情報処理学会論文誌 Vol. 51 No. 7 1234–1244 (July 2010)

シミュレーテッドアニーリングプログラミングの温度並列化

三木光範[†1] 松井勇樹[†2] 小畑拓也[†4]
廣安知之[†3] 吉見真聡[†1]

本論文では，シミュレーテッドアニーリングプログラミング（SAP）を温度並列化した温度並列 SAP を提案し，その有効性を検証する．SAP とは，シミュレーテッドアニーリング (SA) を木構造が扱えるように拡張した自動プログラミング手法であり，温度と呼ばれる制御パラメータにより，解は改良方向だけでなく確率的に改悪方向へも遷移する．この SAP において，解を探索する上で，重要となる温度の決定は容易ではない．そこで，本研究では，温度スケジュールの決定に有効な温度並列アルゴリズムを SAP に適用した．提案手法を自動プログラミングにおける代表的なベンチマーク問題に適用し，並列 SAP，および重要温度領域を用いた一定温度での逐次 SAP との比較を行った．その結果，並列 SAP および逐次 SAP と比べて，温度並列 SAP が最も良好な性能を示した．また，TPSAP は逐次 SAP より解の探索時間が非常に短く，かつ逐次 SAP と同様にノード数の少ない解の生成が可能である．

Temperature-Parallel Simulated Annealing Programming

Mitsunori Miki,[†1] Yuki Matstui,[†2] Takuya Kobata,[†4]
Tomoyuki Hiroyasu[†3] and Masato Yoshimi [†1]

In this paper, we propose Temperature-Parallel Simulated Annealing Programming(TPSAP), that is an extension of the Simulated Annealing Programming(SAP) using the Temperature-Parallel concept. SAP is an automatic programming method that expands the Simulated Annealing(SA) in order to handle the tree structures, and the solution changes not only to the improvement direction but also to the deterioration direction in probabilistic for temperature. For SAP, the determination of the important temperature parameter for the effective searching of a solution is not easy. We applied the Temperature-Parallel algorithm which was effective in the determination of temperature scheduling. We compared TPSAP with Parallel SAP(PSAP) and SAP for some benchmark problems of automatic programming. As a result, TPSAP is found to give the best solution in the same numbers of evaluation. The time to obtain the the optimum solution with TPSAP is shorter than SAP, and the optimum solution obtained with TPSAP have as small node sizes as SAP.

1. はじめに

ロボットの行動を制御する行動規則や関数などのプログラムをコンピュータによって自動生成する研究が注目されている．これは，コンピュータを用いることにより，あらかじめ人が想定できない状況に対応できるプログラムを生成できることや，複数台のロボットが協調行動するような大規模・複雑なプログラムを容易に生成できるからである．

このようなプログラムをメタヒューリスティック手法を用いて自動生成する自動プログラミングの代表的な手法として，Koza により提案された遺伝的プログラミング（Genetic Programming: GP）[1] がある．GP は遺伝的アルゴリズム（Genetic Algorithm: GA）[2] の遺伝子型を，木構造などの構造的表現が扱えるように拡張した手法であり，LISP の S 式のような木構造で記述できるプログラムを自動生成することが出来る．自動プログラミングにおける最適化問題として，ロボットの制御プログラム[3),4)] や，株価予測[5]，画像処理[6] など様々な分野に応用されているが，その際，用いられる探索手法として GP が一般的である．しかし，多くの最適化問題には，問題に適した探索手法が用いられることから，自動プログラミングにおいても，問題に応じて探索手法を使い分けることで，より効率的に最適化が行える可能性がある．

これに対して，著者らはシミュレーテッドアニーリング (Simulated Annealing: SA)[7] を木構造が扱えるように拡張した手法であるシミュレーテッドアニーリングプログラミング (Simulated Annealing Programming: SAP)[8] の研究を行っている．SAP は GP の問題点であるブロート[⋆1]の発生原因である交叉を用いないことで，ブロートが生じることなく GP と同等の性能が得られることを報告した[8]．また，SAP は比較的簡単な問題に対して，少ないノード数で良好な解を生成できることから，リアルタイムでの最適化に適していると考

†1 同志社大学理工学部
Faculty of Science and Engineering, Doshisha University
†2 同志社大学大学院工学研究科
Graduate School of Engineering, Doshisha University
†3 同志社大学生命医科学部
Faculty of Life and Medical Sciences, Doshisha University
†4 同志社大学工学部
Faculty of Engineering, Doshisha University

⋆1 探索が進むにつれてプログラムサイズ (ノード数) が劇的に増大すること．ブロートの発生は探索の停滞や評価時間の増大に繋がる．

えられる．この SAP では，温度と呼ばれる制御パラメータにより改良方向だけでなく確率的に改悪方向へも遷移するメカニズムを持つため，SAP の探索性能は温度パラメータが大きく影響する．そのため，探索に適した温度スケジュールの決定が重要となるが，適切な温度スケジュールの決定は予備実験や人間の経験的判断が必要となるため，容易ではない．

そこで本稿では，SAP の基となる SA において，適切な温度スケジュールを自動的に決定する有効な手法である温度並列 SA(Temperature Parallel SA: TPSA)[9),10)] を SAP に適用した温度並列 SAP(Temperature-Parallel SAP: TPSAP) を提案する．そして，TPSAP の有効性を検証するために，自動プログラミングの代表的なベンチマーク問題に対して，並列 SAP および逐次 SAP と比較を行う．なお，本稿では木構造で表すことのできるプログラムを対象とする．

2. シミュレーテッドアニーリングプログラミング (SAP)

SAP は，金属の焼き鈍しを模倣した進化的最適化手法である SA を木構造が扱えるように拡張したプログラム探索手法であり，GP における突然変異をベースに探索を行う．

SAP は，GP の突然変異と同様に，現在の解から次解候補を生成する．そして，温度と呼ばれる制御パラメータによって次解候補へ確率的に遷移する．これにより，局所解を持つ問題に対しても最適解を得ることが期待できる．

以下に詳細を示す．

STEP 1　初期設定

初期解をランダムに生成し，その評価を行う．ただし，初期解の解の深さは 2 以上とする[⋆1]．

STEP 2　生成処理

現在の解に対して，GP の突然変異と同様の操作を行うことで次解候補を生成し，その解候補を評価する．具体的な生成方法は，現在の解に対してランダムに突然変異点を選択し，その点を根とする部分木を削除する．次に削除した部分に，ランダムに生成した部分木を挿入し，次解候補を生成する (**図 1**)．

STEP 3　受理判定，状態遷移

現在の解の評価値 E と次解候補の評価値 E' との差分 $\Delta E(= E' - E)$ および温度パラメータ T を基に，次解候補に遷移するか否かの判定（受理判定）を行う．受理判定

