論　文

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

小西　健三[†]　　屋鋪　正史[††*]　　瀧　　和男[†††]

An Application of Temperature Parallel Simulated Annealing to the Traveling Salesman Problem and Its Experimental Analysis

Kenzo KONISHI[†], Masafumi YASHIKI[††*], and Kazuo TAKI[†††]

あらまし　本論文では組合せ最適化問題のための新しい汎用アルゴリズムとして提案済みの温度並列SA法を，巡回セールスマン問題に適用し，本アルゴリズムを実験的に評価した．温度並列SA法は既にさまざまな応用問題に適用／利用されているものの，他の優れた発見的手法との比較が行われていない，いずれの適用例も最適解未知の問題である，という点から詳細な評価が行われていなかった．そこで本論文では温度並列SA法を実質的に組合せ最適化問題のベンチマークである巡回セールスマン問題に適用し詳細な評価を行った．まず温度並列SA法とその巡回セールスマン問題への適用方法について報告する．そしてランダムデータを用いた評価結果から，2-opt近傍という非常に基本的かつ古典的な近傍を用いているにもかかわらず，他の手法と比較して優れた結果が得られることを確認した．また実行時間についても並列計算機上での実行により，実用的な時間で実行可能であることを示すことができた．更に実用面での有効性を調べるために，ベンチマーク問題集であるTSPLIBから50個のデータを選び実験を行った結果，ランダムデータと同様良好な結果を得た．

キーワード　温度並列SA法，並列処理，組合せ最適化，巡回セールスマン問題

1. まえがき

近年，工学のさまざまな分野でNP困難[5]のクラスに属する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．そこで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発見的手法の重要性が高まってきている[6], [7], [13], [18].

我々は並列処理と非常に親和性の高い，組合せ最適化問題のための新しい汎用アルゴリズムとして温度並列シミュレーテッドアニーリング法（Temperature Parallel SA：以下，温度並列SA法またはTPSA）[12]を提案し，その最適化能力について[14]で報告した．[14]ではブロック配置問題と呼ばれるLSI-CAD分野の代表的な組合せ最適化問題に対して，逐次SA法との比較を交え報告した．[14]の結論を簡単にまとめると以下の二つになる（詳細は[14]参照）．

- アニーリングステップ数を等しくした場合の比較で，温度並列SA法は逐次SA法より優れた最適化能力をもつ．
- 計算機資源（CPU時間）を等しく（実行時間を等しく）した場合の比較で，温度並列SA法は逐次SA法より優れた最適化能力をもつ．

しかし，更にアルゴリズム自体の詳細な評価を行うためには，以下のような条件を満足している問題を扱う必要があると考えた．

（1）　最適解（あるいは下界）が既知

（2）　比較すべきアルゴリズムの実装方法が明確

本論文ではこれらの条件を満足する問題として，巡回セールスマン問題（以下，TSP）を対象とし，温度

† 神戸大学大学院自然科学研究科知能科学専攻，神戸市
Division of Intelligent Science, Graduate School of Science and Technology, Kobe University, Rokkodai-cho, Nada-ku, Kobe-shi, 657 Japan
†† 神戸大学大学院自然科学研究科情報知能工学専攻，神戸市
Division of Computer and Systems Engineering, Graduate School of Science and Technology, Kobe University, Rokkodai-cho, Nada-ku, Kobe-shi, 657 Japan
††† 神戸大学工学部情報知能工学科，神戸市
Department of Computer and Systems Engineering, Faculty of Engineering, Kobe University, Rokkodai-cho, Nada-ku, Kobe-shi, 657 Japan
* 現在，シャープ株式会社

並列SA法の詳細な実験的解析を行うことを目的とした．TSPは実質的に組合せ最適化問題のベンチマークであり，古くからさまざまな手法が報告されている [16], [18]．また，専用，汎用を問わず優れた手法も数多く報告されており [16]，比較評価も行いやすいと考えたためである．

従来のTSPの近似解法の研究は，以下の二つに大別できると考えられる．すなわち，(a) TSP専用の近似解法（2-opt法，Lin-Kernighan法など）の研究，(b) 汎用の近似解法の研究（SA, Tabu Search, GAなど），である．このうち (b) に属する研究は，大きな枠組をもつ解法の中にTSP依存の部分を組み込み実装を行う．そして実質的に組合せ最適化問題のベンチマークであるTSPに適用することで，他の問題に適用する際のさまざまな知見を得，アルゴリズムの絶対性能を実験的解析により評価するものである．本論文もこの (b) に含まれるものである．

以下，本論文の内容を示す．まず2.では，SA法，温度並列SA法について簡単に説明する．次に3.ではTSPの定義と温度並列SA法のTSPへの適用方法について示す．そして4.では温度並列SA法と他のアルゴリズムとの比較評価をランダムデータを用いて行う．更に5.ではTSPのベンチマーク集であるTSPLIBに適用した結果を報告する．最後に6.でまとめを行う．

