Vol.2013-MPS-95 No.5
2013/9/26

予測交通量に基づくアントコロニー最適化法による時間依存 TSP の解法と広域道路網への適用

落合純一[†1]　　狩野均[†2]

本論文では，予測交通量を用いてアントコロニー最適化法（ACO）で時間依存 TSP（TDTSP）を解く手法を提案する．TDTSP とは，旅行時間が変化するタイプの TSP であり，ルーティング問題やスケジューリング問題など，いくつかの現実世界の問題をモデル化することができる．現実のルーティング問題を考えると，都市間の経路を求めてから旅行時間を計算する必要がある．旅行時間の変化間隔で良い解を求めるためには，探索の高速化が必要となるため，ACO のフェロモンの初期値に偏りを与えることで，探索領域の削減を行った．TSP のベンチマーク問題を用いて TDTSP を作成して評価実験を行った結果，提案手法は解の精度を落とすことなく収束が早まっていることを確認した．最後に，現実の道路網と交通量データを用いて実験を行い，現実の問題にも適用できることを確認した．

Solving Time-Dependent Traveling Salesman Problem using Ant Colony Optimization Based on Predicted Traffic and Its Application to a Real Road Network

JUNICHI OCHIAI[†1]　HITOSHI KANOH[†2]

In this paper, we propose an ant colony optimization based on the predicted traffic for time-dependent traveling salesman problems (TDTSP). TDTSP is a TSP where travel times between cities changes with time. Prediction values required for searching is assumed to be given in advance. In real problems, the travel time changing is associated with paths between cities and needs to be calculated by a shortest path algorithm. A faster algorithm is required to find the best solution every changing of travel times. We previously proposed a method to improve the search rate of MAX-MIN Ant System (MMAS) for static TSPs. In the current work, the method is extended so that the predicted travel time can be handled and formalized in detail. We also present a method of generating a TDTSP to use in evaluating the proposed method. Experimental results using benchmark problems and real world data suggested that the proposed method is better than the conventional method in the rate of search.

1. はじめに

アントコロニー最適化法（ACO）は，蟻の採餌行動をモデル化したメタヒューリスティクスであり，様々な組合せ最適化問題に適用されている[1][2]．ACO では，探索領域の情報をフェロモンと見なし，複数の蟻がフェロモンに基づいて解を作成する．そして，作成された解の情報がフェロモンにフィードバックされ，更新されたフェロモンを利用して蟻が解を作成することで探索が進む．よって，フェロモンの扱い方は ACO の性能に大きく影響を与える[1]．

現在までに ACO の性能向上に関する様々な論文が発表されており，多くのものは目的関数に時間要素が含まれていない静的な問題を対象としている．しかし，現実問題への適用を考える場合，目的関数に時間要素が含まれる動的な問題を解かなければいけないことが多々ある．そこで，本論文では時間依存巡回セールスマン問題（TDTSP）を対象とする．TDTSP とは都市間のコストが変化するタイプの巡回セールスマン問題（TSP）であり，ルーティング問題やスケジューリング問題など，いくつかの現実世界の問題をモデル化することができる[3]．動的な問題を解く一般的な方法は，コストが変化するごとに再探索を行うものである[4][5][6]．しかし，この方法で見つけた解は最良解とは限らない．

本論文では，TDTSP のコストを旅行時間とし，予測交通量[7]を用いて ACO で TDTSP を解く手法を提案し，広域道路網に適用する．提案手法は，旅行時間の変化間隔で精度の高い解を求めるために，著者らの以前の研究[8]を改良し，TDTSP に適用したものである．従来手法[8]は，通常の TSP を対象に，Nearest Neighbor（NN）法と 2-opt により複数の局所最適解を作成して初期解集合とし，初期解集合に含まれる辺のフェロモンの値を大きくすることで探索領域を狭める手法である．提案手法は，TDTSP 向けに NN 法を改良し，改良した NN 法を用いて初期解集合を作成する．

TDTSP を広域道路網に適用する場合，旅行時間の変化ごとに都市間の経路を求める必要がある．従来研究[9][10]では，解の探索前に都市間の経路を求めており，その計算時間に関して言及されていない．各都市間の各時間の経路をすべて求めると，計算時間と記憶容量の無駄が大きい．提案手法は，解の探索と同時に，必要に応じてダイクストラ法を実行することで，都市間の経路を求める．

以下では，まず研究分野の概要として，TDTSP，ACO，関連研究について述べる．次に提案手法について述べる．最後に評価実験として，TSP のベンチマークを用いた実験，現実の道路網と交通量データを用いた実験について述べ，提案手法の有効性を示す．

