連続最適化問題への温度並列シミュレーテッドアニーリングの応用

Application of The Temperature Parallel Simulated Annealing to Continuous Optimization Problems

三木光範 , 笠井 誠之（知的システムデザイン研究室）
Mitsunori MIKI , Masayuki KASAI（Intelligent Systems Design Laboratory）

Abstract The temperature parallel simulated annealing(TPSA), which has been found so far to be very effective for combinatorial optimization problems using parallel computers, is applied to continuous optimization problems. A new treatment for determinig the neighborhood, the maximum and the minimum temperature is considered based on the design space and the accuracy of the solutions. The continuous optimization problems are the minimization of one of the standard test functions and the minimum-volume design of structures. The number of temperature stages is 64, and the computer used is a PC cluster with 8 processors. The numerical experiments showed that the perfomance of TPSA was remarkable compared with the conventional SA with very slow cooling for the standard test function, and was very good for the realistic structural optimization problems. Further, the parallel efficiency is fairy good.

1　はじめに

連続設計変数の空間における最適化問題に対しては，これまで非線形数理計画法による解法が主流であった．設計空間が単峰で，しかも目的関数の勾配が連続である場合にはこのアプローチは極めて有効であるが，そうでない場合には遺伝的アルゴリズム（GA），シミュレーテッドアニーリング（SA），あるいはタブーサーチ（TS）など，いわゆるヒューリスティックサーチ（heuristic search）[1] とよばれる方法が用いられることも多くなってきた．このことは，従来の方法で解ける問題の範囲を超えて，連続変数空間といえども目的関数が極めて複雑な挙動を示す最適化問題が取り扱われ始めたことを意味している．なかでも，GA と SA はこのような手法の双璧であり，連続最適化問題に対しても多くの研究が行われてきた [2].

複雑な連続最適化問題に対して GA と SA のどちらが有効かという問題は難しい．なぜなら，これらの方法はいずれもその問題に適したスキームを採用し，多くのパラメータを適切にチューニングしなければ良い結果が得られないからである．しかしながら，一般的に言えることは，設計空間内に多くのサイズの大きい局所解領域が存在する場合には GA が有効であり，一方，設計空間全域的には単峰に近いが，サイズの小さい局所解領域が無数に存在する場合には SA が有効である [3].

SA[4] は，炉内の固体の熱的平衡状態をシミュレーションするための単純なアルゴリズム [5] を基本として最適化問題を解く方法であり，多くの組合せ最適化問題の解法として有用である [6]．SA の長所は，(a) ほとんど任意の非線形性を持つコスト関数を処理できる，(b) ほとんど任意の境界条件や制約条件が処理できる，(c) 他の非線形最適化アルゴリズムと比較してコード化は極めて容易である，(d) 最適解の発見が統計的に保証されている，ことである．一方，SA の短所は，(a) 最適解を求めるのに要する時間が長い，(b) 特定の問題対してチューニングするのが容易でない，(c) 過大評価されて用いられ，結果の解釈が間違っている場合がある，(d) 誤った利用によりエルゴード性を失う，すなわち冷却が早く，シミュレーテッドクエンチング（simulated quenching）になっている．この場合は最適解が求められる統計的な保証はない，ことである [7]．要するに，GA より単純なアルゴリズムで，計算機へのインプリメントも極めて容易であるが，良い最適解を得るには非常に長い時間がかかる，すなわち計算負荷が極めて高いということである．

計算時間が長いことは SA の最大の欠点である．たとえば，巡回セールスマン問題では SA で良好な近似解を得る計算量よりも完全な総当たり計算の方が計算量が少ない [8]．さらに厄介な問題は，適切な冷却スケジュール（cooling schedule）（温度スケジュールともいう）が不明で，かなりの予備実験を行わなければならない点である．

SA の計算時間を短縮するには二つのアプローチがある．ひとつはできるだけ高速の冷却スケジュールを考えることであり，もう一つは並列処理である．前者については Rosen と中野の解説 [9] に詳しく述べられているが，対数型の Boltzmann アニーリングより，双曲線型の高速アニーリング，さらには指数型の超高速アニーリングを用いることにより，SA の高速化が可能となる．一方，SA の並列処理は並列計算機の発達とともに有効なアプローチとして多くの研究がなされている [10]．この中で最も典型的な方法は異なった初期値で通常の SA を並列に行い，ある時間ごとに最も良好な解を全プロセッサに渡し，並列に近傍探索するものである．一方，SA と GA を組み合わせた方法は，それぞれの方法の長所を活かし，さらに並列化が容易であることから多くの研究が行われている [11,12,13]．しかしながら，いずれの方法においても SA の冷却スケジュールが経験的にしか与えられないという問題は常に残る．

これに対して，温度並列 SA[14] は並列処理との高い親和性を持っているだけでなく，温度スケジュールが原理的に不要であるという極めて優れた特長を有している．このため，温度並列 SA はこれからの SA の発展に欠かせない手法と考えられるが，これまでは LSI ブロック配置問題 [15] や巡回セールスマン問題 [16] に応用されただけで，その有効性は広く確認されておらず，特に連続最適化問題への応用はこれまでなされていなかった．