なお本論文は，並列処理シンポジウムJSPP'96投稿論文 [20] の一部を抜粋し，大幅な計測データの追加と加筆を行ったものである．

2. SA法と温度並列SA法

2.1 S A 法

シミュレーテッドアニーリング法（以下，SA法）[13] は，広範囲の組合せ最適化問題に適用可能な汎用の近似解法であり，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]．SA法は繰返し法の一種であるが，評価関数値の改良方向への遷移のみではなく，改悪方向への遷移を確率によって制御するところに特徴がある．改悪方向への遷移を認めることにより局所解に陥りにくくし，良質の解を得ることができる．しかしSA法で得られる解は，SA法を特徴づける「温度」と呼ばれる制御変数のスケジュール（以下，温度スケジュール）に大きく依存することも報告されている [1]．温度スケジュールに関する研究も行われているが，さまざまな問題に適用可能な温度スケジュールは明らかにされていない．

また一般にSA法で良質の解を得るには，実行時間の長さが問題となっており，通常は得られる解品質と実行時間のトレードオフ点を採用することになる．SA法の長い実行時間を解決するためのアプローチとして種々の並列SA法も提案されている [2]．しかし，SA法はマルコフ連鎖を次々とたどる処理であるため強い逐次性をもち，並列化は容易ではない．また，スピードアップを得るために漸近収束を保証する理論を崩す手法も存在するが，その評価結果の是非は分かれている [17]．

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，解品質の劣化の少ない並列SA法ということになるが，これに対する一つの新しいアプローチとして温度並列SA法 [12], [14] がある（図1）．温度並列SA法は，SA法や従来の並列SA法と比較して次のような優れた点をもっている [12], [14]．

- 温度スケジュールの自動化
- アルゴリズムの時間的一様性：時間の関数となるパラメータが存在しない．
- 並列処理との高い親和性：後述するように，高いスピードアップが得られる．

以下，温度並列SA法について簡単に説明する．温度並列SA法の最大の特徴は「温度スケジュールの自動化」にある（図1）．逐次SA法では温度を温度スケ

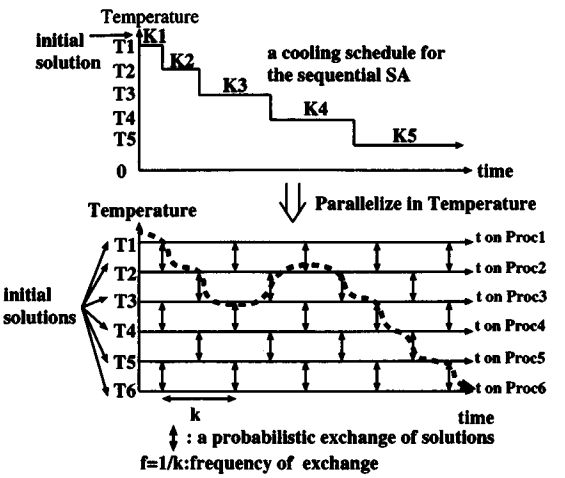

図1 逐次SA法と温度並列SA法
Fig. 1 Simulated Annealing and Temperature Parallel Simulated Annealing.

ジュールに従って単調減少させる．一方，温度並列SA法では，相異なる温度を担当するそれぞれのプロセスで，一定温度下でのアニーリングを同時並行に行う．このとき，逐次SA法で温度 T から T' に冷却することは，温度並列SA法では温度 T を担当するプロセスと温度 T' を担当するプロセスの間で解を交換することに相当する（図1）．この解の交換は，各プロセスが一定温度のSA処理を一定回数（以下，交換周期と呼ぶ．図1中の k）繰り返した後に行う．今，温度 T を担当するプロセスが評価関数値 E の解をもち，温度 T' を担当するプロセスは評価関数値 E' の解をもつとすると，上述の交換確率は式（1）に従う [12], [14].

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

但し，$\Delta E = E' - E, \Delta T = T' - T$ である [12], [14]．この確率的交換を行うことで，複数の解が自分自身に適した温度スケジュールを自動的に決定することになる．

3. TSPへの適用

3.1 巡回セールスマン問題

本論文で扱う対称巡回セールスマン問題は以下のように定義される．N 個の点 $V = \{v_1, v_2, \cdots, v_N\}$ と距離関数 $d(v_i, v_j)$ が与えられたとき，すべての点をただ一度経由する巡回路 π（Hamilton閉路）のうち，式(2)で与えられる巡回路長を最小にするものを求める．

$$\sum_{i=1}^{N-1} d(v_\pi(i), v_\pi(i+1)) + d(v_\pi(N), v_\pi(1)) \quad (2)$$

ここで $v_\pi(i)$ は，ある巡回路 π 上で i 番目の点を表す．また対称巡回セールスマン問題においては，$1 \leqq i, j \leqq N$ なる i, j に対して $d(v_i, v_j) = d(v_j, v_i)$ である．以降，本論文では対称巡回セールスマン問題をTSPと略す．