2. 研究分野の概要

2.1 時間依存巡回セールスマン問題（TDTSP）

巡回セールスマン問題（TSP）は完全グラフ $G=(N,A)$ として表現できる[11]．ここで，N は頂点集合，A は辺集合である．TSP では頂点を都市と呼び，辺にはコストが与えら

†1 筑波大学大学院システム情報工学研究科コンピュータサイエンス専攻
Department of Computer Science, Graduate School of Systems and Information Engineering, University of Tsukuba
†2 筑波大学システム情報系情報工学域
Division of Information Engineering, Faculty of Engineering, Information and Systems, University of Tsukuba

れている．TSP の目的は，総コストが最小となるハミルトン閉路を求めることである．TSP のベンチマークとして TSPLIB[12]が一般的に用いられている．TSPLIB で公開されている問題は，都市の番号と座標，都市間の距離の計算方法が定義されており，距離をコストとして用いる．

TDTSP は，都市が解に現れる位置によって次の都市へのコストが変化するタイプの TSP であり，最初に出発する都市が固定されているハミルトン閉路の中で，総コストが最小のものを求める問題である[3]．現実のルーティング問題を考えたとき，日本では道路交通情報通信システム（VICS）により 5 分間隔で最新の交通情報を得ることができる．そこで本研究では，コストを旅行時間とし，時間間隔 ΔT で旅行時間が変化するものとして，TSPLIB の問題から TDTSP を作成した．都市 i を出発する時間を T_i とするとき，最初の都市を出発してから旅行時間が変化した回数 $n_{\mathrm{up}}(T_i)$ はガウス記号[]を用いて式(1)で表す．

$$n_{\mathrm{up}}(T_i)=\left[\frac{T_i}{\Delta T}\right] \tag{1}$$

$n_{\mathrm{up}}(T_i)$ 回目の都市 i, j 間の旅行時間 $t_{ij}(n_{\mathrm{up}}(T_i))$ は，1 つ前の都市 i, j 間の旅行時間 $t_{ij}(n_{\mathrm{up}}(T_i)-1)$ に基づいて計算する．旅行時間の計算式を式(2)(3)に示す．

$$t_{ij}(n_{\mathrm{up}}(T_i))=\begin{cases} d_{ij} & \text{if} \quad n_{\mathrm{up}}(T_i)=0 \\ \min(d_{ij}, t_{ij}(n_{\mathrm{up}}(T_i)-1)\cdot C_f') & \text{otherwise} \end{cases} \tag{2}$$

$$C_f'=1+C_f\cdot R_{\mathrm{uni}} \tag{3}$$

d_{ij} は TSPLIB で定義されている都市 i, j 間の距離，C_f は範囲 [0,1] のパラメータ，R_{uni} は範囲 [−1,1] の一様乱数である．都市間の距離は制限速度で進んだ場合の旅行時間と仮定し，その都市間の旅行時間の最小値とする．都市間の旅行時間 $\{t_{ij}(0)=d_{ij}, t_{ij}(1), t_{ij}(2), \cdots\}$ は，解の探索前に計算し，正確な予測旅行時間として総旅行時間の計算に用いた．解 s の総旅行時間 $T(s)$ の計算方法を式(4)(5)に示す．

$$T(s)=\sum_{i=1}^{n} t_{i,i+1}(n_{\mathrm{up}}(T_i)) \tag{4}$$

$$T_i=\begin{cases} 0 & \text{if} \quad i=1 \\ T_{i-1}+t_{i-1,i}(n_{\mathrm{up}}(T_{i-1})) & \text{otherwise} \end{cases} \tag{5}$$

2.2 アントコロニー最適化法（ACO）

ACO とは，蟻の採餌行動をモデル化したメタヒューリスティクスであり，様々な組合せ最適化問題に適用されている[1][2]．一般的な ACO のアルゴリズムを図 1 に示す[1]．

まず，フェロモンを均一に初期化する．その後，複数の蟻が解を作成し，作成された解に基づいてフェロモンの更新が行われ，終了条件を満たすまでこの 2 つの処理が繰り返される．これにより，徐々にフェロモンに偏りが生じて探索が進む．

これらの処理を改良することで様々な ACO が提案されており[1]，性能が良い代表的な ACO として Ant Colony System（ACS）[13]と MAX-MIN Ant System（MMAS）[14]がある．ACS は収束が早い長所があるが，MMAS と比較すると解の精度が落ちる場合がある．MMAS は安定して精度の良い解を発見できるが，他の ACO より収束が遅い問題点がある．

```
Procedure ACO ( )
  Set parameters;
  Initialize pheromone trails;
  while ( terminal condition not met ) {
    Construct ant solutions;
    Apply local search;   / * option * /
    Update pheromone trails;
  }
```