本研究は，このような背景の下に，連続変数を持つ最適化問題に温度並列 SA を応用する方法を提案し，二つの代表的な連続最適化問題に適用した結果を基に温度並列 SA の有効性を検証することを目的とする．また，それらの結果を基に温度並列 SA の改良についても示唆する．

2 温度並列 SA

温度並列 SA[14] は，複数のプロセッサに異なる温度を与え，各プロセッサは一定温度でアニーリングを行い，一定の間隔で隣接する温度のプロセッサ間で解の交換を行う方法である．この方法の特長は，(a) 温度を解自身が決定するので温度スケジュールの自動化が図れる，(b) 時間的に一様なので任意の時点で終了が可能であり，また，継続すれば解の改善を続けることができる，(c) 解の品質を劣化させることなく，温度数までの並列化が可能である，という点にある．

図 1 は温度並列 SA の概念図であり，通常の SA と

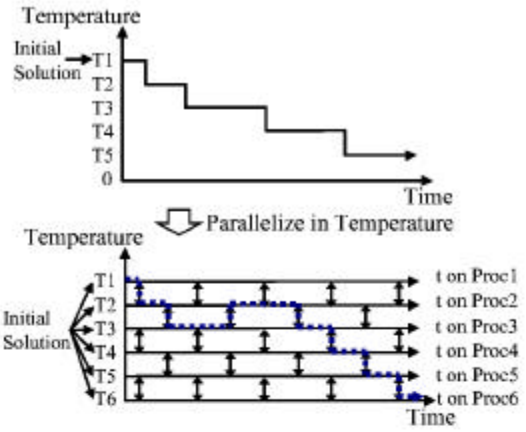

図 1: 逐次 SA と温度並列 SA

比較している．上側に示した通常の SA では実験的に決めた単調減少の温度スケジュールを用いるのに対して，温度並列 SA では各プロセッサは一定の温度を担当し，解が自身のエネルギーを基準として適切な温度を選ぶ．このため，温度スケジュールは不要となる．しかも，最低温度での解の状態を観察していれば終了判定は容易である．すなわち，最低温度での解の更新が長い期間にわたってなければ最適解が求められたと判断できる．一方，通常の温度スケジュールを持つ SA では，最適解かどうかに拘わらず温度が下がれば解の更新がなくなる．したがって，できるだけ緩慢に冷却する必要があるが，対象としている問題に対してどの程度が緩慢な冷却かという判断は，多くの実験を行ってみなければ分からない．ここに温度並列 SA の最大の特長がある．

温度並列 SA における隣接温度での解の交換は式 (1) を用いて確率的に行う．すなわち，隣接する温度 T と T' における解のエネルギーを E と E' とすると，高温部に低いエネルギーの解が存在した場合には無条件で解を交換し，それ以外でも温度差とエネルギー差などから計算される確率で解を交換する．これによって，低温部にエネルギーの低い解が集まるが，確率的にはそうでない場合もあることになる．

$$P_{EX}(T,E,T',E') = \begin{cases} 1 & if \quad \triangle T \cdot \triangle E < 0 \\ \exp\left(-\dfrac{\triangle T \cdot \triangle E}{T \cdot T'}\right) & otherwise \end{cases} \tag{1}$$

一方，各一定温度における SA は通常の方法で行う．すなわち，近傍探索における受理確率は式 (2) で与えられる Metropolis 規準 [4] を用いる．

$$P_{AC} = \begin{cases} 1 & if \quad \triangle E < 0 \\ \exp\left(-\dfrac{\triangle E}{T}\right) & otherwise \end{cases} \qquad (2)$$

$$\triangle E = E(\boldsymbol{x_{new}}) - E(\boldsymbol{x_{old}})$$

3 連続最適化問題への温度並列SAの適用

SAは組合せ最適化問題の有力な解法として提案され，広く用いられてきた．しかしながら，連続最適化問題においても対象とする問題の複雑度が高い場合には多く用いられている [3,7,17,18,19]．SAが有効な問題は，目的関数の勾配が連続でなく，設計空間内において大局的には単峰に近いが，局所的にはサイズが比較的小さい局所解領域が極めて多いような問題である．

連続最適化問題と組合せ最適化問題ではSAにおける近傍の概念およびその定義が異なる．組合せ最適化問題では，解の変更に必要な最小の操作の集合を近傍と定義することが一般的である．たとえば，巡回セールスマン問題ではグラフの４つのノードと２つの辺の関係を組み替えて，新しいグラフを作ることが可能である．これはX-交換（X-change）[20]とよばれ，これによって得られる新しいグラフの集合をX-交換近傍（X-chnage neighborhood）とよぶ．この近傍をもとにして近似的に最適なグラフを求める解法が2-Opt法 [21]である．こうして組み合わせ最適化問題では近傍は厳密に定義できる．