3.2 近傍構造

繰返し法に属するアルゴリズムを構築する際に決定すべき事項として近傍がある．ここでは温度並列SA法をTSPに適用する際に用いた近傍について説明する．

TSPの近傍にはさまざまなものがあるが [18]，本論文では温度並列SA法というアルゴリズムの絶対性能を詳細に評価するために，最も古典的で基本的な近傍である2-opt近傍を用いた．もちろん温度並列SA法は汎用アルゴリズムであるので，他の近傍を用いることも可能である．

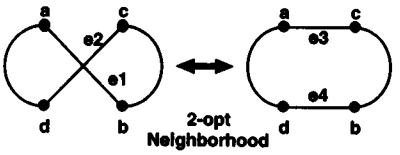

図2 2-opt近傍
Fig. 2 2-opt neighborhood.

ここで2-opt近傍について説明する．今，ある巡回路 π の2-opt近傍 $\mathcal{N}_2(\pi)$ とは，π 上の二つの枝 $e_1 = (a, b)$, $e_2 = (c, d)$（但し，b, d は π 上で a, c の次の点）を，$e_3 = (a, c)$, $e_4 = (b, d)$ に置き換え，bc間（あるいはda間）のすべての点を逆順にした巡回路の集合である（図2）．

本論文における実装では2-opt近傍を用い，巡回路長が減少していれば巡回路を更新するという手続きを繰り返す．この際，SA法で用いる確率に従って，改悪方向への更新も受理するようにしている．

3.3 データ構造

繰返し法に属する手法では近傍の探索を効率良く行う必要がある．本論文では3.2で示した2-opt近傍を用いており，これを効率良く探索することのできるデータ構造としてK-d木を採用した．K-d木についての詳細は [3], [4] に譲るが，ここではK-d木上で定義されている関数を用いて，2-opt近傍が効率良く探索可能なことのみを示す [3], [4].

最初に本論文で使用する基本的な関数，変数の説明を以下に与える．

next(v)： 現在の巡回路上で点 v の次の点を返す関数

dist(v_1, v_2)： 点 v_1, v_2 間の距離関数

flip(a, b, c, d)： b = **next(a)**，d = **next(c)** なる関係にある4点a, b, c, dを与えると，図2の2-opt交換を行う関数

Tour[i]： 巡回路を表す配列．Tour[i] は現在の巡回路 π 上で $v_\pi(i)$ を保持

次にK-d木上で定義されている二つの関数 **deletept(v)**, **undeletept(v)** について説明する．今，N 個の点 $\{v_1, v_2, \cdots, v_N\}$ に対してK-d木が既に構築されているとする．このとき **deletept(v_i)** は点 v_i をK-d木上で消去する．逆に **deletept(v_i)** されている点 v_i に対して **undeletept(v_i)** を行うことで，再度K-d木上に復帰させることができる（図3(a)）．ま

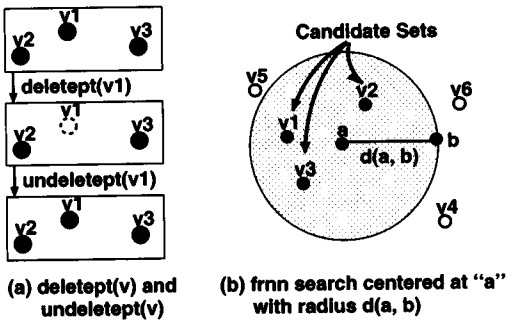

図3 K-d木上の deletept(v), undeletept(v), frnn search
Fig. 3 deletept(v), undeletept(v), and frnn search on the K-d tree.

た関数 **frnn**(c, r, $\mathcal{S}$) (Fixed Radius Near Neighbor search) は，点 c を中心として半径 r 以内に存在する点を $\mathcal{S}$ に登録する機能をもつ(注1) (図 3 (b))．

上述した K-d 木上の三つの関数を用いて 2-opt 近傍を効率良く探索できる手法について説明する．図 2 を用いて説明すると，任意の点 a を選択すると点 b は b=**next**(a) となり一意に決定される．しかし点 c は a,b を除いた残りの $N-2$ 点すべてが候補となり，探索空間が膨大なものになる．そこで，効率良く点 c を決定するために，本論文の実装においては以下の手続きを用いた [3]．

```
1   for given a, b = next(a);
2   deletept(a); deletept(b);
3   frnn(a, dist(a,b), N2);
4   select c ∈ N2;
5   d = next(c);
```

