0,..., 1.0 $\searrow$	473.1 1334.8	9.4 1195.8	170.0 1257.7
(d)	15	99	0.8, 0.8,..., 0.8	5669.7	360.9	725.2
	15	994	1.0, 1.0,..., 1.0 $\searrow$	163.3 1363.7	11.4 1222.9	102.0 1282.9

表 3: 例 2 の実験結果
Table 3: Results of sample 2.

の幅 (W), 及びペアコスト (P) をそれぞれ単独に用いて行った実験の平均と標準偏差 (SD) である。実験回数は (a),(b) がそれぞれ 50 回, (c),(d) がそれぞれ 10 回である。

GA の制御パラメータの値は例 1 の実験結果を参考に定めた。個体数, 世代数は, ブロック毎のコンパクション回数の合計がほぼ同一となるようにした。交叉には一様交叉を用い, 交叉率 P_c や突然変異率 P_m の初期値は, ブロックに含まれる素子数や世代数に応じて設定した。また, y 方向の格子数, 及び突然変異の際の α については, 例 1 と同じ値を用いた。

(b) は 4 ブランチずつ重複部分を持たずに実行したものの, (c), (d) は回路全体に対して GA を適用した場合のものである。対象とする素子数が増えたとコンパクションやコスト計算に時間がかかるため, CPU 時間が大きくなっている。表 3 より, ブロック配置の重複走査の方法はコスト全般に関して有効であることが分かる。

図 10 は, 表 3 の (a) のパラメータ値を用い, 4 種類のコスト関数 (重みは図 7 の場合と同じ値) を用いて 50 回実験し, その中の最良解を図示したものである (コストは $L = 5334.1\mu m, W = 378.5\mu m, P = 934.2\mu m, O = 55.4\mu m$)。

4. 整列配置

配置結果の質を判断する 1 つの基準として, 素子が縦または横に整列しているか否かがある。それは, 整列していた方が配線しやすい, 無駄な面積が少なくて済む等の理由による。整列性の良さは, 一般には仮想配線長や配置面積の減少をもたらすと考えられるが, その他, 実際の配線のし易さに関する種々の要素も考えられ, 一概に特定の関数で判断することは難しいと思われる。

この章では, 横方向, 即ち x 軸方向の整列性を高めるために一種の遺伝的操作を導入し, 横方向の整列度を GA によって進化させる方法を提案する。そして, 整列配置が各コスト関数値にどのように影響するかを調べる。尚, 本稿では横方向の整列だけを扱うので整列の対象とする

図 10: 例 2 の配置結果例
Fig. 10: A placement for sample 2.

素子は, 種類が同じで且つ高さがほぼ同じ素子に限定する。

< 4.1 > 方法 基本的には, GA の各世代において, 各解候補の y 格子座標の微妙なズレを, 世代を経る毎になくしていき, 徐々に横方向の整列性を高めていくという方法である。 y 格子座標の微妙なズレをなくする方法としては, 各解候補の染色体 (個体) に対して, 予め指定された確率 (整列配置率と呼ぶ) に従い, 隣合うブランチ内で y 格子座標が近接している素子のペアを探し, もしそのようなペアが存在すればその 2 素子の y 格子座標を同一の値とする, という操作を行う (以下では y 座標調整と呼ぶ)。但し, 一旦, 座標調整の対象となった 2 素子の y 格子座標は, 現在調べている個体に対する上記操作の間は, 再変更しないものとする。この制約によって, 例えば, 同じ素子の y 格子座標がその両隣りの素子の影響を受けて上下するような無駄がなくなる。

上記の操作は染色体の多様性をなくしていく, 或いは抽象化していく一種の遺伝的操作と考えられる。以下に, 図 3 のフローを補足する形で処理概要を述べる。

- (1) 整列配置用の入力データとして, 整列配置率 P_s と y 格子座標の近接の度合い c (小さな整数) を与える。
- (2) 新世代生成の突然変異に続く遺伝的操作として, 各々の子に対して, 次の (i), (ii) をブランチ総数 $\times P_s$ 回繰り返す。
 - (i) 隣合う 2 ブランチ B_1, B_2 をランダムに選択する。
 - (ii) ブランチ B_1, B_2 から素子のペア $e_1 \in B_1, e_2 \in B_2$ をランダムに選択し, 条件 $0 < |y(e_1) - y(e_2)| \geq c$ をチェックする。ここで $y(e)$ は素子 e の y 格子座標を表す。ある一定回数 (後述の実験では 10 回) までの実行とし, そのうち 1 回でも条件を満たすペアが見つければ y 座標調整し, (i) へ。

