

マルチスレッディングに基づく 並列フロアプランニング手法の開発と評価

佐藤友昭[†] 稲木雅人[‡] 永山忍[‡] 若林真一[‡]

[†]広島市立大学情報科学部 [‡]広島市立大学大学院情報科学研究科

1 はじめに

LSI 設計において、機能モジュールの配置を決めるフロアプラン設計は配置面積や配線長に影響し、チップのコストや性能を決定する重要な設計工程である。フロアプランの最適化問題であるフロアプランニング問題は NP 困難な問題に属するため、シミュレーテッド・アニーリング法 (以下, SA 法) 等の探索的近似解法が適用される。しかしながら、これらの手法を用いても膨大な計算時間を要するため、現在様々な高速化手法が研究されている。

一方、近年、CPU 性能の向上は頭打ちになりつつある。このため、CPU 開発メーカーはこれまでの CPU 単体 (シングルコア) の性能向上から、CPU に複数のプロセッサコアを搭載し、チップレベルマルチプロセッサ化を図ることで性能向上を実現する方向に方針転換した [3]。すでに 4 個のコアを 1 チップ化した CPU が製品化されており、今後は 8 個以上のコアを 1 チップ化した“メニーコア”プロセッサも製品化されると予想されている。こういった状況から、マルチコア CPU を前提とした並列 CAD アルゴリズムの開発が重要になってきた。

本研究では、同時並列処理能力を持つマルチコアプロセッサ向けの新しい並列 SA 法を提案し、フロアプランニングに適用する。

2 フロアプランの定式化

フロアプランとは、モジュール対の相対位置関係のことである。与えられたモジュール集合に対し、様々な大きさのモジュールが互いの重複を許さない二次元平面上への配置を求める問題をフロアプランニング問題という。入力として、モジュールを表す矩形ブロックの集合 $M = \{b_1, b_2, \dots, b_n\}$ が与えられる。 $b_i \in M$ とすると各矩形ブロック b_i に対して矩形ブロックの幅 w_i 、高さ h_i を持つ。フロアプランにおいて、矩形ブロック対 $b_i, b_j \in M$ に、以下の関係が 1 つ以上与えられることは、矩形ブロック対 b_i, b_j が互いに重ならないことと等価である。

$$x_i + w_i \leq x_j, \quad y_i + h_i \leq y_j$$

Development and Evaluation of a Parallel Floorplanning Method Based on Multithreading
Tomoaki SATO[†], Masato INAGI[‡], Shinobu NAGAYAMA[‡] and Shin'ichi WAKABAYASHI[†]

[†]Faculty of Information Sciences, [‡]Graduate School of Information Sciences, Hiroshima City University, Hiroshima, Japan

3 多スタート並列 SA 法

3.1 概要

SA 法は近似解法の一つであり、目的関数値の改善方向への遷移を現在解に微小変形を施した近傍解の質や温度に応じた確率に従って許すことを特徴としている。その特徴から十分な時間をかけることで実用的な解への収束が保証される。SA の高速化の一手法として、SA を並列化することが考えられ、これまでいくつかの方式が提案されている。最も一般的な並列 SA は、各プロセッサ (コア) がそれぞれ SA を実行し、適宜、コア間で解を交換するというものである [4]。この方式の並列 SA では、基本的にはコア数に比例した並列度が達成できる、という利点がある一方、解の収束特性が逐次 SA と異なること、および温度スケジューリングの調整が必要という問題点がある。これらの問題点を解消するため、小西らは温度並列 SA を提案している [5]。温度並列 SA では、コアごとに異なる温度が設定され、適宜、コア間で解を交換することで、解の改良を進める。温度並列 SA はコア数に比例した並列度が達成でき、かつ、温度スケジューリングをあらかじめ設定しておく必要がない、という利点がある。一方、高い温度が設定されたコアについては、アルゴリズム実行の最終段階では、解の改良に寄らなくなり、並列度が低下する、という問題点がある。

本稿では、新しい並列 SA として多スタート並列 SA を提案する。多スタート並列 SA は、実行の初期段階では多スタートによる並列化、終期段階では近傍探索の並列化により、SA の並列実行を実現する。

SA 法の初期段階では受理確率が非常に高いため、近傍解への更新が高確率で発生する。そのため、この段階で近傍解の生成を並列化する必要はない。また、SA 法で主に用いられている幾何冷却法 (温度 $T = T \times \beta (0 < \beta < 1)$) と呼ばれる温度スケジューリングでは、処理の終期段階に比べると初期段階の処理は非常に短い時間であるため、計算時間を短縮する必要性は少ない。このため、複数の初期解を用意し、複数の SA を独立に実行することで、解空間を大域的に探索する。一方、SA 法の終期段階では受理確率が十分に低くなるため、同時に複数の近傍解を生成して高速化を図る並列近傍探索を採用した。幾何冷却法では、処理の終期段階で非常に計算時間を要するため、受理確率が十分に低くなる終期段階にのみ並列近傍探索を使用することで、十分な計算時間の短縮が期待できる。

3.2 アルゴリズム

STEP1 各コアに異なった初期解 x_i を生成し、各初期解に応じた初期温度 t_i を設定する。

STEP2 各コアは受理確率が十分に低くなるまで独立に SA 法を行う。

STEP3 中間解を収集し、SA 法を実行していた隣接コア間で最良解の比較を行い、良質のものを採用する。

STEP4 比較を行ったコアを並列近傍探索のための共有コアとして定め、多スタート法による SA 法を半分にして、共有コアと現在解、最良解を共有しながら SA 法を行う。