つまり点 b=**next**(a) として，点 c の候補集合を，点 a を中心とし半径 **dist**(a,b) の円内に存在する点（但し，a,b は関数 deletept により消去されている）の集合 $\mathcal{N}_2$ に限定することになる．これは巡回路長が短くなるような 2-opt 近傍に遷移するためには，

$$dist(a,b)+dist(c,d) > dist(a,c)+dist(b,d)$$

が成立する必要があり，そのためには

$$dist(a,b) > dist(a,c) \text{ または } dist(c,d) > dist(b,d)$$

の少なくとも一方が成立する必要があるという事実に基づいている．本論文の実装では前半の不等式に当てはまる点を，点 c の候補として探索していることになる．

3.4 温度並列 SA 法のアルゴリズム

本節では本論文で実装した温度並列 SA 法のアルゴリズムの詳細について述べる．温度並列 SA 法は一定温度の逐次 SA 法が同時に複数個実行されるので（逐次プログラムとして実装する場合はループに展開する），アルゴリズムの記述は主に交換周期（図 1 の k）内の一定温度の SA 処理部分とした（並列計算機上での実装方法については別途報告予定）．なお，本リストで新たに用いる変数，関数は以下のとおりである．

T_i : i 番目のプロセスが担当する温度

len_i : i 番目のプロセスが担当する巡回路の巡回路長を保持する変数．本論文では式 (1) の評価関数値 E は巡回路長に相当する．

get_gain(a,b,c,d) : 2-opt 交換によるゲインを計算する関数．ここで 2-opt 交換におけるゲインとは，図 2 を用いて説明すると

$$dist(a,c)+dist(b,d)-dist(a,b)-dist(c,d)$$

で表される量である．すなわち 2-opt 交換前後での巡回路長の変化を表す．

exchange_solution() : 温度並列 SA 法の解交換を行う関数

rand() : [0:1) の一様乱数を生成する関数

以上の説明から，温度並列 SA 法のアルゴリズムは以下のようになる（STOP_CRITERION, STOP_CRITERION2 については **3.5** で具体的に説明する）．

```
procedure TPSA for TSP
1  Set Initial Tour and len_i;
2  for (sc = 0; sc < STOP_CRITERION;
       sc++)
3    parallel_do (for Each Process with T_i)
4      j = 0;
5      for (sc2 = 0;
            sc2 < STOP_CRITERION2;
            sc2++)
6        a = Tour[j]; b = next(a);
7        deletept(a); deletept(b);
8        frnn(a, dist(a,b), N2);
9        undeletept(a); undeletept(b);
```

(注1)：ここでは frnn を 2-opt 近傍の探索に限定しているので，一般形については [3] を参照．

```
10          c = {x | x ∈ N2, ∀y ∈ N2\x,
                    dist(a,x) ≦ dist(a,y)};
11          d = next(c);
12          gain = get_gain(a,b,c,d);
13          if ((gain < 0) ||
14              (rand() < exp(−gain/Ti)))
15              flip(a,b,c,d);
16              leni + = gain;
17          j = (j+1)mod N;
18      parallel_do end
19      exchange_solution();
20  end
```

3.5 パラメータの設定

本節では，温度並列SA法のパラメータについて説明する．温度並列SA法はパラメータとして，交換周期 k，温度数，終了条件（交換回数）をもつ．ここでは予備実験の結果より得られたパラメータの値と，**3.4** で用いたSTOP_CRITERION, STOP_CRITERION2との関係について説明する．

交換周期 k： 都市数の20倍の遷移 [11]．つまり **3.4** で示したリスト中のSTOP_CRITERION2は都市数 N の問題に対して $20N$ となる．

温度数： 32温度 [14]

終了条件： 温度数の5倍だけ解交換が行われた時点．つまり **3.4** で示したリスト中のSTOP_CRITERIONは温度数（ここでは32温度）×5となる．

ここで終了条件についてのみ説明を行う．温度並列SA法では，異なる温度を担当するすべてのプロセスそれぞれに初期解を与える．この際，最も高温の温度を担当するプロセスに与えられた初期解が，最終的な局所探索を行う最低温度を担当するプロセスに到達するには，最低でも温度数だけの解交換が必要である．このことから温度並列SA法の終了条件は温度数の定数倍であると考えた．そして予備実験により，得られる巡回路長と実行時間の関係から，温度数の5倍が妥当な値であるという結果を得た．

3.6 比較用アルゴリズムの実装

本論文で実装した温度並列SA法の実験的解析を行うに当って，比較用アルゴリズムとして以下のものを同じデータ構造（K-d木）の上に実装した．以下，近傍と共に示す．まずTSP専用の近似解法として2-opt法 (2-opt近傍)，3-opt法（3-opt近傍）[16], [4]，Lin-Kernighan法（LK近傍）（以下，LK）[15]，また汎用の近似解法としてSA法（2-opt近傍）[13]，Tabu Search（2-opt近傍）（以下，TS）[6], [7]を選んだ．これらを選んだ理由は，いずれも実装方法がほぼ明らかなこと，従来の報告で実行時間と得られる巡回路長の両面から考えた場合，平均して良好な結果が得られているためである．これらのうちパラメータを含む解法については，別途実験を行うことでチューニングを施した．

