モジュールの重なりを許さない力学的モデルによるモジュール配置手法

山崎博之[†,☆] 三上直人[†] 高橋篤司[†]

モジュール間の配線を考慮したモジュール配置問題において，配線をバネと見なした力学的モデルを構築し，力学的安定点にモジュールを配置する手法がいくつか提案されている．しかし，それらの手法では配置過程や最終配置において，モジュールの重なりを許すため，後行程で多大な時間をかけて重なりを除去する必要がある．本稿では，配置過程および最終配置において，モジュールの重なりを許さない，力学的モデルを用いたモジュール配置手法を提案する．提案手法では，初期配置から各モジュールを力学的な安定点に向かって移動させるが，初期配置，移動方法，モジュール形状を工夫することでモジュール移動におけるデッドロックの発生を抑制し，モジュールの重なりを許さずにより良い解を高速に得ることを可能にする．実験では，従来手法と同程度の質の解を従来よりも高速に求められることを確認した．

A Module Placement Algorithm by Force-directed Method without Overlapping

Hiroyuki Yamazaki,[†,☆] Naoto Mikami[†] and Atsushi Takahashi[†]

For a placement problem to minimize the total wire length among modules, force-directed models are often used. However, since the conventional force-directed based algorithms permit overlaps of modules, it takes a long time to remove overlaps of modules in a placement that is obtained by those algorithms. In this paper, we devise initial placement, movement of modules, and shape of modules so that modules can move smoothly and so that good placement can be obtained. In experiments, our algorithm is faster than other force-directed based algorithms, while the total wire length is comparable to that by other force-directed based algorithms.

1. はじめに

集積回路設計において，機能モジュールの配置場所を決定する問題は古くから研究されており，面積最小化を主眼においた手法や，配線長最小化を主眼においた手法が数多く提案されている．

近年のディープサブミクロン LSI 設計では，全遅延の中で配線遅延が占める割合が高くなり，配線遅延がチップ性能を支配するようになってきたため，配線長最小化を主眼においたモジュール配置手法の重要性が高まっている．

この配線長最小化を主眼においた手法には，最小カットに基づく手法[1),3),6)] や，力学的なモデル用いる手法[4),5),11),14)]，両者を組み合わせた手法[2),7)] などがある．また，Sequence-Pair[9)]，Bounded-Slicline-Grid[10)] に代表されるモジュール配置の表現法を用い，シミュレーテッドアニーリング法などの確率的探索手法で最適解を探索する手法もあるが，多大な時間を必要とする．

最小カットに基づく手法[1),3),6)] では，分割線を通る配線の数が少なくなるようモジュール集合を再帰的に 2 分割し，分割された各領域にモジュールを割り当てる．一般に，各領域の形状はモジュールの形状と一致しないため，無駄な空き領域が少なくなるよう領域の大きさを定めると，モジュールに重なりが生じるため，後工程でこの重なりを除去しなければならない．力学的モデルと組み合わせて分割を行う方法[2),7)] でも同様の問題が発生する．

力学的モデルに基づく手法の 1 つである Force-Directed Relaxation（FDR）法[5),11)] では，配線をモジュール間を結ぶバネと見なし，モジュールにバネの復元力を働かせ，モジュールを力学的な安定点に移動

† 東京工業大学
Tokyo Institute of Technology
☆ 現在，鉄道情報システム株式会社
Presently with Railway Information Systems Co., Ltd

させることで配置を求める．FDR 法では，モジュールを大きさを持たない点として扱っており，モジュールがチップの中心付近に集中した重なりを多く含む配置を出力するため，後工程でモジュールを大きく動かし，この重なりを除去しなくてはならない．文献 4), 14) では，モジュールを拡散させる力を加えることで，モジュールのチップ中心付近への集中を緩和させているが，モジュールの重なりを完全になくすことはできない．

これらの手法ではモジュールの重なりを含む配置を生成するため，後工程でそれら重なりを除去する必要があるが，この工程でのモジュール移動は，配線長を増大させるほか，最終配置を求めるための計算時間も増大させてしまう．

本稿では，これらの問題点を解決するため，モジュールの安定点への移動の際にモジュールの重なりを許さない力学的モデルに基づくモジュール配置手法を提案する．提案手法では，FDR 法などと同様に，配線をバネと見なし，初期配置から各モジュールを力学的な安定点に向かって逐次的に移動させることで配置を求めるが，その移動の際にモジュールの重なりを許さない．モジュールの重なりを許さないことによるモジュール移動の妨げを最小限に抑えるため，初期配置，モジュール移動方法，モジュール形状を工夫し，モジュールを動きやすくすることで，良い解を高速に得られるようにする．

以下，2 章で本稿で扱う問題を定式化し，3 章で提案手法を解説する．4 章で提案手法と従来手法の比較実験の結果を示し，5 章で結論を示す．

2. 準　　備

本稿では，与えられたチップ内にモジュールを配置する問題を考える．すなわち，チップ面積の最小化は考慮せず，モジュールの重なりがなく配線長が最小であるモジュールのチップ内への配置を求めることを目的とする．

問題の入力は，チップの幅と高さ，配置対象となるモジュールの集合 $V = \{v_1, v_2, \ldots, v_m\}$（モジュール数 m，それぞれ幅と高さが与えられる），チップ外周部に置かれる固定モジュール（I/O パッド）の集合 $P = \{v_{m+1}, v_{m+2}, \ldots, v_{m+o}\}$（固定モジュール数 o，それぞれ固定座標が与えられる），モジュール間を接続するネット集合，である．問題の出力は，各モジュールの座標であり，モジュール間を結ぶ配線の総配線長を最小化目標とする．ただし，モジュールの重なり，モジュールのチップ外へのはみ出しは許容しない．

入力されるモジュールは矩形とするが，アルゴリズム中では，面積が等しい円形モジュールとして扱うことがある．モジュール直径は，矩形モジュールの場合は長辺の長さ，円に変換した場合は円の直径とする．また，モジュール半径は，モジュール直径の半分とする．

3. 提 案 手 法

3.1 提案手法の特徴

提案手法では，FDR 法のようにモジュール間を結ぶ配線をバネと見なし，バネに引かれる方向に逐次的に各モジュールを動かすことで，モジュールが力学的な安定点に置かれた配置を出力する．提案手法が FDR 法をはじめとした他の力学的モデルを用いた方法と異なるのは，モジュールの重なりをつねに許さないことである．しかし，モジュールの重なりを許さないことは，配線長を短くするようなモジュールの移動を妨げるため，配線長が十分小さくなる前に安定状態に陥り，配線長が大きい状態でモジュールが移動しなくなることにもつながる．そのため，より滑らかな移動を可能とするよう 3 つの工夫を加える．

