

SIMD 2-opt 法の GPGPU への適応と評価

坂本 亮[†] 小西 克巳[‡]工学院大学大学院工学研究科情報学専攻[†]工学院大学情報学部コンピュータ科学科[‡]

1. はじめに

巡回セールスマン問題[1](Traveling Salesman Problem)は、都市の集合とそれらの間の移動コストが与えられた時に、すべての都市をちょうど一巡して出発地に戻る移動距離を求める問題である。この問題は NP 困難問題の一つであると知られており、様々なアプローチでの解法が研究されている。また、近年 CPU に変わる演算資源として、GPGPU を始めとするアクセラレータが注目されているが、局所探索に GPGPU を使用したもの、また大規模問題に対して適用したものは、まだ数がないのが現状である。本稿では、巡回セールスマン問題のヒューリスティックアルゴリズムである 2-opt 法に SIMD を組み込んだ SIMD 2-opt 法を提案し、それを GPGPU へ適応し、評価を行う。

2. OpenCL

OpenCL とは、ヘテロジニアス環境を対象とした並列化フレームワークである。現在、計算資源として注目されている GPU は、分野によっては従来の CPU より高速な処理が可能ではあるものの、CPU と同様のプログラムでその性能を引き出すことは困難であり、GPU 向けにプログラムの書き換えと最適化が必要不可欠である。しかしながら、GPU のプログラム動作環境は、各ベンダの独自仕様が非常に強く、あるプラットフォームで動作するプログラムは、異なるプラットフォームで動かすためにまた書き換えが必要になるという問題があった。OpenCL はそういった問題を解決するために提唱されたものである。この規格に則って作成されたプログラムは、ほぼプログラムの書き換えなしに他のプラットフォームでの動作を可能とする。OpenCL は制御を担当する CPU 側のホスト部と、実際に演算を実行するデバイス部に分けられており、ホスト部は C/C++ で、デバイス部は OpenCL C 言語と呼ばれる C 言語を拡張した言語でプログラムを記述する。デバイス側で実行されるプログラムはカーネルと呼ばれる纏まりで呼称され、このカーネル内では、各デバイスが有するハードウェア SIMD 機構などを記述し、使用することが可能である。本稿では、この OpenCL を用いて GPGPU 上に SIMD 2-opt 法を実装し、評価を行った。

3. 問題設定

N 都市が与えられた時の巡回セールスマン問題の場合を考える。 V をグラフの点集合とし、 C_{ij} を枝 (i, j) の重み、 x_{ij} を枝 (i, j) が解に含まれるとき 1、そうでないとき 0 の値を取る 0-1 変数とするとき、

次式のように定式化される。

式 1: 巡回セールスマン問題の定式化

$$\begin{aligned} \min & \sum_{i \in V} \sum_{j \in V} C_{ij} x_{ij} \\ \text{subject to} & \sum_{j=1}^N x_{ij} = 1 \quad (\forall i \in V) \\ & \sum_{i=1}^N x_{ij} = 1 \quad (\forall j \in V) \\ & \sum_{i \in S} \sum_{j \in \bar{S}} x_{ij} \geq 1 \quad (\forall S \subset V, S \neq \emptyset) \\ & x_{ij} \in \{0, 1\} \quad (\forall i, j \in V) \end{aligned}$$

ただし、 V は都市を表すノードの集合である。巡回セールスマン問題は問題のサイズが大きくなるにつれ、組み合わせ数が爆発的に増加する。 n 都市の巡回回路数は、 $(n-1)!/2$ 通りとなる。このような組み合わせ爆発が発生することより、すべての解を列挙して最適な解を得ることは計算量的に不可能となる。そのため、厳密解を得られることは保証されていないものの、実用的な時間で解を得るための近似解法が多数研究されている。

5. 提案手法

5.1. SIMD 2-opt 法

本稿では、近似解法アルゴリズムの一つである 2-opt 法を使用する。2-opt 法は解の精度は保証されないが高速であるという特徴を持つ。以下に 2-opt 法を示す。

Step1. 適当な巡回回路を作成する。

Step2. 作成された巡回回路の中から、任意の 2 本の枝を選択する。

Step3. 選択された枝 $(i, j), (k, l)$ において、移動コスト C が

$$C_{i,k} + C_{j,l} < C_{i,j} + C_{k,l}$$

となるような場合、枝を入れ替える。

Step4. そのような枝がなくなるまで Step2,3 を繰り返す

2-opt 法は改善しうる枝が発見された場合に移動を行うが、この際の移動戦略として、即時移動戦略と最良移動戦略の二つが存在する。本稿では、局所探索範囲内でもっとも良い改善枝を選択して移動する最良移動戦略を採用している。以上は 2-opt 法のアルゴリズムであるが、本稿ではこのアルゴリズムの改良手段として、SIMD 2-opt 法を提案する。SIMD とは複数のデータに対して同一の処理を行うという演算方式である。2-opt 法では前述の通り、4 都市間の距離の演算が主な計算である。この演算を SIMD アーキテクチャである GPU にオフロードすることで高速化を図る。2-opt 法は高速な結果が得られるが、問題規模が増加するにつれて計算時間が指数的に増加するという特徴がある。結果として、大規模問題への適用は困難となってしまふ。本稿で

