1J-02

ppOpen-AT で自動生成される混合精度演算プログラムの性能評価

梛尾駿太[1]　満田晴紀[2]　片桐孝洋[3]　大島聡史[3]　永井亨[3]
名古屋大学　情報学部　コンピュータ科学科[1]
名古屋大学　大学院情報学研究科[2]
名古屋大学　情報基盤センター[3]

1. はじめに

ポストエクサスケールコンピューティングに向けて計算機が多様化している．現在注目すべき計算手法の１つとして，混合精度演算の活用がある．混合精度演算は低精度演算による演算高速化が期待できる反面，演算誤差の拡大があり，速度と演算精度の観点からの性能評価が必須となる．また任意のプログラムに混合精度演算を適用する場合，適用可能箇所を探す作業コストが大きい．そこで本研究では，混合精度演算を任意のプログラムに適用する場合においてソフトウェア自動チューニング（AT）技術を活用し作業コストの削減を狙う．加えて，既存の AT 言語である ppOpen-AT が自動生成すると想定するプログラムについて，混合精度演算の観点から性能評価を行う．

2. ppOpen-AT

2.1 概要

ppOpen-AT は数値計算ミドルウェア ppOpen-HPC においてプログラムに AT 機能を提供する目的で開発された AT 言語である．FIBER（Framework of Install-time, Before Execute-time, and Run-time auto-tuning）[1] 方式による AT 言語である．FIBER とは，汎用的な AT 機能の付加を支援する AT フレームワークのことで，以下の 3 つのタイミングで AT を行う：インストール時，実行起動前，実行時．

ppOpen-AT では，ソフトウェア開発者はプログラム上で AT を適用したい箇所に ppOpen-AT のディレクティブを挿入する．プログラムに ppOpen-AT 専用のプリプロセッサを適用して，ソフトウェア開発者がディレクティブで指定した箇所の情報を基に以下のコードを自動生成する：

（１）複数の最適化候補コード

（２）最適コードを探索する AT 機能を含むプログラムコード

以上の仕組みにより AT 機能に関するコードを自動生成するため，ソフトウェアに ppOpen-AT による AT 機能が低いコストで付加できる．

2.2 ppOpen-AT における演算精度を考慮した実行時間最適化

我々は論文[2]において ppOpen-AT を用いて演算精度の最適化手法を提案した．

本手法では，①変数／配列，②ブロック，③関数／サブルーチンを最適化対象とする．これらをディレクティブで指定し，2.1 節の仕組みで複数の最適化候補のコードを自動生成して，システム内部で最適なコード選択を性能パラメタ化する．この際，性能パラメタの精度を変更することによって，演算精度・実行時間・消費電力量の観点から対象ソフトウェアが最適になるような AT を行う仕組みである．

（１）ソフトウェア開発者，およびエンドユーザは，事前に基準となる演算に対する相対的な演算精度劣化の許容値（許容相対誤差）を指定する．

（２）性能パラメタとして①変数／配列，②ブロック，③関数／サブルーチンの 3 種類を指定する．ユーザは混合精度演算を適用したい箇所に，演算精度の変更対象の①～③に対応する，3 種類のディレクティブのうち，適するディレクティブを挿入する．

（３）ppOpen-AT のプリプロセッサを適用すると，挿入したディレクティブに従って候補コードが生成される．

ここで注意は，（３）の候補コード生成時に，対象となるコード領域（AT 領域）の前後で，精度変換に伴う代入のコードが自動生成される．例えば，倍精度（DP）が元のプログラムとすると，単精度（SP）との混合精度演算を行う際には，AT 領域に入る前に，DP から SP への配列等への代入，および，AT 領域から出た後に SP から DP への代入のコードが自動生成される．そのため，オリジナルコードに対して，これらの代入

Performance Evaluation of Auto-generated Programs for Mixed-precision Computations by ppOpen-AT
1 Shunta Nagio, Computer Science, School of Informatics, Nagoya University
2 Haruki Mitsuda, Graduate School of Informatics, Nagoya University
3 Takahiro Katagiri, Satoshi Ohshima, Toru Nagai, Information Technology Center, Nagoya University