モジュール移動　モジュールの移動時に他のモジュールに衝突する場合は，モジュールを被衝突モジュールを避ける方向に移動させるとともに，被衝突モジュールも退かせる．重なりの発生を抑制するとともに，モジュールの安定点への移動における通り道を確保する．

初期配置　十分な間隔をあけてモジュールを配置し初期配置とする．モジュール移動の初期段階では，モジュールの衝突がほとんど発生せず，移動が妨げられにくくなる．

モジュール形状　モジュールの形状を矩形から同じ面積を持つ円に変換する．矩形に比べモジュール衝突時に，被衝突モジュールに沿ってより大きく移動することを可能とする．

矩形モジュールの形状変更に関しては，文献 12) においても矩形の角を丸く変形する手法を提案している．しかし，モジュールの重なりに対して高いコストを加えることで重なりを抑制しており，モジュールの重なりが生じる可能性がある点で本稿の提案手法とは異なる．

なお，矩形モジュールを円に変換し提案手法を適用した場合，最後にモジュールを矩形に戻す必要がある．円形モジュールとして扱う限り重なりは生じないが，矩形に戻す際にモジュールに若干の重なりが生じる．そのため，重なり除去工程を必要とするが，重なり除去に要する時間，配線総長の増加に対する影響は，他

の手法に比べて軽微である．また，回路の性質により，円に変換するよりも矩形のまま扱ったほうが，総配線長が小さい場合があるため，円に変換する場合，変換しない場合の両者について議論する．

3.2 力学的モデル

モジュール v_i と v_j の間のバネ定数 $k_{i,j}$ は，文献4)，11) などと同様に以下のように定める．

$$k_{i,j} = \sum_{e \in E_i \cap E_j} \frac{1}{|e|}$$

ただし，E_i は v_i を含むネットの集合で，$|e|$ はネット e に属するモジュールの数である．$E_i \cap E_j = \emptyset$ の場合は，$k_{i,j} = 0$ となる．

各モジュール v_i の2次元平面上の座標を (x_i, y_i) と記すとき，モジュール v_i はバネの復元力により次の力 $\boldsymbol{f}_i = (f_i^x, f_i^y)$ を受ける．

$$f_i^x = \sum_{v_j \in V \cup P} k_{i,j}(x_i - x_j) \quad (1)$$

$$f_i^y = \sum_{v_j \in V \cup P} k_{i,j}(y_i - y_j) \quad (2)$$

モジュール v_i の移動は，移動ベクトル $\boldsymbol{a}_i = (a_i^x, a_i^y)$ を用いて行う．v_i の移動ベクトル $\boldsymbol{a}_i$ は，v_i の移動開始時にバネの復元力による復元ベクトル $\boldsymbol{s}_i = (s_i^x, s_i^y)$ と他のモジュールが衝突したことによって加えられた衝突ベクトル $\boldsymbol{c}_i = (c_i^x, c_i^y)$ の和として与えられる．

$$\boldsymbol{a}_i = \boldsymbol{s}_i + \boldsymbol{c}_i \quad (3)$$

このとき，v_i は他のモジュールと衝突しない限り，現在位置から $\boldsymbol{a}_i$ だけ移動する．

バネの復元力による復元ベクトル $\boldsymbol{s}_i$ は，Newton-Raphson 法[13] を簡素化した以下の式で定義する．

$$s_i^x = 0.5 f_i^x / \left(\frac{\partial f_i^x}{\partial x_i}\right) \quad (4)$$

$$s_i^y = 0.5 f_i^y / \left(\frac{\partial f_i^y}{\partial y_i}\right) \quad (5)$$

係数 0.5 は大きくすると，早く収束する代わりに解の質が悪くなり，小さくすると，収束が遅くなる代わりに解の質が向上する傾向にある．実験的に調べたところ，0.5 よりも大きくすると，急激に解の質が悪化し，0.5 よりも小さくした場合の解の質の向上は緩やかであったため，係数として 0.5 を採用した．この $\boldsymbol{s}_i$ の定義は，FDR 法[11] でモジュールを安定点に近づけるために移動させる距離を定める式と同じである．

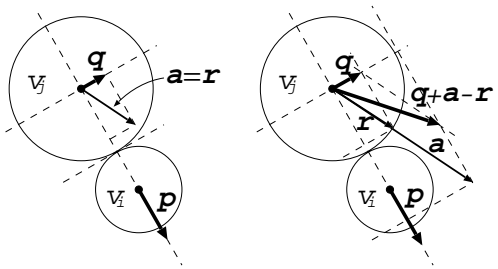
衝突ベクトル $\boldsymbol{c}_i$ は，モジュール v_i が前回移動してから後に，他のモジュールが v_i に衝突することによって与えられるベクトルの和である．

モジュール v_j が，モジュール v_i に衝突したときに与えられるベクトルについて考える．v_j が v_i に衝突したとき，v_j が安定点に近づくためには，v_j は v_i を迂回して移動するとともに，v_i は v_j の進行方向から退くことが望ましい．したがって，衝突時の v_j の移動ベクトルを2つに分解し，一方を v_i の衝突ベクトルに加え，他方を v_j の新たな移動ベクトルとする．ただし，衝突時の移動ベクトルが v_j の半径に比べて大きい場合，v_i の退行および v_j の迂回は，v_j が安定点に近づくためには大きすぎる可能性が高い．そこで分解するベクトルの大きさには上限を与える．

v_j が v_i との衝突時に持つ移動ベクトルを $\boldsymbol{a}$，衝突時の v_i，v_j の中心間を結ぶ方向を中心間方向，中心間方向と垂直な方向を回避方向，大きさ $\min(|\boldsymbol{a}|, v_j$の半径) の $\boldsymbol{a}$ 方向のベクトルを $\boldsymbol{r}$ とする．このとき，$\boldsymbol{r}$ を中心間方向の $\boldsymbol{p}$ と回避方向の $\boldsymbol{q}$ に分解する．この操作を衝突分解と呼ぶ．中心間方向に分解された $\boldsymbol{p}$ を v_i の衝突ベクトルに加え，$\boldsymbol{q} + (\boldsymbol{a} - \boldsymbol{r})$ を v_j の新たな移動ベクトルとする（**図 1** 参照）．

また，モジュールが矩形で，v_i，v_j が横方向（縦方向）のモジュール外周で接している場合，v_j が v_i の外周に沿って移動するよう，$\boldsymbol{q}$ の x（y）成分のみを $\boldsymbol{q}$ とする（**図 2** 参照）．移動ベクトルを中心間方向と外周方向に分解することも可能であるが，$|\boldsymbol{r}| \leq |\boldsymbol{q}|$ となる可能性があるなど，v_j の持つ移動ベクトルが大きくなりすぎるため採用しない．

また，v_j が同時に複数のモジュールに衝突するとき



(a) small vector (b) large vector

図 1 ベクトルの衝突分解（円）
Fig. 1 Vector decomposition (circle).

