4K1-3

アントコロニー最適化法とダイクストラ法のハイブリッド化による時間依存 TSP の解法

Solving Time-Depenedent Traveling Salesman Problems using Ant Colony Optimization and Dijkstra's Algorithm

落合 純一[*1] Junichi Ochiai

狩野 均[*2] Hitoshi Kanoh

[*1] 筑波大学大学院システム情報工学研究科
Graduate School of System and Information, University of Tsukuba

[*2] 筑波大学システム情報系
Faculty of Engineering, Information and Systems, University of Tsukuba

In this paper, we propose a hybrid method for time-dependent traveling salesman problems (TDTSP) using ant colony optimization and Dijkstra's algorithm. TDTSP is a TSP where travel times between cities vary from hour to hour. The travel time changing is associated with paths between cities. We previously proposed a method to solve TDTSP efficiently. In the previous work, the travel times were calculated based on euclidean distance. In the current work, the method is extended so that the travel times can be calculated by Dijkstra's algorithm. We also present a method of generating virtual maps to use in evaluating the proposed method. Experimental results using virtual maps with 51 to 318 cities suggested that the proposed method is better than conventional methods 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，様々な最適化問題に適用されている[Dorigo 04]. ACO では，探索空間にフェロモンが分布され，複数の蟻がフェロモンを利用して解を作成し，作成した解の情報がフェロモンにフィードバックされる. 解の作成とフェロモンの更新を繰り返すことで，徐々にフェロモンの偏りが大きくなり探索が収束する. 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[Dorigo 04].

著者らは以前に，時間依存巡回セールスマン問題 (TDTSP) を対象とした ACO の高速化手法を提案した[落合 11][Kanoh 12]. 通常の ACO は，フェロモンを均一に初期化することで，探索初期は幅広く探索し，探索が進むとフェロモンに偏りが生じる. 一方著者らは，偏りを与えてフェロモンを初期化することで，予め探索領域を狭めて探索速度を向上させた. この手法の対象問題である TDTSP とは，都市間の旅行時間が時々刻々と変化するタイプの巡回セールスマン問題 (TSP) である[Gutin 02]. ここでは，都市間のユークリッド距離に乱数をかけることで，都市間の旅行時間を変化させ，TDTSP を作成した. しかし，地図への応用を考えると，都市間に道路網が存在するため，都市間の経路を求めて，その経路から旅行時間を計算する必要がある.

本稿では，都市間の経路をダイクストラ法で求めることで，実際の地図に応用可能な TDTSP の解法を提案する. 以下では，まず関連研究について述べる. ここでは，TDTSP，ACO の高速化，地図への応用例について述べる. 次に提案手法について述べる. 最後に，評価実験について述べる. ここでは，仮想地図の作成方法，仮想地図を用いた実験結果について述べる.

連絡先：落合純一，筑波大学大学院システム情報工学研究科，〒305-8573 茨城県つくば市天王台 1-1-1，029-853-6909，ochiai@kslab.cs.tsukuba.ac.jp

2. 関連研究

2.1 時間依存巡回セールスマン問題 (TDTSP)

TSP は，複数の都市を巡る巡回路のうち，総旅行時間が最小となる巡回路を求める問題である. 現実世界では，交通量の変化に伴って，都市間の旅行時間も変化する. TDTSP は，都市間の旅行時間が時々刻々と変化するタイプの TSP であり，交通量の変化を考慮した TSP と考えることができる. TDTSP では都市間の旅行時間が変化するため，同一の巡回路でも最初に出発する都市が異なると総旅行時間が異なってしまう. よって，出発都市は固定されている[Gutin 02].

本研究では地図への応用を目的としているため，道路の旅行時間が一定期間で変化する状況を仮定する. ただし，道路の旅行時間は探索開始前に予測されたものであるとし，予測旅行時間に基づく総旅行時間の最小化を目指す. 51 都市から 318 都市の問題を TSP のベンチマーク[TSPLIB]から選択し，問題ごとに道路網を作成することで実験を行った.

