温度並列シミュレーテッド・アニーリング法とその応用

小西 健三† 瀧 和男‡

†神戸大学大学院自然科学研究科
††神戸大学工学部情報知能工学科
〒657 神戸市灘区六甲台町 1-1
Tel: 078-803-1189, Fax: 078-803-1218
E-mail: {konishi,taki}@seg.kobe-u.ac.jp
URL: http://picasso.seg.kobe-u.ac.jp

あらまし 本稿では組合せ最適化問題のための新しいヒューリスティックスである温度並列シミュレーテッド・アニーリング法（温度並列 SA 法）のアルゴリズムとその応用例について報告する．温度並列 SA 法は，(a) 温度スケジュールの自動化，(b) 時間的一様性，(c) 並列処理との高い親和性，という 3 つの著しい特徴を持つ．本稿ではまず温度並列 SA 法のアルゴリズムとその特徴について述べ，次に解品質を劣化させることなくスピードアップが可能な並列計算機上での高効率実装について述べる．さらに温度並列 SA 法の工学的応用として，LSI セル配置問題への適用例，LSI ブロック配置問題への適用例について紹介する．さらにその際の評価結果から，逐次 SA 法と同じ CPU 資源の下でも温度並列 SA 法が優れた最適化能力を持つことを示す．そして最後に巡回セールスマン問題への適用を行い，他の優れたヒューリスティックスとの比較評価結果から，温度並列 SA 法が非常に優れた最適化能力を持つことを示す．

キーワード 温度並列シミュレーテッド・アニーリング法，組合せ最適化，並列処理

Temperature Parallel Simulated Annealing and Its Applications

Kenzo KONISHI† Kazuo TAKI‡

†Graduate School of Science and Technology, Kobe University
‡Department of Computer and Systems Engineering, Kobe University
Rokkodai-cho, Nada-ku, Kobe, JAPAN 657
Tel: +81-78-803-1189, Fax: +81-78-803-1218
E-mail: {konishi,taki}@seg.kobe-u.ac.jp
URL: http://picasso.seg.kobe-u.ac.jp

Abstract This paper describes an algorithm of the Temperature-Parallel Simulated Annealing(TPSA) and its applications. TPSA is a new heuristics for combinatorial optimization problems. TPSA has three special features that are constructing an appropriate cooling schedule automatically, time-homogeneity, and affinity to parallel processing. This paper also briefly describes the efficient implementations of TPSA on distributed memory parallel machines, which can reduce the execution time while keeping the same convergence quality. And then the applications to the LSI standard cell placement problem and block placement problem are reported. Experimental results shows that the convergence capability of the concurrent TPSA algorithm is better than the conventional SA, spending the same CPU time. Finaly, we apply TPSA to the Traveling Salesman Problem. Comparison with TPSA of other good heuristics shows TPSA have good optimization capability.

key words Temperature parallel simulated annealing, Combinatorial optimization, Parallel processing

1 はじめに

近年，工学で扱われる様々な領域で組合せ最適化問題に対する解法の重要性が増してきている．しかしNP完全性 [10] などの計算量の理論より，多項式オーダで最適解を得るアルゴリズムは存在しないことが予想される．そのため実用面において今後ますます大規模化すると思われる組合せ最適化問題に対して，最適解に近い解を実用的な時間で得ることのできる（メタ）ヒューリスティックスの重要性が高まってきており，今日までに様々な手法が報告されている [29]．

本稿でアルゴリズムとその応用について報告する温度並列シミュレーテッド・アニーリング法（以下，温度並列SA法）[20–22,25] は，並列処理と非常に親和性の高いメタヒューリスティックスの一つである．温度並列SA法は名前の通り並列SA法の一種であるが，温度スケジュールが自動化されるという大きな特徴を持つ．ここで温度並列SA法が従来の並列SA法と異なるのは，単に並列処理との親和性が高いのみではなく，並列アルゴリズムにより解の精度をあげることをも目的としていることである．この特徴により，さまざまな問題に対して最適解への収束性が従来より向上することが期待される．

以下，本稿の構成について説明する．2では温度並列SA法のアルゴリズムとその特徴について説明する．3では温度並列SA法の並列実装方法の簡単な説明を行う．4では温度並列SA法の工学的応用としてのLSI配置問題への適用とその評価結果について報告する [6,7,24,25]．5では温度並列SA法をTSPへ適用し，その詳細な評価結果について報告する [26,27]．