図 1　一般的な ACO のアルゴリズム

2.3 関連研究

2.3.1 フェロモンの初期値の偏りによる ACO の高速化

通常の TSP を対象に，フェロモンの初期状態に偏りを与え，ACO の探索を高速化する研究がある[8] [15][16]．通常の TSP を対象とした場合，フェロモンは辺の重みとして値を持つ．一般的な ACO は，すべての辺のフェロモンの値が均一に初期化される．一方，辺ごとにフェロモンの初期値を変えることで，探索領域を狭めることができる．

著者らは，Nearest Neighbor（NN）法[11]と 2-opt[11]で複数の局所最適解を初期解集合として作成し，初期解ごとの精度に基づいてフェロモンの初期値に偏りを与える手法を提案した[8]．フェロモンの初期化法を式(6)(7)に示す．

$$\tau_{ij}=(1-r)\tau_0+r\frac{1}{n_0}\sum_{k=1}^{n_0}\delta\tau_{ij}^k \tag{6}$$

$$\delta\tau_{ij}^k=\begin{cases} \dfrac{1}{f(s_0^k)} & \text{if} \quad (i,j)\in s_0^k \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

τ_{ij} は都市 i, j 間のフェロモンの値，τ_0 は ACO のフェロモンの初期値，都市 n_0 は初期解の数，r は範囲 [0,1] のパラメータ，$f(s)$ は解 s の目的関数値，s_0^k は k 番目の初期解である．r は初期解の重みを表しており，r が 1 に近いほど，初期解を用いて追加されるフェロモンの値が大きくなる．

Tsai らは，NN 法を改良した DNN 法を提案し，DNN 法を用いてフェロモンの初期状態に偏りを与えた[15]．DNN 法は，都市をランダムに 1 つ選択し，その都市から最も離れた都市を選ぶ．この 2 つの都市を出発都市として，すべての都市が選択されるまで，交互に最も近い都市を選択する．よって，DNN 法では 2 つの経路が作成される．TDTSP を対象とする場合，最初の都市から最も離れた都市に関して，出発する時間を決めることができるため，DNN 法は TDTSP に適用できない．

また，Tsai らは DNN 法の提案の他に，NN 法で作成した複数の初期解を用いてフェロモンの初期状態に偏りを与える手法も提案している[15]．この手法は，初期解集合に含まれる辺の重複度によって，フェロモンの初期状態に偏りを与えている．提案手法も NN 法を用いて初期解を作成するが，提案手法は初期解の精度を考慮してフェロモンの初期値に偏りを与える点で，Tsai らの手法と異なる．

Dai らは，最小全域木（MST）[17]を用いて，式(8)によりフェロモンの初期状態に偏りを与えた[16]．

$$\tau_{ij}=\begin{cases} (\tau_0)^{1/\theta} & \text{if} \quad (i,j)\in \mathrm{MST} \\ \tau_0 & \text{otherwise} \end{cases} \tag{8}$$

θ は 1 以上のパラメータである．τ_0 は 1 未満の値であるため，MST に含まれる辺のフェロモンの値は τ_0 より大きくな

ることから，θ は MST の重みとなる．Dai らの手法は ACS をベースとしており，提案手法は MMAS をベースとしている．フェロモンの初期化法を評価するため，Dai らの手法を MMAS に適用した．このとき，基準となる τ_0 が ACS と MMAS で異なるため，τ_0 を式(9)で計算した．

$$\tau_0 = \frac{1-\sqrt[n]{0.05}}{(n/2-1)\cdot\sqrt[n]{0.05}} \cdot \frac{1}{\rho \cdot f(s_{\mathrm{NN}})} \tag{9}$$

n は都市数，ρ は ACO の一般的なパラメータであるフェロモンの蒸発率，s_{NN} は NN 法で作成した解である．

2.3.2 時間依存問題の広域道路網への適用

TDTSP を広域道路網に適用するとき，都市間には道路網が存在し，都市間の経路から旅行時間を計算する必要がある．TDTSP を広域道路網に適用した研究は見られないが，時間依存配送計画問題を広域道路網に適用した研究として[9][10]がある．Haghani らは遺伝的アルゴリズムを用いて時間依存配送計画問題を解いており[9]，Donati らは ACO を用いて時間依存配送計画問題を解いている[10]．どちらの手法も広域道路網への適用実験を行っており，最短経路問題を解いて都市間の経路と旅行時間を計算している．しかし，この計算は解の探索前に行われており，計算時間に関して言及されていない．各都市間の各時間の経路と旅行時間をすべて求めると，計算時間と記憶容量に無駄があるため，提案手法では必要に応じて最短経路問題を解く．