⋆1 ルートノードの深さを 1 とする

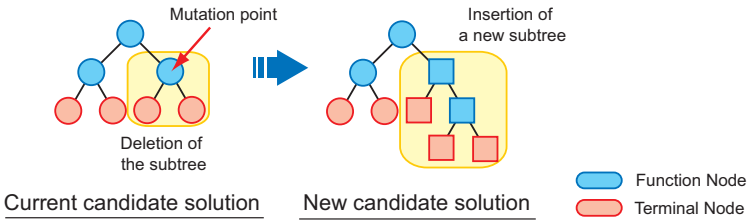


図 1　SAP における次解候補の生成方法
Fig. 1　A generation method of a candidate solution in SAP

には式 (1) に示す Metropolis 基準[7)] を用いる．

$$P_{AC} = \begin{cases} 1 & if\ \Delta E < 0 \\ exp(-\frac{\Delta E}{T}) & otherwise \end{cases} \quad (1)$$

ここで，P_{AC} は受理確率であり，次解候補が改良方向へ生成された場合は無条件で受理し，改悪方向へ生成された場合でも確率的に受理する．改悪方向への受理確率 P_{AC} は，改悪幅 ΔE が小さいほど高く，また，温度パラメータ T が大きいほど高いという特徴を持つ．

なお，STEP 2 および 3 の一連の処理をアニーリングと呼ぶ．

STEP 4　クーリング (冷却)

一定期間アニーリングを行った後，温度パラメータ T_k を小さくする．冷却後の温度 T_{k+1} は式 (2) に示す指数型アニーリングを用いて決定する．なお，γ はクーリング率である．

$$T_{k+1} = \gamma T_k \quad (0.8 \leq \gamma < 1) \quad (2)$$

STEP 5　終了判定

アニーリングを定められた回数行えば，探索を終了する．

3. SAP の温度並列化

3.1 温度パラメータの役割

SAP（SA）の大きな特徴は，次解候補が改悪方向へ生成された場合でも，その解候補への遷移を確率的に認めることである．その確率は式 (1) の Metropolis 基準により決定され，

改悪方向への遷移は温度パラメータ T に依存する．そのため，温度パラメータのスケジューリングは，解探索に大きな影響を与える．この温度スケジュールは一般的に，温度を高温から低温にするクーリングと温度を固定した一定温度の 2 通りに分けられる．以下に特徴を示す．

- クーリングを用いた温度スケジュール
  高温である探索序盤では，改悪方向への遷移を認めやすくなるため大域的な探索が行える．また，低温である探索終盤では，改悪方向への遷移が認めにくくなるため局所的な探索が行える．そのため，局所解に陥りにくい探索を行うことが出来る．しかし，低温時に局所解に陥ってしまうと，局所解から抜け出せないことや，低温時での探索ではプログラムサイズの増大[8] といった問題を引き起こす．
- 温度を固定した温度スケジュール
  探索に有効な温度領域（以下，重要温度領域）を用いた一定温度での探索は，クーリングを用いた場合と比べ，良好な性能が得られる[8]．しかし，重要温度領域を発見するためには，膨大な予備実験や人間の経験的な判断が要求される．

3.2　温度並列化の概要

SAP の基となる SA の既存研究において，適切な温度スケジュールの決定に有効な手法として，温度を並列化して計算を行う温度並列 SA が提案されている．温度並列化は，複数のプロセスに異なる温度を与え，各プロセスは一定温度で並列に探索を行い，一定の間隔で隣接するプロセス間の解の交換を確率的に行う手法である．温度並列化の特徴を以下に示す．

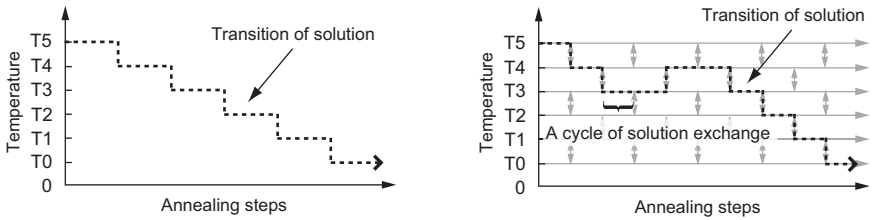
- 温度スケジュールの自動化
  現在の解が，一定温度に設定された各プロセス間を自律的に移動することから，温度スケジュールの自動化が図れる．
- 時間一様性
  探索の終了を任意の時点で行うことができ，またその時点からの探索の継続を行い解の改善を続けることができる．
- 並列処理との高い親和性
  解の品質を劣化することなしに，温度数まで並列実行が可能．

3.3　温度並列 SAP

温度並列 SA を SAP に適用した温度並列 SAP について以下に詳細を示す．

STEP 1　初期設定

各プロセスに異なる温度および異なる初期解を与える．各温度は，最高温度と最低温度および最高温度と最低温度の間を使用するプロセス数で等比的に分割した温度とする．



(a) temperature schedule with a cooling　(b) temperature schedule with a temperature-parallel

図 2　クーリング及び温度並列化を用いた温度スケジュール
Fig. 2　temperature schedule with a cooling and a temperature-parallel

STEP 2　温度を固定した SAP（生成処理，受理判定，状態遷移）

各プロセスが，与えられた温度を基に一定温度を温度スケジュールとした SAP で解の探索を行う．具体的には，2 章で示した STEP 2 および 3 のアニーリング（生成処理，受理判定，状態遷移の一連の処理）を一定周期まで繰り返し行う．

STEP 3　交換判定，解交換

アニーリングを一定周期 (解交換周期 k) まで繰り返した後，解の交換を行う．隣接するプロセスが持つ解の評価値 E と E' との差分 ΔE，および隣接するプロセスが持つ温度 T と T' との差分 ΔT により，プロセス間で解の交換を行うかどうかの判定を行う．交換判定には，式 (3)[9] を用いる．

$$P(\Delta T, \Delta E) = \begin{cases} 1 & if\ \Delta T \cdot \Delta E \leq 0 \\ exp(-\frac{\Delta T \cdot \Delta E}{T \cdot T'}) & otherwise \end{cases} \quad (3)$$

温度並列化では，通常のクーリング処理で温度 T から T' に冷却すること (**図 2**-a) が，温度 T のプロセスと温度 T' のプロセスの間で解を交換すること (図 2-b) に相当する．また，温度スケジュールを設定することは，温度並列化ではプロセス間で解の交換をいつ行うかを指定すること (図 2-b) に相当する．なお，図 2 は，横軸に探索回数，縦軸（対数軸）に温度を示す．

