Memoirs of the Faculty of Science
Kochi University (Information Science)
Vol.29 (2008), No.5

ネットリスト接続の削除と再構成による高速配置法

A Very Fast Placement Method based on Net-list Connectivity Deletion and Reconstruction

竹内 豪　　豊永 昌彦

Tsuyoshi.Takeuchi　Masahiko Toyonaga

高知大学理学部数理情報科学科

Faculty of Science, Kochi University

あらまし

近年，ハードウェアの高性能，小型化のために半導体回路の微細化が進んでいる．しかし，これに伴って回路の配線抵抗増大による信号遅延や不良発生が深刻になっている．これを防ぐには配置配線設計の見積もりが重要になるが，微細化による回路規模の増大により，配置処理に時間がかかるため見積もりは困難である．

そこで本論文では，レイアウト設計における配置処理を高速に行う手法を提案する．ネットリストの接続関係から各素子を周辺と内部に分類して残りをレイアウト中央に配置して再構成するという高速配置手法を考案した．また，単純な回路による実験により，厳密解との差異が平均で 6%程度の結果をほぼ O(N)で得られた．

1　はじめに

パソコンやゲーム機などハードウェアの高性能化と小型化が進み，機器に搭載される半導体回路の製造微細加工技術が進歩している．これに伴い回路の配線抵抗が増加し，近接配線間の寄生的な電荷容量による信号遅延や不良発生が深刻になる．微細化の影響は，製造パターンを設計するレイアウト段階において決まる．問題となる信号への影響は主に配線経路によって見積もることができる．そこで高速高性能な LSI 回路を設計するためには，レイアウト設計後の回路パターンに依存する特性をより早く反映した設計方法が不可欠となる．

微細化における信号遅延の詳細は，配線経路により決まるが，配線の概要情報は LSI 回路上の端子位置の設計（配置設計）で見積もることができる．

しかし，半導体製造の微細化は回路規模を増大させるため，配置処理も長時間要する．

そこで本論文で，著者は LSI レイアウト設計における配置設計を極短時間におこなう高速配置手法について検討をおこなう．

超高速な配置手法を確立すれば，配置設計の入力である回路素子とそれらの信号接続情報（ネットリスト）が得られる論理設計後，直ちに配線概要を見積もることができ，設計フローの大幅な改善が期待できる．

素子数Nで構成されるネットリストの配置問題は，NP完全であることが既に知られている．そのため，様々な近似的な配置手法が提案されてきたが，これらはいずれも素子数をNとして$O(N^2)$～$O(N^3)$程度の処理回数で 2 次元の位置を決めていく方法が主であった．

著者は，$O(N^2)$より小さな時間で配置を完了する方法を考案した．

提案手法では，まずネットリストの接続情報に基づいて2次元配置問題を段階的な1次元問題に分解する．次に各 1 次元化された配置問題をネットリストの接続から順次解いていく．各 1 次元化された配

置問題は，素子数$N^{1/2}$の１次元への割付であるため，高速化が可能である．

簡単な配置問題について，提案手法を適用したところ，ほぼ厳密解となる配置結果が得られた．

以下，本論文の構成は２章において従来の配置手法の概要を説明する．続く３章で提案する配置手法の説明を行う．４章では，提案手法を簡単な配置へ適用する実験方法とその実験結果について述べる．まとめは，５章に述べる．

2　配置問題と従来の配置手法

2.1 配置問題

配置問題とは，素子と素子間の接続が記述されたネットリストを入力として，回路となる２次元の配置領域に素子を配置する問題である．

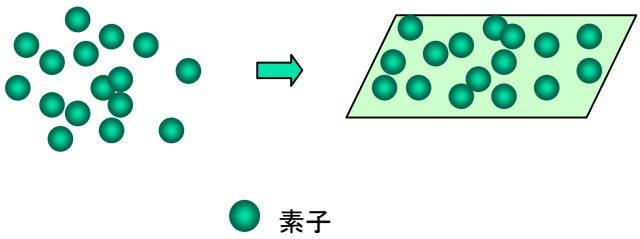


図1　配置問題の概容

レイアウト設計における配置問題は，配置設計の後に続く配線設計が 100%の結線と回路面積を有効に利用するための配置位置を決定する問題である．以下のようにして定式化することができる．

