5J-06

# 複数の自動並列化情報を用いたスレッド並列化に関する一検討

坂本 龍介 †1　小松 一彦 †2　佐藤 雅之 †3　小林 広明 †3

東北大学工学部機械知能航空工学科 †1　東北大学サイバーサイエンスセンター †2
東北大学情報科学研究科 †3

## 1 はじめに

近年の HPC システム向けプロセッサのコア数は増加しており，効率的にコアを活用するためにはマルチスレッド並列化が必要である．マルチスレッド並列化はコンパイラの自動並列化機能によっても実現することができるが，コンパイラの性能に依存してしまう．これを解決するために，高い並列化性能を持つコンパイラから得られる自動並列化情報を活用して OpenMP 化を行う方法が提案されている [1]．しかしながら，1 つのコンパイラのみから得られた並列化情報を利用していたため，そのコンパイラの並列化性能に依存してしまう問題がある．本報告では複数のコンパイラの自動並列化情報を用いてマルチスレッド並列化を行う手法について検討を行う．

## 2 自動並列化情報を用いた OpenMP 並列化

コンパイラによる自動並列化は自動で並列化を行うため，並列化にかかる人的コストを抑えつつ，並列化を実現できる．しかしながら，その並列化性能はコンパイラに依存してしまう．そこで，高い並列化性能を持つコンパイラからの自動並列化情報を活用して OpenMP 並列化を行う手法が提案されている [1]．コンパイラから得られた自動並列化情報を活用することで，マルチスレッド並列化が可能な箇所を容易に特定し，OpenMP 並列化に必要となる人的コストを抑えつつ，他の並列化性能が低いコンパイラを持つシステムに対しても高い並列性能を実現できる．

しかしながら，従来の方法では 1 つのコンパイラによる自動並列化情報のみを利用して OpenMP 並列化を行っているため，必ずしもコンパイラが理想的に並列化を実現してくれるとは限らず，並列化性能は利用するコンパイラの性能に依存してしまう．

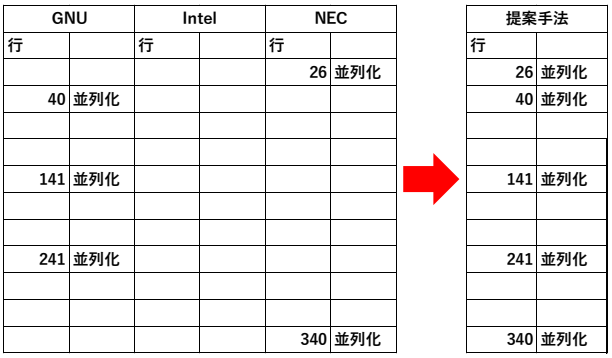


図 1: 複数の並列化情報

## 3 複数の自動並列化情報を用いた OpenMP 並列化

本報告では，複数のコンパイラから得られる並列化情報を組み合わせることによって，OpenMP 並列化を行う手法を提案する．複数のコンパイラから得られる並列化情報において，コードの特定の箇所について並列化するかしないかの判断が分かれる場合がある．その判断の背景として，並列化可能箇所を特定した上で並列化によるオーバーヘッドによる性能低下を懸念し敢えて並列化を行わなかった場合と，コンパイラの並列化性能が悪くそもそも並列化可能であると判断できなかった場合の 2 通りが考えられる．本報告では後者の可能性に着目し並列化箇所を可能な限り増やすことで性能向上を試みる．具体的には，複数の自動並列化情報を活用することで，あるコンパイラが並列化できていない並列化可能箇所については別のコンパイラによる並列化情報を利用することにより OpenMP 並列化を行う．これにより，1 つのコンパイラが理想的な並列化を行わない場合においても，そのコンパイラの性能に依存せず最適な並列化に近い性能を実現する．

---

**A Study on Thread Parallelization by Using Multiple Automatic Parallelizing Information**

Ryusuke Sakamoto†1, Kazuhiko Komatsu†2, Masayuki Sato†3, Hiroaki Kobayashi†3
Department of Mechanical and Aerospace Engineering, Tohoku University†1
Cyberscience Center, Tohoku University†2
Graduate School of Information Sciences, Tohoku University†3