2 SA法と温度並列SA法

2.1 SA法と並列SA法

シミュレーテッド・アニーリング法（以下，SA法）[23] は，広範囲の組合せ最適化問題に適用可能な汎用アルゴリズムである．アルゴリズムの頑強さ，実装の容易さ，良質な解が得られやすい，などの理由から，現在までにさまざまな問題への適用例が報告されている [1,8]．SA法はローカルサーチ（繰り返し法）の一種であるが，貪欲なローカルサーチとは異なり目的関数値の改悪方向への遷移を確率に従って許すことにその特徴がある．SA法のアルゴリズムを図1に示す．ここで $\mathcal{N}(x)$ は解 x の近傍，$f(x)$ は目的関数，$[a]^+$ は $\max\{a,0\}$ を表す．また本稿で扱う組合せ最適化問題は，一般性を失うことなく最小化問題であるとする．

このように非常に優れた特性を持つSA法ではあ

procedure Simulated Annealing
1 x_1 := some initial solution
2 **for** k = 1 **to** ∞ **do**
3 choose a $y \in \mathcal{N}(x_k)$ randomly
4 set $x_{k+1} := y$ with probability
5 $\exp\left\{-[f(y^i) - f(x_k^i)]^+/T_i\right\}$
6 otherwise, set $x_{k+1} := x_k$

図 1: SA法

るが，良く知られている問題点として，(a) 改悪方向への解の受理確率を制御する温度と呼ばれる変数（以下，温度）のスケジュールが困難であること，(b) 実行時間の長さ，があげられる．(a) の温度スケジュールについては，SA法の解品質，実行時間の両方を決定付ける重要な要因である．しかしながら実際の問題に対して万能の温度スケジュールを設定することは，それ自体が非常に困難な問題であることも知られている [1]．実践的なスケジュールとして広く用いられているのは，幾何学的冷却である．これは [18,19] で種々の冷却法が試された後で，実用的な温度の制御法として推奨されているものである．

(b) の実行時間については，その解決策の一つとしてSA法の並列化に関する研究が多く報告されている [2,13,33]．SA法の並列化で最も多いアプローチが，複数の状態遷移を（CPU台数分だけ）同時に扱うというものである．しかしながらこの方法では各瞬間で目的関数値を正確に計算できないという問題点がある．一般にこのような実装においては，並列処理により得られるスピードアップと解品質の良さとがトレードオフの関係になる．

2.2 温度並列SA法

そこで望まれる手法としては，並列処理によるスピードアップが得られやすく，解品質の劣化が少ない並列SA法ということになるが，これに対する一つのアプローチとして温度並列SA法 [22,25] が提案されている（図2）．

温度並列SA法は，異なるプロセッサに対して高温から低温まで別々の一定温度を与え，それぞれのプロセッサでSA処理を並行に進める．つまり各プロセッサは異なる中間解を持つことになる．そして隣接する温度を担当するプロセッサ間で，一定期間ごとにそれぞれの解の目的関数値を比較する．このとき高温側の解が低温側の解より良い解を持っているならば，これ

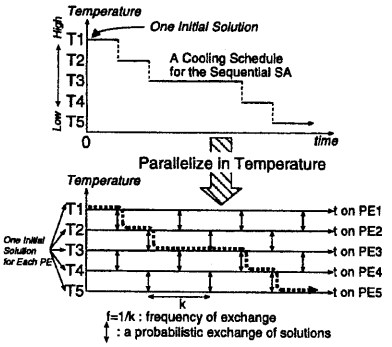

図 2: 温度並列 SA 法

らのプロセッサ間で解を交換する．逆の場合は確率的に交換を行う．今，温度Tを担当するプロセッサが目的関数値fの解を持ち，温度T'を担当するプロセッサが目的関数値f'の解を持つとする．このとき解の交換確率は式（1）に従う [22,25].

$$P(\Delta T, \Delta f) = \begin{cases} 1 & \Delta T \cdot \Delta f < 0 \\ \exp\left(-\frac{\Delta T \cdot \Delta f}{T \cdot T'}\right) & otherwise \end{cases} \quad (1)$$

ここで$\Delta f = f' - f, \Delta T = T' - T$である [22,25].

このように温度並列 SA 法では，高温を担当するプロセッサで多様な探索が，低温のプロセッサで集中的な探索が行われ，処理の進行に伴って良い解が適宜選択されることになる．一般にメタヒューリスティックスを構築する際には，集中化と多様化という 2 つの相反する概念をバランス良く適用することが重要であると言われている [28]．この点でも温度並列 SA 法は非常に優れたアルゴリズムであると言える．

また図 2に示すように，良質の解が高温を担当するプロセッサに存在する場合には，しだいに低温のプロセッサへ解の改善をともないながら移動してゆく．アルゴリズムの途中段階からは最低温度を担当するプロセッサ上に，その時点で最良の解が現れることになる．このようにして良い解に至るための温度スケジュールの自動化が実現される[1]．

以上で述べた温度並列 SA 法の特徴を整理すると，以下の 3 つにまとめることができる [22,25].