3. 提案手法

3.1 基本戦略

時間依存配送計画問題を対象とした ACO の研究[10][18]や，ACO のフェロモンの初期値に偏りを与える研究[15][16]では，ACS を対象としている．ACS の特徴として収束の早さがあるが，MMAS と比べると精度が落ちる場合がある．一方，MMAS は他の ACO と比べて精度の良い解を得られるが，収束に時間がかかる問題点がある[1]．そこで，提案手法は MMAS をベースとし，フェロモンの初期状態に偏りを与えることで，精度を落とさずに収束を早めることを目指す．

通常の TSP を対象とした著者らの手法[8]では，NN 法と 2-opt により複数の局所最適解を作成し，初期解集合とする．通常の TSP に対する 2-opt は，変更する辺の距離の差を計算するだけで解の評価ができる．一方，TDTSP を対象とする場合，変更が生じる都市以降の経路に関して，総旅行時間を計算し直す必要があり，計算量が大きくなる．よって，提案手法では 2-opt を用いず，NN 法のみで複数の初期解を作成する．

TDTSP は出発都市が固定されているため，通常の TSP を対象とした NN 法では 1 通りの解しか作成できない．そこで，出発都市から最初に訪問する都市をランダムに選択することで，最大 $(n-1)$ 通りの解が存在するように NN 法を改良した．提案手法で用いた NN 法のアルゴリズムを図 2 に示す．この NN 法では，最適解の 2 番目の都市が選ばれる確率は $n_0/(n-1)$ であり，n_0 が小さいほど見逃しが発生する確率が高くなる．よって，本研究では n_0 を $(n-1)$ とした．

3.2 提案手法のアルゴリズム

TDTSP の目的は，解の総旅行時間を最小にすることであるため，蟻による解の作成処理を TDTSP 向けに変更した．蟻 k が都市 i を時間 T_i に出発するとき，次に移動する都市として都市 j を選ぶ確率を式(10)(11)に示す．

```
Procedure NearestNeighbor ( second_city )
  s = (1, second_city);;
  N' = N − {1, second_city};
  i = second_city;
  T = t_1i(0);
  while ( N' is not empty ) {
    j = arg min_{k∈N'} t_ik(n_up(T));
    s ← s ⊗ j;     /* j is added to s * /
    N' ← N' − {j};
    T ← T + t_ij(n_up(T));
    i = j;
  }
  s ← s ⊗ 1;
  T ← T + t_i1(n_up(T));
  return s as a solution;
```

図 2　提案手法で用いた NN 法のアルゴリズム

$$p_{ij}^{k}(n_{\mathrm{up}}(T_i)) = \begin{cases} \dfrac{(\tau_{ij})^{\alpha}\cdot\{\eta_{ij}(n_{\mathrm{up}}(T_i))\}^{\beta}}{\sum_{l\in N^k}(\tau_{il})^{\alpha}\cdot\{\eta_{il}(n_{\mathrm{up}}(T_i))\}^{\beta}} & \text{if} \quad j\in N^k \\ 0 & \text{otherwise} \end{cases} \tag{10}$$

$$\eta_{ij}(n_{\mathrm{up}}(T_i)) = \frac{1}{t_{ij}(n_{\mathrm{up}}(T_i))} \tag{11}$$

N^k は未訪問都市集合，α と β は ACO の一般的なパラメータである．よって，フェロモンの値が大きければ大きいほど，旅行時間が小さければ小さいほど高い確率となる．提案手法のアルゴリズムは図 3 となる．s_{ib} はイテレーションごとの最良解を意味しており，s_{gb} は探索開始時からの最良解として最後に出力する．$\tau_{\max}$ と $\tau_{\min}$ は，MMAS で定義されているフェロモンの最大値と最小値である．

3.3 広域道路網への適用

本論文で広域道路網を考えるとき，TDTSP と区別するため，頂点を交差点，辺をリンクと表現し，リンクの旅行時間をリンク旅行時間と呼ぶ．また，都市は交差点上にあるものとする．本研究では，都市間の経路と旅行時間を高速に求めるため，リンク旅行時間が変化する ΔT ごとにダイクストラ法[17]を実行し，求めた経路と旅行時間を保持する．ただし，すべての都市間のすべての時間に対してダイクストラ法を実行すると無駄が大きいため，都市間の旅行時間が必要になった時点で，その都市間の経路が計算されていない場合に限りダイクストラ法を実行する．都市間の旅行時間が必要になるタイミングは，主に蟻による解の作成時であり（式(10)），1 対 1 の都市間の旅行時間が必要ではなく，1 対多の都市間の旅行時間が必要となる．よって，出発する都市から経路と旅行時間を確定する手順でダイクストラ法を実行することで，1 回のダイクストラ法で 1 対多の都市間の経路と旅行時間を求める．