また温度並列SA法が並列（並行）アルゴリズムであるにもかかわらず，TSPの並列処理近似解法を比較アルゴリズムとして実装していないのは，報告例が少ないこと，並列計算機のアーキテクチャ依存のものが多いこと，実装方法（特にパラメータ）が明確でないなどの理由からである．更に比較対象の逐次プログラムの並列化も考えられるが，これらは近傍の構造から，並列処理によるスピードアップと大域的な最適化という二つの要求が衝突するため，効率の良い実装は不可能であると判断したため，実装を行っていない．一方，Multiple-Runと呼ばれる手法に属するものは一つの逐次手法を複数の初期解から出発するため並列化は容易であるが，純粋な並列アルゴリズムではないと判断して実装は行わなかった．

4. 実験的解析

4.1 実行環境

本論文の逐次プログラムはすべてサン・マイクロシステムズ社のSPARC station 2（SPARC, 40 MHz）上で実行した．また温度並列SA法を並列実行する場合は，(株) 富士通研究所の並列計算機AP1000（SPARC 25 MHz）[10]上で行った(注2)．

4.2 逐次プログラムの評価

本節では温度並列SA法と他のアルゴリズムの比較評価を行う．比較用アルゴリズムがすべて逐次プログラムであるため，温度並列SA法も逐次プログラムとして実装することが望ましいと考えた．そこで上述プログラムリストの3行目部分を，ループに展開することで逐次プログラムとして実装を行った．以下，このプログラムを逐次プログラムとして実装した温度並列SA法ということで，C-TPSA（Concurrent -TPSA）と呼ぶことにする．

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．

(注2)：プログラムはすべてcc -O2で最適化コンパイルしたものを使用した．

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. 表 1 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 [8], [9] からの超過パーセンテージ (以降,「下界からの距離」と呼ぶ), 実行時間の平均値を示す. なお本論文で用いる「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.

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 1 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 1 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 4 に示す (数値の詳細については付録の表 5 に示す). 図 4 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, 解の分布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. ところで, 温度並列 SA 法は温度数までの並列性を内在しているが [14], 本節では逐次アルゴリズムとして実装しているためその並列性を利用していない. そこで次節において, 本アルゴリズムを並列計算機上で実装した場合の他のアルゴリズムとの比較評価を行う.

表 1 下界からの平均距離と平均実行時間
Table 1 Average percentage over the Held-Karp lower bound and average running times on SPARC station2 (SPARC 40 MHz).

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

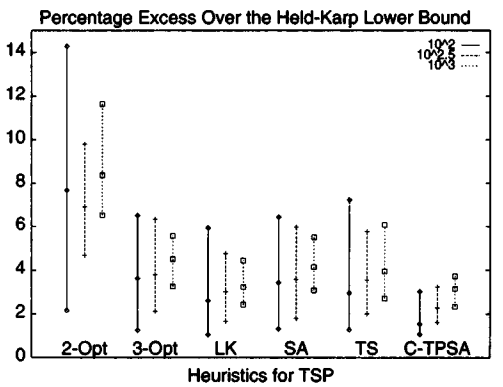

図 4 100 回の実行に対する下界からの距離の分布
Fig. 4 Distribution of the percentage over the Held-Karp lower bound for each heuristics.

4.3 並列プログラムの評価

温度並列 SA 法は, 各プロセスが一定温度での SA 処理を並列に実行することが可能である. つまり温度数までの並列性を内在していると言える. またプロセッサ間通信も交換周期で行うだけでよく, 並列処理と非常に親和性が高いアルゴリズムである. 本論文では温度数を 32 としているため, 32 台の CPU が必要となるが, そのような並列計算機は一般的に分散メモリ型並列計算機であると考えた.

我々は分散メモリ型並列計算機上での温度並列 SA 法の効率の良い実装方法として, 非同期温度交換モデルを提案し, 試作した [19], [20]. ここでは実装方法については省略し結論のみを示す. 表 2 に温度並列 SA 法をそれぞれ 1CPU, 32CPU で実行した場合の結果を示す. 表中 25/40 の行については後述する. 表 2

表 2 AP1000 (SPARC 25 MHz) 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)
Table 2 Average percentage over the Held-Karp lower bound and average running times in seconds on the parallel computer AP1000(SPARC 25 MHz).

	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.

から得られる結論は，32温度の温度並列SA法の実行を32CPUの下で実行すると，解品質を劣化させることなく，最悪でもC-TPSAの約33倍のスピードアップが得られるということである [19], [20].

以上の結論を前提として，温度並列SA法と他のアルゴリズムとの比較を行う．まず，並列計算機上での実験を行った結果が表2であり，この結果と表1の結果を比較する．ここで表2の結果を表1の結果と比べるためには，表2の32CPUの行の実行時間を25/40倍したものを用いればよい(注3)．すると32CPUを用いた温度並列SA法の実行時間はそれぞれ，表2の25/40の行に示した実行時間となる．これらの結果を表1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をも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づけられる．

表3　TSPLIBの50データに対して20回の実行を行った場合の最適解からの距離
Table 3　Percentage over the optimal tour length during 20 runs for TSPLIB 50 data.