一方，連続最適化問題では設計変数が連続であり，上のように簡単に近傍を定義することができない．連続設計変数空間における解の変更は，近くであろうと遠くであろうと数値を変更するだけであり，組合せ最適化問題でのように，操作的に近傍を定義することはできない．そのかわり，物理的に意味のある，すなわち目的関数の連続性における近傍を考えることは容易である．組合せ最適化問題での近傍は目的関数とは何の関係もない．そこで，連続最適化問題では一般に現在の解を中心とし，移動距離に関する確率分布を与え，近傍を定義する．この確率分布としてはBoltzmannアルゴリズムでは正規分布 [19]，FSAではコーシー分布 [17]，VFSAではペンの先の形のように中央部で尖っており，両側で分布を切断した特殊な形の確率分布 [3]を，そして適応的な一様分布 [18]が用いられる場合もある．これらの研究から明らかになったことは，用いる確率分布は近傍探索が十分に行えるように，中央部で厚く，離れたところで急速に減少する，あるいは確率を0にするものが良好で，しかもその分布の幅を受理確率によって適応的に変化させるものが最良であるということである．すなわち，近傍探索において受理の頻度が高すぎると近傍が小さすぎて無駄な探索が多くなり，一方受理が少なすぎると近傍が大きすぎて無駄な探索が多くなることになり，受理の頻度を基に近傍の大きさを適応的に変化させることが重要となる．

しかしながら本研究では近傍として最も単純な正規分布を用い，しかもその幅を温度の関数とした．すなわち，現在の解からの摂動 $\triangle x$ を式 (3) の分布で与えた．これはBoltzmannアルゴリズムで用いられる分布である．

$$g_k(\triangle \boldsymbol{x}) = \frac{1}{(2\pi T_k)^{D/2}} \exp\left(\frac{-|\triangle \boldsymbol{x}|^2}{2T_k}\right) \qquad (3)$$

ここでTkはk番目の温度であり，kは通常のSAでは温度スケジュールの番号を表し，温度並列SAでは複数の温度の一つを表す．

次に最高温度を考える．組合せ最適化問題では一般的に，最大の改悪が生じる状態遷移がある値（たとえば50％）の確率で受理されるという考え方が用いられる [14]．しかしながら，連続最適化問題ではそうした考え方はもはや有効ではない．その理由は，一般的に最大の改悪が不明であること，および近傍を定義した確率分布で最大を考えると非現実的な改悪を考えなければならなくなるからである．たとえば，正規分布を用いれば近傍の最大は無限大の距離になってしまい，一様分布を考えてもその両端部分の値をとる確率は小さく，これがn次元となるとそれが同時に生じる確率はほとんど0となってしまうからである．したがって，連続最適化問題では組合せ最適化問題とは異なるアプローチで最高温度を決める必要がある．

ここでは次のようにして最高温度を決定した．すなわち，設計変数がとりうる値で決まる設計空間は工学的問題では現実的な制約条件から通常はある程度の大きさで与えることが可能であり，この設計空間を最高温度状態で充分に探索可能であるという条件を基に決定する．ここでは近傍の分布として式 (3) の正規分布を用いているため，各変数の標準偏差がその変数の設計領域の 1/4 となるようにした．これにより，現在の解が設計空間の中央にあるときにはそれが端まで移動する確率は 4.6 %であり，現在の解が設計空間の端にある時に他の端まで移動する確率は 0.01 %程度となる．最高温度における２次元の

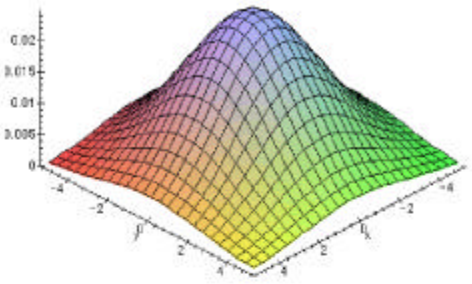

図 2: 最高温度時の正規分布 (2 次元)

近傍の確率分布を図 2 に示す．これより，最高温度では設計空間が十分に探索できていると考えられる．

温度並列 SA では温度スケジュールが不要であるかわりに最低温度を決める必要がある．組合せ最適化問題では最小の改悪がある確率で受理されるという規準で考える [14] が，連続最適化問題ではそうした考え方はできない．これは最高温度と同様である．このため，最低温度は解の精度を規準に決定した．すなわち，式 (3) で与えられる近傍が充分小さい温度ということである．工学的には温度が下がりすぎて近傍が非現実的に小さくなっていることは意味がない．多くの SA の応用で温度スケジュールだけを適当に決めて，かなりの繰り返しを行って解が変動しなくなったのを観察して停止条件としている場合も多いが，そのときの温度を調べると非現実的に小さくなっており，最適解が見いだされたから解が変動しなくなっているのではなく，温度が下がりすぎたから変動しなくなったいるだけ，という場合も多いように思われる．これは意味のないことである．ここでは正規分布の標準偏差が設計空間の 1/10000 程度になる温度を最低温度と考えた．

次にエネルギーについて考える．制約条件のない最適化問題では目的関数に適当なスケーリングファクターを乗じたものをエネルギーとすればよい．制約条件がある最適化問題では目的関数に制約条件に関するペナルティー関数を加えた疑似目的関数を作成し，それに適当なスケーリングファクターを乗じたものをエネルギーとすればよい．この場合のスケーリングファクターは受理確率を規準に決める．