温度スケジュールの自動化 逐次 SA 法で問題となる温度スケジュールは，温度並列 SA 法ではプロセッサ間で解の確率的交換を行うことにより自動化される．

[1]一定温度での SA 処理と解交換による最適解への漸近収束性の理論については [22] を参照されたい．

```
procedure Temperature Parallel SA
 1 set x_1^i := initial solutions (0 ≤ i < P)
 2 for j = 1 to SC1 do
 3     parallel_do for each PE_i with solution S_i
 4         for k = 1 to SC2 do
 5             choose a y^i ∈ N(x_k^i) randomly
 6             set x_{k+1}^i := y^i with probability
 7                   exp{-[f(y^i) - f(x_k^i)]^+ / T_i}
 8             otherwise, set x_{k+1}^i := x_k^i
 9         exchange solutions
```

図 3: 温度並列 SA 法のアルゴリズム

アルゴリズムの時間的一様性 温度並列 SA 法は時間的に一様なアルゴリズムである．このことはプログラムを任意の地点で終了することができ，また継続すれば解の改善が続けられることを示している．これに対して通常の逐次 SA 法では，得られた解が不満足なときは温度を再び上げることが必要となる．しかもこのとき温度をどの程度上げるべきかが問題となる．

並列処理との高い親和性 並列プログラムで CPU 台数に対してスピードアップが得られない最大の理由はプロセッサ間通信である．そのためプロセッサ間通信処理は並列アルゴリズムの設計／実装において最も工夫を要する部分でもある．しかしながら温度並列 SA 法では各プロセッサ上で独立に一定温度のアニーリング処理が行われるため，プロセッサ間通信が必要となるのは解交換の瞬間のみである．よって温度並列 SA 法は並列処理と非常に親和性の高い並行アルゴリズムであると言える．

2.3 温度並列 SA 法のアルゴリズム

2.2で説明した温度並列 SA 法の具体的なアルゴリズムを図 3に示す．図 3の 4〜8 行目がそれぞれのプロセッサ（図 3中では P 個のプロセッサ）で実行される一定温度の SA 処理部分である．実際，一定温度であることを除いて 5〜8 行目は図 1中の 3〜6 行目と同じ処理である．そして 9 行目が隣接するプロセッサ間で同期を取った後，解を交換する処理を表す．

2.4 温度並列 SA 法のパラメータ

一般にメタヒューリスティックスは，いくつかのパラメータにより動作が決定付けられる．このことはメタヒューリスティックスの実用化にはパラメータの適正化が不可欠であることを意味する．しかしながら全

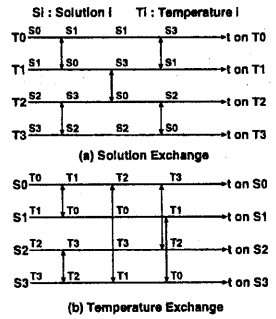

図 4: 解交換と温度交換による温度並列 SA 法の実現

ての問題に対するパラメータの適正値を事前に得ることは原理的に不可能である．そこで通常は系統的な数値実験によりパラメータが決定される．温度並列 SA 法は以下に示す 3 つのパラメータを持ち，これらを適正化する必要がある．

交換周期 図 3中の SC2 に相当．これは問題の入力サイズ（近傍のサイズ）n に対してαn 程度であるとしている．

温度数 図 3中の P に相当．我々の実験では，32 または 64 が適切であるという結果を得ている．

終了条件 図 3中の SC1 に相当．許される実行時間と望まれる解品質により変化する．ただし途中段階で最良の解は常に最低温度を担当しているプロセッサが受け持っているため，それを観測しつつ必要に応じて停止させることができる．

3 温度並列 SA 法の実装法

3.1 並列実装における問題点と同期温度交換

温度並列 SA 法はモデル上は図 2に示すようにプロセッサに温度を固定して解の交換を行う（図 4(a)）．ここで一般に問題の入力サイズ n に対して解の領域計算量は $O(n)$ であることを考慮すると，温度並列 SA 法の通信コストは $O(n)$ となり，問題の大規模化とともに通信コストが大きくなる．そこで解を交換する代わりに，解をプロセッサに固定して温度交換を行うモデルを考える（図 4(b)）．この場合には通信コストは $O(1)$ となり効率的であるが，図 4(b) にも示すように通信相手となるプロセッサが時間とともに変化するという問題点が生じる．これは一つのプロセッサが「どのプロセッサがどの温度を担当している」というテーブルを用意し管理することで解決できる．ここで述べた実装方法を同期温度交換モデルと呼ぶ．

