3K-4

巡回セールスマン問題における2-opt法のGPGPUによる高速化

福田 悠太 † 大川 猛 † 大津 金光 † 横田 隆史 †
† 宇都宮大学大学院工学研究科情報システム科学専攻

1 はじめに

組み合わせ最適化問題のひとつである巡回セールスマン問題 [1] の近似解を求める際，都市数が多くなると探索空間も膨大となるため，多くの計算時間が必要となる．また，都市数が数万都市以上の規模になると，高速な近似アルゴリズムである 2-opt 法を用いても計算時間が膨大となる．一方，近年は高速な描画処理を得意とする GPU に汎用的な数値計算を行わせる GPGPU が注目されている．GPGPU を活用することにより，CPU を用いる場合よりも大幅な速度向上を実現することができる．本稿では，巡回セールスマン問題の近似アルゴリズムである 2-opt 法の計算を GPU を用いて高速化を図る．

2 巡回セールスマンと 2-opt 法

巡回セールスマン問題とは，複数の都市を 1 回ずつ巡回して出発地点に戻る際の最短経路を求める問題である．巡回セールスマン問題の都市数を N とすると，その巡回路の総数は $\frac{(N-1)!}{2}$ となり，都市数が多い巡回セールスマン問題の最適解を求めることが困難である．このため，ある程度の誤差を許容する近似解を短時間で求めることが重要となる．

巡回セールスマン問題の近似解を高速に求めるための手法として局所探索法がある．局所探索法とは，ある 1 つの巡回路を局所的に変化させて，その巡回路のコストを改善していく手法である．局所探索法で巡回路を変化させる手法のひとつとして，2-opt 法が挙げられる．2-opt 法は，図 1 のように，2 つの辺をつなぎ変えることで別の巡回路を生成する．つなぎ変えることで全体のコストが小さくなる場合，巡回路を更新する．これにより，解を徐々に改善していく．すべての 2 辺の組み合わせを探索して，解の改善ができなくなったところで探索を終了する．

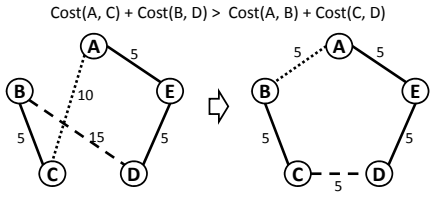

図 1: 2-opt 法

2-opt 法を用いることにより，巡回セールスマン問題の近似解を高速に求めることができる．しかし，数万都市規模の巡回セールスマン問題では，探索空間が膨大となるため，2-opt 法を用いる場合でも探索に時間がかかる．そこで本研究では，探索時間の削減を行うために，GPGPU を活用して探索の高速化を図る．

3 GPGPU による並列探索

本研究で使用する NVIDIA 社の GPU は数百から数千個の CUDA コアを搭載している．このたくさんの CUDA コアは SIMD 動作し，大量のデータを一度に並列処理できる．GPU による並列処理で高い性能を実現するためには，GPU 内の CUDA コアを最大限活かせるように，プログラムを並列化する必要がある．

巡回セールスマン問題では，1 個ずつ 2 辺の組み合わせのコスト計算を行う場合，コストを削減できる巡回路を発見するのに時間がかかることがある．そこで，コストを削減できる巡回路の高速に発見するために，GPU 上で複数のスレッドを実行することで，複数の 2 辺の組み合わせのコストを並列計算する．都市数が N のとき，ある辺 (i_0, i_1) に対して，もうひとつの入れ替え可能な辺 (j_0, j_1) の組み合わせは $N-3$ 通り存在する (隣接する 2 辺を入れ替えても巡回路は変化しないため)．そこで，図 2 のように，$N-3$ 個のスレッドを GPU 上で実行し，各スレッドがそれぞれ異なるパターンの組み合わせで 2 辺のコストを並列計算する．また，並列コスト計算のフローチャートを図 3 に示す．

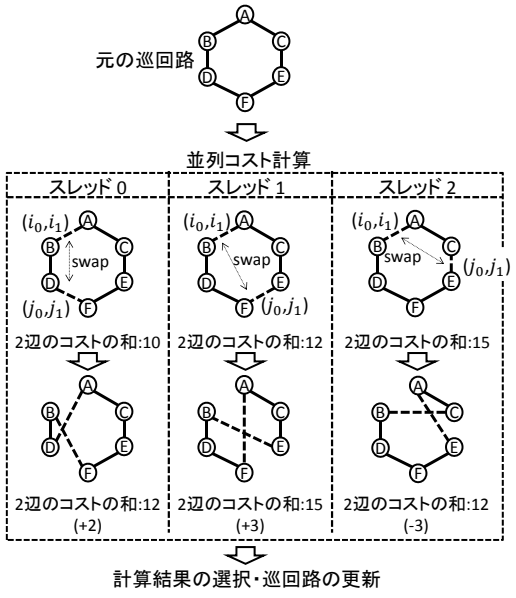

図 2: コスト計算の並列化

Speed-Up of 2-opt Local Search in Traveling Salesman Problem by GPGPU
†Yuta Fukuda, Takeshi Ohkawa, Kanemitsu Ootsu and Takashi Yokota
Department of Information Systems Science, Graduate School of Engineering, Utsunomiya University (†)

