

はじめに

自動チューニング
とは？

逐次実験計画につ
いて

性能評価方法

まとめ

オンライン自動チューニングのための Bayes 統計に基づく逐次実験計画法

武藤 克弥

富山県立大学 電子・情報工学科

October 23, 2020

背景

ソフトウェア開発ではパラメーター決定のために実験を行うことが必要になることがある。しかし、実験結果は実験の条件に依存するため、実験条件が適切でないと、不適切な決定を下しかねない。このため、適切な実験条件を提示し、それに基づいた決定を可能にするソフトウェア自動チューニング (以下自動チューニングとする) を用いる。しかし、測定のばらつきに影響され、あるモデルから最適のパラメータを推定するといった場合にもばらつきが発生するという問題があった。

目的

Bayes 統計に基づく自動チューニングのアルゴリズムを提案し、誤差・ばらつきを含むモデルあるいは観測において、低コストかつ安定的にデータを得られることを示す。

自動チューニングとは？

3/16

プログラムの実装

同じ処理を実行するソースコードは複数あることが多い
→どれを選ぶかによって性能が変わってくる.

チューニングとは

- (1) 同じ処理をするソースコードをいくつか用意して、それぞれの性能を試行実験で評価
- (2) 最適なものを選ぶ
→これを自動化する (チューニング自体もプログラムで記述)

はじめに

自動チューニング
とは？

逐次実験計画につ
いて

性能評価方法

まとめ

逐次実験計画

得られた実験データに応じて、それ以降の実験内容を決める実験計画のこと。

少ない実験回数で情報を効率的に集められる利点があり，Bayes 統計が良く用いられる。

Bayes 統計と自動チューニング

Bayes 統計では統計分布の変数 (正規分布だと平均と分散) の値が分からなくても，ある確率分布に従うと表現する．これを事前分布と呼ぶ．その事前分布と観測値を Bayes の定理で結合し事後分布を得る．

本研究では性能予測モデルを事前分布とし，実際の観測値と合わせることで事後分布を得る．それに基づいて自動チューニングアルゴリズムを設計する．

はじめに

自動チューニング
とは？

逐次実験計画につ
いて

性能評価方法

まとめ

プログラムで実装方法が複数存在するライブラリ関数を呼び出すことを考える.

→最適な実装を選んで呼び出す

このとき、どの実装が最も短い時間で呼び出せるかは分からないため、ある反復呼び出しの一部を実行時間の性能評価として用いる.

最適な逐次実験計画 (1)

6/16

はじめに

自動チューニング
とは？

逐次実験計画につ
いて

性能評価方法

まとめ

まず、逐次実験計画の最適解は最後の呼び出しから順に求めていく。

$\mu_i^{(k)}$... 最後から k 回目の呼び出し直前の事前分布における
実装 i の所要時間の期待値

$\mu_i^{(1)}$... 最後に実装 i 呼び出す場合の所要時間の期待値
→ $\min_i \{\mu_i^{(1)}\}$ となる i を選ぶ。

最後から 2 番目を呼び出すまでにかかる時間を $\omega_i^{(2)}$ とした場合、

$$\omega_i^{(2)} = \mu_i^{(2)} + E(\min_j \{\mu_j^{(1)}\})$$

($E(x)$: x の期待値) となる。

→ 最適な i を選んだ状態で、次に j を選ぶことで最適となることを意味する

最適な逐次実験計画 (2)

7/16

はじめに

自動チューニング
とは？

逐次実験計画につ
いて

性能評価方法

まとめ

同様に最後から k 回目の呼び出しの場合,

$$\omega_i^{(k)} = \mu_i^{(k)} + E(\min_j \{\mu_j^{(k-1)}\}) \quad (1)$$

で i を最後から k 回目を選ぶことで最適となる。しかし、これを精度よく計算するのは困難。

→近似式を提案

近似式

$$\omega_i^{(k)} = \mu_i^{(k)} + (k-1)E(\min_j \{\mu_j^{(k-1)}, \mu_{min'}^{(k)}\}) \quad (2)$$

これは最後から $k-1$ 回の呼び出しまでは全て同じ実装 (実行時間が最短なもの) を選んでいる。

($k-1$ で最短になる実装を選んで ($k-1$) 回分けておく)

実験の例題

$N \times N$ の正方行列の積 $A = BC$ を計算するプログラム

各行列サイズ N に対して 576 通りのサブルーチンを 1000 回呼び出す.

はじめに

自動チューニング
とは？

逐次実験計画につ
いて

性能評価方法

まとめ

実験に用いる事前分布

事前サンプリングによる統計的推定で事前分布を構成.