配置問題

1) ネットリストで与えられた全ての素子が重複することなく配置領域内に割り付ける．

2) 配線設計が容易となる配置を導出する．

配置手法の配置評価は主に「推定総配線長」でおこなう．評価関数 cost の例を式(1)に示す．

$$cost = \sum_{i \in N} l(i) \qquad (1)$$

ここで，N は総接続数，l(i)は i 番目の接続の配線長である．cost は，総配線長として定義する．

この評価関数 cost を最小化する配置解を得ることにより各配線長が短くなり，2)の配線設計の容易化が満たされる．

2.2 従来の配置手法

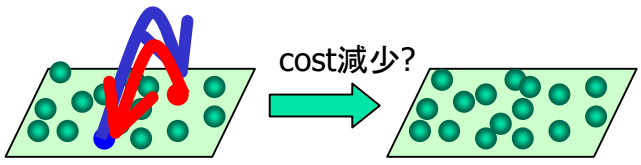
配置問題の評価関数を最小にする解を求める手間は，NP 完全といわれている．そこで以下のような近似解法が提案されてきた．

(a) Pair-wise 法

ネットリストのすべての素子をまず重複すること無く配置領域に並べる．次に素子２つ（ペア）を選択してそれぞれの位置を入れ替える．このとき，配置評価関数がコストを評価し，改善されていればそのまま交換して改善がなければ元に戻す．

これを任意回数繰り返すことで，評価関数がより小さくなる配置位置を得ることができる．

素子数Nに対して，処理の手間は，およそ$O(N^2)$である．



(b) Simulated-annealing(SA)法

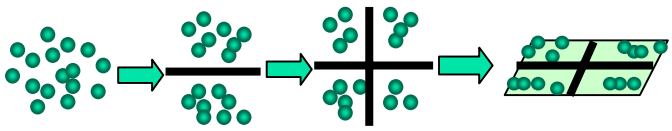
多数の要素の組み合わせを，それらの要素からなる原子・分子の物質とみなして，物理シミュレーション法を適用する方法である．

物質において，高温から低温まで冷却することで，

内部エネルギーが最小となる分子構造をとることをまねて，評価関数をエネルギー関数とみなし，擬似温度Ｔにおける熱平衡となる配置組合せ解を求め，Ｔを高温から低温まで冷却することで評価関数最小となる近似最適解を求める．素子数Ｎに対して，処理の手間は，およそO(N^3)といわれている．

(c) Mincut 法

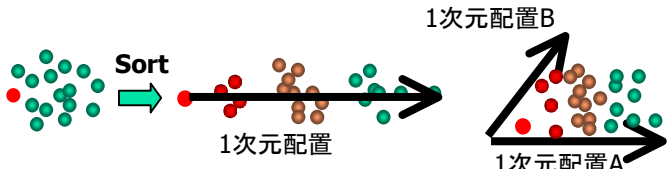
ネットリストの素子群を接続数が最小となる２グループに分割し，それを２次元上の２領域に配置する．これを繰り返すことにより個別の素子レベルまで配置位置を決めていく．素子数Ｎに対して，処理の手間は，およそ O(NlogN)である．



(d) ペアの１次元配置による配置法[2]

まずネットリストの素子をランダムに１つ選び，それを基準にした他の素子のグラフ距離を求める．

次に，距離でソートを行い，１次元接続順ｘを得る．これを２回繰り返して得られた２組の順番 x, y を各素子が配置領域に割り付けられる座標 x, y とみなして２次元配置を得る．



2.3 従来手法の問題点

物理シミュレーション法に基づくシミュレーテドアニーリング法は，初期解に依存せず様々な解を選ぶことができため解空間における検索範囲が極めて広い．そのため処理回数を十分にとれば，高品質な配置解を得ることが期待でき，既に多くの設計システムにおいて配置改善手法として採用されている．しかし、処理時間について素子数Nに対してO(N^3)以上かかるため，膨大な素子数を扱う配置設計で短時間に解を得ることができない．

