数理モデル化と問題解決 33－16
（2001．3．16）

# 巡回セールスマン問題に対する確率・遺伝的ハイブリッドアルゴリズム

軍司栄一, 富田悦次, 若月光夫
電気通信大学大学院 電気通信学研究科 電子情報学専攻

**あらまし** 組み合わせ最適化問題の近似解法として, ボルツマンマシンの概念を基礎として考案された確率アルゴリズムに対し, さらに大域的な解の更新を可能とする遺伝的アルゴリズムを組み合わせることにより, 近似精度を向上させることが期待できる. 本稿では, 巡回セールスマン問題を対象として, そのようなハイブリッドアルゴリズムを提唱し, その有効性を実験的に示す.

# A Randomized and Genetic Hybrid Algorithm for the Traveling Salesman Problem

Eiichi Gunji, Etsuji Tomita, and Mitsuo Wakatsuki
Course in Communications and Systems Engineering,
Graduate School of Electro-Communications, UEC

**Abstract** A randomized algorithm which is based upon Boltzmann machines is expected to be improved by combining a genetic algorithm which enables global changes of solutions. We present such a hybrid algorithm for the traveling salesman problem and show the effectiveness of the hybridization by computer experiments.

## 1 まえがき

NP 困難問題では, 多項式時間内に厳密解が得られるようなアルゴリズムは期待しがたい. そこで, より短い時間内に実用上十分厳密解に近いとみなせるような近似解を得ようとする近似解法が重要となってくる. この内, ボルツマンマシンを利用したアニーリング法は有力なアルゴリズムであるが, 高精度の解を得るためには一般に長時間が必要とされる. そのための対策として, 例えば文献 [1] では, 近似最大クリークを抽出するために'極の緩和'を新たに導入したアルゴリズム RaCLIQUE が提案されている.

この他の NP 困難問題として, 文献 [2] では, 巡回セールスマン問題 (TSP) に対し同様の概念を導入したアルゴリズム RaTSP を提唱し, 他の手法と比較してかなり良好な結果が得られたと報告されている.

本稿では, このような局所探索を基にした確率アルゴリズムに対し, 大域的な解の更新を可能とする遺伝的アルゴリズム (GA) を組み合せたハイブリッドアルゴリズムを提唱し [3], そのようなハイブリッド化の有効性を実験的に示す.

## 2 TSP とボルツマンマシン

都市の集合 $V = \{v_1, \cdots, v_N\}$ と要素 $d_{i,j}$ を都市 $v_i$ と $v_j$ のユークリッド距離とする $N$ 次の正方行列 $D = \{d_{i,j}\}$ が与えられているとする. $V$ の要素に対する順序付け $\langle v_{\pi(1)}, \cdots, v_{\pi(N)} \rangle$ を $V$ の巡回路 (tour) と呼び,

$$\sum_{i=1}^{N-1} d_{\pi(i),\pi(i+1)} + d_{\pi(1),\pi(N)}$$

をこの巡回路の長さと定義する. この時, 最短の $V$ の巡回路を探し出すというのが TSP である.

### 2.1 諸定義

TSP にボルツマンマシンを適用するために, 状態, エネルギー, および近傍を定義する.

**状態** $V$ の巡回路 $\langle v_{\pi(1)}, \ldots, v_{\pi(N)} \rangle$ を $N$ 次元のベクトル

$$x = (x_1, x_2, \ldots, x_N)$$
$$x_i = \pi(i) \quad 1 \leq i \leq N$$

で表したものを状態と呼ぶ.

**エネルギー** 状態 $x$ で表される巡回路の長さをその状態のエネルギー $E(x)$ と定義する.

**近傍** 状態 $x = (x_1, \ldots, x_{i-1}, x_i, \ldots, x_j, x_{j+1}, \ldots, x_N)$ から 2-opt 法により遷移した先, つまり

$$\mathcal{N}(x,i,j) = (x_1, \ldots, x_{i-1}, x_j, \ldots, x_i, x_{j+1}, \ldots, x_N)$$

で定義される状態 ($x_j, x_i$ 間は逆順) の和集合を $\mathcal{N}(x)$ とする. 但し, $i > j$ の場合には, $x_1$ を $x_N$

の次の要素であるとみなして $\mathcal{N}(x,i,j)$ を決定する. また, 状態 $x$ から状態 $\mathcal{N}(x,i,j)$ への状態遷移を行なうとき,

$$e_i = d_{x_i,x_{i-1}}, \quad e_j = d_{x_j,x_{j+1}}$$
$$e'_i = d_{x_i,x_{j+1}}, \quad e'_j = d_{x_j,x_{i-1}}$$

とおくと, エネルギーの増分は $\Delta E(x,i,j) = e'_i - e_i + e'_j - e_j$ で与えられる.