4. 対象問題

テスト問題は，自動プログラミングの代表的なベンチマーク問題である Santa Fe trail 問題[1)]，Wall-following 問題[1)]，および複雑さが異なる 2 つの Symbolic Regression 問題（Simple Symbolic Regression 問題[1)]，Complex Symbolic Regression 問題[11)]）とする．Santa Fe trail 問題，及び Wall-following 問題は解の評価に影響を及ぼさないノードが発生する（構文的イントロンが発生する）問題である．一方，Symbolic Regression 問題は全てのノードが解の評価に影響を及ぼす（構文的イントロンが発生しない）問題である．

4.1 Santa Fe trail[1)]

Santa Fe trail 問題とは，1 匹の人工蟻が**図 3** に示す 32 × 32 のマス目上に配置された餌を，限られたエネルギー内でできるだけ多く獲得するプログラムを生成する問題である．人工蟻は餌上を通ることにより餌を獲得することができ，終端記号が 1 つ実行されるごとにエネルギーは 1 消費する．人工蟻の初期エネルギーは 400 である．

この問題に用いる非終端記号は {IF_FOOD_AHEAD, PROGN2, PROGN3}，終端記号は {MOVE, RIGHT, LEFT} とした．if_food_ahead は引数を 2 個持ち，人工蟻の 1 マス前方に餌があれば第 1 引数，無ければ第 2 引数を実行する．progn$N$ は引数を N 個持ち，第 1 引数, 第 2 引数，…，第 N 引数の順に実行する．

評価関数 E_{val} は式 (4) より，餌の総数である F_{max}(= 89) から人工蟻が獲得した餌の数 F を引いたものであり，E_{val}=0 を最適解とする最小化問題である．

$$E_{val} = F_{max} - F \tag{4}$$

4.2 Wall-following[1)]

Wall-following 問題[1)] とは，ロボットが**図 4** に示す不規則な壁に囲まれた部屋で，壁に沿って移動することを目的とするプログラムを生成する問題である．部屋の環境は壁の横幅，および縦幅は 27.6[feet]，タイルは一辺 2.3[feet] の正方形であり，Koza の文献[1)] と同等の環境である．ロボットは 12 個の距離センサ（壁までの距離を測るセンサ）と 2 個の安全距離センサを持ち，前進，後退，左旋回および右旋回ができる．

この問題に用いる非終端記号は｛IFLTE，PROGN2｝，終端記号は｛S00，S01，S02，S03，…，S11，MSD，EDG，SS，MF，MB，TR，TL｝である．IFLTE は引数を 4 つ持ち，第 1 引数と第 2 引数の値を比較し，第 1 引数の方が小さい，もしくは同値の場合は第 3 引数を実行し，そうでなければ第 4 引数を実行する．ここで，第 1 引数，第 2 引数が行動を意味する終端記号（MF，MB，TR，TL）の場合は，S02 と S03 を比較し，その小さい方の値を返す[1)]．PROGN2 は引数を 2 つ持ち，第 1 引数，第 2 引数の順に実行する．S00~ S11 は距離センサの出力値，SS は 12 個の距離センサの中の最小値である．MSD（2.0）および EDG（2.3）は安全距離を表している．MF は前進（1.0[feet]），MB は後退（1.3[feet]），TR は右旋回（30[deg]），TL は左旋回（30[deg]）である．この問題では IFLTE の連鎖により，構文的イントロンが発生する．

評価関数 E_{val} 式 (5) は，壁際に設置されたタイルの枚数である N_{max}(= 56) からロボットが通過した壁際のタイルの枚数 N を引いたものであり，E_{val}=0 を最適解とする最小化問題である．

$$E_{val} = N_{max} - N \tag{5}$$

4.3 Simple Symbolic Regression[1)]

Simple Symbolic Regression とは，未知の関数 $y = f(x)$ に対して n 個の入出力データを用いて関数 f を同定する問題である．同定する目的関数は式 (6) に示す f_{obj} である．その概形を**図 5** に示す．

$$f_{obj} = x^4 + x^3 + x^2 + x \tag{6}$$

この問題に用いる非終端記号は｛+，-，*，%，sin，cos，exp，rlog｝，終端記号は｛x｝とした．なお，%は剰余，rlog は自然対数である．

評価関数 E_{val} は式 (7) より，-1 から 1 の間を 0.1 刻みにした 21 個の入力に対する出力誤差の和とし，$E \leq 0.01$ を最適解とする最小化問題である．ここで，$prog$ は生成されたプログラムを示す．

$$\begin{cases} E_{val} = \sum_{i=0}^{20} \left| prog(x_i) - f_{obj}(x_i) \right| \\ x_i = 0.1i - 1 \end{cases} \tag{7}$$

4.4 Complex Symbolic Regression[11)]

Complex Symbolic Regression 問題[11)] は，Symbolic Regression 問題の中でも複雑な関数を同定する問題であり，本実験では式 (8) に示す関数を目的関数 f_{obj} とする．目的関数 f_{obj} の概形を**図 6** に示す．

$$f_{obj}(x) = x^3 cos(x) sin(x) e^{-x} (sin^2(x) cos(x) - 1) \tag{8}$$

この問題に用いる非終端記号は｛+，×，−，%，sin，cos，exp，$rlog$｝，終端記号は｛x，0.1，0.2，0.3，0.4，0.5，0.6，0.7，0.8，0.9，1.0｝である．

評価関数 E_{val} は式 (9) より，0 から 10 の間を 0.1 刻みにした 101 個の入力に対する出力誤差の絶対値の総和であり，$E_{val} \leq 2$ を最適解とする最小化問題である．ここで，$prog$ は

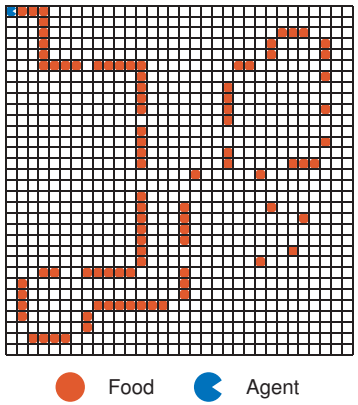


図 3　Santa Fe trail 問題
Fig. 3　Santa Fe trail problem

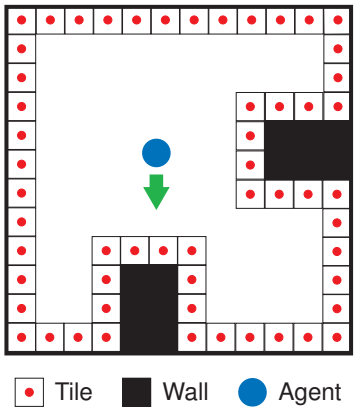


図 4　Wall-following 問題
Fig. 4　Wall-following problem

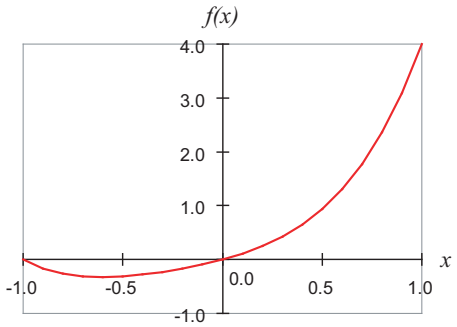


図 5　Simple Symbolic Regression 問題の目的関数の概形
Fig. 5　The target function for the simple symbolic regression problem