のための実行が，オーバヘッドとして追加される．そのため，混合精度による速度向上のメリットが，このオーバヘッドによる実行時間の増加を上回る場合のみ，速度向上の恩恵がある．

4.性能評価

4.1 評価方法

本研究では，全球雲解像モデル NICAM (Nonhydrostatic ICosahedral Atmospheric Mode)を対象プログラムとして評価実験を行った．NICAM のベンチマークパッケージの一つである nicam_dckernel_2016 の中で，雲の微小な物理演算を行う physicskernel_microphysics に存在するサブルーチン mp_nsw6 内の３重ループを対象とした．

ここでは，論文[2]で最も効果があったブロック方式を採用した．サブルーチン mp_nsw6 内にある対象ループ内を計算のまとまりごとに 36 ブロックに分割し，はじめにそれぞれのブロックの演算を SP に低精度化する．続いて，その中から速度向上率が大きかった 4 ブロックを選び同時に 2 ブロックを SP 化する 6 通りと，同時に 3 ブロックを単精度化する 4 通りの実験を行う．

また，36 ブロック全てを同時に SP 化した場合と，精度変更せず全てのグループが DP の場合の 2 通りもテストケースとして実験を行った．

4.2 性能評価環境

本実験では，名古屋大学情報基盤センターに設置されているスーパーコンピュータ「不老」TypeI サブシステムを用いた．詳細は以下の表 1 の通りである．

表 1.　実験環境

プロセッサ名	A64FX
1 ノードあたりのプロセッサ数	1
1 ノードあたりのコア数	48 コア+2 アシスタントコア(I/O 兼計算ノードは 48 コア+4 アシスタントコア)
周波数	2.2GHz
1 ノードあたりの理論演算性能	倍精度 3.3792 TFLOPS 単精度 6.7584 TFLOPS 半精度 13.5168 TFLOPS

表 1 から，DP と SP の TFLOPS 性能の違いは 2 倍であるため，FLOPS 値においては，混合精度演算による速度向上の上限は 2 倍となる．

4.3 評価結果

はじめに，全ブロックが倍精度の場合，実行時間は 0.0819 秒だった．これに対し，36 ブロックそれぞれを SP 化した結果，実行時間が最も短かったのはブロック 17，31 を SP 化した場合で，実行時間は 0.0733 秒だった．実行時間が短かった上位４ブロック 3，10，17，31 を選び，同時に 2 ブロックを SP 化した 6 通りの結果のうち，最も実行時間が短かったのはブロック 10 と 17 を同時に SP 化した場合で，0.0729 秒だった．最後に，同時に 3 ブロックを SP 化した 4 通りの結果のうち，実行時間が短かったのはブロック 10 と 17 と 31 を同時に SP 化した場合で，0.0727 秒だった。複数ブロックを SP 化することで，実行時間は短縮した．

表 2.　実行時間と allDP に対する速度向上率

	allDP	1 ブロックのみ	2 ブロック同時	3 ブロック同時
実行時間(s)	0.0819	0.0733	0.0729	0.0727
速度向上率	1.000	1.117	1.123	1.127

5.おわりに

本発表では，AT 言語 ppOpen-AT における混合演算の最適化において，今まで評価されていなかった，複数個所指定の効果について予備実験を行った．なお詳細な性能分析結果は，当日の発表で行う予定である．

謝辞

本研究は JSPS 科研費 JP19H05662 の助成を受けたものです．研究内容についてコメントを頂いた星野哲也准教授に感謝の意を表します．評価プログラムについて助言をいただいた山梨祥平氏に感謝します．

参考文献

[1] T. Katagiri, D. Takahashi , Japanese Autotuning Research: Autotuning Languages and FFT, Proc. of the IEEE, Vol. 106, Issue 11 , pp. 2056-2067 (2018)

[2] 山梨，八代，片桐，永井，大島，ppOpen-AT における演算精度と消費電力を考慮した自動チューニング方式の提案，情処研報，2021-HPC-182,12，pp.1-9 (2021)