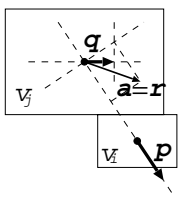


図 2 ベクトルの衝突分解（矩形）
Fig. 2 Vector decomposition (rectangle).

は，$\boldsymbol{r}$ を衝突されたモジュール数で等分し，それぞれを v_j と衝突されたモジュールの中心間方向，回避方向に分解する．中心間方向に分解されたベクトルを，それぞれのモジュールの衝突ベクトルに加え，回避方向のベクトルの和を $\boldsymbol{q}$ とする．

以降，大きさ $\min(|\boldsymbol{a}|,$ モジュール v_i の半径) のベクトル $\boldsymbol{a}$ 方向のベクトルを $\boldsymbol{r}(v_i,\boldsymbol{a})$，ベクトル $\boldsymbol{a}$ をモジュール v_i，v_j の中心間方向，回避方向に衝突分解したベクトルをそれぞれ $\boldsymbol{p}(v_i,v_j,\boldsymbol{a})$，$\boldsymbol{q}(v_i,v_j,\boldsymbol{a})$ で表す．

3.3 提案手法の流れ

提案手法の流れは以下のとおりである．

(1) 初期配置生成.
(2) 固定モジュールの拡大率を固定して，力学的な安定点を探索.
(3) 固定モジュールを拡大率を変化させながら，力学的な安定点を探索.
(4) 配置領域外にはみ出しているモジュールを領域内に押し込む.
(5) （モジュールを円に変換した場合）円を矩形に戻し，重なり除去.

まず，十分な間隔を空けてモジュールを配置し初期配置とする．初期配置では，チップ外周部に置かれる固定モジュールは，すべてのモジュールを囲むよう相似拡大して配置される．次に，固定モジュール位置の拡大率を固定し，力学的な安定点を探索する．次に，固定モジュール位置の拡大率を徐々に小さくしながら，モジュールの位置を修正していく．このステップでは，モジュールの力学的な安定点への移動と固定モジュール位置の拡大率の縮小を交互に行い，モジュールが移動しなくなるまで続ける．次に固定モジュールを本来の位置に固定し，チップの外に出ているモジュールに中へ押し込むための力を加える．最後に，モジュールを円に変換した場合は，円を矩形に戻し，重なり除去を行う．

3.4 モジュールの力学的安定点への移動

提案手法では，モジュールを 1 つずつ逐次的に移動ベクトルに従って移動させることで，力学的安定点を探索する．

モジュール v_i は，移動開始時に式 (3) により移動ベクトル $\boldsymbol{a}_i$ が与えられる．v_i の移動に応じて，この移動ベクトルは減少し，移動ベクトルが十分小さくなったとき，v_i の 1 回の移動を終了する．

v_i は，移動中に他のモジュールに衝突しなければ，現在位置から $\boldsymbol{a}_i$ だけ移動する．$\boldsymbol{a}_i$ だけ移動する前に，他のモジュールに衝突する場合は，他のモジュールに接触するまで移動した後，移動ベクトルの一部を衝突したモジュールに与え，再び移動を試みる．

モジュール v_i を，$\boldsymbol{a}_i$ だけ移動すると，途中でモジュール v_j に衝突する場合，v_i の移動は以下のように行う（v_j に衝突する以前に他のモジュールには衝突しないとする）．

まず，v_i を v_j に接触するまで移動する．次に，その移動に対応するベクトル $\boldsymbol{a}'$ を移動ベクトル $\boldsymbol{a}_i$ から減じる．残りの移動ベクトルを $\boldsymbol{a}=\boldsymbol{a}_i-\boldsymbol{a}'$ としたとき，$\boldsymbol{r}(v_i,\boldsymbol{a})$ を，中心間方向の $\boldsymbol{p}$ と回避方向の $\boldsymbol{q}$ に衝突分解し，$\boldsymbol{p}$ を v_j の衝突ベクトルに加え，v_i を $\boldsymbol{q}+\boldsymbol{a}-\boldsymbol{r}(v_i,\boldsymbol{a})$ だけ移動させる．移動中に v_i が再び他のモジュールと衝突するならば，v_i を他のモジュールに接触するまで移動し，残りの移動ベクトルに対し衝突分解を行う．

以上の操作を各モジュールに対して順に各 1 回適用する操作を 1 パスとし，移動が収束する（モジュールが移動しなくなる）まで繰り返す．

モジュールを安定点に移動するアルゴリズムを**図 3** に示す．アルゴリズム中の $\mathrm{mov}(v_i,\boldsymbol{a})$ は，モジュール v_i を現在位置から $\boldsymbol{a}$ だけ移動させる関数である．ただし，他のモジュールに衝突する場合は，他のモジュールに接触するまで移動させる．また，$\boldsymbol{a}$ から移動した分を減じたベクトルを戻り値として返す．

本手法でモジュール移動が収束したとき，モジュールの動きが完全には停止せず，小刻みな振動を繰り返すように動く場合がある．そのような状況に陥った場合も収束と見なすため，

```
1. すべてのモジュール v_i に対して，c_i = 0;
2. while（収束していない）{
3.     for（各モジュール v_i）{
4.       v_i の移動ベクトル a_i を求める;
5.       c_i := (0, 0);
6.       do{
7.         a_i := mov(v_i, a_i);
8.         if(a_i が十分小さくはない）{
9.           r := r(v_i, a_i);
10.          a_i := a_i − r;
11.          n := v_i が衝突したモジュールの数;
12.          for（v_i が衝突した各モジュール v_j）{
13.            c_j := c_j + p(v_i, v_j, r/n);
14.            a_i := a_i + q(v_i, v_j, r/n);
15.          }
16.        }
17.    } while（a_i が十分小さくはない）
18.  }
19. }
```

図 3 力学的安定点への移動アルゴリズム
Fig. 3 Algorithm toward balanced situation.