同期温度交換モデルは [6,7,16] などで用いられている方法である．我々の実験では，AP1000 上 32CPU で直列プログラム[2]と比較して，約 30 倍のスピードアップという良好な結果が得られている [26]．

3.2 非同期温度交換

我々は 3.1の同期温度交換モデルを改良した非同期温度交換モデルの提案を [26] で行っている．これは同じ回数の SA 処理が行われても，低温を担当するプロセッサと高温を担当するプロセッサでは処理時間が異なるという事実に基づいている．なぜなら高温を担当しているプロセッサでは状態遷移の受理率が高いため，解の更新処理に多くの時間を要するためである．

以上の事実に基づき，最低温度を担当するプロセッサが交換周期に達したタイミングで，全プロセッサが交換を行うモデルが非同期交換モデルである．非同期温度交換モデルによる実装では，同じく AP1000 上 32CPU で約 33 倍程度という非常に良好なスピードアップが得られている [26]．

4 温度並列 SA 法の LSI 配置問題への応用

4.1 LSI セル配置問題への応用と評価結果

[6,7] では，マクロブロックを含まないスタンダードセル型の LSI を対象とした LSI 配置問題に温度並列 SA 法を適用している．配置対象のセルは，高さが一定で幅がさまざまな形状を持ち，それらの集合であるセル列がチップ上に複数構成されるように配置する（図 5(a)）．まずランダムな初期配置から始めて，総配

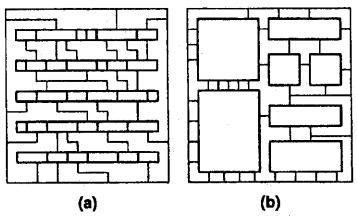

図 5: LSI 設計方式：(a) スタンダードセル方式，(b) ビルディングブロック方式

線長の見積り値が最小になるよう，温度並列 SA 法で配置を改善する．各種パラメータ等の設定方法は [5] に準じている．配置改善のためのセル位置の交換と移

[2]並列プログラムの並列実行部分を逐次プログラムのループに展開したもの．

動は乱数を用いて行い，高温域ではチップ全体を対象とする移動・交換を，低温域ではセル列内に限定した移動・交換を行っている．

セル配置では，MCNC ベンチマークのデータ (125 セル，147 ネット) 等を使用し，プロセッサ数 (温度数) を 63 とした．内側のループ (新しい配置の生成と受理の可否の計算)100 回に対して 1 回の割合で隣接温度間の解交換を試み，ループ 20,000 回実行後の配置結果では，初期配置に比べてチップ面積が 56%の減少となった [6]．

その後の改良により，752 セル，904 ネットのデータ (primary1) において，最小のチップ面積を実現した [7]．処理時間の短縮が次の課題であるが，同期を必要としない単純な計算の部分で，記述に用いた並列論理型言語 KL1 [34] の処理が遅いことが分かっており，改良すべき点となっている．

4.2 LSI ブロック配置問題への応用と評価結果

4.2.1 LSI ブロック配置問題への応用

[24,25] では，マクロブロックを対象とした配置問題に温度並列 SA 法を適用している．配置対象のブロックは，形状固定で配置位置が任意である．セル配置への応用と同様にランダムな初期配置から始めて，目的関数値が最小となるよう繰り返し改善を行う．配置改善のため 2 ブロック間の位置の交換，1 ブロックの移動，回転，鏡像反転を行っている．

4.2.2 逐次 SA 法との評価結果

温度並列 SA 法と逐次 SA 法を適用した LSI ブロック配置プログラムを作成し，SA 処理数を等しくした場合の最適化能力の比較に加えて，計算機資源を等しくした場合（詳細は下記参照）の比較を行う．

温度並列 SA 法 1(以下，TPSA1) 32 温度の温度並列 SA 法で 3000 回の解交換を行う．1 台の計算機上で並行に実行される．

温度並列 SA 法 2(以下，TPSA2) 32 温度の温度並列 SA 法で 3000 回の解交換を行う．マルチワークステーション（7CPU）上で並列に実行される．

逐次 SA 法 1(以下，SA1) TPSA2 で得られた最良の解が経過したアニーリングステップ数（およそ (3000 × (5 × ブロック数))）だけ逐次 SA 法を実行する．

逐次 SA 法 2(以下，SA2) TPSA2 の全ての解が経過したアニーリングステップ数の総和（およそ (3000× (5 × ブロック数) × 32)）だけ逐次 SA 法を実行する．

