

マルチ GPU 環境における 機械学習ハイパーパラメータの自動チューニング(2)

藤家空太郎[†] 多部田敏樹[†] 藤井昭宏[†] 田中輝雄[†] 加藤由花^{††} 大島聡史^{†††} 片桐孝洋^{†††}

[†]工学院大学 ^{††}東京女子大学 ^{†††}名古屋大学

1 はじめに

自動チューニングとはコンピュータの環境やプログラムに合わせて最適な性能パラメータを自動で探索する技術である [1]. 我々は反復次元探索を用いた自動チューニングの研究を進めている [2].

現在, ノード内外の複数の GPU を利用できるマルチ GPU 環境において, 機械学習プログラムのハイパーパラメータの最適化をする研究に取り組んでいる [3]. チューニング対象とする機械学習プログラムは同一のハイパーパラメータを用いても毎回教師データが変わることから同一の結果にならないため, 自動チューニングの結果に揺らぎがある. この揺らぎに対して, これまで我々は推定したパラメータに対して追加測定を行い, 自動チューニングの安定性を高める手法を提案してきた [2].

反復次元探索によるパラメータの推定をフェーズ 1 (Ph.1), 本報告での追加測定をフェーズ 2 (Ph.2) とする. 本研究の目的は, Ph.2 をマルチ GPU 環境で並列化し, 自動チューニング実行時間を短縮すること, および Ph.2 を機械学習プログラムに適用し, 自動チューニングが結果に揺らぎのあるアプリケーションに効果があるかを検証することである.

2 反復 1 次元探索によるパラメータ推定

反復 1 次元探索とは, パラメータ空間で方向探索と 1 次元探索を繰り返す手法である. パラメータ空間は複数のパラメータを軸として, 各パラメータの取り得る値 (離散点) からなる. 方向探索は初めに一つの基準点を決め, 基準点の各方向隣の点を調べ, 最も性能の良い点を見つける. 1 次元探索では方向探索で見つけた点の方向にある点を調べ, 最も性能の良い点を新たな基準点とする. この 2 つの探索を基準点の移動がなくなるまで反復する. この手法は各点を一度しか測定しないため, 確実に安定した値を取得している保証はないが, 実行結果が悪くなる点は調べずにできるだけ多くの点を調べている.

Auto-Tuning of Hyperparameters of Machine Learning on Multi GPU Environment (2)

Sorataro Fujika[†], Toshiki Tabet[†], Akihiro Fujii[†], Teruo Tanaka[†], Yuka Kato^{††}, Satoshi Ohshima^{†††}, Takahiro Katagiri^{†††}

[†]Kogakuin University, ^{††}Tokyo Woman's Christian University, ^{†††}Nagoya University

3 追加測定による自動チューニングの効果検証

3.1 Ph.2 のアルゴリズム

自動チューニングにおける Ph.1 と Ph.2 の位置付けを図 1 に示す. Ph.1 はマルチ GPU 環境で全方向の探索を並列化した [3]. 一方 Ph.2 は Ph.1 の結果をもとに同じマルチ GPU 環境での並列実行する.

図 2 に Ph.2 の追加測定の手法を例示する. 縦軸は性能, 横軸は実行回数を示す. Pa1~Pa4 はそれぞれ Ph.1 で推定した最良から 4 番目までのパラメータとする. まずステップ 1 で最も性能の良かった Pa1 を並列に複数回同時に測定する (今回は 10 回). 測定結果から最大, 最小の値を除外して平均を求めることで極端なデータを削除して安定した値を取得する. Pa1 の値を求めた平均に更新する. Pa1 の値が大きくなり Pa2 が最も性能の良い点となる. 次にステップ 2 で最も性能の良くなった Pa2 を測定して値を更新する. 以上を繰り返し, 最も性能の良い点が変わらなくなるまで値を更新する. ステップ 4 では再び Pa2 を測定している. ステップ 5 で値の更新後も Pa2 が変わらず最も性能の良い点であることを判断し, 終了する.

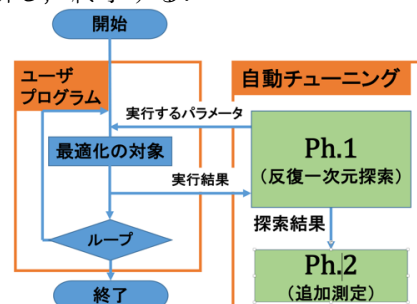


図1 自動チューニングにおける
Ph.1 と Ph.2 の位置付け

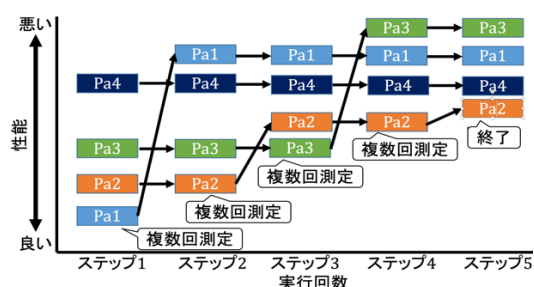


図2 Ph.2 の追加測定例

3.2 今回の評価対象

本研究では加藤らのグループが作成した歩行者経路予測アプリケーションの機械学習プログラム[4]に自動チューニングを適用した。このプログラムは予測した到達地点と実際の到達地点の差である `final_error` [m] で精度をはかる。`final_error` の値が小さくなるほど性能が良いことになる。