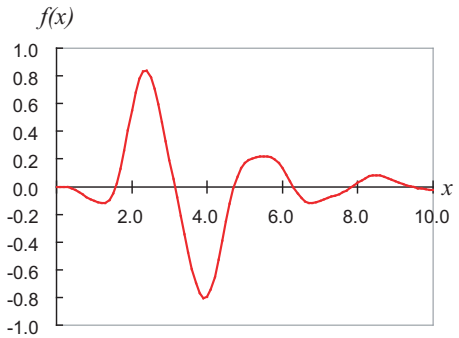


図 6　Complex Symbolic Regression 問題の目的関数の概形
Fig. 6　The target function for the complex symbolic regression problem

生成されたプログラムを示す．

$$\begin{cases} E_{val} = \sum_{i=0}^{100} |prog(x_i) - f_{obj}(x_i)| \\ x_i = 0.1i \end{cases} \tag{9}$$

表 1　各並列手法に用いるパラメータ
Table 1　Parameters for Santa Fe trail

Problem	Total steps	T_{max}	T_{min}
Santa Fe trail	200,000	128.4	0.271
Wall-following	400,000	80.8	0.271
Simple Symbolic Regression	100,000	888.2	0.008
Complex Symbolic Regression	400,000	1684.6	0.021

5. 温度並列化による有効性の検証

5.1 実験概要

SAP における温度並列化の有効性を検討するために，以下に示す並列手法に対して，4 章に示した各対象問題を適用し，比較実験を行う．

- 温度並列 SAP（以下，TPSAP）
  n 温度の温度並列 SAP を，プロセス数 n で実行する．解交換は解交換周期 k 毎に行う．なお，各プロセスに用いた温度は，最高温度，最低温度および，その間を等比的に分配した温度である．
- 解交換を行わない温度並列 SAP（以下，TPSAP-NoExchange）
  n 温度の温度並列 SAP を，プロセス数 n で実行する．解交換は行わない．なお，各プロセスに用いた温度は，最高温度，最低温度および，その間を等比的に分配した温度である．
- 並列 SAP（以下，Parallel SAP : PSAP）
  クーリングの温度スケジュールを用いた逐次 SAP を，プロセス数 n で同時に実行する．解交換は行わない．なお，各プロセスにおける初期温度は最高温度である．

各対象問題に用いたパラメータを**表 1** に示す．ここでの，Total steps は総評価計算回数，T_{max} および T_{min} は最高・最低温度を表す．また，各手法において用いたプロセス数は 16，解交換周期は一般的に用いられる値である $40^{10)}$ とし，PSAP に用いられるクーリング数は 32 とした．なお，最高温度は解交換周期内に最大改悪を 50%認める温度，最小温度は解交換周期内に最小改悪を 1 回認める温度とした．

5.2 並列手法における性能比較

各対象問題において，各手法を 50 試行した時の成功率の履歴を**図 7** に示す．なお，図 7 は横軸に探索回数 (=総評価計算回数 / プロセス数)，縦軸に成功率 (= 最適解を得た試行数

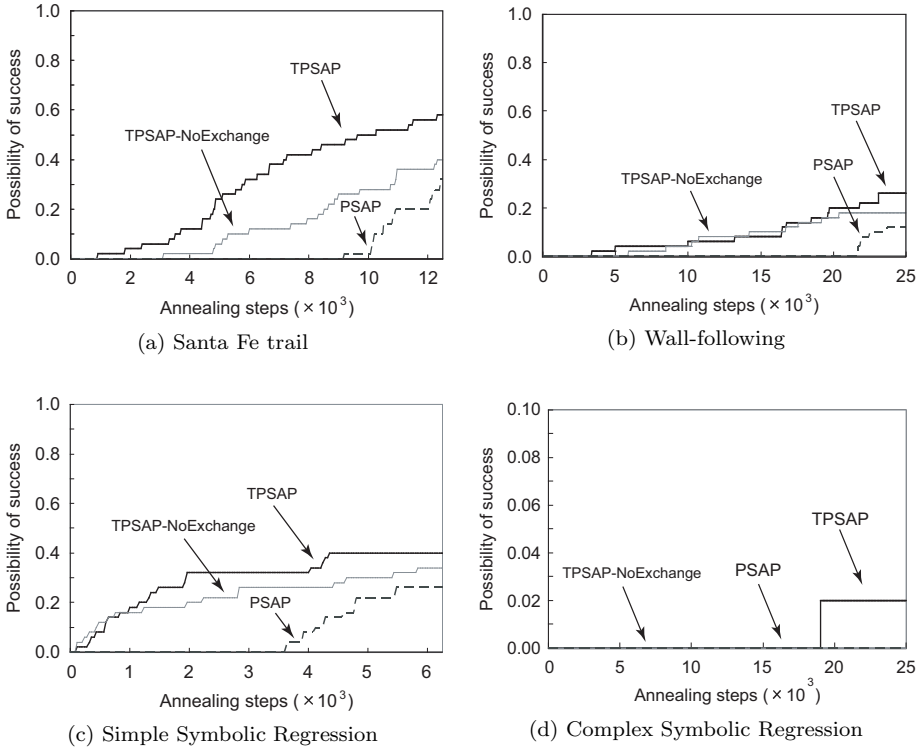


図 7　並列手法における TPSAP の有効性
Fig. 7　Effectiveness of TPSAP in Parallel-Methods

/ 試行数) を示し，成功率が高いほど解探索性能が高いことを表す．ここでの各手法における最良解は，全プロセス中の探索において最も良好な評価値（最良値）を得たの解とする．

図 7 より，(a)~ (d) の各対象問題において，TPSAP は他の並列手法である TPSAP-NoExchange および PSAP より高い解探索性能が得られることがわかった．また，Santa Fe trail 問題，Wall-following 問題および Simple Symbolic Regression 問題において，最適解が得ることができた探索回数は TPSAP および TPSAP-NoExchange は探索の序盤に対し，PSAP では探索の終盤であることがわかる．