収束条件： 10 パスごとに総配線長を記録し，最近の 10 記録において，総配線長が前回のよりも悪くなった記録が 5 つ以上ある場合，

とする．

なお，各パスにおけるモジュール v_i の選択順序としては，移動ベクトルが大きい順，かかる力が大きい順，接続しているバネ定数の総和が大きい順，中心から近い順，それらの逆順などが考えられる．しかし，実験的には，それらの間に有意な差は見られなかった．

3.5 初期配置の生成

初期配置は以下のように生成する．

(1) 格子間隔がモジュールの直径の最大値の 10 倍の $m \times m$ の格子を作成．
(2) モジュールの順列 Γ_1，Γ_2 をランダムに生成．
(3) すべてのモジュールを以下の操作により格子点上に配置：モジュール v が Γ_1 で x 番目，Γ_2 で y 番目のとき，x 番目の縦格子，y 番目の横格子の交差する格子点に v を配置．
(4) すべてのモジュールを囲むように固定モジュールの位置を相似拡大する．

初期配置においてモジュール間の間隔が狭い場合，モジュール移動の初期段階からモジュールの衝突が多発し，配線長を短くするための動きが妨げられるため，最終配置における総配線長が長くなる傾向がある．逆に広くした場合，初期段階では衝突は発生しにくくなるが，計算時間が増大する．実験により，モジュールが置かれる平面の広さと最終配置の総配線長の関係を調べたところ，格子間隔をモジュール直径の最大値の 10 倍以上としたとき，最終配置における総配線長はほとんど変化しなかったため，格子間隔をモジュール直径の最大値の 10 倍とした．

チップ領域は入力として与えられ，固定モジュールの位置は領域外周部に固定されている．固定モジュールの位置を相似拡大せず，他のモジュール位置のみを変化させた場合も，固定モジュールのチップ上における位置は，他のモジュール配置に影響を与える．しかし，初期段階からより正確に考慮されるよう提案手法では，固定モジュールの位置を相似拡大し，徐々に本来の位置に戻す．

3.6 力学的な安定点の探索

力学的な安定点は，固定モジュールの拡大配置の方法により異なる．

提案手法では，固定モジュール位置の拡大率を初期配置で定めたまま固定した状態で，モジュールの移動をモジュールが移動しなくなるまで繰り返し，安定点をまず求める．次に，固定モジュールを徐々に本来の位置に近づけながら，モジュールの位置を修正していく．

固定モジュール位置の拡大率の修正は，モジュール移動操作のパスが終了するごとに，すべてのモジュールをちょうど囲むように，固定モジュール位置の拡大率を修正することにより行う．拡大率を小さくすれば，モジュールが固定モジュールに引かれる力が弱まるため，モジュールはチップ中心へ移動する．その結果，拡大率が小さくなる．この循環を繰り返すことで，固定モジュールの拡大率が小さくなっていく．このとき，固定モジュールで囲まれる領域内にモジュールが片寄って存在すると，拡大率の縮小率が小さくなり収束が遅くなるため，モジュール集合の重心と領域重心が一致するようモジュールの相対位置を保ったまま平行移動させる操作を，拡大率の修正前に行う．

この操作は，モジュールの移動が収束したとき終了するが，このとき，すべてのモジュールがチップ内に収まっているとは限らず，固定モジュールの位置も本来の位置に戻るとは限らない．そのため，チップ内に収まっていないモジュールを押し込む操作が必要になる．

なお，固定モジュール位置の拡大率を初期配置で定めたまま固定した状態で安定点を求める操作を省略すると，初期配置で，望ましいと思われる配置位置と正反対の場所に置かれていたモジュールが，望ましい位置からほど遠い位置で止まってしまうことが多くなる．実験によると，この問題は，初期配置の拡大率を上げることでも解決できるが，操作を省略しない場合と同程度の質の解を得るためには，拡大率を相当大きくする必要があり，計算時間が増大する．また，必要な拡大率は回路によってまったく異なる．よって，一度安定点を求めてから，徐々に拡大率を下げる方式を採用した．

3.7 チップ内へのモジュールの押し込み

力学的安定点の探索の後，チップ内に収まりきらないモジュールがあった場合，それらのモジュールに対して**図 4** のようにバネを追加して，チップ外に出てい

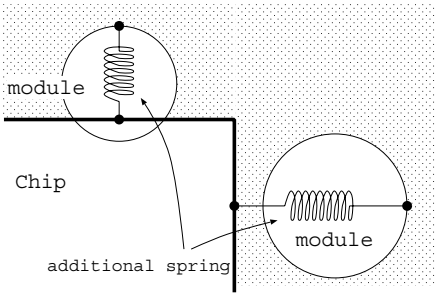


図 4 追加バネの挿入
Fig. 4 Insertion of additional spring.

るモジュールをチップ内に引き込む力を働かせる．固定モジュールは本来の位置に固定し，追加バネのバネ定数を変化させながら力学的安定点を探索する．

追加バネに与えるバネ定数は，引き込もうとしているモジュールが外に出ている間，k_{avg} = (追加バネを除くバネのバネ定数の総和)/m ずつ増加させ，モジュールがチップ内に収まったら，バネを取り除く（ただし，再度押し出される場合に備え，このときのバネ定数は記録しておき，チップ内に収まっている間，そのバネ定数を k_{avg} ずつ減少させる）．各モジュールの追加バネのバネ定数の更新は，そのモジュールの移動開始時に行う．

なお，追加バネによる力は，式 (1)，式 (2) の (f_i^x, f_i^y) にのみ加え，式 (4)，式 (5) での偏微分 $(\frac{\partial f_i^x}{\partial x_i}, \frac{\partial f_i^y}{\partial y_i})$ の計算では，追加バネの存在を無視する（無視しない場合，バネ定数の増加とともに偏微分値も増加し，移動ベクトルがほとんど増加しなくなってしまう）．また，この操作ではモジュールの細かい移動が必要になるため，復元ベクトルを定める式 (4)，式 (5) の係数を 0.5 ではなく 0.05 とする．アルゴリズムの終了条件は，モジュールが

矩形の場合： 全モジュールがチップ内に収まっている

円の場合： チップからの横（縦）方向のモジュールのはみ出しが，チップの幅（高さ）の 5%以内とする．円の場合の終了条件が緩いのは，多少のはみ出しは，後の重なり除去工程で取り除けるためである．

3.8 重なり除去

モジュールを矩形として扱う場合，前節までのアルゴリズムでモジュールの重なりや，モジュールのチップ外へのはみ出しのない配置を得ることができる．一方，モジュールを円に変換した場合，モジュールを元の矩形に戻したときに若干の重なりが生じる．そこで，以下の重なり除去アルゴリズムを用い，モジュールの重なりやモジュールのチップ外へのはみ出しのない配置を求める．

提案する重なり除去アルゴリズムは 2 つのステップからなる．初めにモジュール拡散アルゴリズム[8] を拡張した方法により重なりを減らし，次に局所的な重なり除去を繰り返し残った重なりを取り除く．

まず，最初のステップであるモジュール拡散アルゴリズムについて説明する．モジュール拡散アルゴリズムは，チップ領域の外側を含む配置領域をタイルに切り分け（**図 5** 参照），この各タイル内のモジュールの密度が均等になるようにモジュールを動かす．モジュールの移動は East，South，West，North の 4 つのフェーズに分けて行われる．各フェーズではモジュール v_i はそのフラグ g_i $(0 \leq g_i \leq 1)$ により移動するかどうかを決定する．East フェーズでは，g_i が 0.5 より大きいモジュールは右方向に 1 タイルだけ移動させ，g_i が 0.5 より小さいモジュールはそのフェーズでは移動させない．East フェーズにおける g_i は，全タイルの理想的な密度と East フェーズ後の密度の差の二乗和を最小化するように以下のように決定する．