一方，Pair-wise 法，Mincut 法は，ペア１次元配置法は処理時間は短いが，初期解から改善する解のみを限定して検索するため，高品質な解が得られない．

そこで著者は，新たな視点から配置問題を見直し，高速・高品質な配置法を検討した．次章において詳細を述べる．

3　新配置法の検討

3.1 簡単な配置問題の考察

配置問題を考察するため，図 1(a)に示した簡単な配置問題について考察をおこなう．図１に各素子を四角で，接続を実線で表した．図１は，７ｘ７の内部素子をもつ配置問題の接続を示すと同時に，各素子の最適配置解に相当する．

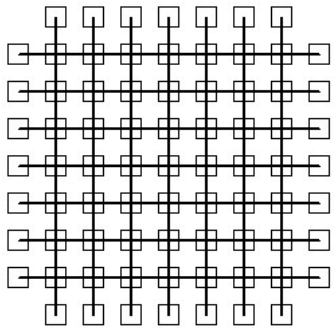


図1.　簡単な配置問題のネットリスト

図１に示したネットリストにおいて「周辺に配置すべき素子の接続数は，内部素子に比べて小さい」

傾向が見られる．いま，ネットリストから接続数の少ない素子（接続数１以下）を除去したとすると，図２に示すような新たなネットリストが得られる．

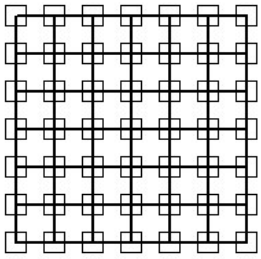


図2. 接続数の少ない素子を削除したネットリスト

さらに図２において，先ほどと同様に，接続数の少ない素子（接続数３以下）を削除したとすると，図２と同様なネットリストが得られることは明らかである．もし，ネットリスト接続数によってネットリストから素子をくり返し削除すれば，最終的には極小数の素子群が残る．これらを「中央に配置すべき素子」として解釈できる．

一度，このようにして分解したネットリストについて，中央に残された素子を配置し，その周囲に削除した順番に素子を戻していくことができれば，かなり高速な配置手法が得られそうである．

以下，ここまでの考察を実現するアルゴリズムについて説明する．

3.2　新配置アルゴリズム

前節の考察から，著者は次のような配置アルゴリズムを提案する．

Peeler Algorithm

Step1. 接続レベル L=1 とする．

Step2. ネットリスト NT の全ての素子数 N を計算する．N が 0 以下なら Lmax=L として終了する．

Step3. 各素子の接続数を計算し，一番少ない素子群 S(L)を選択．

Step4. 素子群 S(L)を除いた新たなネットリスト NT を作る．

Step5. L=L+1.　Step2 へ行く．

Peeler アルゴリズムを以下説明する．

まず，step1 において周辺素子と内部素子を分類するための接続レベル L を 1 とおく．次に，Step2 でネットリスト NT の全ての素子数 N を計算し，N が 0 以下であればプログラムを終了する．N が 0 以上であれば Step3 で各素子の接続数を計算し，接続数が少ない素子群を選択する(この素子群を S(L)とする)．

素子群 S(L)が全て見つけ終われば，ネットリスト NT から S(L)を削除し，残った素子で新たな NT を作成する．

その後，接続レベル L を 1 上げてからステップ 2 に戻り繰り返す．

すなわち，接続数の少ない素子群から順にレベルを決めてネットリストから除去することを繰り返し，素子群のレベル分けを行う処理である．

Core-Growth Algorithm

Step1. 配置領域中央周辺に空領域を設定する．

Step2. L=Lmax として S（L）の素子配置を空領域内で求める．

Step3.域内で配置改善を行う．

Step4. L=L－1．L が 0 なら終了．

Step5. 素子配置領域周辺に空領域を設定する．

Step6. S（L）の素子配置を空領域内で求め，Step3 へ行く．

配置の構成を行う Core-Growth アルゴリズムを以下説明する.