2.2 ACO の高速化手法

著者らは，TDTSP を対象とした ACO の高速化手法を提案した[落合 11][Kanoh 12]. 環境が変化する問題に対しては，環境が変化するごとに再探索を行うことが一般的であり，環境が変化する前に良い解を見つける必要がある. そこで，予測旅行時間を用いて良い解をなるべく早く求めるため，ACO の高速化を行った. この手法は，代表的な ACO である MMAS[Stützle 00]をベースとしている. 通常，MMAS のフェロモンの初期化は，全ての辺のフェロモンの値を均一に初期化する. 一方著者らは，TSP の代表的な Greedy 法である Nearest Neighbor 法 (NN 法) で初期解を複数作成し，その初期解集合に含まれる辺のフェロモンの初期値を大きく，それ以外の辺のフェロモンの初期値を小さくした. NN 法で作成された解は最適解の辺を含みやすい

ため, NN 法で複数通りの解を作成することで, 最適解の辺を網羅しつつ探索領域を狭めることに成功した.

TSP を対象とした NN 法は, 出発都市をランダムに選び, 旅行時間が最小の未訪問都市に移動していくことで解を作成する. よって都市数を n とすると, 最大 n 通りの解が存在する. しかし, TDTSP の出発都市は固定されているため, NN 法をそのまま適用すると 1 通りの解しか存在しない. 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ぶことで, 最大 $(n-1)$ 通りの解が存在するように NN 法を変更した.

2.3 時間依存問題の地図への応用例

TDTSP を地図に応用した例は見られないため, 時間依存配送計画問題を地図に応用した研究[Donati 08][Haghani 05]について述べる. 配送計画問題とは, デポと呼ばれる配送センターから複数の顧客に荷物を運ぶとき, 配送車の台数を最小化し, かつ総旅行時間が最小となる顧客の訪問順を求める問題である. よって, 時間依存配送計画問題は TDTSP を拡張した問題と考えることができる. [Donati 08]と[Haghani 05]は, どちらも顧客間の経路は探索開始前に求められており, その経路の計算時間に関しては言及されていない. しかし現実問題への応用を考えると, 解の探索と同時に経路を求めなければならない. そこで本研究では, 解の探索と同時に, 都市間の経路を必要に応じて求めた.

3. 提案手法

3.1 都市間の経路と旅行時間

実時間で時間依存最短経路問題を解くことは困難であるため, 道路の旅行時間が変化する一定期間 ΔT ごとに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. ただし, ΔT ごとの全都市間に対して経路を求めるのは無駄が大きいため, 必要になった場合にのみ経路を求める. ダイクストラ法を実行することで, 経路と同時に旅行時間が求まる. 都市 i を時刻 T_i に出発するとき, ダイクストラ法で求めた都市 j への経路を $P_{ij}(n_{\mathrm{up}}(T_i))$, 旅行時間を $t_{ij}(n_{\mathrm{up}}(T_i))$ で表す. ここで, $n_{\mathrm{up}}(T_i)$ は何回目の旅行時間の変化かを表す番号であり, 以下の式から求まる.

$$n_{\mathrm{up}}(T_i)=\left\lfloor \frac{T_i}{\Delta t} \right\rfloor$$

$t_{ij}(n_{\mathrm{up}}(T_i))$ はダイクストラ法で求めているため, $t_{ij}(n_{\mathrm{up}}(T_i))$ の計算途中では道路の旅行時間は変化せず, $P_{ij}(n_{\mathrm{up}}(T_i))$ が長ければ長いほど誤差が大きいと考えられる. そこで, $P_{ij}(n_{\mathrm{up}}(T_i))$ に沿って都市 i から都市 j に向かうとき, 道路の旅行時間を変化させて求めた都市 i, j 間の旅行時間を $t^*_{ij}(T_i)$ で表す. $n_{\mathrm{up}}(T_i)$ が等しくても, T_i が異なることで $t^*_{ij}(T_i)$ も異なる値になることが考えられるため, $t^*_{ij}(T_i)$ は保存することができない. 解の総旅行時間, 都市の出発時刻は正確に旅行時間を計算する必要があるため, $t^*_{ij}(T_i)$ を用いて計算する. 一方, 3.4 節で述べる蟻による解の作成は, $t_{ij}(n_{\mathrm{up}}(T_i))$ を用いて近似的に計算する.