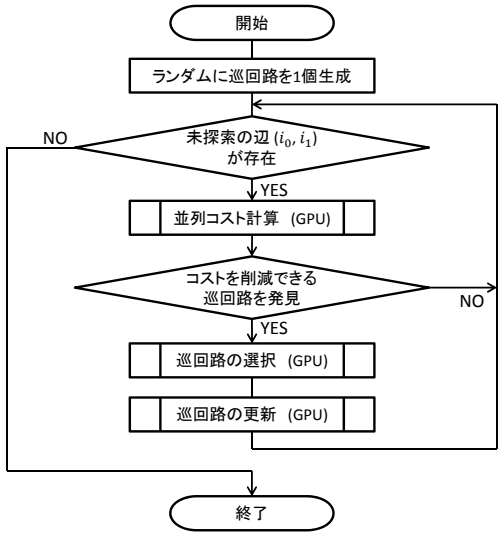

図 3: GPU による並列探索のフローチャート

コストの並列計算によって得られた結果の中にコストを削減できるものが存在する場合，これらの結果の中から 1 個を選択する．このとき，CPU で 1 個ずつコスト計算を行う場合と同一の近似解を得るために，コストを削減できる結果の中から一番最初に発見したものを 1 個選択する．選択した結果をもとに，巡回路を 2 辺を入れ替えた後のものに更新する．一方，コストの並列計算の結果の中にコストを削減できるものが存在しない場合，計算結果の選択や巡回路の更新は行わず，さらに別の異なる 2 辺の組み合わせのセットに対してコストの並列計算を試みる．すべての 2 辺の組み合わせを探索して，コストを削減できなくなったら，探索を終了する．

GPU で並列探索を行う場合 (GPU 探索)，コストの並列計算によって，削減できる巡回路が発見しやすくなる．しかし，CPU で 1 個ずつ探索を行う場合 (CPU 探索) に，コストを削減できる計算結果がすぐに見つかることがある．この場合，CPU 探索によるコスト計算にかかる時間は，GPU 探索による並列コスト計算にかかる時間よりも短くなる．ランダムに生成された初期解から探索を開始するときの巡回路の更新速度は，探索序盤では CPU 探索，探索終盤では GPU 探索のほうが速くなる．そこで，探索開始時は CPU 探索を行い，GPU 探索での更新速度が CPU 探索での更新速度よりも速くなったときに，GPU 探索へ切り替える．切り替えポイントを決めるために，まず CPU 探索および GPU 探索における巡回路を 1 回更新するまでにかかる時間を求める．CPU 探索で巡回路を 1000 回ずつ更新したとき，CPU 探索が GPU 探索よりも更新時間が短くなるパターンの発生確率が，ある一定の閾値を下回ったときに CPU 探索から GPU 探索に切り替える．今回は閾値を 60%に設定して，CPU 探索と GPU 探索を切り替えたときの探索時間を計測した．

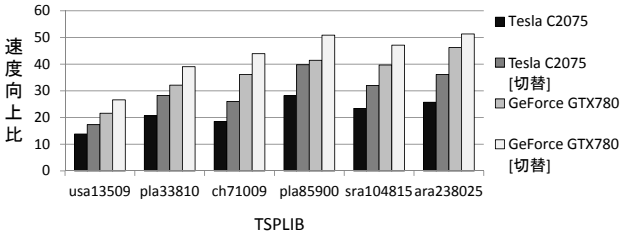

図 4: GPU による並列探索の性能

4 GPU による並列処理の性能

今回提案した手法の性能を調べるために，CPU 探索，GPU 探索，および CPU 探索と GPU 探索を切り替えた場合の探索時間の計測を行った．GPU 探索では，Tesla C2075 (1.15GHz, 448CUDA コア) および GeForce GTX780 (0.86GHz, 2306CUDA コア) の 2 種類の GPU を使用した．そして，Xeon W3565 (3.20GHz, 1 コアのみ使用) の CPU を用いて CPU 探索をした場合との性能を比較した．入力用の都市データには，ベンチマーク問題集 (TSPLIB) [2] から 6 個の問題を使用した．また，開発環境として CUDA[3] を用いた．CPU 探索に対する GPU 探索および探索切り替え時の速度向上比を図 4 に示す．図 4 より，都市数が多いほど速度向上比も高くなり，Tesla C2075 では最大 40 倍，GeForce GTX780 では最大 50 倍程度の速度向上を達成したことが分かる．また，CPU 探索を途中から GPU 探索に切り替えることにより，探索開始時から GPU 探索を行う場合よりも 1.3 倍程度の高速化を達成したことが分かる．

5 おわりに

本研究では，数万都市規模の巡回セールスマン問題の近似解を GPGPU によって高速に求めるための手法を提案し，実装を行った．この結果，GeForce GTX780 の GPU を活用することで，最大 50 倍の速度向上を達成した．GPU を用いて 2-opt 法におけるコスト計算を並列に行う際，現状ではチップ外の低速なメモリへのアクセスが多くなっている．そのため，今後は GPU によるコスト計算の最適化を行うことで，さらなる高速化を目指していく．

謝辞

本研究は，一部日本学術振興会科学研究費補助金 (基盤研究 (C)24500055，同 (C)24500054, 同 (C)25330055, 若手研究 (B)25730026) の援助による．

参考文献

[1] 山本芳嗣, 久保幹雄, “巡回セールスマン問題への招待,” 朝倉書店, 1997.

[2] G.Reinelt, “TSPLIB - A Traveling Salesman Problem Library,” ORSA Journal on Computing, Vol.3, No.4, pp.376-384, 1991.

[3] NVIDIA, “NVIDIA CUDA C Programming Guide Version 5.5,” 2013.