Data	TPSA			LK	SA	3Op.
	Best	Ave.(s.d.)	Wor.	Ave.	Ave.	Ave.
att48	0.00	0.19(0.18)	0.66	1.60	1.42	2.04
*att532	1.66	2.67(0.49)	3.24	2.04	3.43	3.36
d1655	2.29	2.87(0.21)	3.21	3.36	3.06	4.24
d493	1.23	1.59(0.16)	1.95	1.94	2.57	2.96
d657	0.76	1.43(0.58)	2.47	2.51	2.49	3.90
eil101	0.00	0.17(0.19)	0.48	2.71	1.28	3.26
eil51	0.00	0.21(0.07)	0.23	1.74	1.66	2.23
eil76	0.00	0.09(0.18)	0.56	1.95	2.20	3.48
fl417	0.06	0.60(0.49)	1.52	1.55	3.82	1.80
gil262	0.55	1.53(0.41)	2.14	1.82	3.14	3.70
kroA100	0.48	0.64(0.14)	0.94	1.30	1.49	2.50
kroA150	0.21	1.42(0.63)	2.53	1.76	3.22	2.34
kroA200	0.58	0.98(0.53)	2.34	1.56	2.42	2.37
kroB100	0.37	0.61(0.15)	0.93	1.67	2.14	1.79
kroB150	0.33	0.93(0.37)	1.42	1.29	3.43	3.16
kroB200	1.24	2.06(0.75)	4.01	2.09	3.82	4.19
kroC100	0.00	0.41(0.28)	0.86	1.53	2.16	3.36
kroD100	0.28	0.71(0.62)	2.22	1.52	2.86	2.76
kroE100	0.16	0.65(0.33)	1.14	1.55	2.75	2.96
lin105	0.55	1.08(0.41)	1.76	1.34	3.33	2.20
linhp318	2.99	3.62(0.44)	4.11	4.36	4.64	5.28
nrw1379	1.59	2.05(0.23)	2.28	2.21	2.61	3.65
*p654	0.90	1.72(0.44)	2.61	1.33	5.47	2.75
pcb1173	2.50	2.85(0.20)	3.12	3.04	3.77	5.01
pcb442	0.58	0.90(0.20)	1.35	2.48	2.18	3.63
pr107	0.08	0.38(0.17)	0.54	0.94	3.06	0.52
†pr124	1.26	1.84(0.47)	3.36	1.64	3.62	1.90
pr136	0.25	0.50(0.16)	0.81	2.09	2.04	6.49
pr144	0.00	0.45(0.30)	1.63	1.82	1.30	0.82
pr152	0.00	0.31(0.23)	0.77	1.34	1.83	1.89
pr226	1.21	2.44(1.13)	4.43	1.03	4.06	2.57
*pr2392	2.43	3.09(0.29)	3.83	3.04	3.68	4.36
pr264	0.20	1.32(1.35)	4.51	3.02	3.41	2.51
pr299	0.93	1.79(0.49)	2.83	2.01	2.94	3.58
pr439	1.00	2.84(1.28)	4.94	2.45	4.41	4.30
pr76	0.10	0.19(0.08)	0.43	1.53	1.61	3.15
rat575	1.27	1.95(0.27)	2.41	2.34	3.04	3.98
rat783	1.76	2.26(0.30)	2.99	2.75	3.15	3.70
rd100	0.57	1.45(0.54)	2.55	1.21	3.44	2.93
*rl1304	3.23	5.37(1.12)	6.83	3.37	7.00	5.76
*rl1323	2.91	3.76(0.54)	4.70	2.57	4.62	3.79
*rl1889	2.48	3.73(0.74)	5.04	3.51	5.03	4.22
ts225	0.40	0.91(0.39)	1.59	1.49	1.43	1.52
u1060	1.71	2.33(0.26)	2.78	2.76	3.43	3.38
u1432	0.97	1.26(0.12)	1.49	2.37	1.50	1.96
u159	0.39	0.62(0.29)	1.33	1.96	2.65	3.59
u1817	2.21	3.11(0.51)	3.98	4.24	4.54	4.01
u574	2.10	2.62(0.29)	3.05	2.67	3.48	3.71
u724	1.43	2.09(0.30)	2.62	2.32	2.98	3.28
*vm1084	2.51	3.16(0.32)	3.51	2.45	3.80	3.29

表中 * はLKの方が良質の結果を示したもの．

5. TSPLIBの実験結果

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 [16]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．実験結果を表3に示す．なお実験結果は各アルゴリズムで各データに対して20回の実行を行った場合の平均であり，温度並列SA法のみ最良値，平均値（標準偏差），最悪値を示したものである．また，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．