組合せ最適化問題では，先に述べたように，最大の改悪が生じる状態遷移が 50 %の確率で受理されるというような考え方でスケーリングファクターを決める．連続最適化問題ではこうした考え方はできないので，次のように考える．すなわち，解の摂動と目的関数の関係が分かっている場合にはある程度の値で起こり得る可能性のある目的関数の改悪が評価できるので，その改悪を受理する確率が 50 %になるように決める．一方，そうでない場合は，近傍を定義した確率分布を用いて実際に試行を行い，ある試行回数のなかで生じる目的関数の最大の改悪の平均値を実験的に求め，それを 50 %で受理するようにスケーリングするのが合理的と考えられる．ここではこの考え方を用いた．

4 並列計算機への実装

温度並列 SA では解の交換の時にプロセッサ間通信が発生するだけなので，並列計算機との親和性が良好である．ここでは 8 プロセッサの PC クラスタ (Pentium II, 350 MHz × 8) を用いて計算を行った．プロセッサ間通信は Fast Ethernet と PVM を用いて行った．

一つの温度での SA を一つのプロセスに割り当てる．したがって，温度数が 8 までのときは一つのプロセッサに一つの温度プロセスが実行されているが，温度数が多くなると一つのプロセッサで複数の温度プロセスが実行される．温度数が 64 の場合は一つのプロセッサで 8 プロセスが実行されている．

5 数学的関数の最小化問題

連続最適化問題として最初に種々のヒューリスティックアルゴリズムの性能検証に用いられる典型的な数学関数 [22] の最小化問題を考える．ここでは式 (4) で表される 5 次元の Rastrigin 関数を用いる．

$$f(x_i|_{i=1,N}) = (N \times 10) + \left[\sum_{i-1}^{N} \left(x_i^2 - 10\cos(2\pi x_i)\right)\right] \tag{4}$$

この関数は 2 次元で示すと図 3 のように，多くの比較的大きなサイズの局所解領域を持つ．このような関数の最小化問題は SA より GA で解くのが有効である．しかしながら，ここでは SA にとって難しい問題をあえて選んで温度並列 SA の有効性を検証することにした

各設計変数の範囲は-5.12 から +5.12 とし，近傍は式 (3) で与えられる正規分布とした．先に述べたように，設計変数の範囲の 1/4 が近傍生成における正規

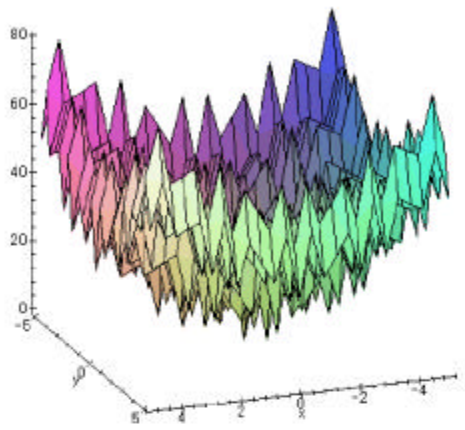

図 3: Rastrigin 関数の景観 (2 次元)

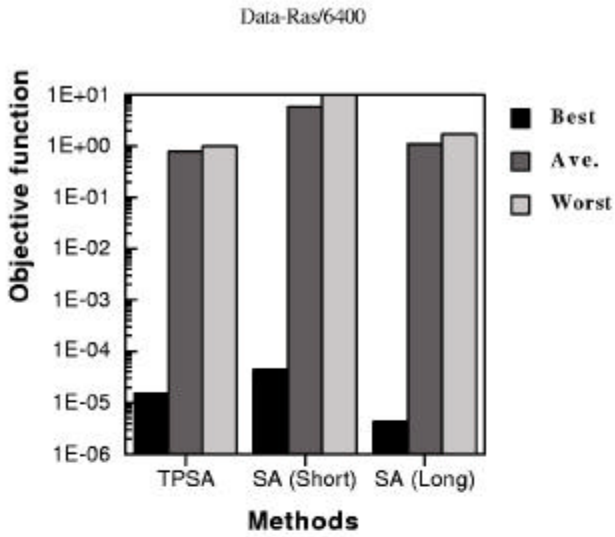

図 4: 手法の比較 (64000 × 64)

分布の標準偏差になるように最高温度（=6.554）を決めた．一方，最低温度は，先に述べたように，解の精度で決める．ここでは正規分布の標準偏差が 0.001 となるように決めた．これで分解能は設計空間に対して 1.0E-4 となる．また，このとき最低温度は 1.0E-6 となる．

目的関数からエネルギーへの変換は次のようにした．すなわち，このような問題では設計変数の摂動と目的関数の変化には直接の関係はなく，最大の改悪は実験的に求めることになる．ここでは，式 (3) の近傍を用い，実際に乱数を用いて 100 回の試行を 5 回行い，100 回の試行における近傍での目的関数の最大の改悪の平均を実験的に得た．この値は 66.4 であり，この改悪が 50 %の確率で受理されるようにエネルギーのスケーリングファクターを決めた．この値は 0.684 となる．