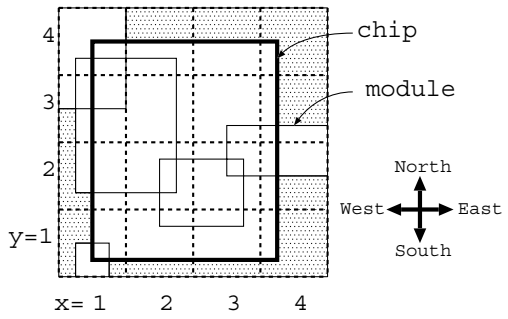


図 5 配置領域のタイルへの切り分け
Fig. 5 Division of placement area into tiles.

East フェーズにおいて，各モジュール v_i が，確率 g_i で右方向に 1 タイルだけ移動し，確率 $1 - g_i$ で移動しなかったとすると，タイル (x, y) のモジュール密度 $A(x, y)$ は次のように表せる．

$$A(x,y) = \sum_{v_i \in M_{x,y}} (1 - g_i) h_{i,x,y} + \sum_{v_i \in M_{x-1,y}} g_i h_{i,x-1,y}$$

$M_{x,y}$ はタイル (x, y) と重なっているモジュールの集合，$h_{i,x,y}$ はモジュール v_i がタイル (x, y) に占める面積の割合である．第 1 項は移動しないモジュールが占める割合を表しており，第 2 項はタイル $(x-1, y)$ から移動してくるモジュールが占める割合を表している．タイル (x, y) の理想的なモジュール密度 $B(x, y)$ は，タイル (x, y) が完全にチップ内にあれば $B(x, y) = 1$，チップ外にあれば $B(x, y) = 0$，チップ外周の境界部分にあればチップ内部分が占める割合を $B(x, y)$ とする．このとき，全タイルの理想的な密度と East フェーズ後の密度の差の二乗和は

$$F = \sum_{x,y} (A(x,y) - B(x,y))^2$$

となり，これを最小にする 0 以上 1 以下の実数 g_i $(i = 1, 2, \ldots, m)$ を次の線形連立方程式を解くことにより求める．

$$\frac{\partial F}{\partial g_i} = 0 \text{ for } i = 1, 2, \ldots, m$$

同様に South，West，North フェーズでは，モジュール v_i をそれぞれ下，左，上方向に 1 タイルだけ動かすかどうかを g_i を用いて決定する．モジュール拡散アルゴリズム[8] では，この 4 つのフェーズを重なりが除去されるまで繰り返し行う．

以上が文献 8) のモジュール拡散アルゴリズムであるが，提案手法では以下のように修正した．まず，モジュールの移動量を (1 タイルの辺の長さ) × $(1 - 0.8|L_i|/|E_i|)$ とする．1 タイルの辺の長さは，モジュールの中で最も短い辺の半分の長さとした．$|E_i|$ は v_i を含むネットの数，$|L_i|$ はモジュール v_i の移動により配線長が増加するネットの数である．モジュールの移動による配線長の増加を抑えるため，多くのネットを配線長を増加させるようなモジュールの移動を抑制している．また，g_i がちょうど 0.5 の場合は，v_i にかかるバネの復元力の和が，移動方向の正成分を持つ場合だけ移動する．これにより，他のモジュールと重ならずに孤立しているようなモジュールが，その場にとどまらず，配線長が短くなる方向に移動する．g_i が 0.5 未満の場合には，v_i の移動は行わない．また，各フェーズの始めにモジュールの回転を試し，回転によりモジュールどうしの重なりや，モジュールとチップ外領域との重なりが少なくなる場合には，回転させる．4 つのフェーズの繰返しは，重なり面積がモジュールの総面積の 0.5%以下になったとき終了する．

次に，逐次的な方法で残った重なりを除去する．まず，**図 6** 左のようにチップ外に出ているモジュールをチップ内に移動し，図 6 右のように 2 つのモジュールが重なっている部分については，両モジュールを重なりが少ない軸方向へ半分ずつ移動する．この操作をすべての重なりがなくなるまで繰り返し適用する．

以上でモジュールどうしの重なりや，モジュールのチップ外へのはみ出しのない配置が得られる．

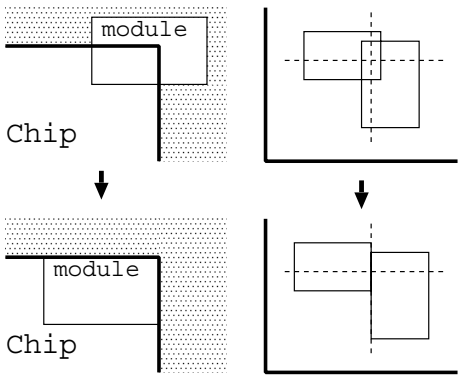


図 6 残った重なりの除去
Fig. 6 Removal of left overlaps.

4. 実験結果

提案手法の性能を確認するため，提案手法に加えて，力学的モデルに基づく従来手法として FDR 法[11] と Eisenmann 法[4] を，PC（Pentium III 600 MHz）上に実装し，提案手法との比較を行った．

FDR 法，Eisenmann 法の実装で用いる各種のパラメータは，各論文で標準とされているものを用いた．また，FDR 法，Eisenmann 法では，モジュールが重なった配置を出力するため，それらの重なりは先に述べた重なり除去アルゴリズムを用いて取り除く．

実験に用いた 3 種類の MCNC のベンチマーク回路の諸元を**表 1** に示す．矩形パッキングアルゴリズムの性能評価実験などでは，チップ形状の指定は無視されることが多いが，本稿では，チップ面積の最小化は考慮しないため指定されたチップ形状，面積，固定モジュール位置で実験を行った．ただし，playout.xlii では，I/O パッドの位置が固定されていないため，チップ外周上のランダムな位置に固定した．また，playout.xlii は極端に小さいモジュールを含んでいたため，その 1 辺の数倍の長さをモジュール拡散で用いるタイルの 1 辺の長さとした．

実験結果を**表 2** に示す．表中の “配線長” は，総配線長，括弧内の “除去前”，“移動距離” は，それぞれ重なり除去工程前の総配線長，重なり除去工程でのモジュールの総移動距離である．また，“時間”，“配置”，“除去” は，それぞれ総所要時間，各手法にかかる時間，重なり除去工程にかかる時間である．各ネットの配線長はネット端子を囲む最小矩形の半周長で評価した（モジュールにおける端子位置はモジュールの中心とした）．実験では，回路，アルゴリズムの 12 通りの組合せについて，それぞれ 15 種類の初期解を用いた．各組合せの最初の行は 15 回の平均値，2 番目，3 番目の行はそれぞれ総配線長が最大，最小となった初期解に対する結果を示す．