### 2.2 ボルツマンマシン

ボルツマンマシンでは, 対象とする問題の構造を, 状態 $x$ に対するエネルギー $E(x)$ の定義された系として表現する. 系の各状態 $x$ は, その 1 近傍 $\mathcal{N}(x,i,j)$ へ確率的な遷移を繰り返す. このとき, $E(x)$ を最小化することが問題を解くことに相当する. 状態遷移によるエネルギーの増分は

$$\Delta E(x,i,j) = E(\mathcal{N}(x,i,j)) - E(x)$$

で表される. このとき, 状態遷移を採用する確率は, ボルツマン分布を用いて

$$p(\Delta E) = \begin{cases} exp\left[\frac{-\Delta E}{T}\right] & \text{if } \Delta E \geq 0 \\ 1 & \text{otherwise} \end{cases}$$

とする. ここで, $T$ は温度と呼ばれる 0 以上の実数値をとるパラメータである.

### 2.3 アニーリング法

アニーリング法とは, ボルツマンマシンを用い, 系を高い温度からゆっくりと冷却していき, エネルギーを低い状態に収束させることで解を探索する手法である. 温度の下げ方に十分に遅い関数を用いた場合, 確率 1 で厳密解が求まることが証明されている.

## 3 RaTSP

### 3.1 極の緩和

RaCLIQUE はアニーリング法と同様ボルツマンマシンを基本とした確率的な近似アルゴリズムであるが, アニーリングは行わず, 温度を低く保ったまま探索を行う. この方法では, 系を特定の状態に収束させるのではなく, 次々と新たな局所探索を行っていくことを目的としている. しかし, 通常のボルツマンマシンでは低温下で極小値から抜け出すには膨大な試行回数を必要とする. そこで, 極小値から抜け出すのを容易にするため, そのアルゴリズムにおいて望ましいと考えられる遷移に対し採用確率を高くしている. これを極の緩和と呼ぶ [1].

この概念を TSP に応用したアルゴリズムが RaTSP[2] である.

### 3.2 状態遷移の評価関数

遷移確率をある状態を採用する確率 $p$, すなわち

$$p(x,i,j) = \begin{cases} exp\left[\frac{-F(x,i,j)}{T}\right] & \text{if } F(x,i,j) \geq 0 \\ 1 & otherwise \end{cases}$$

によって定義する. ここで,$F(x,i,j)$ は状態 $x$ からその近傍 $\mathcal{N}(x,i,j)$ への状態遷移に対する評価関数であり, その状態遷移が望ましいものであるほど小さい値をとる. 文献 [2] では極の緩和を実現する評価関数として, 次の式 $F = F_p$ が与えられている.

$$f_1 = \begin{cases} (R \cdot e_i/e'_i)(e'_i - e_i) & (e'_i < e_i) \\ (e'_i/e_i)(e'_i - e_i) & (e'_i \geq e_i) \end{cases}$$

$$f_2 = \begin{cases} (R \cdot e_j/e'_j)(e'_j - e_j) & (e'_j < e_j) \\ (e'_j/e_j)(e'_j - e_j) & (e'_j \geq e_j) \end{cases}$$

$$F_p(x,i,j,R) = f_1 + f_2$$

上式中の $R$ が緩和項と呼ばれる, 極の緩和を司るパラメータであり, 1 以上の実数値をとる. この値が大きいほど, 極小値から抜け出すのが容易となる.

この評価関数 $F_p(x,i,j,R)$ はそれぞれの経路長変化においてそれが短くなるようならその扱いを大きくするだけでなく, 経路長の比を評価に取り入れており, 経路長変化の割合が大きいものも大きく扱っている.

## 4 GA とのハイブリッド化

RaTSP では, 局所解が深い場合には, そこから抜け出すのに多大な試行回数が必要となる. そこで, RaTSP を複数並列に動作させ, ある回数以上解の更新が行なわれなければ GA で使用される操作の一つである交叉によって局所解から強制的に抜け出すように変更を加え, 解精度の向上を図る [3].

### 4.1 交叉における親の選択

交叉を管理するため, cross_wait を設定し, ある時点において解の更新されない回数がこれを超えている個体が 2 つ存在する時, 交叉を行う.

### 4.2 共有サブツアー

共有サブツアーとは2つの巡回路に共通して含まれる区間のうち極大のものを指す.