表 1: 温度並列 SA 法と逐次 SA 法の比較評価

	#SA	Cost Values	Min.
	hp		
TPSA1	201006	9307161	318
TPSA2	201224	**9283042**	121
SA1	201224	10555674	10
SA2	7000414	9949273	389
	xerox		
TPSA1	190930	23523006	745
TPSA2	206540	**23358268**	280
SA1	206540	33046949	24
SA2	6617230	26692333	918
	ami33		
TPSA1	496290	3578236	1243
TPSA2	441650	**3565491**	430
SA1	441650	4779578	34
SA2	14840222	3908557	1566

以上，4 種のプログラムを 3 つの MCNC ベンチマークデータ（hp, xerox, ami33）に対して適用した際，SA 処理数 (表中の SA 回数)，目的関数値，実行時間を表 1に示す．

TPSA1 と SA1 の最適化能力の比較 表 1の hp, xerox, ami33 それぞれの TPSA1 と SA1 の目的関数値より，1CPU で同数のアニーリングステップ数の下では TPSA1（並行実行）が SA1 より明らかに優れていることがいずれの結果についてもわかる．

図 6に温度並列 SA 法で得られた最良の解が経過した温度スケジュールの一例を示す．温度スケジュールは何度かの加熱，冷却過程を繰り返した後，最低温度に落ち着くという特徴を示している．この温度スケジュールは逐次 SA 法で用いられる単調減少の温度スケジュールとは全く異なるタイプのものである．このような温度スケジュールが，温度並列 SA 法の優れた最適化能力の原因であると考えられる．また表 1の

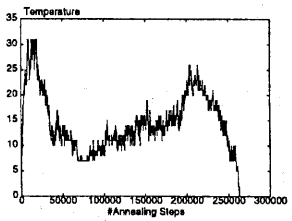

図 6: 温度スケジュールの例

TPSA1 と SA1 の実行時間を比較すると，いずれのデータに対しても TPSA1 は SA1 の約 30 倍である．これはアニーリングを行うプロセスが温度数分だけ，一台の計算機で並行実行されているためである．

TPSA1 と SA2 の最適化能力の比較　これはともに1CPU実行で，ほぼ同じCPU時間をかけたときの解品質の比較である．まず表1においてTPSA1の目的関数値とSA2の平均評価値を比較すると，hp,xerox,ami33いずれの結果についてもTPSA1の方が優れていることがわかる．さらにSA2の最低目的関数値と比較してもやはりTPSA1の方が僅かに優れている．このことから1台のCPUで同じ計算時間をかける場合でも，逐次SA法より温度並列SA法の方が良質の解が得られることを示している．

またTPSA1では32温度分を並行実行しているため，一温度分のCPU時間はSA2の約1/32である．それにもかかわらずより良い解に到達しているのは，図6のような温度スケジュールにより，一温度あたりの実行時間を大幅に（1/32以下に）短縮しているためと見ることもできよう．

4.2.3 配置品質の評価

温度並列SA法で得られた配置結果に対して，市販の配線ツール[3]を用いて実配線を行った．そして現在までに発表されている他の配置プログラムとの比較を行った．比較結果を表2に示す．それぞれのシステムの制約条件，実行環境などが異なるため直接的な比較は難しいが，本プログラムによる配置結果が良質なものであることが判る．

表 2: 他の配置プログラムとの比較

Data	System	area[mm^2]	wire length[mm]
hp	TPSA	12.31	236
	BB [31]	12.15	278
	GEN [9]	12.89	-
xerox	TPSA	25.99	617
	BB [31]	26.17	628
	MBP [35]	25.79	601
	GEN [9]	29.33	-
ami33	TPSA	2.20	97.7
	BB [31]	2.24	109
	MBP [35]	2.42	91

5 温度並列SA法の性能評価

5.1 評価方法

ここでは巡回セールスマン問題（以下，TSP）[32, 36]を用いて，計算機実験による比較評価（実験的解析）[17–19]により，温度並列SA法の性能を評価する．

比較プログラムとしては，TSPのヒューリスティックスとして良好であると報告されているもので，かつ実装方法が明確なものという指標に基づき，2opt, 3opt（[3,4]に基づく実装），Lin-Kernighan法（以下，LK）[30]，Tabu Search [11,12]，SA法[23]を用いた．このうちSA法，Tabu Searchにはパラメータを含んでおり，それぞれ予備実験によりチューニングが施されている．

また温度並列SA法の逐次プログラムは，図3の3行目部分をループに展開した直列プログラムを実装した．以下，この逐次プログラムをC-TPSA(Concurrent TPSA)と呼ぶ．

また実験データとしてはTSPの実験でよく用いられる$[0:1]^2$の一様乱数で生成されたデータを用いる．都市数は10^2, $\lfloor 10^{2.5} \rfloor$（以下，$10^{2.5}$），10^3のものを用意した．