総配線長を平均値で比較すると，FDR 法の結果が最も悪く，Eisenmann 法は，提案 2 手法の良い方の結果と同等かやや劣る傾向にあった．FDR 法では，重なり除去前後で配線長が大きく異なり，その相関も小さいことが分かる．Eisenmann 法では，ami33 において重なり除去後に平均値で配線長が減少しているが，これは重なり除去工程で配線長を考慮したことによる．

提案 2 手法では，回路により優劣が異なっているため，様々な特性を持つ回路をランダムに生成し配置の質の傾向を調べた．その結果，チップ形状が正方形に

表 1 MCNC ベンチマーク回路
Table 1 Statistics of MCNC benchmark circuits.

回路名	モジュール数 [個]	I/O パッド数 [個]	ネット数 [個]	チップ形状 [mm × mm]	モジュール面積総和 [mm^2]	密度
ami33	33	42	123	2.05 × 1.46	1.156	0.39
ami49	49	22	408	7.67 × 7.84	35.44	0.59
playout.xlii	62	192	1611	17.82 × 8.81	88.22	0.56

表 2 MCNC ベンチマーク回路に対する実験結果
Table 2 Experimental result on MCNC benchmark circuits.

回路名 (縦横比)	手法	配線長 (除去前,移動距離) [mm]	時間 (配置, 除去) [s]
ami33 (0.71)	提案手法 (円)	78.05 (76.85, 1.99)	0.64 (0.36, 0.29)
	配線長最大	80.98 (79.26, 2.36)	0.84 (0.45, 0.39)
	配線長最小	75.51 (75.12, 1.99)	0.66 (0.38, 0.28)
	提案手法 (矩形)	76.95 (76.95, —)	0.56 (0.56, —)
	配線長最大	79.57 (79.57, —)	0.46 (0.46, —)
	配線長最小	72.41 (72.41, —)	0.84 (0.84, —)
	Eisenmann 法	76.15 (78.55, 3.13)	0.80 (0.38, 0.42)
	配線長最大	82.70 (81.89, 1.27)	0.56 (0.41, 0.15)
	配線長最小	72.67 (71.74, 2.98)	0.69 (0.35, 0.34)
	FDR 法	84.23 (55.86, 13.54)	1.38 (0.50, 0.87)
	配線長最大	91.25 (55.22, 13.61)	1.38 (0.50, 0.88)
	配線長最小	78.89 (55.50, 13.76)	1.20 (0.44, 0.76)
ami49 (0.98)	提案手法 (円)	953.56 (938.56, 11.06)	1.58 (0.58, 0.99)
	配線長最大	972.04 (967.39, 11.75)	1.53 (0.49, 1.04)
	配線長最小	916.40 (912.30, 9.46)	1.47 (0.68, 0.79)
	提案手法 (矩形)	911.26 (911.26, —)	1.44 (1.44, —)
	配線長最大	981.09 (981.09, —)	2.03 (2.03, —)
	配線長最小	861.00 (861.00, —)	1.91 (1.91, —)
	Eisenmann 法	920.27 (820.26, 35.19)	5.02 (0.49, 4.53)
	配線長最大	991.19 (783.33, 37.28)	3.64 (0.32, 3.32)
	配線長最小	856.61 (749.62, 26.24)	3.07 (0.46, 2.61)
	FDR 法	1178.1 (194.61, 107.42)	9.84 (0.86, 8.98)
	配線長最大	1398.7 (195.87, 112.12)	10.51 (0.88, 9.63)
	配線長最小	1035.0 (196.61, 106.90)	8.82 (0.91, 7.91)
playout.xlii (0.49)	提案手法 (円)	6227.0 (5967.6 , 33.15)	2.89 (1.06, 1.83)
	配線長最大	6386.4 (5939.3 , 31.66)	2.65 (0.92, 1.73)
	配線長最小	6019.5 (5896.8 , 30.77)	3.13 (1.03, 2.10)
	提案手法 (矩形)	6526.1 (6526.1 , —)	3.09 (3.09, —)
	配線長最大	6944.8 (6944.8 , —)	4.47 (4.47, —)
	配線長最小	6180.0 (6180.0 , —)	2.80 (2.80, —)
	Eisenmann 法	6760.8 (6207.9 , 53.69)	4.77 (1.63, 3.14)
	配線長最大	7044.5 (6685.8 , 56.99)	4.89 (1.89, 3.00)
	配線長最小	6454.8 (6083.5 , 45.05)	3.40 (1.48, 1.92)
	FDR 法	8168.4 (2360.3 , 201.63)	26.89 (14.10,12.79)
	配線長最大	8785.4 (2498.3 , 199.38)	26.79 (14.06,12.73)
	配線長最小	7437.1 (2350.4 , 200.80)	26.43 (14.24,12.19)

近い場合は，モジュールを矩形のまま扱う方が良く，チップ形状が細長い場合は，モジュールを円に変換して扱う方が良かった．これは，追加バネによる力を加える前の段階では，モジュールが円形に集まった形で安定する傾向にあるが，追加バネによりモジュールをチップ内に押し込める場合，チップ形状が正方形よりは細長いとき，モジュール形状が円形よりは矩形のときにより多くのモジュールがより大きく移動し配線長が増加するためと考えられる．表 1 の ami49 のチップ形状は正方形に近く，playout.xlii のチップ形状は細長い．ami49，playout.xlii のチップ面積をほぼ一定に保ったまま，チップ形状および，固定モジュール位置を修正して行った実験においてもその傾向が裏付けられた．その結果を**表 3** に示す．

重なり除去を必要とする 3 手法に関して，重なり除去前までにかかる計算時間は，FDR 法，Eisenmann

表 3 MCNC ベンチマーク回路に対する実験結果
Table 3 Experimental result on MCNC benchmark circuits.