は以下の Karp の分割法を適用することにより、問題規模の分割を行う。

5.2. 領域分割

本稿では、問題分割の手法として、Karp の分割法[2]を使用する。Karp の分割法は、R.M.Karp により提案された分割統治手法である。以下にアルゴリズムを示す。

- Step1. 各都市をバケットに分割する
- Step2. バケットに分割された各都市間でバケットごとの最短巡回路を求める。
- Step3. 各バケットから代表都市を算出し、代表都市間の巡回路を作成する。
- Step4. バケットごとの巡回路と代表都市間の巡回路を合わせてオイラー閉路を作成する。
- Step5. 適当な都市から出発し、重複した経路を近道することにより全巡回路を得る。

Karp の分割法は、正方領域内に均一に都市が分布している場合、よい結果が得られる事が知られている。しかしながら、現実の問題においては、均一に分布している問題よりも分布の偏りが発生している場合が多い。そのため、偏りを均一にするよう動的に分割点の移動を行うようアルゴリズムを変更した。

6. 数値実験と評価

実行環境を以下に示す。CPU,GPU 共に領域分割数は 8 とする。また、CPU においては 8 スレッドによる並列化、SIMD 化の最適化を施した。

CPU: Intel Core i7 2600K(3.4Ghz * 4)

GPU: NVIDIA GeForce GTX580

また、実験においては The Traveling Salesman Problem[3]にて公開されている問題を使用した。

6.1. TSP Art への適用

TSP Art とは、絵画を巡回セールスマン問題の経路で表現したものである。以下のような画像が生成される。各問題における最短巡回路長は既知である。本実験の処理時間は表のようになった。また、最短巡回路長との誤差は表のようになった。他の都市数であっても誤差に変化は見られないため、10 万都市の場合のみを記載する。GPU では、10 万都市の時に CPU の約 1.8 倍の高速化を得られている。また、最短巡回路長に対し約 6%の誤差の解を得ている。

図 1:TSP Art の例



表 1: TSP Art の処理時間

	CPU	GPU
10 万都市	229.503	113.944
12 万都市	348.389	169.758
14 万都市	422.273	265.870
16 万都市	635.624	328.4
18 万都市	802.924	404.607
20 万都市	995.756	484.102

表 2: 10 万都市における誤差

	CPU	GPU
精度	106%	106%

6.2. VLSI TSP

VLSI TSP とは、現実の問題に近い、VLSI の基盤データセットである。この時の処理時間は表のようになった。また、最短巡回路長との誤差は表のようになった。本実験では、GPU は CPU に対し、最大で約 4 倍の高速な結果を得られている。この実験においては最短巡回路長との誤差が非常に大きくなっているが、それは問題における都市の分布の形状が起因していると考えられる。

図 2: VLSI TSP の例

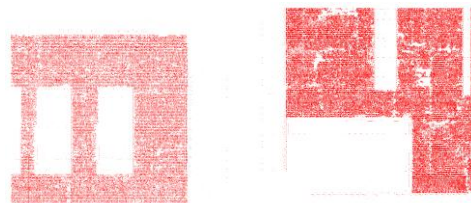


表 3:VLSI TSP の処理時間

	CPU	GPU
Sra104815	315.831	156.618
Ara238025	2431.035	758.4
Lra498378	15412.512	3849.535

表 4: sra104815 における誤差

	CPU	GPU
精度	119%	119%

7. まとめ

本稿では、SIMD 2-opt 法を GPGPU 上で OpenCL を用いて実装し、その性能を評価した。処理時間としては現状の CPU よりも高速な結果を得られているが、本手法では、一部の問題を解く際に最短巡回路長との誤差が大きくなってしまふ問題がある。これは分布の形状に起因すると考えられる。こういった問題に対応するために領域分割手法を改善することは今後の課題である。

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] R.M.Karp, Math Oper, “Probability analysis of partitioning algorithms for the traveling-salesman problem in the plane”, Mathematics of Operations Research, vol. 2. no. 3, pp. 209-224, 1977

[3] The Traveling Salesman Problem, <http://www.tsp.gatech.edu/>