温度数は 64 とした．組合せ最適化問題では温度数は 32 程度あれば良好な結果が得られている [14]．中間の温度は最高温度と最低温度，ならびに温度数から温度が等比的に並ぶように決めた．解の交換周期は 40 とし，計算の繰り返しは 6400 回および 12800 回行った．また，温度並列 SA の性能を評価するために通常の SA を行った．この場合，温度並列 SA と同じ条件とした．すなわち，冷却における温度段数は 64 とし，決めた繰り返し数で最低温度に到達するように冷却率を決め，指数型の冷却を行った．ここでは冷却率は 0.78 となる．たとえば計算繰り返し数が 12800 回の場合では 200 回ごとに温度が一段低くなる．通常の SA を用いる場合には計算回数に関して 2 通りの考え方がある．すなわち，並列処理と同じ計算量となるようにして，温度並列 SA と同じ計算回数の試行を温度数（=64）だけ行う方法と，通常の SA の温度スケジュールを極めて緩慢にして温度数（=64）倍の時間をかける方法である．ここでは前者を SA (Short)，後者を SA (Long) とよぶ．なお，試行はすべて 5 回行い，その最良値，平均値，最悪値で議論する．

図 4 は総計算回数が 64000 × 64 回の場合の得られた最適解の目的関数の値を示したものである．まず，平均値で考えると，温度並列 SA は最も優れた性能を示している．Rastrigin 関数の最小値は 0 であり，1 は 2 番目に良好な最適解である．このため，温度並列 SA は 2 番目の最適解を確実に見つけていることが分かる．これに対して 6400 × 64 試行の通常の SA では極めてまれに真の最適解の近傍に達しているが，大部分はかなり悪い局所解にトラップされていることが分かる．一方，冷却時間を 64 倍した通常の SA では温度並列 SA に近い性能を示している．これは，SA では冷却が緩慢であればあるほど，良好な結果が得られることから予想される結果である．しかしながら，温度並列 SA は 64 倍の時間をかけた通常の SA より優れた性能を示していることは驚くべきことである．

この傾向は図 5 に示す総計算回数が 128000 × 64 回の結果を見ればさらに明確となる．温度並列 SA ではすべての試行で真の最適解の近傍に達しているのに対して，長時間の通常の SA では図 4 の結果とほとんど変わらない．また，128000 回の繰り返しの通常の SA を 64 試行行った結果も図 4 と比較して目立った改善はない．一方，最良値は偶然に得られるもので計算スキームや計算関数には関係ないことがわかる．

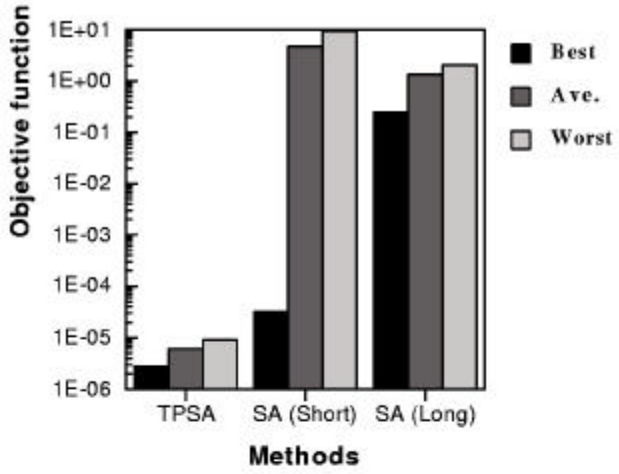

図 5: 手法の比較 (128000 × 64)

この結果より，温度並列 SA は少し時間をのばせば性能が劇的に上昇することが分かった．これこそ，温度並列の特長である．すなわち，通常の SA は最初に決められた温度スケジュールで行うので，総計算回数が決まってしまい，それを延長してもすでに温度が下がっているので意味はない．このため，多くの予備実験を行って温度スケジュールを最適化しなければならない．たとえばここでの結果では SA (Long)，すなわち 64 倍の時間をかけて冷却しても解の改善は図 4 および 5 より僅かであり，さらに長時間が必要と考えられる．一方，温度並列 SA では 64000 回での結果を見て，まだ不足と考えられればそれをそのまま継続するだけで図 5 の結果を得ることができる．しかも，すべての試行がほとんど同じになったことから真の最適解が求められたと判断できる．

図 6 は 128000 × 64 のケースについて計算時間を比較したものである．温度並列 SA と SA (Short) を比較すると，温度並列 SA は解の交換という操作のための時間とプロセッサ間通信のための時間が余分にかかっているにも拘わらず，計算時間は 1.49 倍になっているだけである．これだけの時間増分で温度並列 SA は非常に大きな性能向上を示すことが分かる．一方，SA (Long) では非常に長い時間をかけて緩慢に冷却しているにも拘わらず，解の改善は少ない．このことから温度並列 SA は高い並列性で良好な最適解が出るまでの時間を非常に短縮できることが分かる．