ダイクストラ法で都市間の経路を求めるとき，最も旅行時間が大きい都市まで計算するのではなく，都市が n_{cl} 個確定した段階で計算を終了することで，ダイクストラ法の高速化を行った．n_{cl} は，通常の TSP を対象とする ACO で用いられている Candidate List[1]に関するパラメータである．Candidate List とは，都市ごとに近い都市を上位 n_{cl} 個並べたものであり，利用するためにはソートが必要になるが，提案手法ではダイクストラ法がソートの役割を果たしている．

```
Procedure ProposedMethod ( )
  Input TDTSP;
  Set parameters;
  Construct n_0 solutions using NN heuristic;
  Initialize pheromone trails using (6) and (7);
  while ( terminal condition not met ) {
    for ( k = 1; k ≤ m; k++ )
      Construct solution s^k using (10) and (11);
    s_ib = arg min_k T(s^k);
    if ( T(s_ib) < T(s_gb) ) {
      s_gb = s_ib;
      Calculate τ_max and τ_min;
    }
    Update pheromone trails;
  }
  Output best so far solution s_gb;
```

図 3　提案手法のアルゴリズム

よって，ダイクストラ法を実行したとき，都市間の経路と旅行時間の他に，Candidate List も保持する．

TDTSP の解を作成するとき，次の都市を訪問するごとにその都市の出発時間を計算する必要がある（式(5)）．この計算には，ダイクストラ法で求めた都市間の旅行時間は使用せずに，都市間の経路に沿ってリンク旅行時間から計算する．ダイクストラ法で求めた都市間の経路は，その時間帯における静的環境での最短経路であるため，都市間の経路が長くなるほど，都市間の旅行時間には誤差が含まれる．広域道路網の予測旅行時間の偏りが大きい場合，予測旅行時間が小さい時間帯に遠くの都市に訪問してしまい，現実的ではない解を求めてしまうことが考えられる．よって，都市の出発時間を計算するときは，ダイクストラ法で求めた経路に沿って計算する．これにより，解の総旅行時間も経路に沿って計算されたことになる．

この計算は解の評価に必要なものであり，フェロモンの更新により探索にも大きく影響する．しかし，経路に含まれる交差点数だけ計算量が必要であるのに対し，ダイクストラ法で計算済みの都市間の旅行時間は定数時間で利用できる．都市間の旅行時間に関して，蟻による解の作成の予備実験を行った結果，得られる解の精度に有意差はなかった．よって，蟻による解の作成で利用する都市間の旅行時間は，ダイクストラ法で計算された近似的なものとする．

4. 評価実験

本研究では，提案手法の有効性を確認するため，TSP のベンチマークを用いて都市間の旅行時間を変化させた実験（4.1 節），現実の道路網と交通量データを用いた実験（4.2 節）を行った．プログラムは Visual C++ 2010 64bit で作成し，実験環境は Windows7 64bit，Core i7-860，16GB RAM である．各実験では，乱数のシードを変えて 50 回実験を行い，その平均値で比較を行った．

4.1 TSP のベンチマークによる実験

4.1.1 実験方法

TDTSP の例題として，TSPLIB[12]で公開されている問題の中から表 1 の 10 問を用いた．これらの問題は小さいサイズの問題であるが，セールスマンが 1 日に訪問できる顧客の数は高々200 ほどであることから，評価実験に適していると考えた．日本では道路交通情報通信システム（VICS）

表 1　実験に用いた TSP のベンチマーク問題

問題	都市数	最適解	ΔT
eil51	51	426	5
eil76	76	538	5
eil101	101	629	5
kroA100	100	21282	300
u159	159	42080	300
d198	198	15780	300
kroA200	200	29368	300
pr299	299	48191	300
lin318	318	42029	300
d493	493	35002	300

により 5 分間隔で最新の交通情報を得ることができる．各問題の最適解から，eil51，eil76，eil101 の旅行時間の単位を分と仮定して ΔT を 5 とし，それ以外の問題は単位を秒と仮定して ΔT を 300 と設定した．C_f（式(3)）に関しては，現実の交通量データを確認して 0.1 と設定した．ΔT と C_f の値から，各問題について TDTSP を作成した．

従来手法として，ACS，MMAS，Dai らの手法（DAI）を提案手法と比較した．ただし，DAI は MMAS をベースとするように変更し（式(9)），TDTSP の最小全域木はプリム法[17]により近似的に求めた．予備実験から得られた各手法のパラメータ[1][8][16]の値を表 2 に示す．

