巡回セールスマン問題を解く 枝組立交叉（EAX）の一性能改善施策

吉川　克哉[††]　　髙橋　良英[††]

巡回セールスマン問題（TSP:Traveling Salesman Problem）を解く枝組立交叉（EAX:Edge Assembly Crossover）の性能改善策として、アーント・コロニー最適化手法（ACO:Ant Colony Optimization）による突然変異方式 EAX を提案する。本方式では局所最適解に陥った EAX の解探索空間を ACO により拡げ解の探索を続行する。中規模 TSPLIB データ d198、kroA200 を用いた C 言語実験の結果、提案方式は集団サイズが小さい場合でも最適解を 100%探索できること、探索時間についても、EAX 単独で最適解を得るのにかかった時間の約 60%で探索できることを確認した。

A Performance Improvement of Edge Assembly Crossover to Solve the Traveling Salesman Problem

Katsuya Yoshikawa[††]　and Ryouei Takahashi[††]

This paper proposes an improved method of EAX (Edge Assembly Crossover) with ACO (Ant Colony Optimization) mutation operation to solve the TSP (Traveling Salesman Problem). This method enables ACO to enlarge the solution space for EAX to search for the optimum solution successively, when EAX falls into local optimum solutions and cannot get rid out of them. C experiments using the medium sized TSP data such as d198 and KroA200 show that the proposed method can find out the optimum solution 100% even though the population size of EAX is small and that for finding the optimum solution the proposed method requires only 60% of the computational time that is necessary for EAX alone to find it.

1. はじめに

巡回セールスマン問題（TSP:Traveling Salesman Problem）とは、セールスマンが n 箇所の都市を 1 回だけ巡回するための最短経路を求める、代表的な組合せ最適化問題である。TSP の最適解の近似を効率的に得る手法として遺伝的アルゴリズム(GA:Genetic algorithm)が有効であることが知られている。GA の探索性能は、遺伝的操作である交叉に依存するため、TSP に対して多くの交叉が提案されている。その中でも枝組立交叉(EAX:Edge Assembly Crossover)は、探索率、探索時間の両面において優秀である。

これまで、TSP を解く枝組立交叉の性能改善対策として、集団サイズが少ない場合でも集団の多様性を確保できるように Lin-Kernighan 法を突然変異手段とする EAX 手法が提案されている。突然変異は局所最適解に陥った EAX の解空間をその近傍まで拡げる手段であり、それを初期値として EAX が解の再探索を行うので、SA（シミュレーテッドアニーリング）の一実現手段と解釈できる。

本研究では、アーント・コロニー最適化手法（ACO: Ant Colony Optimization）を突然変異手段とする EAX 手法を検討する。今回は、TSP のベンチマーク（TSPLIB）を用いた EAX の性能改善結果を報告する。

2. 枝組立交叉（EAX）

EAX では、２つの親（tuor-A、tuor-B）の枝情報を和した枝集合ＧAB を生成する。ＧAB は A と B の枝が交換して現れる幾つかの部分巡回路（AB-Cycle）に分割される。AB-Cycle と親個体を重ね合せ、緩和個体（部分巡回路集合）を生成する。緩和個体数 N は AB-Cycle の和 k とすると、その組み合わせの数は N=2k となる。本検討では AB-Cycle ごとに緩和個体を生成する方式とした。親 A と親 B はそれぞれ緩和個体を生成し、全体で 2k 個の緩和個体を生成する。2k 個の緩和個体のそれぞれに修正操作を施し、巡回路を 2k 個生成する。その中から最も短い経路を次世代の子として生成する。

3. アーント・コロニー最適化手法（ACO）

アーント・コロニー最適化手法（ACO）は蟻が巣から餌を探して巣へ戻るという蟻の餌採取活動に見られる群知能を模倣した最適化モデルである。

[††] 八戸工業大学大学院　電子電気情報工学専攻
Hachinohe Institute of Technology

4. EAX の現状と問題点

4.1 EAX の現状調査

EAX の現状性能調査　EAX の現状と、問題点を調査するために、TSPLIB にある 198 都市問題(d198)で C プログラム開発実験を行った。実験では、EAX の主な起動パラメータの一つである集団サイズを 50、100、150 と変化させ測定した。