3.2 巡回路の総旅行時間

解 s の総旅行時間 $T(s)$ を以下の式で定義する.

$$T(s)=\sum_{k=0}^{n-2} t^*_{k,k+1}(T_k)+t^*_{n-1,0}(T_{n-1})$$

$$T_k=\begin{cases} 0 & \text{if } k=0 \\ T_{k-1}+t^*_{k-1,k}(T_{k-1}) & \text{otherwise} \end{cases}$$

出発都市の出発時刻 T_0 は 0 と設定し, 順に都市の出発時刻を求め, 最後に出発都市への旅行時間を加えることで $T(s)$ が求まる.

3.3 Nearest Neighbor 法 (NN 法)

[落合 11]と同様に, 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 NN 法を利用する. ただし, 提案手法では最も近い都市の選択に以下の式を用いる.

$$k=\arg\min_{j\in N'} t^*_{ij}(T_i)$$

ここで, N'は未訪問都市の集合である. 最も近い都市が選択され, その都市に移動するごとに時刻の更新を以下の式を用いて行う.

$$T_k=T_i+t^*_{ik}(T_i)$$

3.4 ACO による解の作成

蟻が都市 i にいるとき, 次に移動する都市として j を選ぶ確率 p_{ij} を以下の式で定義する.

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

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

ここで, τ_{ij} は都市 i, j 間のフェロモンの値, α と β は定数である. η_{ij} はヒューリスティクス値と呼ばれる問題ごとに異なる変数であり, 総旅行時間が短い巡回路を求める TDTSP の場合, η_{ij} は都市 i, j 間の旅行時間の逆数となる. 移動する都市が決定した場合, NN 法と同様に, $t^*_{ij}(T_i)$ を利用して時刻を更新する.

ACO では, 解の作成処理の高速化のため, Candidate List[Dorigo 04]がよく用いられており, 本研究でも利用した. Candidate List とは, 都市ごとに近い都市を上位 N_{CL} 個並べたものであるため, 利用するためにはソートが必要となる. しかし本研究では, ダイクストラ法がソートの役割を果たしており, 上位 N_{CL} 個の目的地が確定した段階でダイクストラ法を中断できるため, ダイクストラ法の高速化にもつながる. 本研究では N_{CL} を 20 と設定した.

4. 評価実験

4.1 仮想地図

仮想地図の作成は以下の手順で行う.

(1) 幅 Δd の格子状の道路網を作成
(2) TSP のベンチマーク問題を入力
(3) 都市以外の交差点をランダムに削除
(4) 削除した交差点につながっている道路を削除
(5) 道路の旅行時間を設定

1 本の道路の長さとなる Δd は，TSP のベンチマーク問題によって異なる値にする．問題の都市は交差点上に設定し，都市が交差点と一致しない場合は都市を一番近い交差点に移動する．このとき，複数の都市による交差点の重複は許可するものとした．また，問題に記されている最初の都市を出発都市とする．次に，仮想地図を現実の地図に近づけるため，全交差点の N_{del} %の交差点をランダムに選んで削除する．その時，都市が設置されていない交差点を選ぶ．本研究では，$N_{\mathrm{del}} = 20$ として仮想地図を作成した．交差点を削除した後，その交差点に接続している道路を削除する． 最後に，存在する道路の旅行時間を設定する．本研究では，旅行時間の単位を秒と仮定し，旅行時間を以下の式で変化させた．