まず Step1 で配置領域を作成し,その中央周辺に空領域を設定する．次に，Step2 で，その空領域内に接続レベル L が最大となる S(L)を配置する．Step3 で，同域内で配置改善を行う.

さらに Step4 で，接続レベルを 1 下げる．このとき L=0 であれば終了する．そうでなければ，Step5 で,配置が決まった素子群の周辺に新たに空領域を設定する.次に Step6 で素子群 S(L)を空領域内で配置して，配置改善のため Step3 へ行く.

すなわち,接続数の少ない素子群から順に削除した素子群を 2 次元平面上に再構築していく処理である.

3.3 ネット削除(Peeler)の例

最初に接続数が少ないとして削除する素子をレベル 1 （L=1）の素子群と呼ぶことにする.

(1)ネットリストを入力，接続レベル L=1 とする.

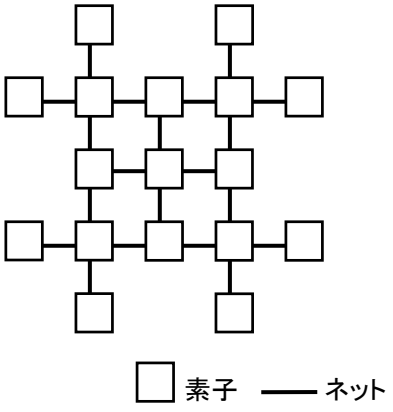


図3. 周辺素子削除の過程 1

(2)ネットの接続数の一番少ない素子にレベル付けをする．以降，この素子群を S (L)とする.

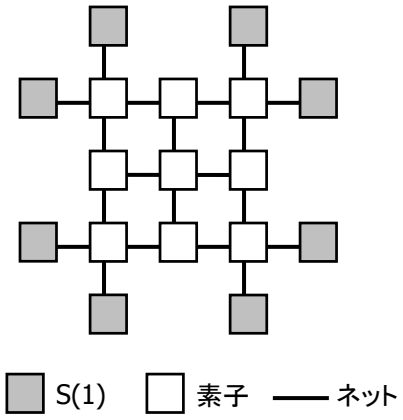


図4. 周辺素子削除の過程 2

(3)回路から素子群 S (L)を削除する.

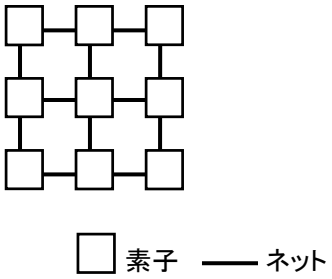


図5. 周辺素子削除の過程 3

(4)接続レベル L を 1 上げる.

(5)回路 G にまだ素子が残っていれば，ステップ 2 に戻る.

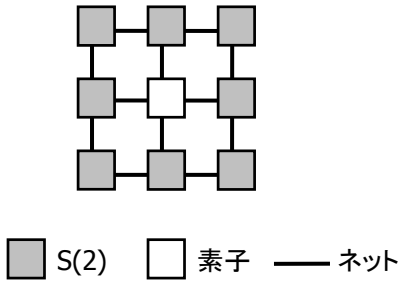


図6. 周辺素子削除の過程 4

(6)これを繰り返せば，最終的に中心の素子が得られる．

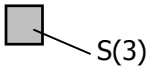


図7. 周辺素子削除の過程 5

(7)回路 G からの素子がなくなれば終了する．

3.3. 配置の構成(Core-Growth)の例

素子のレベル付け，削除の後，レベルの一番高い素子から順にネットリストを再構成していく．その際のアルゴリズムを以下に示す．

(1) 図 7 のようなグリッドを用意する．
(2)最高レベル Lmax を L とする．
(3)S(L)の素子数を N とし，M 個(M≥N)の配置可能領域をグリッドに設ける．

