4K-1

GPU を用いた TSP における遺伝的アルゴリズムの高速化

赤崎　康貴†　　志田　晃一郎‡　　安井　浩之‡

東京都市大学大学院工学研究科†　　東京都市大学知識工学部‡

１．はじめに

巡回セールスマン問題（Traveling Salesman Problem, TSP）とは，各都市を 1 度ずつ通る最短経路を求める問題である．考えられる巡回路数は都市数をＮとすると$\frac{(N-1)!}{2}$通り存在する．そのため厳密解を求めると膨大な時間を必要とするため，一般的に近似解を用いて求める．

本研究では近似解を求める手法として遺伝的アルゴリズム（Genetic Algorithm, GA）の枝組み立て交叉（Edge Assembly Crossover, EAX）[1]を用いる．この手法は，10000 都市を超える問題において良質な個体を生成することを可能としているが，多くの個体を繰り返し進化させる手法であるため多くの実行時間を必要としている．そこで，並列処理を得意とするハードウェアである GPU（Graphics Processing Unit）を用いて GA の高速化を図ることを研究の目的とする．

２．EAX

EAX の基本的な手続きを 2.1 に記し，図１は両親から解候補が生成されるまでの手順を図で表したものである．

２．１　EAX algorithm

i. 両親を tour-A, tour-B とする．GAB を tour-A と tour-B を重ね合わせて構成される多重グラフとする．GAB 上で両親の共通する枝は多重枝として扱う．

ii. GAB 上の枝を AB-cycle へ分割する．AB-cycle とは，tour-A と tour-B を交互にたどって得られる閉路とする．

iii. 複数の AB-cycle の中から任意の数の AB-cycle を選択し E-set とする．ただし，一対の多重枝のみからなる AB-cycle は候補から外す．E-set は AB-cycle に含まれる枝の集合とする．

iv. E-set を tour-A に適応し，中間個体を生成する．まず，E-set に含まれる tour-A の枝を tour-A から取り除く．次に，E-set に含まれる tour-B の枝を tour-A に付け加える．これにより，中間個体を生成する．

v. 部分巡回路からなる中間個体を繋ぎ合わせ，完全な巡回路へと修正する．

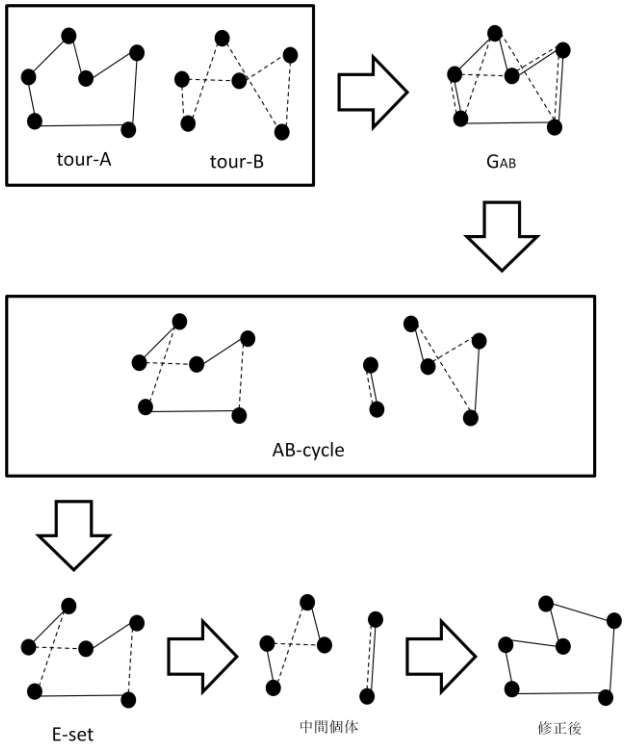

図 1：EAX アルゴリズム

２．２　他手法との比較

EAX と TSP の解法として効率が良いとされている局所探索法の CLK（Chained Lin-Kernighan）との比較を行った．それぞれの実装環境は EAX が 3.06GHz Xeon Processor，CLK が 500MHz Alpha processor（Xeon に比べ 4-5 倍遅い）である．また，表中の err. は最適解からの平均誤差，time は実行時間であり，試行する問題は TSPLIB[2]に記載されている問題を用いることと

Speedup of the genetic algorithm in TSP using GPU.
†Yasuki Akasaki
Graduate School of Engineering, Tokyo City University
‡Koichiro Shida, Hiroyuki Yasui
Faculty of Knowledge Engineering, Tokyo City University