$$t_L(n_{\mathrm{up}}(T)) = [R_{\mathrm{jam}} t_{\min} + 0.5]$$

ここで，L は道路番号，R_{jam} は範囲 [1,2] の乱数，$t_{\min}$ は道路の最小旅行時間である．R_{jam} は渋滞率，$t_{\min}$ は制限速度での旅行時間を表している．道路の旅行時間は 300 秒ごとに変化する状況を想定し，ΔT は 300 とした．図 1 に作成した仮想地図の例を示す．

4.2 比較手法

比較手法として，MMAS の他に，最小全域木を用いたフェロモンの初期化手法[Dai 09] (DAI) を用いる．DAI は，MMAS をベースとしていないため，MMAS に対応するようにフェロモンの初期化を変更した．DAI がベースとして用いている ACO のフェロモンの初期値は，MMAS のフェロモンの最小値に相当する．よって，以下のフェロモンの初期化手法を DAI として用いる．

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

$$\tau_0 = \frac{1}{\rho\, T(s_{\mathrm{NN}})} \frac{\left(1 - \sqrt[n]{0.05}\right)}{\left(\frac{n}{2} - 1\right)\sqrt[n]{0.05}}$$

ここで， MST は最小全域木，θ は MST の重み，ρ はフェロモンの蒸発率，s_{NN} は NN 法で求めた解である．MST は TDTSP に対してプリム法で求めており，最小全域木の近似解として利用した．プリム法の実行時には NN 法と同様に，最も近い都市の選択と，都市の出発時刻の計算に $t^*_{ij}(T_i)$ を用いた．

4.3 実験条件

プログラムは Visual C++ 2010 64bit で作成した．実験環境は，OS が Windows7 64bit，CPU が Core i7-860，メモリ容量が 16GB である．表 1 に TSP のベンチマークと仮想地図の対応を示す．ここでの最良解は，各問題で発見できた一番良い解の総旅行時間である．表 2 に各手法のパラメータの値を示す．ここで，r は NN 法で作成した初期解の辺の重みである[落合 11]．これらの値は予備実験により求めた．探索の終了条件は，全ての問題で 6000 ステップとした．これらの条件の下で，各問題について，手法ごとに 50 回実験を行った．

4.4 実験結果

表 3 に各手法の探索終了時の最良解の精度を示す．μ は平均値，σ は標準偏差であり，それぞれの値は各問題の最良解で割った値である．eil51 と d198 以外の問題は MMAS と提案手法に違いが見られず，DAI は全ての問題で精度が低下している．

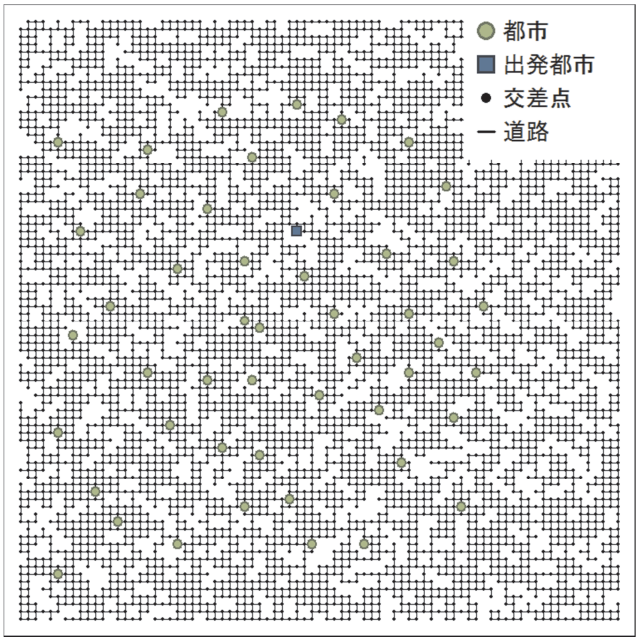

図 1 仮想地図の例 (eil51)