4.1.2 最小全域木と初期解集合の評価

最小全域木と初期解集合を評価するために，次の指標を計算した（式(12)(13)(14)）．

$$R_{\text{cover}} = \frac{\left| A_0 \cap A(s_{\text{pb}}) \right|}{n} \tag{12}$$

$$R_{\text{reduction}} = 1 - \frac{|A_0|}{|A|} \tag{13}$$

$$A_0 = \begin{cases} A(\text{MST}) & \text{for DAI} \\ \bigcup_k A(s_0^k) & \text{otherwise} \end{cases} \tag{14}$$

s_{pb} は実験を通して得られた最良解，$A(s)$ は解 s に含まれる辺集合，A(MST) は最小全域木に含まれる辺集合である．よって，R_{cover} は最良解の辺の見逃しが少ないほど高い値となり，$R_{\text{reduction}}$ はフェロモンの初期値に偏りを与える辺が少ないほど高い値となる．DAI と提案手法の R_{cover} と $R_{\text{reduction}}$ の結果を表 3 に示す．表 3 の結果から次のことが確認できる．

- DAI で利用する最小全域木に含まれる辺の数は $(n-1)$ であるため，DAI の $R_{\text{reduction}}$ は 1 に近い値となるが，R_{cover} の値は提案手法の値よりも低い．
- 提案手法の R_{cover} の値は，DAI の値の約 3 倍に近く，DAI よりも最良解の辺を発見できている．
- 提案手法の $R_{\text{reduction}}$ の値は DAI の値よりも小さく，eil51 では全体の辺の約 2 割が初期解集合に含まれている．

基本的に R_{cover} と $R_{\text{reduction}}$ はトレードオフの関係にあり，R_{cover} は解の精度に，$R_{\text{reduction}}$ は収束の速さに影響する．たとえ $R_{\text{reduction}}$ が高くて探索領域を狭められているとしても，最良解の辺の見逃しが多くて R_{cover} が小さい場合，得られる解の精度が落ちてしまう．したがって，DAI と提案手法を比較すると，DAI の方が収束は早いが，提案手法は DAI よりも精度の良い解を発見できると予想される．

表 2　手法ごとのパラメータの値

パラメータ	ACS	MMAS	DAI	提案手法
蟻の数	10	n	n	n
α	1	1	1	1
β	5	5	5	5
ρ	0.1	0.02	0.02	0.02
n_{cl}	20	20	20	20
q_0	0.9	NA	NA	NA
θ	NA	NA	1.8	NA
r	NA	NA	NA	0.9
n_0	NA	NA	NA	n - 1

表 3　最小全域木と初期解集合の評価

問題	DAI		提案手法	
	R_{cover}	$R_{\mathrm{reduction}}$	R_{cover}	$R_{\mathrm{reduction}}$
eil51	0.33	0.98	0.92	0.79
eil76	0.39	0.99	0.94	0.84
eil101	0.32	0.99	0.94	0.85
kroA100	0.34	0.99	0.93	0.89
u159	0.39	0.99	0.97	0.90
d198	0.33	0.99	0.97	0.94
kroA200	0.32	1.00	0.97	0.92
pr299	0.36	1.00	0.98	0.92
lin318	0.36	1.00	0.97	0.93
d493	0.32	1.00	0.97	0.95

4.1.3 探索の高速化の定量的評価

探索の高速化を定量的に評価するため，式(15)で表す最良解の精度に対する誤差（%）を考える．

$$E(s)=\frac{f(s)-f(s_{\mathrm{pb}})}{f(s_{\mathrm{pb}})}\cdot 100 \qquad (15)$$

$E(s)$ が 10%と 5%となる解を発見できるまでの CPU 時間［秒］を表 4 に示す．表 4 では，問題ごとに各誤差の解が最も早く求まる CPU 時間を太字で表しており，解を発見できなかった場合はハイフンで表している．表 4 から次のことが確認できる．

- DAI は解の精度が悪く，3 つの問題で誤差 10%の解が発見できなかった．
- 収束の早さでは ACS が良い性能を示しているが，誤差 5%の解を発見できた問題数は MMAS と提案手法より少ない．
- MMAS と提案手法を比較すると，すべての問題で提案手法の収束が早い．

以上のことから，提案手法は MMAS と同様に安定して精度の良い解を求めつつ，MMAS より収束を早めていることが確認できる．

4.2 現実の道路網と交通量データによる実験

広域道路網への適用に関する評価として，現実のデータを用いた実験を行った．道路網はカーナビに用いられているナビ研 S 規格地図，交通量データは車両感知器から得られる VICS データを用いた．対象とした地域は，東京都中央区 9km×11km の一般道，VICS データは 2003 年 6 月 17 日（火）のものを予測交通量として用いた．都市は交差点上にランダムに作成し，都市数 51，101，201，301，501 の 5 つの問題を解いた．都市数 101 の最良解を図 4 に示す．