P_s	L (平均) L (SD)	時間 [秒]	W (平均) W (SD)	時間 [秒]	P (平均) P (SD)	時間 [秒]
0.0	5016.8 145.0	317.4	349.2 7.3	288.8	512.2 78.9	308.3
0.5	5021.0 150.6	347.9	346.7 8.6	306.9	463.8 69.1	327.2
0.75	5050.7 139.9	352.5	347.1 10.0	318.7	478.5 71.5	340.3
1.0	5062.1 146.0	360.0	349.9 9.3	325.2	487.3 59.7	347.8

表 4: 整列配置の実験結果
Table 4: Results on the line-up method.

(ii) における y 座標調整では、ブランチ B1 の素子 e_1 とブランチ B2 の素子 e_2 が近接している場合、 $y(e_1)$ を $y(e_2)$ に合わせるか、又は $y(e_2)$ を $y(e_1)$ に合わせるか、のどちらか一方をランダムに決定し実行する。例えば、
個体

0 8 12 27 30 39; 19 26 37; 7 13 40; ...

に対して、1 番目のブランチの第 4 成分と 2 番目のブランチの第 2 成分の差は $|27 - 26| = 1$ と小さいので、近接していると判断し、ランダムに例えば値 26 の方を 27 に変更する：

0 8 12 27 30 39; 19 27 37; 7 13 40; ...

整列配置をブロック配置の重複走査と併用する場合は、ブロック毎に上記 (2) の処置を行う。但し (i) で、隣り合う 2 ブランチ B1, B2 を選ぶ際には、現在配置対象としているブロックのすぐ左隣りの (既に座標が決定している) ブランチを B1 として選択することも許容する。

< 4.2 > 実験 例 2 [図 9] の回路図に対して、表 3(a) の制御パラメータと同じものをを用い実験した。入力時に指定する近接の度合い c は 2 とし、整列配置率 P_s には数種類の値を用いた。ブロック内のブランチ数は 4 なので、 P_s の値は実質的には 0 を含めて 5 通りである。例えば $0.5 \leq P_s < 0.75$ ならば前節の (i), (ii) は $[4 * P_s] = 2$ 回繰り返すことになる。

表 4 は、各 P_s の値毎に各コスト関数を単独に用いて、それぞれ 50 回実験を行った結果 (平均、標準偏差) である。 $P_s = 0.5$ の場合が全般に良い結果となっている。 $P_s = 1.0$ の場合は無理に整列させようという働きがあり、コストの微増をまねいたと考えられる。 $P_s = 0.25$ の場合は殆ど整列の効果がないので省略した。

表 4 より、横方向の整列配置は面積の縮小、及びベア性を高める効果があると考えられる。仮想配線長が減少しないのは縦方向の整列性を考慮していないためと考えられる。

図 11 は、 $P_s = 0.5$ とし、図 10 同様、4 種類のコスト関数 (重みは図 10 と同じ値) を用いて 50 回実験し、その中の最良解を図示したものである (コストは $L = 5207.0\mu\text{m}$, $W = 371.0\mu\text{m}$, $P = 923.3\mu\text{m}$, $O =$

図 11: 整列配置の結果例
Fig. 11: A placement by the line-up method.

204.9 μm)。図 10 に比べて横方向への整列の度合いが高いことが分かる。

5. むすび

本論文では、制約の厳しいアナログ LSI の素子配置問題を、4 種類のコスト関数値の最小化問題に抽象化し、GA の適用を試みた。特に、GA の集団進化の特徴を、ブロック配置の重複走査、及び横方向の整列配置の方法に応用し、それぞれの有効性を実験によって確かめた。

今後の課題として、

- (1) タブーサーチ等、局所探索法とのハイブリッド化による配置結果の性能向上。
- (2) 縦方向 (y 軸方向) への整列配置の研究。
- (3) 配線の混雑度等、配線のし易さに貢献する評価基準を整えること。

などがあげられる。(1) は制約条件付きの 2 次元配置問題に対する効率の良い汎用的探索手法の研究であり、(2)、(3) はアナログ LSI の素子レイアウトアルゴリズムの実用面からの研究である。この他に GA の制御パラメータや、複数のコスト関数に対する重みづけの設定の研究も必要である。