表3の結果より，温度並列SA法で得られた巡回路長はTSPLIBのデータに対しても非常に良好な結果が得られていることがわかる．これは多くのデータで最良値が最適解から1%以内であることからも判断できる．またスペースの都合から他手法の最良，最悪値のデータは割愛したが，ランダムデータを用いた実験で見られた，得られる解の分布が非常に小さいという傾向は，TSPLIBのデータについても見られた．更に表3の温度並列SA法の部分について整理し，表4の

(注3)：$10^{2.5}$, 10^3 のデータについては，実行環境の制約からAP1000上1CPU（SPARC 25 MHz）での実行が不可能であった．しかし試行実験と 10^2 のデータから，SPARC station2（SPARC 40 MHz）での実行時間を40/25倍した時間がAP1000上1CPUの実行時間であると考えてよいという結論を得た．

ようにまとめた．表 4 より 50 データのうち 7 データで最適解に到達，更に全データの 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．

最後に表 3 中の * の付いたデータに対して考察を行う．これらのデータは LK 法が最良の結果を得ているものである．そのようなデータの中の一つである p654 について LK 法で得られた最良の結果を図 5 に示す．

図 5 にも見られるように，表 3 中 * のデータは他のデータと比較して，点が密な部分と疎な部分の差が大きいことがわかった．このようなデータに対して温度並列 SA 法で良好な結果が得られていない理由として近傍の問題が考えられる．2-opt 近傍では疎な部分を中心に考えている場合は全体に摂動を与えるような動作になり，また密な部分を中心に考えている場合は，近くに存在している点集合のみが近傍となってしまう．そのため 2-opt 近傍では大域的な探索がうまく行えていないと考えられる．一方 LK 法で良好な結果が得られているのは，LK 近傍は，幅優先，深さ優先探索を巧みに組み合わせ，改善の見込みがない部分については探索を打ち切る操作を行っており，2-opt 近傍より高度な探索を行っているためであると考えられる．温度並列 SA 法で LK 近傍を用いることも考えられるが，今後の課題とする．

表 4 温度並列 SA 法の TSPLIB データに対する解の分布
Table 4 Distribution of TPSA results for 50 TSPLIB data.

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%)

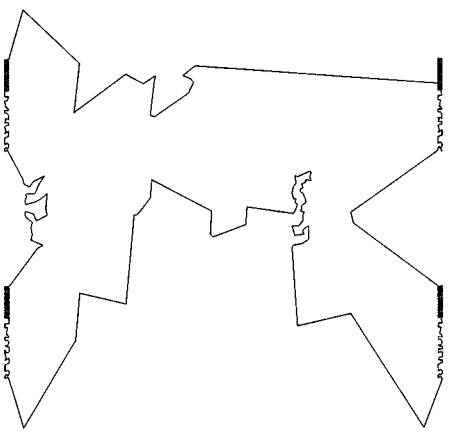

図 5 LK で得られた最良の p654 の出力
Fig. 5 p654 best solution with LK in our experiment.

6. むすび

近年，組合せ最適化問題に対する解法の重要性と共に，最適解に近い解を実用的な時間で得ることのできる発見的手法の重要性が高まってきており，さまざまな手法が報告されている [6], [7], [13], [18].

我々は並列処理と非常に親和性の高い，組合せ最適化問題のための新しい汎用アルゴリズムとして温度並列 SA 法 [12], [14] の最適化能力について [14] で報告した．しかし，更にアルゴリズム自体の詳細な評価を行うためには，以下のような条件を満足する問題を扱う必要があると考えた．

（1） 最適解（あるいは下界）が既知

（2） 比較すべきアルゴリズムの実装方法が明確

そこで本論文では温度並列 SA 法を TSP に適用することで，詳細な評価を行った．まず一様乱数により生成したランダムデータを用いて逐次プログラムと並列プログラムを用いた評価を行った．逐次プログラムを用いた評価結果は，2-opt 近傍という非常に基本的な近傍を用いているにもかかわらず，他のアルゴリズムと比較して非常に良好な結果が得られていることがわかった．また逐次プログラムにおける実行時間の長さも，並列計算機上で実行することにより，実用的な時間で実行可能となることを示した．更に得られる巡回路長に関しては，専用アルゴリズムである LK 法で得られる程度の巡回路長は十分に得られていることも示した．

次に，実用面での有効性を調べるために，TSP のベンチマーク集である TSPLIB から 50 データを選んで実験を行った．ここでも温度並列 SA 法で得られた解が十分良質であることを示した．具体的には，50 データ中 7 データで最適解に到達し，平均値においても，20 データで最適解から 1%以内という結果であった．しかし，点の疎密の偏りが大きなデータに対しては有効な結果が得られなかった．理由としては 2-opt 近傍が考えられ，実際，更に複雑な近傍を用いている LK 法では，良好な結果が得られていることがわかった．温度並列 SA 法で LK 近傍を用いられることも考えられるが，今後の課題とする．