例) 2つの巡回路

$\langle v_0, v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8, v_9 \rangle$
$\langle v_1, v_5, v_2, v_9, v_8, v_7, v_0, v_3, v_4, v_6 \rangle$

があったとすると $\langle v_3, v_4 \rangle$ 及び $\langle v_7, v_8, v_9 \rangle$ が共有サブツアーとなる.

### 4.3 交叉

2つの個体を選び共有サブツアーを抽出したら, 共有サブツアーはそれ毎, 他は各都市毎ランダムに並び替え, 新たな巡回路を生成する.

### 4.4 局所探索の適用

RaTSP において, 確率アルゴリズムに対して緩和項を付加していることからの影響と推測されるが, 十分な局所的改善を行わないうちに探索が終了し, 明らかにローカルミニマムですらない解が出力されることがある.

そこで局所探索を加えることにより探索効率を向上させ, また, 上で挙げたような状態を解として出力する事の無いようにする.(以上を考慮したアルゴリズムが文献 [3] の RaTSP_GA であり, 本稿ではこれを改めて RaTSP_GA0 と呼ぶことにする.)

## 5 RaTSP_GA

RaTSP_GA0 において, 大規模問題や都市配列に偏りのある問題に対して解精度が向上しない場合がある. そのため, 更に以下の3つの変更を加えた. このアルゴリズムを RaTSP_GA とする.

**都市選択方法の変更** 探索する近傍を設定するための都市の選択方法について, ある程度探索を行った後は経路上で近い都市同士だけを選択するように変更した. これは都市配列に偏りのある問題において, それぞれの都市が偏って集まったところにおける最適化が不十分であったことから, よりそれらに集中して探索するのが目的である.

**GA の適用方法の変更** 現在探索中の個体同士を交叉させるのではなく, 現在までに見つかっている解候補のうちで最も良いものと交叉させるようにする. これにより, より良い子個体を生成できるようにするのが目的である.

**温度パラメータの変更** これまでは温度一定のまま緩和項の効果のみで極からの脱出を試みていた. しかし, それだけでなく, ある程度温度を操作することによりそれを補助できると考えられる.

具体的には, ある回数以上遷移が行われなくなったとき, その都度温度をある値 $\Delta T$ ずつ上げていき, 遷移が行われたらその時点で, 温度を1段階下げることにする.

### 5.1 アルゴリズム RaTSP_GA

以上ような変更を加えたアルゴリズム RaTSP_GA は, 次のとおりである. ここで, $switch\_1$ は局所探索から RaTSP への切り替え時期を表すパラメータ, $switch\_2$ は都市選択方法の切り替え時期を表すパラメータ, $switch\_3$ は温度を上昇させる時期を表すパラメータ, $n$ は並行して動作させる RaTSP の数, $rand1[i,j]$ は区間 $[i,j]$ の一様整数乱数,$rand2[i,j]$ は区間 $[i,j]$ の一様実数乱数である. なお, $n = 1$, $switch\_2 = switch\_3 = cross\_wait \geq max\_t$, $switch\_1 = 0$ とおくと RaTSP となる.

**procedure** RaTSP_GA($R, T_0, \Delta T, max\_t$, $cross\_wait, switch\_1, switch\_2, switch\_3, n$)
  **begin**
    **for** $s := 1$ **to** $n$ **do**
      $t2[s] := 0$; $t\_sw3[s] := 0$; $T[s] := T_0$;
      $x[s]$ の初期解をランダムに生成;
    **od**
    **for** $t := 1$ **to** $max\_t$ **do** (1)
      **for** $s := 1$ **to** $n$ **do** (21)
        $i := rand1[1, N]$;
        **if** $t2[s] < switch\_2$ **then**
          $j := i + rand1[1, N-1]$
        **else**
          $j := i + rand1[1, (N-1)^{0.5}]$
        **fi**
        **if** $j > N$ **then** $j := j - N$ **fi**
        **if** $t2[s] < switch\_1$ **then**
          **if** $\Delta E(x, i, j) < 0$ **then** $flag := 1$
          **else** $flag := 0$ **fi**
        **else**
          **if** $exp\left(\frac{-F_p(x,i,j,R)}{T[s]}\right) < rand2[0,1]$ **then**
            $flag := 0$;$t\_sw3[s] := t\_sw3[s] + 1$;

```
        if t_sw3[s] > switch_3 then
          T[s] := T[s] + ΔT; t_sw3[s] := 0 fi
      else
        flag := 1; t_sw3[s] := 0;
        T[s]:=max[T0, T[s] − ΔT] fi
      fi
    if flag = 1 then
      x[s] := N(x[s], i, j);
      if E(x[s]) < E(SOL[s]) then
        SOL[s] := x[s]; t2[s] := −1;
        t_sw3[s] := −1;
        if E(SOL[s]) < E(SOL0) then
          SOL0 := SOL[s] fi
      fi
      t_sw3[s] := t_sw3[s] + 1;
      t2[s] := t2[s] + 1 fi
  od (21)
  for s := 1 to n do (22)
    if t2[s] > cross_wait then
      x[s] と SOL0 を交叉させる;
      t_sw3[s] := 0; t2[s] := 0;
      T[s] := T0 fi
  od (22)
od (1)
end
```