最後に、日ごろ素子レイアウトに関し貴重なアドバイスを頂いている当社半導体システム技術センターの美馬健児主務ならびに川北真裕主務に感謝致します。また本研究の機会を与えて頂いた当社研究開発センター、システム・ソフトウェア生産技術研究所の渡辺貞一所長、新井政彦主幹に感謝致します。

(平成 6 年 2 月 15 受付, 同 6 年 10 月 11 日再受付)

- (1) J. P. Cohoon and W. D. Paris: "Genetic placement", *IEEE Trans. on CAD*, **6**, pp. 956-964 (1987)
- (2) M.R. Garay and D.S. Johnson: *Computers and Intractability: A Guide to the Theory of NP-Completeness*, Freeman, New York (1979)
- (3) D.E. Goldberg: *Genetic algorithms in search, optimization and machine learning*, Addison-Wesley Publishing Company (1989)
- (4) J.H. Holland: *Adaptation in natural and artificial systems*, The University of Michigan Press (1975), The MIT Press (1992, 再刊)
- (5) S. Kirkpatrick, C.D. Gelatt, Jr., and M.P. Vecchi: "Optimization by simulated annealing", *Science*, **220**, pp. 671-680 (1983)
- (6) D. Smith: "Bin packing with adaptive search", *Proc. of the First Int. Conf. on Genetic Algorithms and their applications*, pp. 202-207 (1985)
- (7) G. Syswerda: "Uniform crossover in genetic algorithms", *Proc. of the third Int. Conf. on Genetic Algorithms*, pp. 2-9 (1989)
- (8) 渡辺孝博: 「LSI レイアウト自動設計の現状と可能性」, *信学誌*, **76**, 7, pp. 774-782 (1993)
- (9) 佐々木, 謙佐: 「アナログ・セル生成における素子配置手法」, *信学技報*, VLD90-94, pp. 31-38 (1990)
- (10) 小坪, 須貝, 平田: 「遺伝的要素を取り入れた改良型アニリング法によるブロック配置手法」, *信学論 (A)* **J73-A**, pp. 87-94 (1990)
- (11) 美馬, 福永, 小笠原, 吉束: 「バイポーラアナログ LSI のブロック内レイアウト CAD システム」, *信学技報*, VLD90-23, pp. 27-34 (1990)

付 録

ペアコスト判定の際の許容範囲については、本論文の実験では以下のように設定している。まずペア性の弱い素子に対しては、

$$\begin{aligned}\text{許容範囲の高さ} &= (2 \text{ 素子の高さの和})/2 \\ \text{幅} &= (2 \text{ 素子の幅の和})/2\end{aligned}$$

と定義する。そしてペア性の強い素子に対しては、横方向のペア性が強い場合には、許容範囲の高さ $= (2 \text{ 素子の高さの差の絶対値})/2$ ；縦方向のペア性が強い場合には許容範囲の幅 $= (2 \text{ 素子の幅の差の絶対値})/2$ のようにそれぞれ条件を強める。3つ以上の素子がペア素子を形成している場合は、2つ毎に上記の方法で求める。ペア素子の一方が1組の分割抵抗($< 3 \cdot 2 >$ 節参照)の場合は、分割された複数抵抗を含む最小矩形を元の抵抗と考えると、ペアコストを計算する。尚、分割抵抗は一般に同じ箇所にまとまって配置されることが望ましいので、分割抵抗はペア性の強い素子の集まりと考え、上記の方法でコスト計算の対象に含めている。

半田 恵一 (正員) 1986 年早大数学専攻修士課程了。

1989 年京大数理解析専攻博士課程単位取得退学。同年 (株) 東芝入社。グラフ論、組合せ最適化の応用研究に従事。現在、同社研究開発センター、システム・ソフトウェア生産技術研究所主務。理博 (慶大)。情報処理学会、電子情報通信学会の会員。

本位 田 真一 (正員) 1976 年早大電気工学科卒。1978 年

同大学院修士課程了。同年 (株) 東芝入社。主としてソフトウェア工学、人工知能の研究に従事。1986 年情報処理学会論文賞受賞。現在、同社研究開発センター、システム・ソフトウェア生産技術研究所課長。早大非常勤講師。工博。IEEE, AAAI, 情報処理学会、人工知能学会等の会員。