図 7-(d) に示した Complex Symbolic Regression 問題において，SAP を拡張した全ての並列手法において探索性能が非常に低い結果となった．この Complex Symbolic Regression 問題は，全てのノードが解の評価に影響を及ぼす（構文的イントロンが発生しない）問題の中でも特に複雑な問題であり，SAP の解探索では，良好な探索が行えないと藤田らにより報告[8] されている．このことから，SAP を改良した手法である TPSAP も SAP と同様に，構文的イントロンが発生しない問題の中でも特に複雑な問題において，良好な解探索ができないことがわかった．

5.3　最適解における温度遷移に関する考察

図 7 に示したように，TPSAP は TPSAP-NoExchange および PSAP と比べ，解探索の性能が高いことがわかった．これは TPSAP が解の遷移を制御する温度スケジュールにおいて，探索に適切な温度スケジュールが実現されいると考えられる．そこで，以下に TPSAP で得た最適解の評価値および温度の遷移についての考察を行う．各対象問題において，ある試行における TPSAP で得た最適解の評価値および温度の遷移履歴を**図 8** に示す．図 8 は，横軸に探索回数，左縦軸に評価値，右縦軸に温度を示す．

図 8 より，各対象問題において，最適解は探索序盤では温度が比較的に高温の値を行き来することで，大域的に探索していることがわかる．具体的には，探索回数が Santa Fe trail 問題では $2 \sim 5 \times 10^3$[step]，Wall-following 問題では $0 \sim 9 \times 10^3$[step]，Simple Symbolic Regression 問題では $0 \sim 3 \times 10^3$[step]，Complex Symbolic Regression 問題では $0 \sim 2 \times 10^3$[step] の範囲となる．

探索中盤では，探索序盤の探索によって比較的に良好な解が発見されたことにより，温度が低温に遷移している．これは他のプロセスに比べ，良好な解が発見されたことを意味しており，探索回数が Santa Fe trail 問題では $5 \sim 6 \times 10^3$[step]，Wall-following 問題では $9 \sim 13 \times 10^3$[step]，Simple Symbolic Regression 問題では $3 \sim 5 \times 10^3$[step]，Complex Symbolic Regression 問題では $2 \sim 5 \times 10^3$[step] の範囲となる．

探索終盤では，温度が低温の領域付近に固定され，局所的な探索が行われていることがわかる．中盤までの探索で最適な解を発見されていない Wall-following 問題や Complex Symbolic Regression 問題では，低温での局所的探索によって，徐々に良好な解を発見し，最終的には最適解を得られたことがわかる．

このように，TPSAP では解が必要に応じてプロセス（温度）を移動することによって，解自身が自律的に温度を遷移し，解探索に適した温度スケジュールを実現している．以上のことから，TPSAP は TPSAP-NoExchange および PSAP と比べ，解探索に適した温度スケジュールを実現することにより，効率的な解探索が可能であると言える．

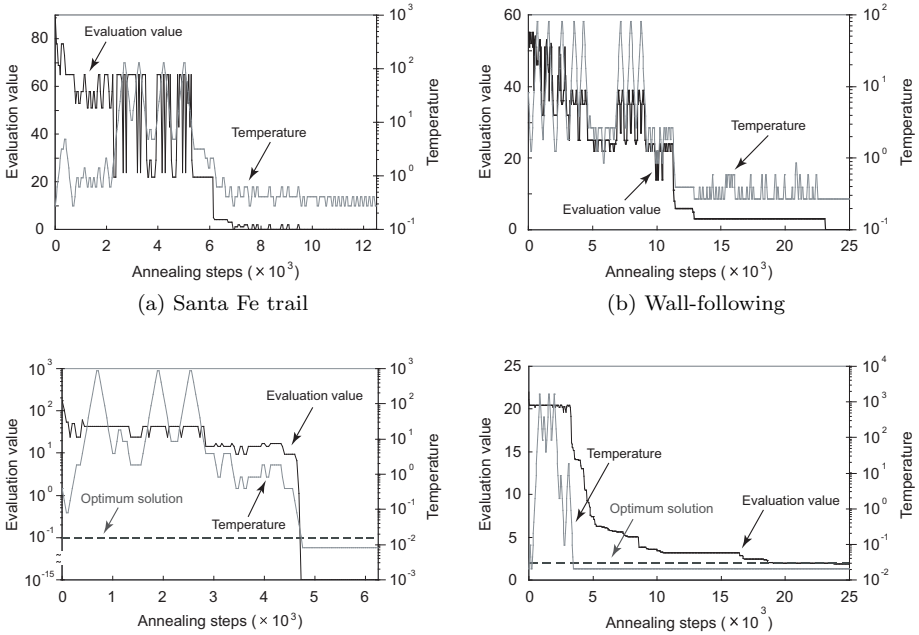


図 8　TPSAP および SAP における評価値および温度の遷移履歴
Fig. 8　History of evaluation value and temperature for the best solution in TPSAP

6. 逐次 SAP との比較

6.1 実験概要

先行研究において，SAP では解探索に有効な温度（重要温度領域）があると報告されている[8]．そこで，1 プロセスで解探索を行う重要温度領域を用いた一定温度での逐次 SAP と，複数プロセスで解探索を行う温度並列 SAP との比較を行う．

各対象問題に用いたパラメータを**表 2** に示す．ここでの，Total steps は総評価計算回数，P_{num} はプロセス数（温度数），T は温度を表す．なお，SAP に用いた重要温度領域は予備実験により求めた値，TPSAP に用いた最高・最低温度は表 1 と同様の値とした．

TPSAP に用いたプロセス数（温度数）は 8，16，32（以下，TPSAP-8，TPSAP-16，TPSAP-32）とし，解交換周期は 40[10] とした．また，実験に用いた計算機の環境は**表 3** に示す通りである．

表 2　TPSAP および SAP に用いるパラメータ
Table 2　Parameters for TPSAP and SAP

(a) Santa Fe trail

Parameter	SAP	TPSAP
Total steps		200,000
P_{num}	1	8，16，32
T	4.0	(128.4，0.271)

(b) Wall-following

Parameter	SAP	TPSAP
Total steps		400,000
P_{num}	1	8，16，32
T	2.0	(80.8，0.271)

(c) Simple Symbolic Regression

Parameter	SAP	TPSAP
Total steps		100,000
P_{num}	1	8，16，32
T	0.5	(888.2，0.008)

(d) Complex Symbolic Regression

Parameter	SAP	TPSAP
Total steps		400,000
P_{num}	1	8，16，32
T	0.25	(1684.6，0.021)