表 1: 用いるコンパイラとそのオプション

| コンパイラ | | |
|---|---|---|
| 種類 | version | オプション |
| GNU | 11.2.1 | -Ofast -ftree-parallelize-loops=24/-fopenmp -fopt-info-all-optimized |
| Intel | 2021.7.0 | -fast -parallel/-qopenmp -qopt-report=3 |
| NEC | 3.5.1 | -O4 -mparallel/-fopenmp -report-all |

表 2: 実行するマシンのスペック

| プロセッサ | | |
|---|---|---|
| 種類 | 製品名 | コア数 |
| VH | Xeon Gold 6226 | 24 |
| VE | Type 20B | 8 |

## 4 評価

### 4.1 実験環境

コンパイルには表 1 に示す GNU，Intel，NEC のコンパイラを用いる．図 1 のように，複数のコンパイラから異なる並列化の情報が手に入った．GNU からは 3 箇所，Intel からは 0 箇所，NEC からは 2 箇所である．複数の自動並列化情報を用いる手法の有効性を確認するために，GNU，Intel，NEC の情報をそれぞれ使って OpenMP 化した 3 つのコード（single_GNU，single_Intel，single_NEC）と，3 つのコンパイラ情報を使ってマージした提案手法による OpenMP 化したコード，自動並列化を用いたコードの 5 種類のコードを用意した．これらのコードをそれぞれ GNU，Intel，NEC でコンパイルし，表 2 に示す Xeon および VE で実行する．実験の対象とするベンチマークとして NAS parallelBT ベンチマークの内の 1 つのプログラムである exact_rhs.f90 を採用する．

### 4.2 結果

BT ベンチマークでの結果を図 2 に示す．縦軸は実行時間を表し，横軸には自動並列化のコード，図 1 よりコンパイラの情報を単体で用いた 3 種類のコード，提案手法のように組み合わせたコードの計 5 種類のコードを実行したプロセッサを示す．図 2 から，Xeon（GNU）の場合,Xeon（Intel）の場合，VE の場合のいずれも提案手法が最も実行時間が短いことが分かる．これは，複数のコンパイル情報を活用することで，多くの箇所を並列化できたためだと考えられる．これにより，1 つの

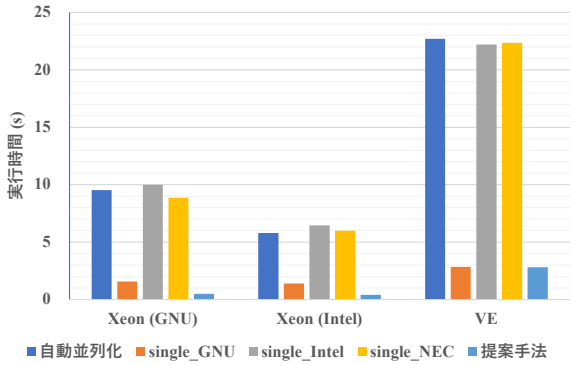


図 2: BT ベンチマークでの結果

コンパイル情報だけでなく，複数のコンパイル情報を活用することで，人間のエフォートを抑制しつつ，高い並列性能を実現できることが明らかになった．

## 5 おわりに

本稿では複数のコンパイラから得られた並列化情報から並列化を行う手法を提案した．具体的には，あるコンパイラが並列化できていないところを別のコンパイラによる並列化情報を利用することで補い合う形を取った．実験では，NAS parallel BT ベンチマークの exact_rhs.f90 プログラムを用いて，GNU，Intel，NEC のコンパイラについて，それぞれの単一の並列化情報を用いた OpenMP 並列化と提案手法を比べた．その結果，複数のコンパイル情報を活用することで，高い並列性能を実現できる可能性があることが分かった．今後は，並列化できる箇所の特定に加えて，その効果も考慮して並列化手法の開発に取り組む．

## 参考文献

[1] Komatsu, K., Egawa, R., Takizawa, H., Kobayashi, H. (2014). A Compiler-Assisted OpenMP Migration Method Based on Automatic Parallelizing Information. In: Kunkel, J.M., Ludwig, T., Meuer, H.W. (eds) Supercomputing. ISC 2014. Lecture Notes in Computer Science, vol 8488. Springer, Cham. https://doi.org/10.1007/978-3-319-07518-1_30