探索の様子を確認するため，CPU 時間［秒］に対する最良解の精度を図 5 に示す．図 5 より，次のことが確認できる．

表 4　近似解を求めるために必要な CPU 時間［秒］

問題	ACS		MMAS		DAI		提案手法	
	10%	5%	10%	5%	10%	5%	10%	5%
eil51	**0.01**	**0.18**	0.14	0.36	0.02	2.36	0.05	0.22
eil76	**0.53**	-	1.00	-	4.87	-	0.62	-
eil101	1.35	-	2.54	8.36	**0.86**	-	1.70	**8.16**
kroA100	**0.14**	1.34	1.22	1.58	0.44	16.48	0.51	**0.88**
u159	8.22	-	6.28	-	23.64	-	**4.90**	-
d198	**0.55**	41.78	5.39	18.42	1.01	-	3.72	**17.01**
kroA200	**1.35**	-	9.91	20.43	-	-	6.39	**17.33**
pr299	80.65	-	36.11	-	-	-	**30.17**	-
lin318	46.74	-	48.21	-	-	-	**38.94**	-
d493	**112.21**	-	144.23	-	174.58	-	115.53	-

図 4　現実のデータを用いた実験で得られた最良解（都市数 101）

- DAI と提案手法のフェロモンの初期化の計算時間は，探索全体に対して無視できる．
- 都市数 51 のサイズが小さい問題では，ACS と DAI が精度の良い解を早く求めている．
- 問題サイズが大きくなると，ACS と DAI の解の精度は落ちている．
- 提案手法は，すべての問題で MMAS と同等の解の精度が得られており，MMAS より収束が早い．

以上のことから，ベンチマーク問題と同様に，提案手法の有効性を確認できる．

5. おわりに

本論文では，予測交通量に基づく ACO による TDTSP の解法を提案した．提案手法は，安定して良い解を求めることができる ACO である MMAS をベースとし，フェロモンの初期値に偏りを与えることで探索領域を狭め，収束を早めるものである．複数の初期解に基づいてフェロモンの初期化を行うため，NN 法の改良を行った．提案手法の有効性を確認するため，TSP のベンチマークを用いて TDTSP を作成した実験と，現実の道路網と交通量データを用いた実験を行った結果，提案手法は解の精度を落とさずに収束を早めていることを確認した．

今後の課題として，実際の旅行時間が変化するごとに再探索を行う手法[4][5][6]と組み合わせることが考えられる．現実の問題を考えると，セールスマンが実際に都市を移動している間でも旅行時間は変化し，予測交通量には誤差が存在する．よって，旅行時間が変化するごとに，最新の交通情報から予測交通量を計算し，セールスマンが訪問している都市から解を探索することが必要と考えられる．

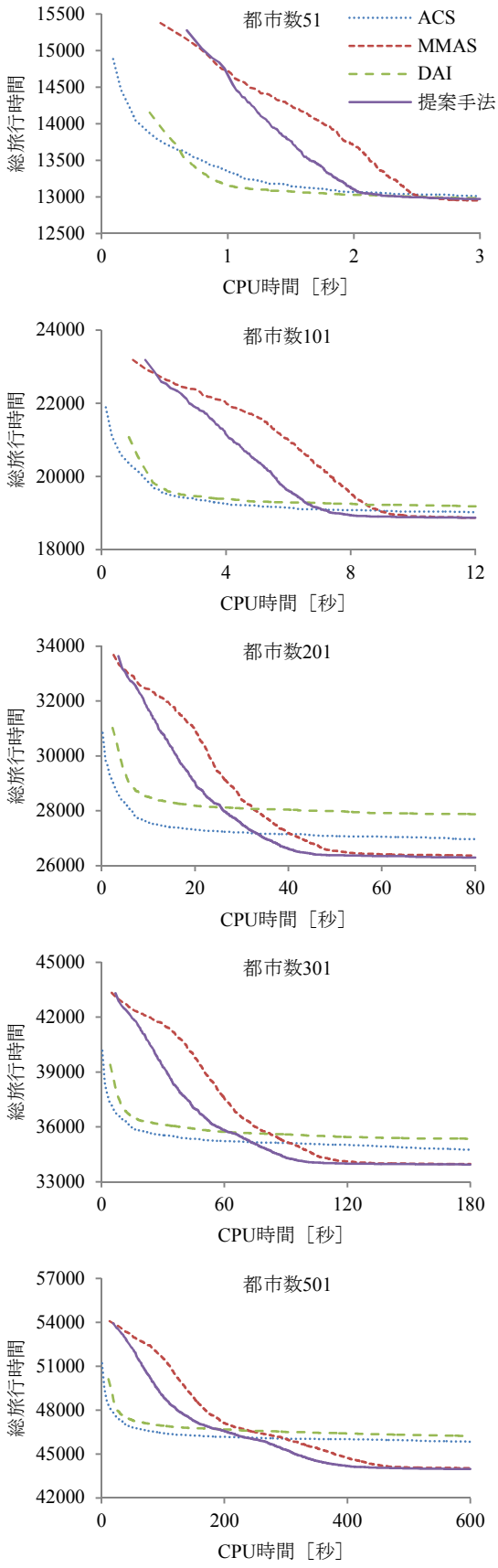

図 5　CPU 時間に対する最良解の精度

謝辞　本研究は JSPS 科研費 23500169 の助成を受けたものです.