表 3　計算機環境
Table 3　Execution environment

CPU	AMD Opteron 1.8GHz × 2
Memory	PC2700 Registeres ECC 2GB
NIC	Gigabit Ethernet
NIC driver	Broadcom Tigon3
Switch	NETGEAR GS524T
Cable	Category 5e
OS	Debian GUN / Linux4.0 amd64
Kernel	Linux 2.6.16 + Xen patch
Xen	3.0.3
DCAST	3.0.5
HPL	1.0a
Compiler	gcc 4.1.2, gfortran 4.1.2
Communication library	mpich 1.2.7

6.2 逐次 SAP との性能比較

各対象問題において，各手法を 50 試行した時の成功率の履歴を**図 9** に示す．なお，図 9 は，横軸に評価計算回数，縦軸に成功率を示し，成功率が高いほど解探索性能が高いことを表す．ここでの各手法における最良解は，全プロセス中の探索において最も良好な評価値

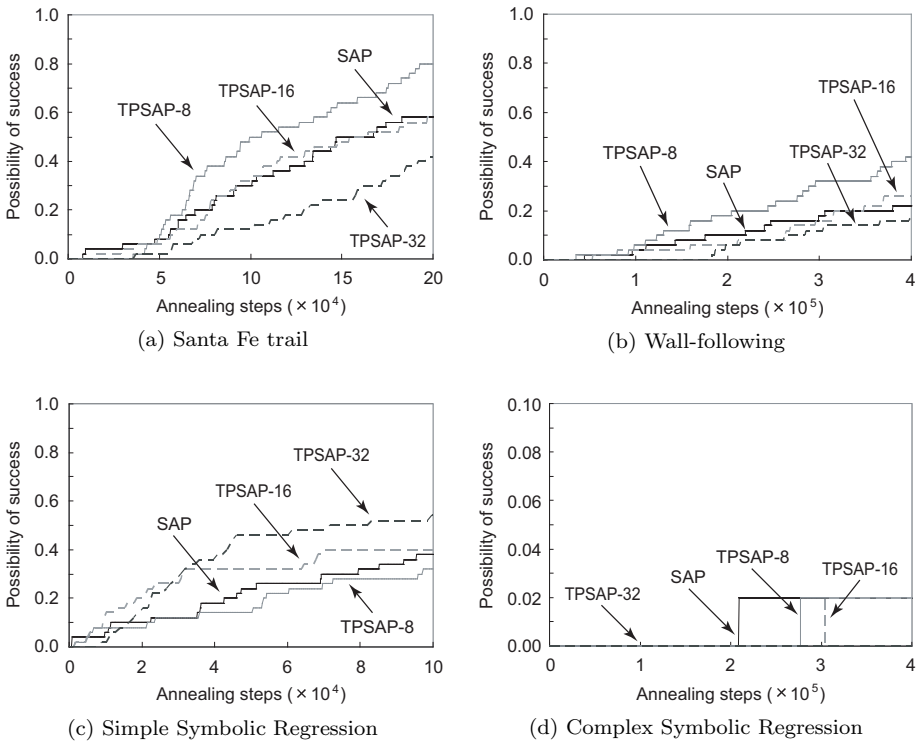


図 9　TPSAP と逐次 SAP との性能比較
Fig. 9　Effectiveness of TPSAP in Parallel-Methods

（最良値）を得たの解とする．

図 9-(a) および (b) より，Santa Fe trail 問題および Wall-following 問題では，TPSAP-8 が最も良好な性能を示しており，また，TPSAP-8 および TPSAP-16 は重要温度領域を用いた一定温度での逐次 SAP より，良好な性能を得られたことがわかる．一方，図 9-(c) より，Simple Symbolic Regression 問題では，TPSAP-32 が最も良好な性能を示しており，また，TPSAP-16 および TPSAP-32 は SAP より，良好な性能を得られたことがわかる．図 9-(d) より，Complex Symbolic Regression 問題では，SAP，TPSAP-8，TPSAP-16 の各手法は同等の性能を示している．

以上のことから，TPSAP は重要温度領域を用いた一定温度での逐次 SAP と比べ，同等

表 4　1 試行における平均解探索時間
Table 4　Average evaluation time

Problem	SAP[sec]	TPSAP-8[sec]	TPSAP-16[sec]	TPSAP-32[sec]
Santa Fe trail	37.950	4.020	2.034	1.235
Wall-following	1029.726	150.644	72.265	37.190
Simple Symbolic Regression	6.965	2.075	0.951	0.455
Complex Symbolic Regression	318.074	69.943	33.764	12.210

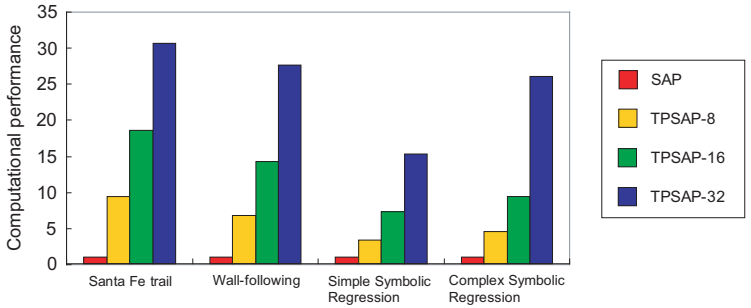


図 10　TPSAP における解探索時間の性能向上率
Fig. 10　Computational performance

以上の性能を得られることがわかった．また，対象問題により，TPSAP は用いるプロセス数（温度数）が探索の性能に影響を与えることがわかった．

6.3　解探索時間

各手法において，解の探索に必要とした 1 試行の平均探索時間を**表 4** に，探索時間の性能向上率を**図 10** に示す．なお，図 10 は横軸に対象問題の種類，縦軸に TPSAP における探索時間の性能向上率（=SAP の探索時間/TPSAP の探索時間）を示し，性能向上率が高いほど探索時間が短縮されたことを表す．

表 4 より，SAP における 1 試行の探索時間は Santa Fe trail 問題では約 40[sec]，Wall-following 問題では約 1030[sec](=17[min])，Simple Symbolic Regression 問題では約 7[sec]，Complex Symbolic Regression 問題では約 320[sec](=5[min]) 必要とした．一方，複数のプロセス用いた TPSAP では，1 試行における探索時間は SAP と比べ，大幅に短縮できることがわかった．

図 10 より，TPSAP は用いるプロセス数（温度数）に応じた探索時間の短縮が行われてい

表 5　最良解のノード数
Table 5　Parameters for Santa Fe trail

(a) Santa Fe trail

Method	N_{min}	N_{mid}	N_{max}
SAP	15	20	35
TPSAP-8	14	21	37
TPSAP-16	14	21	39
TPSAP-32	11	21	42