Data-Ras/6400*2

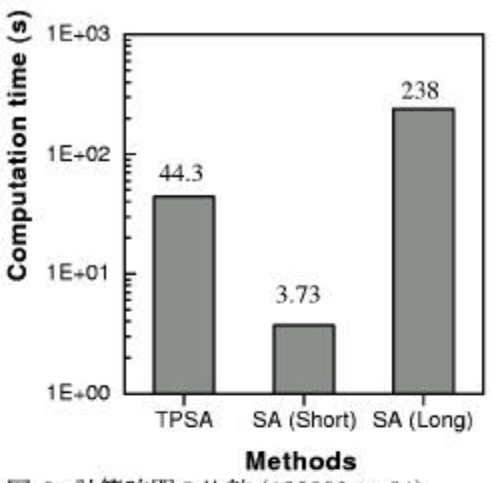

図 6: 計算時間の比較 (128000 × 64)

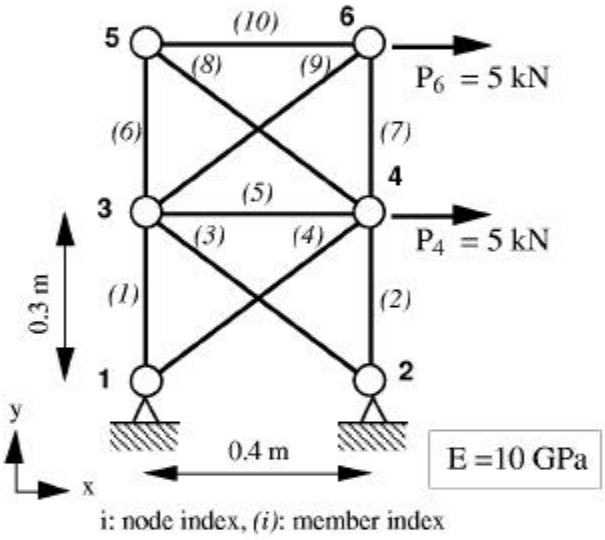

図 7: 10 部材トラス構造と負荷条件

6 構造最適化問題

次に，上の問題とはまったく異なり，現実的な最適化問題での温度並列 SA の性能を検証する．ここでは構造最適化問題として典型的なトラス構造最小体積問題を考える．図 7 は対象とする 10 部材トラスである．問題は，各部材の引張破壊と圧縮の座屈破損が生じないこと，および節点 6 の変位が規定値 (=5.8mm) 以下であること，という制約条件の下で目的関数である構造の体積を最小化することである．設計変数は断面形状を円とする各部材の断面積とする．この問題についての詳しい説明は紙数の関係で文献 [23] に譲る．

近傍の生成には前節と同様に式 (3) の正規分布を用いる．また，その標準偏差が設計領域の 1/4 となるように最高温度を決める．ここでは設計変数の範

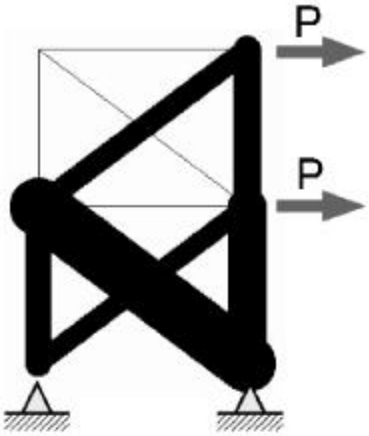

図 8: 最適解の断面積分布

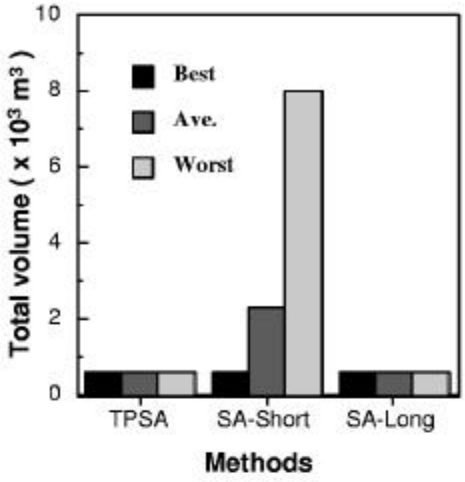

図 9: 手法の比較 (6400 × 64)

囲を 1 から 4000mm^2 と考え，標準偏差が 1000 となるように最高温度を考えた．最高温度は 1.0E6 となる．また，最低温度は解の精度を 1mm^2 と考えることで 1 となる．

すなわち，構造の体積にスケーリングファクターを乗じたものに局所制約に関するペナルティーのエネルギーと，全体制約に関するペナルティーのエネルギーを加えたものをエネルギーと考える．ここで，スケーリングファクターは可能性のある体積の改悪が最高温度においてある一定の確率（=50 %）で受理されるようにする．

局所制約に関するペナルティーのエネルギーは局所制約を破っている部材の数の和にスケーリングファクターをかけて求める．この場合，一つの部材が破損するという状態遷移を最高温度において 50 %の確率で受理するようにそのファクターを決めた．また，全体制約に関しては，制約を破った場合に変位の 2 乗にスケーリングファクターを乗じてペナルティーのエネルギーとした．ここでは変位制約が僅かに破られるという状態遷移を最高温度において 50 %の確率で受理するようにそのファクターを決めた．