参考文献

1) Dorigo, M. and Stützle, T.: Ant Colony Optimization, MIT Press (2004).
2) Dorigo, M. and Stützle, T.: Ant Colony Optimization: Overview and Recent Advances, Handbook of Metaheuristics, International Series in Operations Research & Management Science, Vol. 146, pp. 227-263, Springer US (2010).
3) Gouveia, L. and Vob, S.: A classification of formulations for the (time-dependent) traveling salesman problem, European Journal of Operational Research, Vol. 83, No. pp. 69-82 (1995).
4) Guntsch, M. and Middendorf, M.: Pheromone modification strategies for ant algorithms applied to dynamic TSP, Applications of Evolutionary Computing, Lecture Notes in Computer Science, Vol. 2037, pp. 213-222, Springer Berlin Heidelberg (2001).
5) Eyckelhof, J.C. and Snoek, M.: Ant System for a Dynamic TSP: Ant Caught in a Traffic Jam, Ant Algorithms, Lecture Notes in Computer Science, Vol. 2463, pp. 88-99, Springer Berlin Heidelberg (2002)
6) Mavrovouniotis, M. and Yang, S.: An Immigrants Scheme Based on Environmental Information for Ant Colony Optimization for the Dynamic Traveling Salesman Problem, Artificial Evolution, Lecture Notes in Computer Science, Vol. 7401, pp. 1-12, Springer Berlin Heidelberg (2012).
7) Kanoh, H. and Ochiai, J.: Solving Time-Dependent Traveling Salesman Problems using Ant Colony Optimization Based on Predicted Traffic, Proc. International Symposium on Distributed Computing and Artificial Intelligence, Advances in Intelligent and Soft Computing, Vol. 151, pp. 25-32, Springer Berlin Heidelberg (2012).
8) Kanoh, H., Ochiai, J. and Kameda, Y.: Pheromone Trail Initialization with Local Optimal Solutions in Ant Colony Optimization, International Journal of Knowledge-Based and Intelligent Engineering Systems, accepted (2013).
9) Haghani, A. and Jung, S.: A dynamic vehicle routing problem with time-dependent travel times, Computers & Operations Research, Vol. 32, No. 11, pp. 2959-2986 (2005).
10) Donati, V.A., Montemanni, R., Casagrande, N., Rizzoli, E.A. and Gambardella, M.L.: Time dependent vehicle routing problem with a multi ant colony system, European Journal of Operational Research, Vol. 185, No. 3, pp. 1174-1191 (2008).
11) Gutin, G. and Punnen, P.A.: The traveling salesman problem and its variations, Combinatorial Optimization, Vol. 12, Springer US (2007).
12) Reinelt, G., et al.: TSPLIB, available from 〈http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/〉 (accessed 2013-08-26).
13) Dorigo, M. and Gambardella, M.L.: Ant colony system: a cooperative learning approach to the traveling salesman problem, IEEE Trans. Evolutionary Computation, Vol. 1, No. 1, pp. 53-66 (1997).
14) Stützle, T. and Hoos, H.H.: MAX-MIN Ant System, Future Generation Computer System, Vol. 16, No. 8, pp. 889-914 (2000).
15) Tsai, C.-F., Tsai, C.-W. and Tseng, C.-C.: A new hybrid heuristic approach for solving large traveling salesman problem, Information Sciences, Vol. 166, pp. 67-81 (2004).
16) Dai, Q., Ji, J. and Liu, C.: An Effective Initialization Strategy of Pheromone for Ant Colony Optimization, Proc. 4th International Conference on Bio-Inspired Computing, pp. 1-4, IEEE (2009).
17) Korte, B. and Vygen, J.: Combinatorial Optimization: Theory and Algorithms, Algorithms and Combinatorics, Vol. 21, Springer Berlin Heidelberg (2000).
18) Montemanni, R., Gambardella, M.L., Rizzoli, E.A. and Donati, V.A.: Ant Colony System for a Dynamic Vehicle Routing Problem, Journal of Combinatorial Optimization, Vol. 10, No. 4, pp. 327-343 (2005).