(b) Wall-following

Method	N_{min}	N_{mid}	N_{max}
SAP	13	25	53
TPSAP-8	11	25	47
TPSAP-16	11	23	47
TPSAP-32	11	23	47

(c) Simple Symbolic Regression

Method	N_{min}	N_{mid}	N_{max}
SAP	13	17	26
TPSAP-8	13	25	70
TPSAP-16	13	22	63
TPSAP-32	13	20	41

(d) Complex Symbolic Regression

Method	N_{min}	N_{mid}	N_{max}
SAP	20	39	66
TPSAP-8	7	56	115
TPSAP-16	11	55	106
TPSAP-32	9	40	81

ることがわかる．特に，Santa Fe trail 問題および Wall-following 問題では，逐次 SAP の探索時間に用いたプロセス数（温度数）倍の探索時間の短縮が実現されており，また，Simple Symbolic Regression 問題および Complex Symbolic Regression 問題においても，十分な探索時間の性能向上が得られた．

6.4　最良解におけるノード数に関する考察

図 9 および図 10 に示したように TPSAP は逐次 SAP と比べ，解の探索性能が高く非常に短い探索時間で解の探索が行えることがわかった．ここで，TPSAP が SAP と同様に少ないプログラムサイズ（ノード数）で解を生成できてきる特徴[8] を持つかの確認を行う．各対象問題において，各手法を 50 試行した時の最良解のノード数を**表 5** に示す．ここでの，N_{min} はノード数の最小値，N_{mid} はノード数の中央値，N_{max} はノード数の最大値を表す．

表 5-(a) および (b) より，Santa Fe trail 問題および Wall-following 問題において，最良解におけるノード数の最小値，中央値，最大値が同等であることから，TPSAP は逐次 SAP と同様に少ないノード数で解が生成されていることがわかる．しかし，表 5-(c) より，Simple Symbolic Regression 問題において，最良解におけるノード数の最小値は同等であるのに対し，中央値および最大値が逐次 SAP より TPSAP の方が多い結果となった．また，表 5-(d) より，Complex Symbolic Regression 問題おいて，最良解のノード数の最小値は TPSAP が SAP より少ないのに対し，中央値および最大値は TPSAP の方が大幅に多いことがわかる．

表 6　最適解のノード数
Table 6　Parameters for Santa Fe trail

(c) Simple Symbolic Regression

Method	N_{min}	N_{mid}	N_{max}
SAP	13	17	25
TPSAP-8	13	17	27
TPSAP-16	13	14	26
TPSAP-32	13	16.5	29

(d) Complex Symbolic Regression

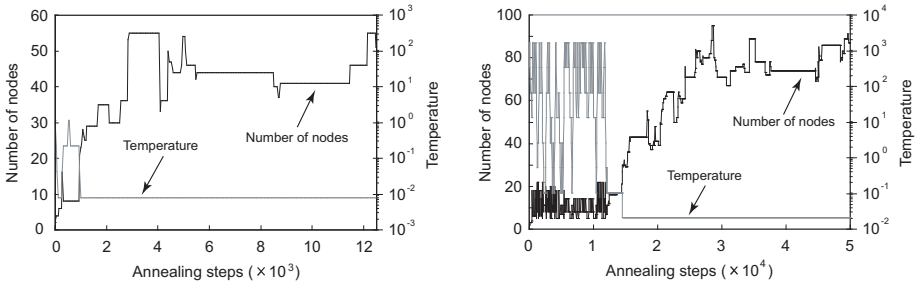
Method	N_{min}	N_{mid}	N_{max}
SAP			35
TPSAP-8			88
TPSAP-16			73
TPSAP-32			-

ここで，Simple Symbolic Regression 問題および Complex Symbolic Regression 問題における成功した解（Simple Symbolic Regression 問題 : 評価値 ≤ 0.01，Complex Symbolic Regression 問題 : 評価値 ≤ 2.0）である最適解のノード数の最小値，中央値，最大値を**表 6** に示す．なお，Complex Symbolic Regression 問題において，50 試行した中で最適解を得た試行は SAP および TPSAP ともに 1 試行のため，ノード数の最小値，中央値，最大値の値は同一の値となる．

表 6-(c) より，Simple Symbolic Regression 問題において，SAP および TPSAP で得られた最適解のノード数の最小値，中央値，最大値が同等であることがわかる．このことから，表 5-(c) において，TPSAP は逐次 SAP よりノード数の多い解を生成する要因として，最適解を得られていない（成功していない）試行で得られた解のノード数が多いため，最良解のノード数の中央値および最大値が増加したと考えられる．また，表 6-(d) より，Complex Symbolic Regression 問題において，最適解のノード数が TPSAP が SAP より多いことがわかる．このことから，表 5-(d) および表 6-(d) から，Complex Symbolic Regression 問題において，TPSAP は SAP よりノード数の多い解を生成する傾向があると言える．

Simple Symbolic Regression 問題および Complex Symbolic Regression 問題において，TPSAP により得た解のノードが SAP より多い要因について，考察を行う．SAP では，解のノード数と温度が密接に関係していると報告されている[8]．具体的には，温度が高温の場合，解のノード数が少なくなり，また，温度が低温の場合，解のノード数が多くなる傾向がある．このことから，TPSAP における解の温度遷移に着目をする．Simple Symbolic Regression 問題および Complex Symbolic Regression 問題において，ある試行における TPSAP で得た最良解のノード数および温度の遷移履歴を**図 11** に示す．なお，横軸に探索回数，左縦軸にノード数，右縦軸に温度を示す．

図 11 より，Simple Symbolic Regression 問題および Complex Symbolic Regression 問



(c) Simple Symbolic Regression　(d) Complex Symbolic Regression

図 11　TPSAP におけるノード数および温度の遷移履歴
Fig. 11　History of evaluation value and temperature for the best solution in TPSAP

題の両問題において，探索序盤（Simple Symbolic Regression 問題：$0 \sim 1 \times 10^3$[step], Complex Symbolic Regression 問題：$0 \sim 1.2 \times 10^4$[step]）である温度が比較的高温の場合は，ノード数が少ないことがわかる．しかし，探索中盤から終盤にかけて，解が設定した最低温度から遷移せず，ノード数が増加していることがわかる．これは，他のプロセスで良好な解が生成されないため，解が低温プロセスから遷移できないためと考えられる．また，図 10 より，Santa Fe trail 問題および Wall-following 問題に比べ，SAP に対する探索時間の短縮が実現されていない要因として，図 11 に示したように探索におけるノード数の増加が考えられる．