計算に用いたスキームならびに計算機は前節と同様である．温度並列 SA によって得られた最適解の一例を図 8 に示す．初期値は乱数で与え，5 回の試行を行ったが，すべて同様の最適解が得られた．総計算量は 6400 × 64 である．通常の SA における冷却率は 0.803 となる．

図 9 に得られた構造の体積を示す．これより，温度並列 SA は SA (Long) と同じ性能を有していることが分かる．一方，温度並列と同じ関数だけのアニーリングを 64 試行した結果より大変良好な結果となっている．この結果から，前節の数学的関数の最小化と同様に，現実的な構造最適化問題においても温度並列 SA が良好な結果を出すことが明らかとなった．なお，この問題では長時間 SA でも真の最適解を見つけているため，温度並列 SA との相違が出なかった．もう少し少ない計算回数で比較すれば前節同様，温度並列 SA と長時間 SA の相違が明白になると予想される．いずれにしても，温度並列は SA としては少なすぎる回数で真の最適解を出す能力があることが認められる．

図 10 は温度並列 SA における一つの温度内でのエネルギーの変化を示したものである．ここでは 64 温度のうち，最高温度の T63（温度=1.0E6），中間の T40（温度=6449），および最低温度の T0（温度=1）を考える．初期解はほぼ同じエネルギーであるが，最高温度ではエネルギーは上昇して一定値の周りを変動し，一方，中間の温度ではエネルギーは減少しつつも最適な値には到達しない．これに対して，最低温度でのエネルギーは解の交換によって着実に減少していることがわかる．温度並列 SA では最低温度での解の挙動を観察することで最適解が得られたことが分かる．ここでは 2000 回ぐらいからあとはほとんど解のエネルギーに変化がなく，このため 3000 回もしくは 4000 回ぐらいで終了判定ができる．

図 11 は温度並列 SA と SA (Short) の場合の温度スケジュールを比較したものである．ここで並列温度 SA では最も良好な解が得られた場合の，その解がたどった温度履歴を示した．これを見ると最適解は結果的に複雑な温度履歴によって得られていることがわかる．しかし，この結果を組合せ最適化問題での温度並列 SA において最良の解が経過した温度スケジュール [14] と比較するとかなり単調であることがわかる．これは，この問題のエネルギー空間が比較的単純な形状をしていることに起因すると思わ

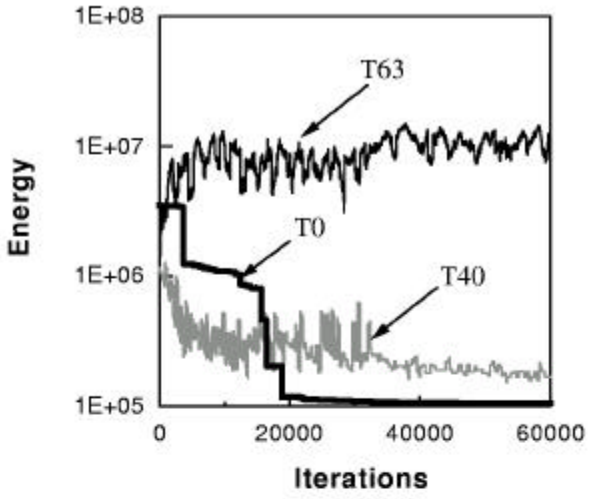

図 10: 一定温度内のエネルギー変化

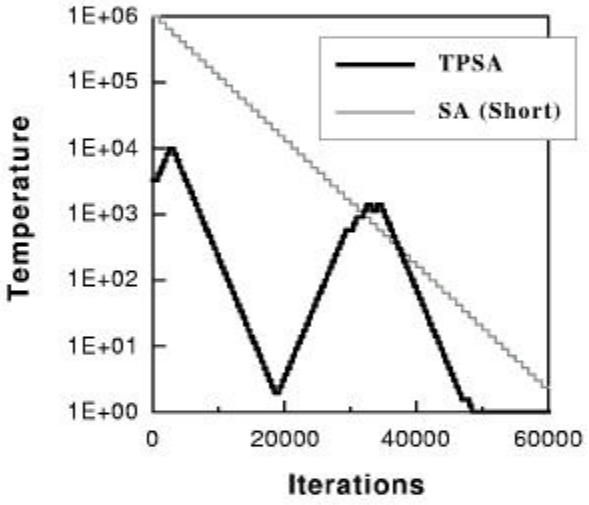

図 11: 温度スケジュール

れる．

7 結論

本研究では，これまで組合せ最適化問題にしか用いられてこなかった温度並列シミュレーテッドアニーリング（TPSA）法を連続最適化問題に適用する場合の問題点を抽出し，それを解決する提案を行い，二つの異なった種類の連続最適化問題に応用し，その手法の有効性を検証した．得られた結論は以下の通りである．

1）連続最適化問題に TPSA を適用する場合の近傍，最高温度，および最低温度について議論を行い，近傍を正規分布で与えた場合の標準偏差を設計空間との関係で決定し，それを用いて最高温度を決定する方法を提案した．また，最低温度は必要な解の精度で決める方法を提案した．また，最大改悪となるエネルギーについては決定した分布を基に実験的に求める方法を提案した．