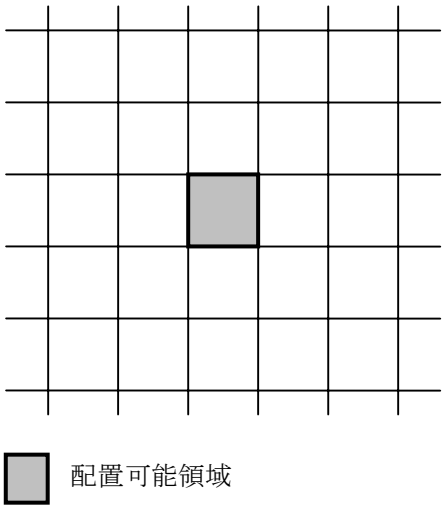


図8. ネットリスト再構成の過程 1

(3) S (L)の素子を配置可能領域に配置する．
(4)配置可能領域内で配置改善を行う．

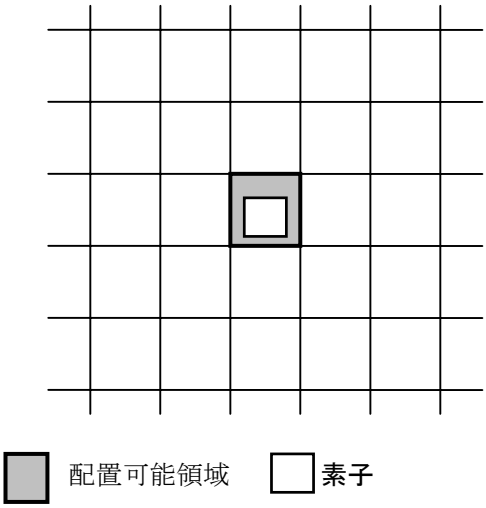


図9. ネットリスト再構成の過程 2

(5)接続レベル L を 1 下げる．
(6)素子を配置した周辺のグリッドに，ステップ 3 の要領で配置可能領域を設定する．

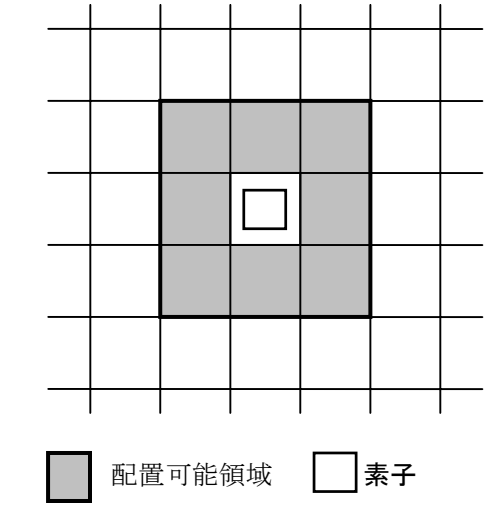


図10. ネットリスト再構成の過程 3

(7) S (L)の素子を配置可能領域に配置する.

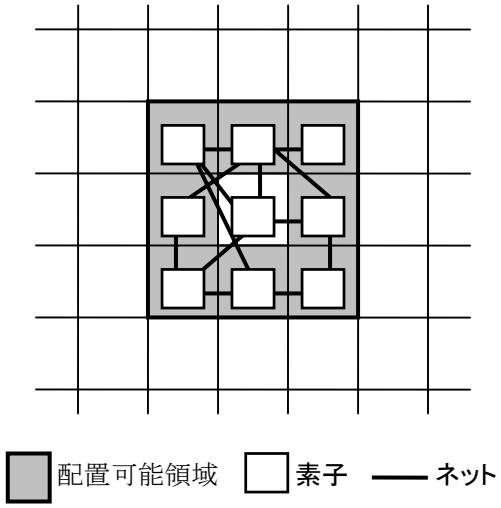


図11. ネットリスト再構成の過程 4

(8) 配置可能領域内で配置改善を行う.