測定結果を図 1 に示す。図 1 の X 軸は集団サイズ、Y 軸は 15 回測定したうち最適解を探索した時の実行時間の平均値と、最適解探索率を示している。図 1 より、集団サイズが 50 のときの探索率は 53%、100 のときの 67%であるが、集団サイズを 150 に増加させると、急激に探索率は 100%に到達することが分かる。また、実行時間は集団サイズにほぼ比例し増加していることが分かる。

以上から、①集団サイズが増えると実行時間が増大すること、②集団サイズの増加に伴い、最適解探索率が向上することが分かった。

4.2 考察

測定結果から、EAX はその集団サイズを大きく設定すれば、最適解を 100%探索できると考えられる。これは、集団サイズにより、集団の多様性を確保しているためと考えられる。計算時間が増加するのは集団サイズを増加させると計算量も増加するためである。

4.3 EAX の問題点

EAX は高い最適解探索性能を持つがそれは集団サイズに大きく依存する。集団サイズが大きければ、集団の多様性が確保されているため、最適解探索率が高くなるが、探索時間については増大する。そのため、集団サイズが小さくても集団の多様性を確保し、尚且つ最適解探索率を低下させない手法が必要である。

5. ACO 突然変異方式による EAX の性能改善

EAX の性能改善に向け、ACO を利用した突然変異手法を提案する。具体的には、EAX で求めた次善解を ACO により突然変異させる。ACO により解の多様性を確保し、EAX で再度、最適解を探索する。収束性については、すでに求められた次善解を多様性が確保された解集合（ACO によって突然変異させた解集合）に混在させることにより、確保する。

ACO 突然変異方式 EAX の最適解探索アルゴリズム（図 2）

（1） EAX（ACO 無効）
- EAX を実行し、解を生成する。最適解が探索できたなら終了する。

（2） ACO（突然変異）
- （1）で生成した次善解を入力し、ACO を実行する。この操作により EAX で生成した次善解を突然変異させる。

（3） Merge（EAX+ACO）
- （1）で生成した次善解（解集合）の中で一番良い物を（2）で突然変異させた解集合に入れる。

（4） EAX（merge 引継ぎ）
- （3）で生成した解を入力とし、EAX を実行する。最適解を得たなら終了する。最適解を得ることができなかった場合は（2）～（4）を繰り返す。

なお、初回のみ（1）で生成した次善解を ACO で突然変異させた解集合に入れている。以降は（4）で生成した次善解を解集合に入れる。

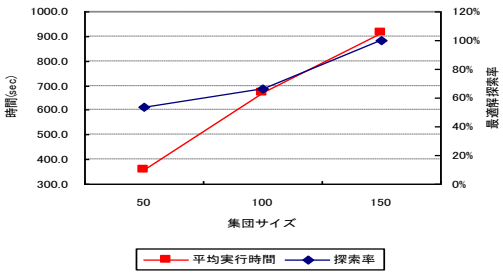

図 1．EAX の実行時間と探索率

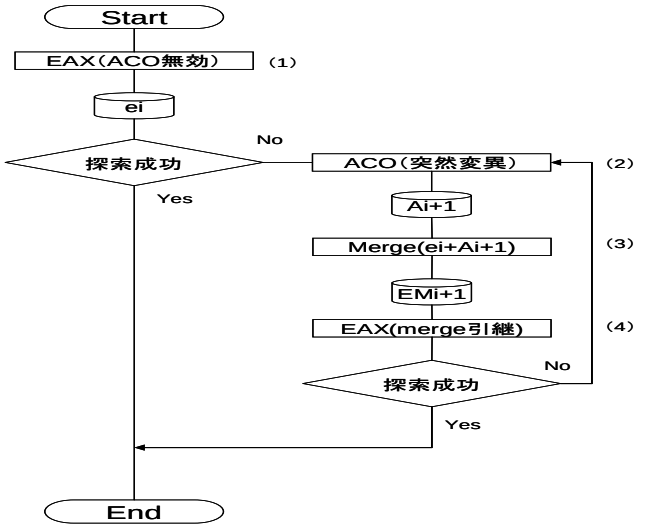

図 2．ACO 突然変異機方式 EAX の最適解探索アルゴリズム

6. ACO 突然変異方式 EAX の性能評価

ACO 突然変異方式 EAX の性能を評価するために、TSPLIB にある 200 都市問題（kroA200、最適解 29368）を利用し、seed-id を 1 から 15 まで変化させた 15 回の独立な実験を行った。