[3]ケイデンス・デザイン・システムズ（株）のBlock Ensembleを使用

表 3: 10^2, $\lfloor 10^{2.5} \rfloor$, 10^3の各データに対して各手法を100回実行した場合のHeld-Karpの下界からの距離の平均値とCPU時間

Heuristics for TSP	下界からの距離(%)			CPU Time (sec.)		
	10^2	$10^{2.5}$	10^3	10^2	$10^{2.5}$	10^3
2-Opt	7.66	6.90	8.37	0.04	0.19	1.04
3-Opt	3.62	3.78	4.49	0.15	1.02	7.32
LK	2.59	3.01	3.23	2.37	26.6	266
SA	3.43	3.58	4.15	35.7	95.9	358
TS	2.96	3.54	3.95	15.2	387	8496
C-TPSA	1.54	2.28	3.01	1258	4945	15267

5.2 逐次計算機上での実験結果

表3に各アルゴリズムに100通りの初期解を与えて実行した場合のHeld-Karpの下界[14,15]からの超過パーセンテージ（以降,「下界からの距離」とよぶ），実行時間の平均値を示す．なお本稿で用いる「下界からの距離」とは，100 (巡回路長/下界値− 1.0) によって得られたものである．

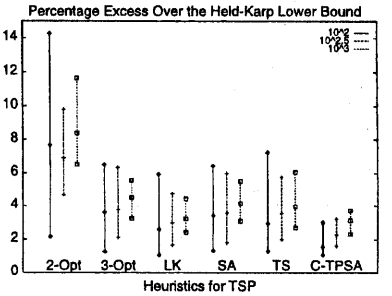

図 7: 10^2, $\lfloor 10^{2.5} \rfloor$, 10^3の各データに対して，各手法を100回実行した場合のHeld-Karpの下界からの距離の最良値，平均値，最悪値とその分布

まず得られる巡回路長について見ると，温度並列SA法で得られた結果（表3中のC-TPSA）は，他のいずれの手法と比較しても下界からの距離が小さいことがわかる．表3のC-TPSA, LKの行を比較すると，いずれのデータに対してもTSP専用アルゴリズムであるLK以上の性能が得られていることがわかる．

以上は平均値に関しての議論であるが，実用面から考えると，さまざまな初期解に対して得られた解の分布が重要なものになってくる．つまり初期解に対する依存性が低いアルゴリズムは，非常に有効であると言える．

そこで100回の実行により得られた最良，平均，最悪の巡回路長の下界からの距離をプロットしたものを図7に示す．図7より，最良，平均，最悪のいずれにおいても本アルゴリズムの優位性は明らかである．特に最悪値が他のアルゴリズムと比較して良い値を示していること，解の分散が非常に小さいことは本アルゴリズムの大きな特徴である「温度スケジュールの自動化」の効果により，初期解に対する依存性が抑えられていることを示している．

しかし実行時間については他手法と比較して非常に長く現実的な実行時間であるとは言い難い．そこで温度並列SA法を並列計算機上で実行することにする．

5.3 並列計算機上での評価

表4: AP1000(SPARC 25MHz)上で実行した場合の平均の下界からの距離と平均実行時間（秒）

	下界からの距離(%)		
	10^2	$10^{2.5}$	10^3
1CPU	1.54	2.28	3.01
32CPU	1.01	2.24	3.11
	実行時間（スピードアップ）		
1CPU	2028(1.0)	*7912(1.0)	*24427(1.0)
32CPU	60.4(33.6)	233.8(33.8)	715.4(34.1)
25/40	37.6	146	447

表中*は予測値．

温度並列SA法は温度数までの並列性を持っている．本節での実験においては温度数が32温度であるので，32PEまで並列実行可能である．ここで3.2で述べた非同期温度交換モデルにより実装を行った場合の結果を表4に示す．ここで表4中25/40の欄は，表3での環境で実行した場合の実行時間予測値を示す．

表4から言えることは，まず温度並列SA法については並列実行による解品質の劣化は見られないということである．そしてこの並列実行の結果を表3の他手法の実行時間と比較すると，温度並列SA法を温度数と同数のCPUを持つ並列計算機上で実行した場合には，専用アルゴリズムには及ばないまでもSA, TSのような汎用の近似解法と十分比較可能な時間であることがわかる．また得られる巡回路長についても，今回実装した専用アルゴリズム程度の解であれば十分に得られると結論付けられる．