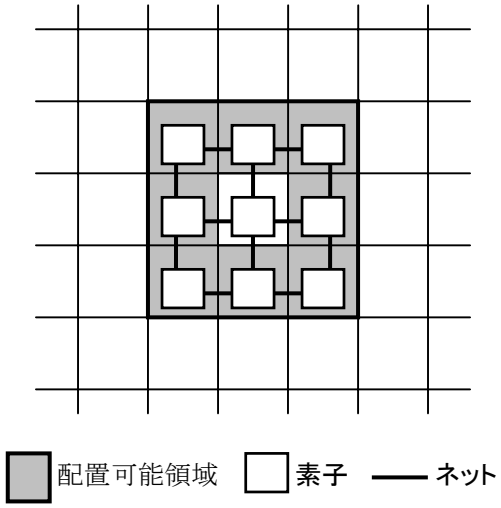


図12. ネットリスト再構成の過程 5

(9)接続レベルが 1 になるまでステップ 5，6，7，8 を行う.

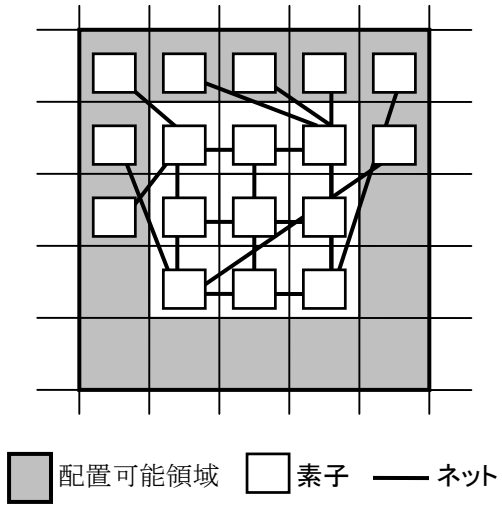


図13. ネットリスト再構成の過程 6

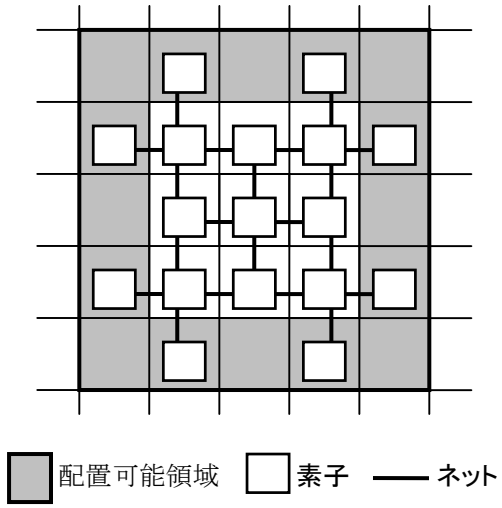


図14. ネットリスト再構成の過程 7

(10)接続レベルが 1 以下になったら終了する.

(11)全素子の配置位置を出力する.

4　実験

提案配置法を実装して実験を行った. 厳密解と提案手法の解と Pair-wise 法の解について, 総配線長を比較する実験を行った．なお、プログラムは Celeron 2.53GHz, 1.25GB RAM の PC で Cgwin 標準の C 言語を用いて作成，実行した.

実験用の回路データとして図 1(a)に示す簡単な回路で 6×6 から 21×21 まで表 1 に示した 6 種類の規模で処理時間を比較した.

4.2. 結果

表 1 は提案手法と Pair-wise 法の配線長を比較した表である．基準として厳密解の値も載せている．処理回数は Pair-wise 法が 10000 回，提案手法が 500 回をそれぞれ解が改善される限り繰り返すようにしている.

表 1　セル数と配線長の比較

セル数 N×N	厳密解 [配線長]	Pair-wise 法			提案手法		
		配線長	厳密解との差異	処理回数	配線長	厳密解との差異	処理回数
6×6	80	96	20%	20000	80	0%	2500
9×9	224	320	43%	20000	230	3%	5000
12×12	440	704	60%	50000	440	0%	8000
15×15	728	1808	148%	90000	786	8%	7500
18×18	1088	3272	201%	150000	1148	6%	10000
21×21	1520	5434	258%	280000	1838	21%	12500

表 1 によると, 厳密解と Pair-wise 法の配線長には厳密解を分母にして平均 122%の差異があったのに対して、厳密解と本提案手法の差異は平均で 6%程度に抑えられている．また，6×6 と 9×9 に関しては厳密解になっている．この結果から，本提案手法の方が Pair-wise 法に比べて厳密解にかなり近いことが分かった.