STEP5 終了条件を満たせば、暫定解を出力して探索を終了する。そうでなければ STEP3, STEP4 を繰り返し実行する。

図 1 に、8 個のコアを持つマルチコアプロセッサ上での提案並列 SA の実行の様子を示す。8 個の独立した SA から実行が開始され、温度が下がってくると 2 個ずつのコアがそれぞれ解を共有し、4 個の SA が並列に実行され、それぞれの SA において、2 個のコアが近傍解を並列探索する。さらに温度が下がると 4 個ずつのコアが SA を実行し、最後は 8 個のコアが 1 つの SA を実行する。

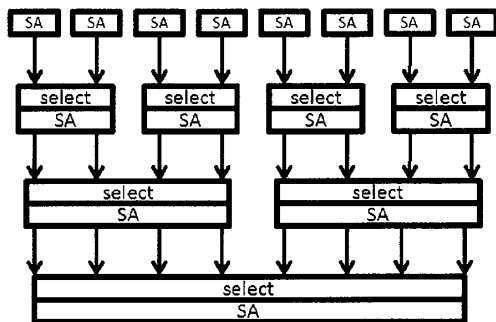


図 1: 多スタート並列 SA の実行例

3.3 並列フロアプランニング手法

多スタート並列 SA を LSI フロアプランニングに応用した。本稿では、フロアプランの表現方法としてシーケンスペア (sequence-pair, SP) [1, 2] を採用した。SP とはモジュール名の 2 本の順列により、モジュールの相対位置を示す表現方法である。また、SP が表すモジュール配置の面積を高速に計算する手法として、文献 [1, 2] で提案されている手法を採用した。紙面の都合上、SP および面積計算手法の説明は省く。SA の近傍解探索は SP のモジュール名の交換とモジュールの 90 度回転を確率的に選択することで実現した。

4 実験と考察

前節で提案した多スタート並列 SA に基づく並列フロアプランニング手法をクワッドコア CPU (Xeon X5482 3.2GHz) が 2 個搭載されたパーソナルコンピュータ上に実装し、評価と実験を行った。スレッドプログラミングにより手法を実現し、SA の近傍解探索において解が受理されるごとにコア間で同期をとっている。

実験においては、200 個のモジュール集合に対して提案 SA を実行し、通常の逐次 SA と比較した。提案 SA と比較対象の逐次 SA の温度スケジュールは同一とし、並列 SA において、いずれかの初期解から最終解までの解の探索過程における解の探索回数は逐次 SA と同一とした。

表 1: 実験結果

| | 面積 | 計算時間 (sec) |
|-------|--------|------------|
| 逐次 SA | 181409 | 519.78 |
| 並列 SA | 181386 | 425.00 |

10 回実験を行った結果の平均を表 1 に示す。表からわかるように、面積は同等であるが、計算時間は並列化を導入したのにも関わらず、提案並列手法のほうが約 18% 短縮されただけであった。理論的には、8 コアによる実行において図 1 に示すように、SA を 8 並列、4 並列、2 並列、単一実行と並列度を下げていき、それぞれの計算時間が同一で、並列化によるオーバーヘッドが無視できると仮定すれば、計算時間は約半分になるはずである。計算時間の短縮が小さい理由として、今回の実験データでは、モジュール間の配線長を考慮していないため、解の改良が進むとシーケンスペアの更新を行っても目的関数値が変化しない場合が頻出し、結果として近傍解探索における解の受理確率が温度の高低に関わらず常に 2.5% 以上となり、近傍解探索における相互排他処理などのオーバーヘッドが相対的に大きかったこと、および、配線長評価を行わない分、目的関数の計算時間が短く、結果として、やはり並列処理のオーバーヘッドが大きくなったことが考えられる。

まとめとして、本稿で提案した並列 SA が有効性を発揮するのは、温度が下がったときに近傍解探索の受理確率が十分に低くなること、および目的関数値の計算に時間がかかる場合であると予想される。SA が適用される実際的な問題はこのような条件を満たすことが多く、今後、そのような問題に対して提案手法の有効性を調べる予定である。

5 あとがき

本稿では、SA の新しい並列化手法として、多スタート並列 SA を提案し、LSI フロアプランニング問題に適用して、評価した。今回の実験評価では、提案手法の有効性を示すことはできなかったが、より実際的な問題に適用すれば有効性が検証できると考えており、今後、評価を進めていく予定である。

本研究の一部は広島市立大学特定研究費により実施された。

参考文献

- [1] 高橋俊彦: “矩形パッキングのための最大重み減少列を求めるアルゴリズム”, 電子情報通信学会 VLSI 設計技術研究会技術研究報告, 1996.
- [2] X. Tang, R. Tian, D.F.Wong: “Fast evaluation of sequence pair in block placement by longest common subsequence computation,” IEEE Trans. CAD, 2001.
- [3] 安田絹子, 小林林広, 飯塚博道, 安部貴之, 青柳信吾: マルチコア CPU のための並列プログラミング, 秀和システム, 2008
- [4] P. Banerjee: Parallel Algorithms for VLSI Computer-Aided Design, Prentice Hall, 1994.
- [5] 小西健三, 瀧和男: “温度並列シミュレーテッド・アニリング法とその応用”, 情報処理学会数理モデル化と問題解決研究会研究報告, 11-1, 1997.