表 1 TSP のベンチマークと仮想地図の対応

ベンチマーク		仮想地図			
問題	n	交差点の数	Δd	$t_{\min}$	最良解
eil51	51	5249	1	40	35529
eil76	76	5249	1	40	43611
eil101	101	5249	1	40	50967
d198	198	8009	30	20	19396
kroA200	200	8009	30	20	40921
lin318	318	13409	30	20	53549

表 2 各手法のパラメータの値

パラメータ	MMAS	DAI	提案手法
蟻の数	$n-1$	$n-1$	$n-1$
α	1	1	1
β	4	8	4
ρ	0.02	0.02	0.02
r	N/A	N/A	0.9
θ	N/A	1.4	N/A

図 2 と図 3 に探索過程の最良解の精度を示す．ここでは eil101 と lin318 の結果を示しているが，他の問題に対しても同様の結果が得られている．収束速度に関しては，DAI，提案手法，MMAS の順で収束が速いことがわかる．しかし，提案手法は解の精度を維持したまま収束を速めているのに対し，DAI は解の精度が低下している．

4.5 フェロモン初期化の評価

MMAS，DAI，提案手法の違いは，フェロモンの初期化のみである．そこで，問題ごとの最良解に基づいて，提案手法と DAI のフェロモンの初期化に用いた，フェロモンの値を大きくする辺の評価を行う．まず重要な点は，最良解の辺の見逃しである．最良解の辺を見逃してしまうと，その辺が探索されにくくなり，解の精度の低下につながる．次に重要な点は，フェロモンの値を大きくする辺の数である．この辺の数が小さければ小さいほど探索領域の削減に成功していることになる．以上の 2 点から，提

表 3 6000 ステップ後の最良解の精度

問題	MMAS		DAI		提案手法	
	μ	σ	μ	σ	μ	σ
eil51	1.01	0.01	1.02	0.01	1.02	0.01
eil76	1.03	0.01	1.04	0.01	1.03	0.01
eil101	1.03	0.01	1.05	0.01	1.03	0.01
d198	1.02	0.02	1.06	0.02	1.03	0.02
kroA200	1.03	0.01	1.05	0.02	1.03	0.01
lin318	1.03	0.01	1.05	0.02	1.03	0.01

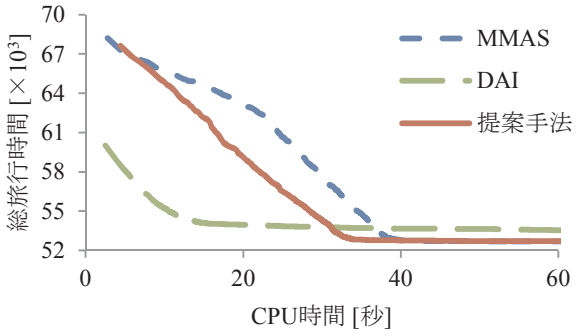

図 2 探索過程の最良解の精度 (eil101)

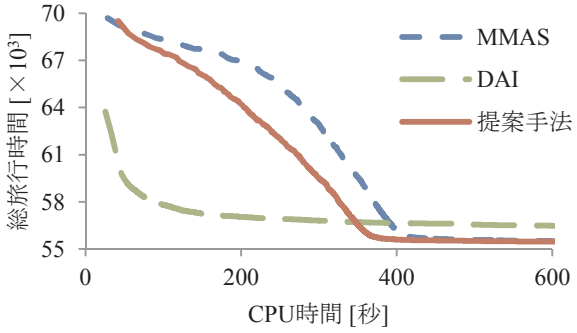

図 3 探索過程の最良解の精度 (lin318)