2）複雑な連続最適化問題として典型的な数学的関数のなかで SA では性能が出にくい問題を選んで提案した方法に基づいて TPSA を行い，TPSA が極めて優れた性能を示すことを明らかにした．すなわち，TPSA は極めて長時間の通常の SA よりも性能が高いことが確認できた．

3）現実的な連続最適化問題として構造最適化問題を考え，これに提案した方法を用いて TPSA を適用し，この方法が長時間の SA と同様に真の最適解を極めて短時間で見いだすことを確認した．

4)TPSA に用いた計算機は 8-CPU の PC クラスタであるが，通常の SA と比較して TPSA は複雑な操作とプロセッサ間通信を行っているにも拘わらず，計算時間は少し増加しただけであった．一方，解の品質は計算時間がはるかに長い長時間の SA と比較しても良好だった．これより，TPSA は連続最適化問題においても極めて短時間で良好な最適解を見つける能力があることがわかった．

参考文献

[1] Reeves, C.R. 編，横山，奈良ら訳：モダンヒューリスティックス，日刊工業新聞社 (1997)

[2] Schwefel, H.P.: Evolution and Optimum Seeking, John-Wiley & Sons, Inc., New York (1995)

[3] Ingber, L.: Genetic Algorithms and Very Fast Simulated Reannealing: A Comparison, Mathematical and Computer Modeling, 16(11), pp. 87-100 (1992)

[4] Kirkpatrick, S., Gelett Jr. C.D., and Vecchi, M.P.: Optimization by Simulated Annealing, Science, Vol. 220, No. 4598, pp. 671-680 (1983)

[5] Metropolis, N. Rosenbluth, A., Rosenbluth, M., Teller, A., and Teller, E.: Equation of State Calculation by Fast Computing Machines, Journ. of Chemical Physics, Vol. 21, pp. 1087-1092 (1953)

[6] Aarts, E. and Korst, J.: Simulated Annealing nad Boltzmann Machines, John Wiley & Sons (1989)

[7] Ingber, L.: Simulated Annealing: Practice versus Theory, Journal of Mathl. Comput. and Modelling, Vol. 18, No. 11, pp. 29-57 (1993)

[8] 文献6の p. 54

[9] Rosen, B.E., 中野良平:シミュレーテッドアニーリングー基礎と最新技術ー，人工知能学会誌，Vol. 9, No. 3, pp. 365-372 (1994)

[10] Holmqvist, K., Migdalas, A., and Pardalos, P.M.: Parallelized Heuristics for Combinatorial Search, in Parallel Computing in Optimization, Migdalas, A. et al. eds, Kluwar Academic Publishers, p. 269 (1997)

[11] Chen, H. and Flann, N.S.: Parallel Simulated Annealing and Genetic Algorithms: a Space of Hybrid Methods, in Parallel Problem Solving from Nature, Davidor, Y. et al. eds., Springer-Verlag, pp. 428-438 (1994)

[12] Yong, L., Lishan, K., and Evans, D.J.,: The Annealing Evolution Algorithm as Function Optimizer, Parallel Computing, Vol. 21, pp. 389-400 (1995)

[13] Kurbel, K., Schneider, B., and Singh, K.,: Solving Optimization Problems by Parallel Recombinative Simulated Annealing on a Parallel Computer - An Application to Standard Cell Placement in VLSI Design, IEEE Trans. on Systems, Man, and Sybernetics - Part B: Cybernetics, Vol. 28, No. 3, pp. 454-461 (1998)

[14] 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応用，情報処理学会論文誌，Vol. 36, No. 4, pp. 797-807 (1995)

[15] 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ー LSIブロック配置問題に関してー，情報処理学会 DA シンポジウム'94, pp. 223-228 (1994)

[16] 小西健三，屋舗正史，瀧 和男：温度並列 SA 法による巡回セールスマン問題の解法，Parallel Comuputing Workshop'96, pp.P2-R-1-8 (1996)

[17] Szu, H. and Hartley, R.: Fast Simulated Annealing, Phissics Letters A, Vol. 122, No. 3,4, pp. 157-162 (1987)

[18] Corana, A., Marchesi, M., Martini, C., and Ri8della, S.: Minimizing Multimodal Functions of Continuous Variables with the Simulated Annealing Algorithm, ACM Trans. on Mathematical Software, Vol. 13, No. 3 pp. 262-280 (1987)

[19] Rosen, B.: functional Optimization based on Advance Simulated Annealing, IEEE Workshop on Phisics and Computation, PhysComp 92 (Dallas, Texas), pp. 289-293 (1992)

[20] Papadimitriou, C.H. and Steiglitz, K.: Combinatorial Optimization -Algorithms and Complexity-, Prentice-Hall, p. 460 (1982)

[21] 情報処理学会編：情報処理ハンドブック，オーム社，p. 119 (1997)

[22] Whitley, D., Mathias, K., Rana, S., and Dzubera, J.: Evaluating Evolutionary Algorithms, Artificial Intelligence, Vol. 85, pp. 245-2761 (1996)

[23] 三木光範：並列分散最適化のためのアルゴリズム-資源の追加と削減に基づく方法-，情報処理学会並列処理シンポジウム JSPP'98, pp. 263-270 (1998)