## 6 計算機実験による評価

実験の結果を表1に示す. TSPLIBは,TSPのベンチマーク問題集の1つである. 実験に用いた計算機のCPUはAMD Athron850MHzである. 結果の上段は解精度 $\left(\frac{\text{得られた解}-\text{厳密解}}{\text{厳密解}} \times 100(\%)\right)$, 下段は実行時間(sec)であり, これはseedを変え10回ずつ実行したデータを平均している. また, 解精度右の括弧の中は厳密解を抽出した回数である.

共通パラメータについては, $T_0$=1, $R$=2.5, $n$=5 (RaTSPについては1), $max\_t$=(都市数)$^4/n$, $switch\_1$=(都市数)$^2$, $cross\_wait$ = (都市数)$^3$ とした. 但し, 保存されている探索中の最適解の更新が (都市数)$^{3.5}/n$ 回以上ない場合は, 最適解が既に求まったと判断し, そこで処理を終了する.

また, RaTSP_GAのみのパラメータについては, $\Delta T = 1$, $switch\_2$=2×(都市数)$^2$, $switch\_3$ = (都市数)$^2$ とする.

表1より, RaTSPよりもRaTSP_GA0が, また, これらに対してRaTSP_GAが全体的に解精度が向上していることがわかる. これは交叉により, より広い空間を探索できるようになったためと考えられる. RaTSP_GAでは更に探索の効率が向上したためと推測される. 実行時間については各々がほぼ同様の時間となっている.

表1 : TSPLIBに対する各アルゴリズムの結果
【上段:解精度, 下段:実行時間 (sec)】

| TSPLIB (厳密解) | RaTSP | RaTSP_GA0 | RaTSP_GA |
|---|---|---|---|
| eil51(426) | 0.37(3) | 0.02(9) | 0.00(10) |
| | 2.7 | 2.6 | 2.5 |
| eil101(629) | 0.05(7) | 0.00(10) | 0.00(10) |
| | 30.9 | 31.7 | 35.3 |
| pr107(44303) | 6.88(0) | 0.77(0) | 0.66(0) |
| | 40.4 | 47.8 | 40.1 |
| pr124(59030) | 0.98(0) | 0.04(5) | 0.02(8) |
| | 73.6 | 58.1 | 68.1 |
| pr136(96772) | 0.51(0) | 0.28(0) | 0.27(0) |
| | 90.1 | 104.6 | 101.1 |
| pr152(73682) | 7.43 | 0.91(0) | 0.78(0) |
| | 169.3 | 129.7 | 120.6 |
| rat99(1211) | 0.68(1) | 0.05(5) | 0.03(6) |
| | 26.1 | 29.1 | 27.2 |

## 7 むすび

局所的な探索を基としたRaTSPに対し大域的な更新を起こすGAを組み合わせたハイブリッド化により, 解精度を向上出来ることを示した. 他のアルゴリズムとの比較は今後の課題である.

**謝辞** 文献[2]を提供していただきました茨城大別役教授, 及び本研究の基礎に貢献していただいた電通大卒研生加賀谷俊介氏(現, 日立ソフト)に感謝いたします.

## 参考文献

[1] 山田義朗, 富田悦次, 高橋治久,"近似最大クリークを抽出する確率アルゴリズムとその実験的評価," 電子情報通信学会論文誌 (D-I),vol.76-D-I,no.2,pp.46-53 (1993).

[2] 新田寛, 別役廣, "確率アルゴリズムによる巡回セールスマン問題の解法," 茨城大学工学部システム工学科テクニカルレポート,IU-T/SE 01/95 (1995).

[3] 軍司栄一, 富田悦次, 若月光夫, 加賀谷俊介, "巡回セールスマン問題における確率及び遺伝的ハイブリッドアルゴリズムとその実験的評価," 電子情報通信学会情報・システムソサイエティ大会 (D-1-3) (1999).