3.3 測定結果

計算環境は名古屋大学の「不老-Type II サブシステム」を用いた [5]。表 1 に並列実行したことによる実行時間の変化とその比率を示す。今回は Ph.2 で並列に 10 回同時に測定を行った。自動チューニング全体の実行時間は、逐次実行の場合 Ph.1 に加えて Ph.2 で 32%増加した。一方、並列実行の場合は Ph.1 に加えて Ph.2 で 72%増加した。これは Ph.1 が並列化で実行時間が 22.6 倍に減少したのに対し、Ph.2 は 10 回の同時測定を並列化したため実行時間の減少が 10%にとどまり、並列効率が悪いからである。

表 1. 各フェーズの逐次と並列の実行時間の変化

	Ph.1	Ph.2 (Ph.1 からの増分)
逐次	11.6 日	3.7 日 (32%)
並列	12.3 時間	8.9 時間 (72%)

Ph.2 では Ph.1 で推定した点を性能の良い点から順に調べていく。今回は 16 点となった。図 3 は Ph.2 の結果、性能の良かった上位 10 点について、Ph.1 と Ph.2 の `final_error` の比較である。縦軸は `final_error`、横軸は Ph.1 の探索において何番目に探索した点かを示す探索履歴の番号である。ここで Ph.1 は測定回数 1 回の結果、Ph.2 は最大、最小を除いた測定結果の平均である。Ph.2 は平均して安定した値を求めているため、Ph.1 に比べ全体的に `final_error` が大きい。しかしこれら 10 個の点は Ph.1 で推定した性能の良かった上位 14 個までの点から選ばれており Ph.1 と Ph.2 で推定した点は大きく変わらなかった。

図 4 に図 3 で示した各点と初期点について Ph.2 の測定結果の分布を示す。初期点は機械学習プログラムで初めに設定されていたパラメータの組み合わせである。初期点に対して推定した点は `final_error` が 0.84~0.91 近辺と優位な位置で安定している。

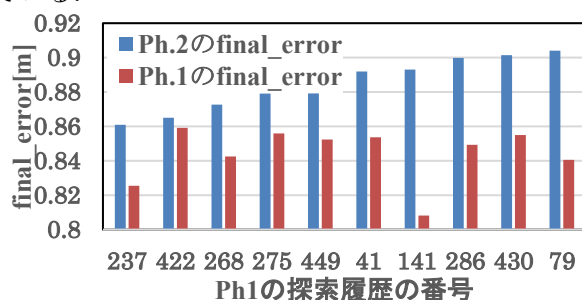


図 3 Ph.1 と Ph.2 での性能変化

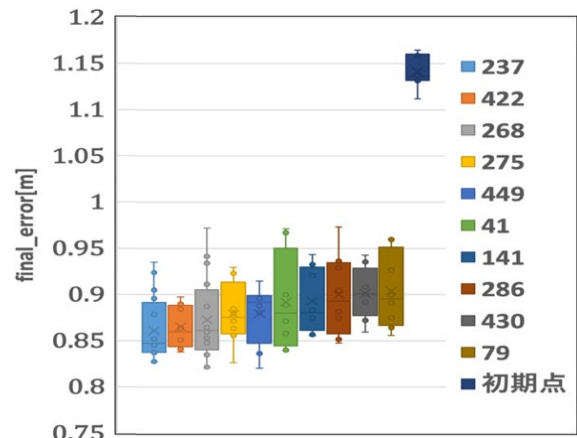


図 4 Ph.2 の `final_error` の分布

4 おわりに

本研究では追加測定の並列化、追加測定を機械学習プログラムに適用し自動チューニングの機械学習プログラムへの効果の検証を行った。結果から 1 回しか測定していない反復次元探索でも、性能に揺らぎのあるプログラムに対して安定して良いパラメータを推定できることがわかった。

Ph.2 は Ph.1 比べて並列化による実行時間の減少が少なかった。今後の課題として Ph.2 の並列性をあげること、有効な並列数を検定し求めることが挙げられる。

謝辞

本研究の一部は、JSPS 科研費 JP18K19782, JP18K11340 の助成および名古屋大学 HPC 計算科学連携研究プロジェクトの支援による。

参考文献

- [1] 片桐孝洋, 「ソフトウェア自動チューニング - 数値計算ソフトウェアへの適用とその可能性 -」, 慧文社, 2004 年 12 月 3 日 初版第一刷発行, ISBN4-905849-18-7.
- [2] 関直人, 性能パラメータ推定における実行時間の揺らぎに頑健な自動チューニング手法, 2019 年度工学院大学大学院修士学位论文(2020).
- [3] 多田敏樹, 藤家空太郎, 藤井昭宏, 田中輝雄, 加藤由花, 大島聡史, 片桐孝洋, マルチ GPU 環境における機械学習ハイパーパラメータの自動チューニング(1), 第 83 回全国大会講演文集 (2021).
- [4] Rina Akabane, Yuka Kato, "Pedestrian Trajectory Prediction Using Pre-trained Machine Learning Model for Human-Following Mobile Robot", IEEE International Conference on Big Data Workshop (IoTDA 2020), pp.3453-3458, (2020).
- [5] 名古屋大学情報連携推進本部. "スーパーコンピュータ「不老」紹介", <http://www.icts.nagoya-u.ac.jp/ja/sc/overview.html>, (最終アクセス日時 2021 年 1 月 4 日).