5.4 TSPLIBを用いた評価

一様乱数で生成されたランダムデータは，理論的には与えられた領域に点が一様に分布している．しかし実際の問題では偏りのあるものも多く存在しており，それらの問題に対しても比較評価を行う必要がある．そこでTSPのベンチマーク問題集であるTSPLIB [32]の中から最適解既知の問題を50個選び追加実験を行った．ここで温度並列SA法はすべて逐次プログラムとして1CPU上で実行したものである．実験結果の詳細は割愛するが，温度並列SA法で得られた巡回路長の最適巡回路長から超過パーセンテージの分布を表5に示す [27]．表5より50データのうち7データで最適解に到達，さらに全データの40%に対して平均で最適解から1%以内，また60%のデータで2%以内の解，という結果が得られた．また他のプログラムとの比較評価から [27]，温度並列SA法で得られている結果が非常に良好であることがわかった．

表5: 温度並列SA法のTSPLIBデータに対する解の分布

条件	データ数/全データ数
最適解に到達	7/50(14%)
平均で最適解から0.5%以内	10/50(20%)
平均で最適解から1.0%以内	20/50(40%)
平均で最適解から2.0%以内	32/50(64%)
平均で最適解から3.0%以内	43/50(86%)

良好な結果が得られていることが判明した．以上の結果より，実用的なデータに対しても温度並列SA法が有効であることを示すことができた．

6 おわりに

近年，工学で扱われる様々な領域で組合せ最適化問題に対する解法の重要性が増してきている．特に，最適解に近い解を実用的な時間で得ることのできる（メタ）ヒューリスティックスの重要性が高まってきており，様々な報告がある [29]．

本稿では，温度並列SA法のアルゴリズムと応用例について報告した．温度並列SA法は新しい並列メタヒューリスティックスであり，解の精度をあげることをも目的にしていることが他の並列SA法とは大きく異なる．また温度並列SA法は以下の3つの優れた特徴を持つ．

- 温度スケジュールの自動化
- 時間的一様性
- 並列処理との高い親和性

本稿ではまず，温度並列SA法のアルゴリズムの説明とその特徴について説明を行った．そして並列計算機上に実装する際の問題点を示し，効率の良い実装方法について簡単に紹介した．さらに温度並列SA法の工学的応用として，セル配置問題，ブロック配置問題への適用例とその評価結果を示した．特にブロック配置問題における評価では，同じ実行時間を与えても，温度並列SA法が逐次SA法よりも良質な解が得られることを報告した．そして最後に，実質的に組合せ最適化問題のベンチマークであるTSPへ適用し，温度並列SA法の詳細な性能評価結果を紹介した．

今回の報告で適用例を示した問題以外にも，まだまだ多くの組合せ最適化問題が存在する．今後，温度並列SA法をそれらの問題へも適用し，他のアルゴリズムとの比較評価を行う必要があると考えている．

参考文献

[1] Aarts, E. and Korst, J. *Simulated Annealing and Boltzmann Machines.* Wiley, NY, 1989.

[2] Banerjee, P. *Parallel Algorithms for VLSI Computer Aided Design.* Prentice-Hall,Inc., 1994.

[3] Bentley, J. L. Experiments on traveling salesman heuristics. In *Proc. 1 st Annual ACM-SIAM Symp. on Disc. Algo.*, pp. 91–99, January 1990.

[4] Bentley, J. L. K-d trees for semidynamic point sets. In *Proc. 6 th Annual ACM Symp. Comp. Geometry*, pp. 187–197, June 1990.

[5] C.Sechen and A.Sangiovanni-Vincentelli. The TimberWolf placement and routing package. *IEEE J. Solid-State Cir.*, Vol. SC-20, No. 2, pp. 510–522, 1985.

[6] Date, H., Matsumoto, Y., Kimura, K., Taki, K., Kato, H., and Hoshi, M. LSI-CAD programs on parallel inference machine. In *Proc. Intl. Conf. on FGCS*, pp. 237–247, 1992.

[7] 伊達博, 瀧和男. 温度並列シミュレーテッド・アニーリング法に基づくスタンダードセル方式LSI配置プログラム. DAシンポジウム'93, pp. 173–176, 8月 1993.

[8] Eglese, R. W. Simulated annealing : A tool for operational research. *Euro. J. Oper. Res.*, Vol. 46, pp. 271–281, 1990.

[9] Esbensen, H. A genetic algorithm for macro cell placement. In *Proc. Euro.DAC*, pp. 52–57, September 1992.

[10] Garey, M. R. and Johnson, D. S. *Computers and Intractability : A Guide to the Theory of NP-Completeness.* Freeman, 1979.

[11] Glover, F. Tabu search - part I. *ORSA J. Computing*, Vol. 1, No. 3, pp. 190–206, 1989.