する．

表１より CLK より EAX の方が精度が高く良質な解を生成していることがわかる．しかし，CPU の性能を考慮すると GA の特徴である多くの個体を用いることから EAX の方が数倍以上計算量が多いことが確認できる．

本研究では EAX の実行時間を短縮するため，並列処理を得意とするハードウェアの GPU を用いて高速化を行うことを目的とする．

表１：EAX と CLK の性能比較

	EAX		CLK	
CityNum	err.(%)	time(s)	err.(%)	time(s)
1084	0.0000	36	0.00	27
4461	0.0001	780	0.15	129
11849	0.0004	2728	0.21	599

３．GPU での実装

３．１　EAX の並列処理手法

まず，EAX の処理において CPU で実装する部分と GPU で実装する部分の割り当てを図２に示す．

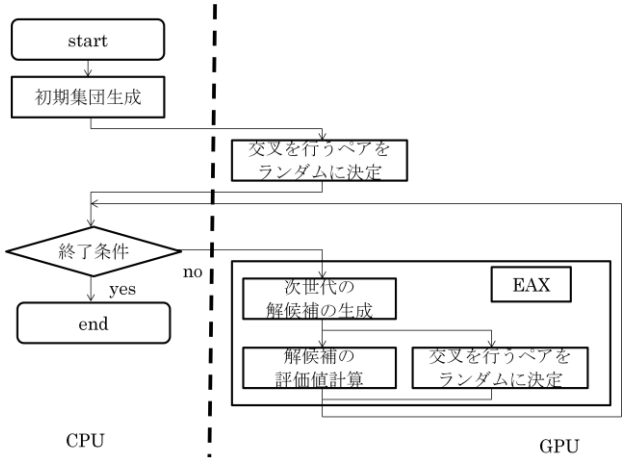

図２：CPU と GPU のそれぞれの処理部分

図２の次世代の解候補の生成を行う処理では親個体数と一対の親で生成する解候補の数をかけた数だけ解候補を生成する処理を行うこととなる．この処理はそれぞれが独立な処理であるため GPU で並列処理を行うことで高速化ができることが期待できる．

３．２　乱数アルゴリズム

EAX では乱数を様々な処理において使用する．しかし，GPU の開発環境である CUDA のライブラリには疑似乱数を生成する関数が存在しない．そこで，本研究では Xorshift[3]によって乱数を生成する．Xorshift は XOR をビットシフトによって生成される乱数のため，高速に生成することができる疑似乱数アルゴリズムである．

４．実験

提案手法を実装し，有効性を検証するため CPU で実装した EAX と提案手法である GPU で実装した EAX の比較を行った．実装環境は CPU が Core i5 760，GPU が Geforce 460 GTX である．比較を行うための問題として TSPLIB より eil101，rat195，rat575 を用いた．なお CPU，GPU 共に E-set を選択するために選択する AB-cycle の数は１とする．

表２：実行結果

	eil101		rat195	
	CPU	GPU	CPU	GPU
平均実行時間	2.802	1.160	9.003	3.453
平均世代数	8.700	8.750	20.000	18.800
最適解を得た数	20	20	19	20
最短距離からの誤差	0.000000	0.000000	0.000108	0.000000
	d493		rat575	
	CPU	GPU	CPU	GPU
平均実行時間	32.415	16.405	67.008	31.755
平均世代数	58.111	59.313	85.300	84.200
最適解を得た数	18	16	20	20
最短距離からの誤差	0.000006	0.000011	0.000000	0.000000

５．まとめ

本研究では TSP における EAX を GPU を用いることで高速化をする手法を提案した．表 2 より CPU が Core i5 760，GPU が Geforce 460 GTX の時において解の精度を損なうことなく 1.9～2.6 倍の処理速度を得ることができた．

今後の課題は３点ある．まずメモリアクセスの最適化である．次に EAX の特徴である大規模 TSP の GPU への実装．最後に他の TSP の近似解法とのハイブリッド処理の GPU への実装である．

参考文献

[1] 永田裕一：局所的な EAX を用いた GA の高速化と TSP への適応．人工知能学会論文誌，Vol.22，No.5，pp.542-552，2007.

[2] TSPLIB：http://www2.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95

[3] G.Marsaglia：Xorshift RNGs. Journal of Statistical Software，Vol.8，Issue 14，pp.1-6，(2003).