回路名（縦横比）	手法	配線長（除去前,移動距離）[mm]	時間（配置,除去）[s]
ami49（0.34）	提案手法（円）	1019.0（ 999.44, 14.98）	2.30（0.70,1.60）
	配線長最大	1058.0（1034.1 , 12.86）	2.01（0.64,1.37）
	配線長最小	970.48（ 934.69, 12.45）	1.58（0.61,0.97）
	提案手法（矩形）	1073.4（1073.4 , —）	2.79（2.79, —）
	配線長最大	1130.3（1130.3 , —）	1.78（1.78, —）
	配線長最小	1008.4（1008.4 , —）	2.08（2.08, —）
ami49（0.56）	提案手法（円）	992.68（ 968.46, 12.93）	1.85（0.63,1.23）
	配線長最大	1032.1（ 999.44, 16.22）	2.69（0.63,2.06）
	配線長最小	942.75（ 919.48, 10.96）	1.66（0.70,0.96）
	提案手法（矩形）	1011.5（1011.5 , —）	2.04（2.04, —）
	配線長最大	1087.1（1087.1 , —）	2.48（2.48, —）
	配線長最小	939.42（ 939.42, —）	1.53（1.53, —）
playout.xlii（0.72）	提案手法（円）	6098.0（5912.4 , 24.01）	2.33（1.11,1.22）
	配線長最大	6256.8（6225.1 , 21.06）	1.92（1.00,0.92）
	配線長最小	5993.4（5820.6 , 23.58）	2.31（1.15,1.16）
	提案手法（矩形）	6224.7（6224.7 , —）	2.85（2.85, —）
	配線長最大	6501.1（6501.1 , —）	3.25（3.25, —）
	配線長最小	5901.5（5901.5 , —）	2.73（2.73, —）
playout.xlii（1.00）	提案手法（円）	6042.7（5919.5 , 23.89）	2.26（1.04,1.22）
	配線長最大	6324.0（6066.1 , 25.45）	2.39（0.85,1.54）
	配線長最小	5724.5（5760.6 , 18.30）	1.86（1.02,0.84）
	提案手法（矩形）	5949.3（5949.3 , —）	2.45（2.45, —）
	配線長最大	6217.2（6217.2 , —）	2.44（2.44, —）
	配線長最小	5742.7（5742.7 , —）	2.85（2.85, —）

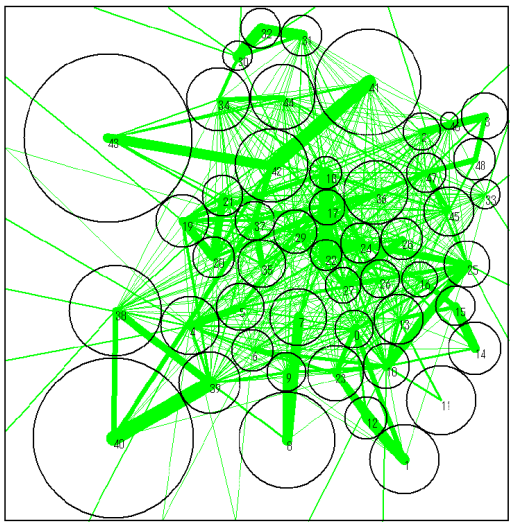


図 7 提案手法（円）：円形モジュールの最終配置（ami49）
Fig. 7 Proposed method (circle): final layout of circular modules (ami49).

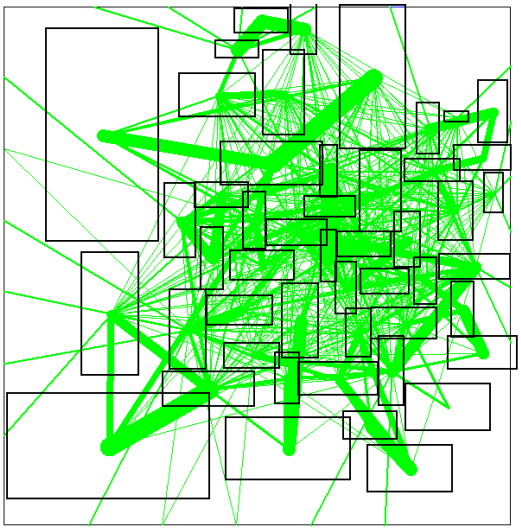


図 8 提案手法（円）：矩形に変換直後の配置（ami49）
Fig. 8 Proposed method (circle): after transformation into rectangles (ami49).

法，提案手法（円）の順に長く，重なり除去にかかる時間は，同様に FDR 法，Eisenmann 法，提案手法（円）の順に長い．モジュール数が多い場合の FDR 法の計算時間の増大は，他の手法に比べ大きくなっているが，これは本実験における実装が悪かったためである．通常の実装をすれば計算時間の増加は他と同程度にとどまると考えられるが，重なり除去と合わせた計算時間で他手法に明らかに劣るため，再実装による実験は行わなかった．

提案手法（円）では，矩形に戻す前の円形モジュールの配置は，**図 7** に示すとおり重なりは存在せず，矩形に戻した場合も**図 8** に示すとおり重なりは少ない

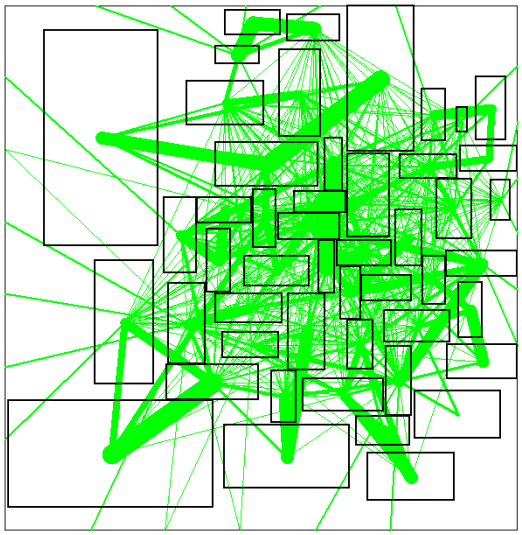


図 9 提案手法（円）：重なり除去後の最終配置（ami49）
Fig. 9 Proposed method (circle): after eliminating overlaps (ami49).

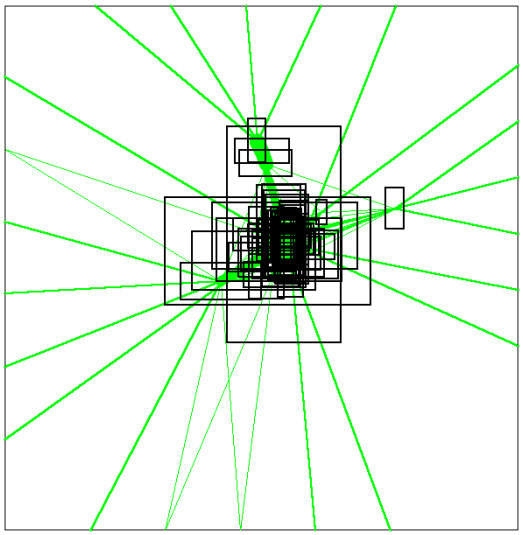


図 10 FDR 法：重なり除去直前の配置（ami49）
Fig. 10 FDR method: before removal of overlaps (ami49).