本検討の ACO 突然変異方式 EAX は、集団サイズが小さい場合でも最適解を探索できる事を確認するために、集団サイズを 100 に固定した。その他の EAX の主な起動パラメータは NCROSS=100、親の選択方法は「親 A は全ての現世代の親、親 B はランダムに選択する」である。ACO の主な起動パラメータはフェロモン更新タイミング＝100、揮発率=0.2、距離の重みづけ=2 である。比較対象の EAX 単独の集団サイズは 15 回の試行で最適解を 100%探索できた 400 とした。

単独 EAX が最適解を探索するのにかかった時間と ACO 突然変異方式 EAX が最適解を探索するのにかかった平均時間を比較した。比較結果を図 3 に示す。図 3 に示す通り ACO は EAX の平均実行時間を 57%(=1420/2478)に削減できた。

次に、seed-id=13 の場合の単独 EAX の最適解探索過程と ACO 突然変異方式 EAX の最適解探索過程の比較結果を図 4 に示す。図 4 から、①ACO 突然変異方式 EAX は探索した次善解を ACO の突然変異で拡張し、再度 EAX で探索することによって、1600 秒で最適解（29368）を探索することが出来ていること、②EAX 単独では、29700 しか探索できていないこと、③EAX が最終的に最適解を探索できた時間は 2790 秒と、ACO 突然変異方式 EAX と比べて時間がかかっている事が分かる。

7. まとめ

ACO 突然変異方式 EAX は、集団サイズが小さい場合でも最適解を 100%探索できること、探索時間についても、EAX 単独で最適解を得るのにかかった時間の約 60%で探索できることを d198 ならびに KroA200 で確認した。

参考文献

[1] 髙橋良英：拡張遺伝子交叉オペレータ交代法による巡回セールスマン問題の解法
[2] 永田裕一：巡回セールスマン問題に対する交叉：枝組立交叉の提案と評価
[3] 伊庭斉志：遺伝的アルゴリズムの基礎　-GA の謎を解く-
[4] Marco Dorigo and Thomas Stützle：Ant Colony Optimization
[5] Huai-Kuang Tsai：An Evolutionary Algorithm for Large Traveling Salesman Problems
[6] David E.Goldberg：Genetic Algorithems

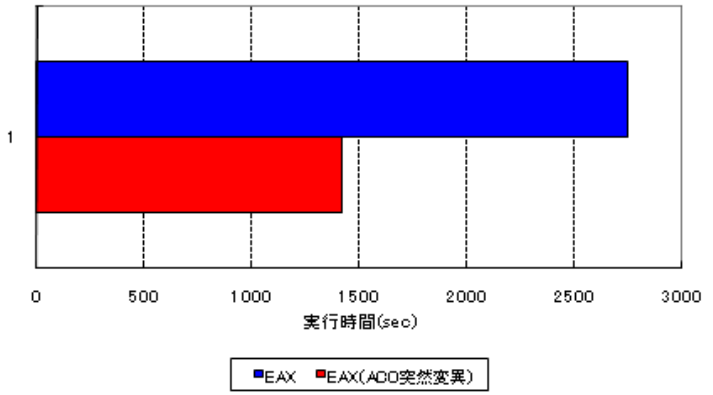

図３．ACO 突然変異方式 EAX と EAX 単独の実行時間の比較

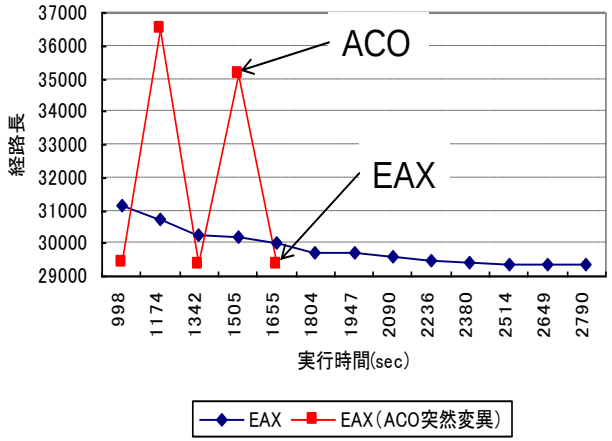

図４．ACO 突然変方式能 EAX と EAX 単独の最適解探索過程の比較