[12] Glover, F. Tabu search - part II. *ORSA J. Computing.*, Vol. 2, No. 1, pp. 4–32, 1990.

[13] Greening, D. R. Parallel simulated annealing techniques. *Physica*, Vol. D-42, pp. 293–306, 1990.

[14] Held, M. and Karp, R. M. The traveling-salesman problem and minimum spanning trees. *Oper. Res.*, Vol. 18, pp. 1138–1162, 1970.

[15] Held, M. and Karp, R. M. The traveling-salesman problem and minimum spanning trees: Part II. *Math. Program.*, Vol. 1, pp. 6–25, 1971.

[16] Hirosawa, H., et al. Folding simulation using temperature parallel simulated annealing. In *Proc. Intl. Conf. on FGCS*, pp. 300–306, June 1992.

[17] Johnson, D. S. Local optimization and the traveling salesman problem. *Proc.17th.Colloquium on Automata,Lang.,and Prog.*, pp. 446–461, 1990.

[18] Johnson, D. S., Aragon, C. R., Mcgeoch, L. A., and Schevon, C. Optimization by simulated annealing : An experimental evaluation; part I, graph partitioning. *Oper. Res.*, Vol. 37, No. 6, pp. 865–892, November-December 1989.

[19] Johnson, D. S., Aragon, C. R., Mcgeoch, L. A., and Schevon, C. Optimization by simulated annealing:an experimental evaluation; part II, graph coloring and number partitioning. *Oper. Res.*, Vol. 39, No. 3, pp. 378–406, May-Jun. 1991.

[20] Kimura, K. and Taki, K. Time-homogeneous parallel annaeling algorithm. Technical Report 673, ICOT, 1991.

[21] Kimura, K. and Taki, K. Time-homogeneous parallel annealing algorithm. In *Proc. 13 th IMACS World Congress on Comp. and Appl. Math.*, July 1991.

[22] 木村宏一, 瀧和男. 時間的一様な並列アニーリングアルゴリズム. 信学NC技法, Vol. NC-90-1, , 1990.

[23] Kirkpatrick, S., Gelatt Jr., C. D., and Vecchi, M. P. Optimization by simulated annealing. *Science*, Vol. 220, No. 4598, pp. 671–680, May 1983.

[24] 小西健三, 瀧和男. 温度並列シミュレーテッド・アニーリング法の評価 – LSIブロック配置問題に適用して –. DAシンポジウム'94, pp. 223–228, 8月 1994.

[25] 小西健三, 瀧和男, 木村宏一. 温度並列シミュレーテッド・アニーリング法とその評価. 情報処理学会論文誌, Vol. 36, No. 4, pp. 797–807, 4月 1995.

[26] 小西健三, 屋鋪正史, 瀧和男. 温度並列SA法のTSPへの適用と分散メモリ型並列計算機上での効率の良い実現手法. 並列処理シンポジウムJSPP'96論文集, pp. 153–160, 6月 1996.

[27] 小西健三, 屋鋪正史, 瀧和男. 温度並列シミュレーテッド・アニーリング法の巡回セールスマン問題への適用と実験的解析. 電子情報通信学会論文誌 D-I, 1996. (*to appear*).

[28] 久保幹雄. Generic local search と life span method. 計測と制御, Vol. 34, No. 5, pp. 353–357, 5月 1995.

[29] 久保幹雄. 離散構造とアルゴリズムIV, 第5章 メタヒューリスティックス, pp. 171–230. 近代科学社, 1995.

[30] Lin, S. and Kernighan, B. W. An effective heuristic algorithm for the traveling salesman problem. *Oper. Res.*, Vol. 21, pp. 498–516, 1971.

[31] 小野寺秀俊, 谷口陽, 田丸啓吉. ビルディングブロックレイアウトのための分枝限定配置手法. 信学論, Vol. J75-A, No. 9, pp. 1487–1495, 9月 1992.

[32] Reinelt, G. *The Traveling Salesman – Computational Solution for TSP Applications.* Springer-Verlag, 1994.

[33] Rosen, B. E., 中野良平. シミュレーテッド・アニーリング – 基礎と最新技術 –. 人工知能学会誌, Vol. 9, No. 3, pp. 365–372, 5月 1994.

[34] Ueda, K. and Chikayama, T. Design of the kernel language for the parallel inference machine. *The Computer Journal*, Vol. 33, No. 6, pp. 494–500, 1990.

[35] Upton, M., Samil, K., and Sugiyama, S. Integrated placement for mixed macro cell and standard cell. In *Proc. DAC*, pp. 32–35, 1990.

[36] 山本芳嗣, 久保幹雄. 巡回セールスマン問題への招待. 朝倉出版, 1997.