ことが分かる．重なり除去後の配置も**図 9** に示すとおり図 8 とほとんど変わらないことが分かる．FDR 法，Eisenmann 法が出力する重なり除去直前の配置は，**図 10**，**図 11** のようになっており，重なり除去にかかる時間がモジュールの重なりの量や重なり除去工程でのモジュールの総移動距離に比例していることが分かる．

矩形のまま扱う提案手法（矩形）と他手法の比較も，重なり除去で用いるタイルの大きさにより，重なり除去にかかる時間が大きく左右されるため単純にはできないが，Eisenmann 法や提案手法（円）と同程度の総計算時間で配置が得られている．ただし，チップの形状が細長い場合は，追加バネによりモジュールをチップ内に押し込むのに時間がかかるため，提案手法（円）に比べて時間がかかる傾向にある．

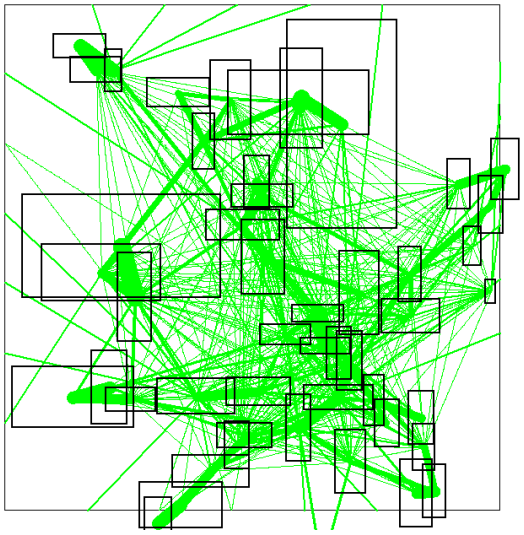


図 11 Eisenmann 法：重なり除去直前の配置（ami49）
Fig. 11 Eisenmann method: before removal of overlaps (ami49).

以上の結果より，提案手法において，チップ形状に応じてモジュールの形状を円もしくは矩形と定めることにより，従来手法と同等の質の配置をより短時間で得られることが確認できた．

5. ま と め

本研究では，モジュールの重なりを許さない力学的モデルに基づくモジュール配置手法を提案し，計算機実験により，提案手法が従来手法で得られる解と同程度の質の解を，より短い時間で得られることを確認した．

今後の課題としては，配線密度やネットに対するタイミング制約の考慮，ソフトモジュールの導入，I/O パッドの位置やチップの形状を固定しないモデルの構築などが考えられる．また，提案手法は，大規模な回路に対しても適用可能と考えるが，提案手法（円）の場合，重なり除去が他の手法と同様に徐々に困難となるため，大規模な回路に対して適用する場合は，重なり除去工程に関してさらなる工夫が必要であると考えられる．

謝辞 本研究を進めるにあたり適切なご助言をいただいた北九州市立大学梶谷洋司教授，中武繁寿助教授に深く感謝する．なお，本研究は CAD21 プロジェク

トの一部である．

参 考 文 献

1) Breuer, M.: Min-cut placement, *J. Design Automation and Fault Tolerant Computing*, Vol.1, pp.343–382 (1977).
2) Cheng, C. and Kuh, E.: Module placement based on resistive Network optimization, *IEEE Trans. Computer Aided Design*, Vol.CAD-3, No.3, pp.218–225 (1984).
3) Dunlop, A. and Kernighan, B.: A procedure for placement of standard-cell VLSI circuits, *IEEE Trans. Computer Aided Design*, Vol.CAD-4, No.1, pp.92–98 (1985).
4) Eisenmann, H. and Johannes, F.: Generic global placement and floorplanning, *Proc. 35th Design Automation Conference*, pp.269–274 (1998).
5) Forbes, R.: Heuristic acceleration of force-directed placement, *Proc. 24th Design Automation Conference*, pp.735–740 (1987).
6) Huang, D.-H. and Kahng, A.: Partitioning based standard cell global placement with an exact objective, *Proc. International Symposium on Physical Design*, pp.18–25 (1997).
7) Kleinhans, J., Sigl, G., Johanes, F. and Antreich, K.: GORDIAN: VLSI placement by quadratic programming and slicing optimization, *IEEE Trans. Computer Aided Design*, Vol.CAD-10, pp.356–365 (1991).
8) Kyung, C., Kraus, P. and Mlynski, D.: Diffusion — An analytic procedure applied to macro cell placement, *Proc. International Conference on Computer Aided Design*, pp.102–105 (1990).
9) Murata, H., Fujiyoshi, K., Nakatake, S. and Kajitani, Y.: VLSI Module Placement Based on Rectangle-Packing by the Sequence-Pair, *IEEE Trans. Computer Aided Design of Integrated Circuits and Systems*, Vol.15, No.12, pp.1518–1524 (1996).
10) Nakatake, S., Murata, H., Fujiyoshi, K. and Kajitani, Y.: Module Packing Based on the BSG-Structure and IC Layout Applications, *IEEE Trans. Computer Aided Design of Integrated Circuits and Systems*, Vol.17, No.6, pp.519–530 (1998).
11) Quinn, N. and Breuer, M.: A force-directed component placement procedure for printed circuit boards, *IEEE Trans. Circuits and Systems*, Vol.CAS-26, No.6, pp.377–388 (1979).
12) Sha, L. and Dutton, R.: An analytical algorithm for placement of arbitrarily sized rectangular blocks, *Proc. 22nd Design Automation Conference*, pp.602–608 (1985).
13) Stark, P.: *Introduction to Numerical Methods*, Macmillan, New York (1970).
14) 岡本 匠, 吉村 猛, 高永 潤, 水牧俊博, 水沼貞幸：2次計画法と矩形パッキングに基づくVLSIフロアプランの一手法，DAシンポジウム2000論文集，pp.79–84 (2000).

(平成13年9月18日受付)
(平成14年3月14日採録)

山崎 博之（正会員）

昭和51年生．平成11年東京工業大学工学部電気・電子工学科卒業．平成13年同大学院理工学研究科電気・電子工学専攻修士課程修了．同年より鉄道情報システム（株）勤務．顧客操作型発券端末のシステム開発に従事．

三上 直人

昭和51年生．平成12年東京工業大学工学部電気・電子工学科卒業．現在，同大学院総合理工学研究科精密機械システム専攻在学中．地理画像処理に関する研究に従事．

高橋 篤司（正会員）

昭和41年生．平成3年東京工業大学大学院理工学研究科電気・電子工学専攻修士課程修了．同年より東京工業大学工学部助手．平成9年より東京工業大学工学部助教授．平成12年より東京工業大学大学院理工学研究科助教授．グラフ理論，組合せアルゴリズム，VLSI自動設計に関する研究に従事．博士（工学）．電子情報通信学会，IEEE各会員．