本論文では温度並列SA法のTSPに対する適用法と評価結果を示したが，TSPは組合せ最適化問題の非常に基本的な問題であり，本論文で得られた結果は他の組合せ最適化問題に対して温度並列SA法を適用する際にも有益であると考えられる．

謝辞 巡回セールスマン問題，およびその評価方法に関して御指導，御助言を頂きました，東京商船大学流通情報工学課程の久保幹雄助教授に深謝致します．また常日ごろ，御助言を頂く神戸大学工学部情報知能工学科の金田悠紀夫教授に感謝致します．最後にAP1000の使用に当り御世話になりました（株）富士通研究所並列処理センターの皆様に感謝致します．

文　献

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

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

[3] J.L. Bentley, K-d trees for semidynamic point sets. Proc. Sixth Annual ACM Symp. on Comp. Geometry, pp.187–197, June 1990.

[4] J.L. Bentley, "Fast algorithms for geometric traveling salesman problems," ORSA J. Computing, vol.4, no.4, pp.387–411, Fall 1992.

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

[6] F. Glover, "Tabu search—part I," ORSA J. Computing, vol.1, no.3, pp.190–206, 1989.

[7] F. Glover, "Tabu search—part II," ORSA J. Computing, vol.2, no.1, pp.4–32, 1990.

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

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

[10] H. Ishihata, T. Horie, S. Inano, T. Shimizu, and S. Kato, "An Architecture of highly parallel computer AP1000," Proc. IEEE Pacific Rim Conf. on Comm., Comput. and Signal Processing, pp.13–16, May 1991.

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

[12] 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，" 信学技報，NC90-1, March 1990.

[13] S. Kirkpatrick, C.D. Gelatt, Jr., and M.P. Vecchi, "Optimization by simulated annealing," Science, vol.220, no.4598, pp.671–680, May 1983.

[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評価，" 情処学論，vol.36, no.4, pp.797–807, April 1995.

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

[16] G. Reinelt, "The Traveling Salesman–Computational Solution for TSP Applications," Lecture Notes in Computer Science 840, Springer-Verlag, 1994.

[17] B.E. Rosen, 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，" 人工知能誌，vol.9, no.3, pp.365–372, May 1994.

[18] 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.

[19] 小西健三，屋鋪正史，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，" Proc. 5th Parallel Computing Workshop, Japan, pp.P2-R-1-8, March 1996.

[20] 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，" JSPP'96, pp.153–160, June 1996.

付　録

実験データの詳細

表A·1 各手法の100回の実行で得られたHeld-Karpの下界からの超過パーセントの最良，平均（標準偏差），最悪

Table A·1 Best, average (standard deviation), and worst percentage excess over the Held-Karp lower bound during 100 runs for TSP heuristics.

Heuristics for TSP	Number of Cities (Points Uniform in the Unit Square)								
	10^2			$10^{2.5}$			10^3		
	Best	Ave.(sd.)	Worst	Best	Ave.(sd.)	Worst	Best	Ave.(sd.)	Worst
2-Opt	2.16	7.66(2.58)	14.27	4.67	6.90(1.21)	9.80	6.52	8.37(1.09)	11.64
3-Opt	1.26	3.62(1.26)	6.51	2.11	3.78(0.89)	6.32	3.25	4.49(0.53)	5.56
LK	1.05	2.59(0.91)	5.93	1.66	3.01(0.61)	4.75	2.42	3.23(0.43)	4.44
SA	1.32	3.43(1.03)	6.43	1.80	3.58(0.81)	5.97	3.08	4.15(0.45)	5.51
TS	1.29	2.96(1.10)	7.23	2.00	3.54(0.77)	5.77	2.71	3.95(0.60)	6.07
C-TPSA	1.07	1.54(0.33)	3.01	1.61	2.28(0.32)	3.23	2.33	3.14(0.25)	3.72
Para-TPSA	1.05	1.47(0.30)	2.27	1.55	2.25(0.30)	3.02	2.31	3.11(0.32)	3.72

（平成8年5月17日受付，9月17日再受付）

小西　健三　（学生員）

平4神戸大・工・システム卒．平6同大大学院修士課程了．現在，同博士後期課程在学中．並列処理，LSI-CAD，組合せ最適化等に興味をもつ．情報処理学会，日本OR学会会員．

屋鋪　正史

平6神戸大・工・システム卒．平8同大大学院修士課程了．同年，シャープ株式会社入社．並列処理，言語処理系に興味をもつ．

瀧　　和男　（正員）

昭51神戸大・工・電子卒．昭54同大大学院修士課程システム工学修了．工博．同年（株）日立製作所入社．昭57（財）新世代コンピュータ技術開発機構に出向．逐次型および並列型推論マシンと並列応用プログラムの研究開発に従事．平2同機構第1研究室室長．平4年9月神戸大学工学部情報知能工学科助教授．平7年4月同学科教授．並列マシンのアーキテクチャ，並列プログラミング，LSI-CAD等に興味をもつ．情報処理学会，IEEE，ソフトウェア科学会，ACM各会員．