分散 σ^2 ... 行列サイズ N ごとに $\log \sigma^2$ の平均をとって $\log N$ に関する区分線形関数で近似.

ν_0 (分散未知モデルで必要となる標本分散) ... N ごとにモーメント法で求め, それぞれで $\log(\nu_0 - 4)$ の平均から得られる値を共通で使用.

平均の期待値 μ_0 ... 推定値を使用.

モデルの重み κ_0 ... モデルの二乗誤差の相乗平均 τ^2 から $\kappa_0 = \sigma^2 / \tau^2$ を求め, $\log \kappa_0$ を $\log N$ に関する区分線形関数で近似.

比較対象 —4 つの手法を比較する—

手法 M_1 : 最初の 576 回で全ての実装を 1 回ずつ実行. その後, 平均所要時間 (その実装のすべての実行に関する平均) が最小のものを選択.

手法 M_2 : 最初の 7 回でモデル構築した後, そのモデルが最短時間と推定するものを選択.

手法 M_{3A} および: 最初の 7 回でモデル構築した後, (3) 式の手法で選択. (分散既知の正規分布モデル)

手法 M_{3B} : M_{3A} と同様. (分散未知の正規分布モデル)

用いた指標 1

相対リグレット rr

$$rr = (T_M - T_{opt}) / T_{opt}$$

T_M : 手法 M による合計所要時間

T_{opt} : 最速ルーチンの平均所要時間に呼び出し回数をかけたもの。

用いた指標 2

相対ロス rl

$$rl = (t_M - t_{opt}) / t_{opt}$$

t_M : 手法 M がもっとも多く選択したルーチンの平均所要時間

t_{opt} : 最速ルーチンの平均所要時間。

実験結果 (1)

12/16

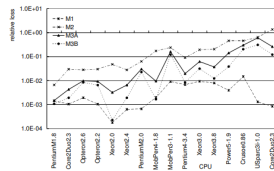
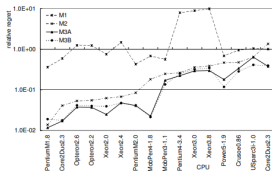


図 1: 各手法の相対リグレットの評価 (呼び出し 1000 回)

はじめに

自動チューニング
とは？

逐次実験計画につ
いて

性能評価方法

まとめ

はじめに

自動チューニング
とは？

逐次実験計画につ
いて

性能評価方法

まとめ

指標	呼び出し回数	M_1	M_2	M_{3A}	M_{3B}
rr	100	—	1.405	1.11	1.11
	1000	1.006	1.358	0.40	0.38
	10000	0.101	1.353	0.19	0.13
rl	100	—	1.352	0.83	0.80
	1000	0.0008	1.353	0.26	0.12
	10000	0.0005	1.353	0.17	0.06

図 2: 呼び出し回数を変えた時の評価 (Core2Duo 2.3Ghz)

指標	呼び出し回数	M_1	M_2	M_{3A}	M_{3B}
rr	100	—	1.8	1.8	1.8
	1000	7.96	0.26	0.253	0.22
	10000	0.80	0.11	0.04	0.05
rl	100	—	0.09	0.04	0.03
	1000	0.007	0.09	0.02	0.01
	10000	0.006	0.09	0.01	0.002

図 3: 呼び出し回数を変えた時の評価 (Pentium4 3.4Ghz)

マシン	M_1	M_2	M_{3A}	M_{3B}
Core2Duo	1.67e-6	1.12e-4	4.69e-4	1.90e-3
Pentium4	2.02e-6	1.36e-4	5.73e-4	2.47e-3

図 4: 各制御手法の 1 呼び出しあたりの計算時間

はじめに

自動チューニング
とは？

逐次実験計画につ
いて

性能評価方法

まとめ

実験結果 (4)

15/16

比較	M_2 vs M_{3A}		M_2 vs M_{3B}	
呼び出し回数	1000	10000	1000	10000
Core2Duo	65	60	110	100
Pentium4	320	145	310	270

図 5: M_{3A}/M_{3B} が M_2 よりも有利になる行列サイズ

はじめに

自動チューニング
とは？

逐次実験計画につ
いて

性能評価方法

まとめ

はじめに

自動チューニング
とは？

逐次実験計画につ
いて

性能評価方法

まとめ

本研究では自動チューニングに Bayes 統計を組み込んだオンライン自動チューニングのための逐次実験計画法を提案し、提案手法の有効性を示した。

今後の課題として事前分布の構成方法の改善や、今回のような離散的ではなく連続的な変数の最適設定などにも取り組んでいきたい。