案手法では初期解集合に含まれる辺，DAI では最小全域木に含まれる辺に対して，最良解の辺を含んでいる割合を表す包含率，フェロモンの値を大きくしない辺の割合を表す削減率を求めた (表 4)．DAI は最小全域木を利用しているため，フェロモンの値を大きくする辺の数は $(n-1)$ となる．よって，削減率は高い値となっており収束は速いが，包含率を高くすることは難しいため解の精度が低下していると考えられる．一方，提案手法は NN 法で初期解を複数作成して多様な辺を求めているため，DAI と比べて削減率は小さいが包含率は高くなっている．このことから，地図を用いた TDTSP に対しても NN 法の解を利用したフェロモンの初期化法は有効であり，探索効率の向上に成功していると考えられる．

5. おわりに

本稿では，ACO とダイクストラ法を用いた，地図上の TDTSP の解法を提案した．これは，都市間の経路を必要に応じてダイクストラ法で求めることで，解の探索と都市間の経路探索を同時に行うものである．ACO のフェロモンの初期化では，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た．提案手法の

表 4 包含率と削減率

問題	DAI		提案手法	
	包含率	削減率	包含率	削減率
eil51	0.31	0.98	0.86	0.87
eil76	0.43	0.99	0.95	0.90
eil101	0.44	0.99	0.97	0.92
d198	0.42	0.99	0.89	0.96
kroA200	0.46	1.00	0.95	0.96
lin318	0.36	1.00	0.89	0.97

有効性を確認するため，TSP のベンチマークを用いて仮想地図を作成して実験を行い，提案手法は解の精度を落とすことなく，収束速度が向上していることを確認した．

今後の課題として，実際の地図と交通量データを用いた評価実験が考えられる．次に，予測交通量の誤差について考察することが考えられる．本研究では，道路の予測交通量は精度 100%として評価実験を行った．しかし，実際に交通量を予測するのであれば誤差が存在し，探索精度に影響を与えると考えられる．

参考文献

[Dai 09] Dai, Q., Ji, J., and Liu, C.: An Effective Initialization Strategy of Pheromone for Ant Colony Optimization, In Proc. of the 4th International Conference on Bio-Inspired Computing, pp. 1-4, IEEE (2009)

[Donati 08] Donati, V., Montemanni, R., Casagrande, N., Rizzoli, E., and Gambardella, L.: Time Dependent Vehicle Routing Problem with a Multi Ant Colony System, European Journal of Operational Research, Vol. 185, pp. 1174-1191, ELSEVIER (2008)

[Dorigo 04] Dorigo, M., and Stützle, T.: Ant Colony Optimization, MIT Press (2004)

[Gutin 02] Gutin, G., and Punnen, A. P. (Eds.): The Traveling Salesman Problem and Its Variations, Combinatorial Optimization, Vol. 12, Kluwer Academic Publishers (2002)

[Haghani 05] Haghani, A., and Jung, S.: A Dynamic Vehicle Routing Problem with Time-Dependent Travel Times, Computers & Operations Research, Vol. 32, pp. 2959-2986, ELSEVIER (2005)

[Kanoh 12] Kanoh, H., and Ochiai, J.: Solving Time-Dependent Traveling Salesman Problems using Ant Colony Optimization Based on Predicted Traffic, In Proc. of International Symposium on Distributed Computing and Artificial Intelligence, pp. 25-32, Springer (2012)

[Sherali 98] Sherali, H. D., Ozbay, K., and Subramanian, S.: The Time-dependent Shortest Pair of Disjoint Paths Problem: Complexity, Models, and Algorithms, Networks, Vol. 31, Issue 4, pp. 259-272, Wiley (1998)

[Stützle 00] Stützle, T., and Hoos, H. H.: MAX-MIN Ant System, Future Generation Computer System, Vol. 16, No. 8, pp. 889-914, ELSEVIER (2000)

[TSPLIB] http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95

[落合 11] 落合, 狩野: 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, 情報処理学会 数理モデル化と問題解決研究会, MPS86BIO27-35 (2011)