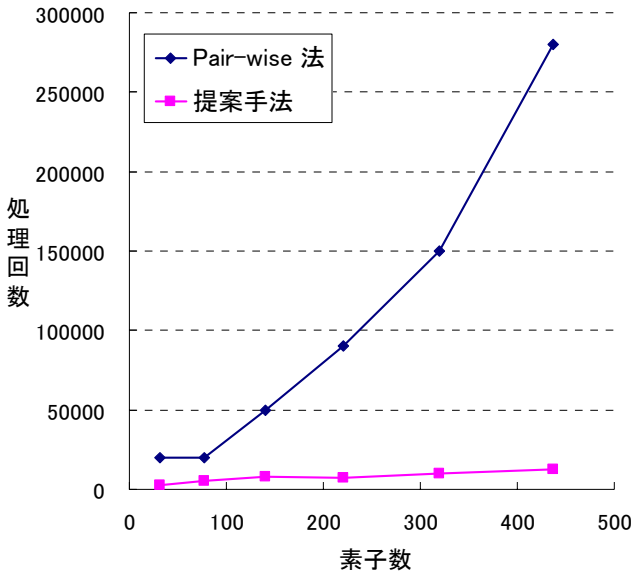
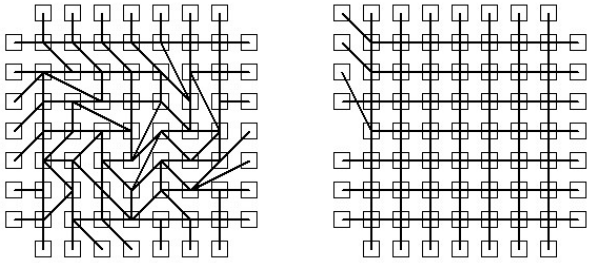


図15.　規模と処理回数

図 15 は表 1 のPair-wise法と提案手法の処理回数をそれぞれグラフに表したものである．図 15 を見ると，Pair-wise法のグラフがほぼ 2 次関数の曲線になっているのに対して, 提案手法は 1 次関数のグラフになっている．この結果から処理時間は Pair-wise法がおよそ$O(N^2)$，提案手法はおよそ O(N)であることが確認できる.

最後に，図 16 に Pair-wise 法と提案手法の 9×9 の回路による配置結果を示す. これらから提案手法を用いた配置結果が, より最適配置に近いことがわかる.



(a) Pair-wise 法　　(b)提案手法

図 16.　配置結果の比較

5　まとめ

ネットリストの接続関係から各素子を周辺と内部に分類して残りをレイアウト中央に配置して再構成するという高速配置手法を提案した. 単純な回路による実験では，処理時間がほぼO(N)であるという結果が得られた．また，解の品質に関しては厳密解との差異が平均で 6%程度という結果が得られた．これは，同時に実験したPair-wise法の処理時間が$O(N^2)$，厳密解との差異が平均 122%であることを考慮するとかなり良好な結果であることが分かる.

今後はより実際的な配置問題で評価していく予定である.

6　謝辞

この論文の作成にあたって, 丁寧にご指導をして下さった豊永先生に深く敬意を表し, 深く感謝致します. また多大な協力をしてくれた同研究室の坂東さんをはじめ，同研究室の方々に感謝します.

参考文献

[1]張 曦，板東修司，豊永昌彦“グラフ距離による高速配置法の検討”平成 18 年度電気関係学会 四国支部連合大会 1-3 (2006 9 月 26 日)

[2]Xin Zang, Masahiko Toyonaga “ A Two-Dimensional Placement by Pair of One-Dimensional SA Solutions”平成 18 年度電気関係学会 四国支部連合大会 17-8(2006 9 月 26 日)

[3] Natarajan Viswanathan, GiJoonNam, Charles J. Alpert, Paul Villarrubia, Haoxing Ren, Chris Chu “RQL: Global Placement via Relaxed Quadratic Spreading and Linearization” DAC 2007, June 4–8, 200