以上のことから，TPSAP は短時間で良好な解を探索することが可能であり，尚且つ，SAP と同様に少ないプログラムサイズ（ノード数）で最適解を生成できてきる特徴を持つと言える．

7. おわりに

本研究では，適切な温度スケジュールを自動的に決定する有効な手法である温度並列 SA のアルゴリズムを SAP に適用した温度並列 SAP を提案し，その有効性について検証を行った．数値実験の結果より，ベンチマーク問題において，提案手法である TPSAP における温度並列化の有効性を示した．これは TPSAP において，解がプロセス間を移動することによって解自身が適切な温度スケジューリングを自動的に決定することにより，効率的に解探索が行われたと考えれらる．また，重要温度領域を用いた一定温度での逐次 SAP と比較した結果，適切なプロセス数（温度数）を用いた TPSAP が良好な解探索性能を示した．TPSAP は SAP と比べ，解の探索時間を大幅に短縮が可能であり，また，SAP と同様にノード数の少ない最適解を生成できることがわかった．

最後に本研究を通して，SAP および SAP の改良手法である TPSAP は，Complex Symbolic Regression 問題に対して，良好な解探索性能が得られないことがわかった．そのため，構文的イントロンが発生しない問題の中でも特に複雑な問題において，SAP および SAP の改良手法は不向きであると考えられる．

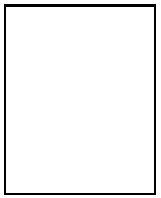
なお，TPSAP は SAP の性能向上を目的とした手法であるため，SAP に適した対象問題に用いることができる．GP や SAP などの自動プログラミング手法には，それぞれ異なる特徴があり，問題に応じて探索手法を使い分けることが望ましい．探索手法の使い分けについては，前報[8] を参照して頂きたい．

参　考　文　献

1) Koza, J.R.: *Genetic Programming: On the Programming of Computers by Means of Natural Selection*, MIT Press (1992).
2) Goldberg, D.E.: *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley Professional (1989).
3) Nordin, P. and Banzhaf, W.: Genetic Programming Controlling a Miniature Robot, *Working Notes for the AAAI Symposium on Genetic Programming* (Siegel, E.V. and Koza, J.R., eds.), MIT, Cambridge, MA, USA, AAAI, pp.61–67 (1995).
4) 片上大輔，山田誠二：ノード使用頻度に依存した交叉による進化ロボティクスの高速化，人工知能学会論文誌， Vol.16, No.4, pp.392–399 (2001).
5) Iba, H. and Nikolaev, N.: Genetic Programming Polynomial Models of Financial Data Series, *Proceedings of the 2000 Congress on Evolutionary Computation CEC00*, La Jolla Marriott Hotel La Jolla, California, USA, IEEE Press, pp.1459–1466 (2000).
6) Daida, J.M., Bersano-Begey, T.F., Ross, S.J. and Vesecky, J.F.: Computer-Assisted Design of Image Classification Algorithms: Dynamic and Static Fitness Evaluations in a Scaffolded Genetic Programming Environment, *Genetic Programming 1996: Proceedings of the First Annual Conference* (Koza, J.R., Goldberg, D.E., Fogel, D.B. and Riolo, R.L., eds.), Stanford University, CA, USA, MIT Press, pp.279–284 (1996).
7) Metropolis, N., Rosenbluth, A., Rosenbluth, M., Teller, A., Teller, E.: Equation of State Calculation by Fast Computing Machines, *Journ. of Chemical Physics*, Vol.21, pp.1087–1092 (1953).
8) 藤田佳久，三木光範，橋本雅文，廣安知之：シミュレーテッドアニーリングを用いた

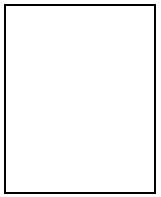
自動プログラミング，情報処理学会論文誌， Vol.48, No.SIG15, pp.88–102 (2007).
9) 小西健三，瀧　和男，木村宏一：温度並列シミュレーテッドアニーリング法とその評価，情報処理学会論文誌， Vol.36, No.4, pp.797–807 (1995).
10) 三木光範，廣安知之，笠井誠之：連続最適化問題への温度並列シミュレーテッドアニーリングの応用，情報処理学会論文誌， Vol.41, No.5, pp.1607–1616 (2000).
11) Salustowicz, R.P. and Schmidhuber, J.: Probabilistic Incremental Program Evolution: Stochastic Search Through Program Space, *Machine Learning: ECML-97* (van Someren, M. and Widmer, G., eds.), Vol.1224, Springer-Verlag, pp.213–220 (1997).

(平成 21 年 12 月 16 日受付)
(平成 22 年 4 月 5 日採録)



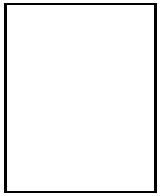
三木　光範（正会員）

1950 年生．1978 年大阪市立大学大学院工学研究科博士課程修了，工学博士．大阪府立大学工学部航空宇宙工学科助教授などを経て，1994 年同志社大学工学部教授．進化的計算手法とその並列化，および知的なシステムの設計に関する研究に従事．著書は「工学問題を解決する適応化・知能化・最適化法」（技法堂出版）等多数．IEEE，情報処理学会，人工知能学会等会員．知的オフィス環境コンソーシアム会長．



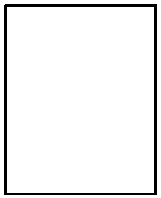
松井　勇樹

1985 年生．2008 年同志社大学工学部知識工学科卒業．同年，同志社大学大学院工学研究科修士課程入学．シミュレーテッドアニーリングプログラミングの研究に従事．2010 年に NEC システムテクノロジー株式会社に入社．



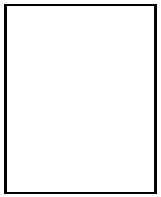
小畑　拓也

1985 年生．2008 年同志社大学工学部知識工学科卒業．シミュレーテッドアニーリングプログラミングの研究に従事．現在は TIS 株式会社に勤務．



廣安　知之（正会員）

1997 年早稲田大学理工学研究科後期博士課程修了．博士工学．同志社大学工学部准教授を経て 2008 年同志社大学生命医科学部教授．創発的計算，最適設計，並列処理などの研究に従事．IEEE，情報処理学会，電気情報通信学会，計測自動制御学会，日本機械学会，超並列計算研究会，日本計算工学会各会員．



吉見　真聡（正会員）

2009 年慶大大学院理工学研究科後期博士課程了．2006 年より，日本学術振興会特別研究員 (DC1)．2009 年より，同志社大学理工学部助教．リコンフィギャラブルシステム，並列処理，知的システムの研究に従事．電子情報通信学会，情報処理学会，人工知能学会各会員．