平成 29 年度　利用報告書

課題名（タイトル）：

Coarray Fortran プログラムの開発および評価

利用者氏名：

〇岩下　英俊*

中尾　昌広*

村井　均*

理研での所属研究室名：

***計算科学研究機構　研究部門　プログラミング開発環境研究チーム**

１．　本課題の研究の背景、目的、関係するプロジェクトとの関係

科学技術計算の並列化には MPI (Message Passing Interface)ライブラリを使うことが現在は一般的だが、一方で PGAS (Partitioned Global Address Space)と呼ばれる考え方の言語とライブラリが少しずつ広まりつつある。中でも、Coarray Fortran (CAF)は性能が出る可能性のある並列プログラミング言語として期待されている。我々のプロジェクトでは、CAF の実用性と性能について検証している。

我々は、開発中の XcalableMP コンパイラ（Omni XMP）に Coarray 機能を組み込み、いくつかの計算機上に実装して CAF プログラムの性能と生産性を評価している。PRIMEHPC FX100（HOKUSAI）については、特徴のある Tofu ネットワークを有しており、MPI との通信性能の比較で興味深いデータが得られているので、継続的に利用させていただいている。

本課題研究では、昨年度に引き続き CAF プログラムの作成・改善と、Omni XMP による評価を行った。これらの作業を通して、性能を出すために必要なコンパイラのキー技術を得ることと、性能を出すための CAF プログラミングのノウハウを得ることを目的とした。

２．　具体的な利用内容、計算方法

（１）Omni XMP Coarray 機能の基本性能の測定

エジンバラ大学が公開している EPCC Fortran Coarray microbenchmark suite（CAF ベンチ）を用いた。このベンチマークセットは CAF で記述されていて、CAF の基本的な記述に対して通信の性能を測定することができる。これを使って、我々は Omni XMP の Coarray 機能の性能を、MPI メッセージパッシングによるものと比較し、その差異の原因を分析した。

（２）性能改善の効果の検証

その結果から、CAF で MPI よりも性能を出すためのプログラミング技術と、CAF 処理系に必要な最適化項目を得た。その検証のため、CAF ベンチを修正し、かつ、Omni XMP を改善して性能測定を行い、同じ条件の MPI の性能と比較した。

３．　結果

（１）Omni XMP Coarray 機能の基本性能の測定

CAF ベンチの中の Ping-pong プログラムの評価で、CAF の PUT/GET と MPI メッセージパッシングの性能上の特徴を知ることができた。図 1 に測定結果を示す。データ量が大きいとき（8kB 以上）のバンド幅は、PUT, GET とも MPI と同等またはそれ以上に高い性能であることを確認した（PUT は最大 18%高速、GET は最大 9.3%高速)。しかし一方、データ量が小さいときのレイテンシについては、PUT, GET とも MPI より数倍大きいことが分かった（8B のデータで PUT は MPI の 2.8 倍遅い)。

PUT のレイテンシ時間の内訳を解析した結果、データ量が小さいときには本来のデータ移動に要する時間は非常に小さく、それに対して、通信完了待ちの時間と、同期に要する時間が、全体のコストのほとんどであることが分かった

（２）性能改善の効果の検証

これら２つのコストの軽減のため、ping-pong プログラムの修正と、CAF 処理系の改善を行った。効果が大きかったのは、以下の組合せであった。

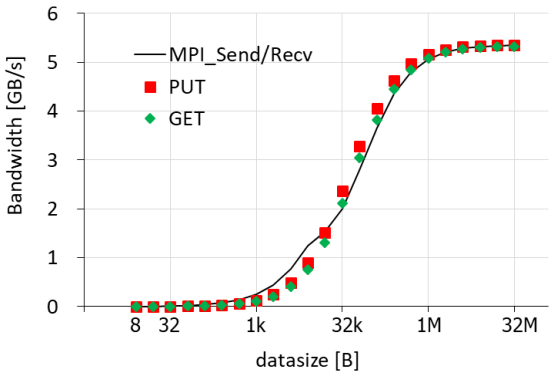
- Ping-pong プログラムの修正。1 回 PUT する度に 1 回同期を行うのではなく、複数回の PUT に対して 1 回同期を行う（多変数化)。これは恣意的な修正ではなく、その方が実プログラムに近

い通信パターンになると考えられる。

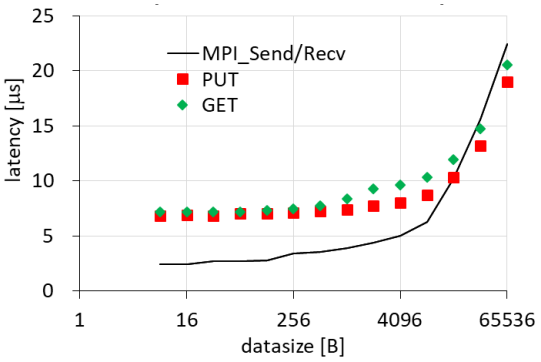
- CAF 処理系の改善。通信の完了待ちを毎回行うのではなく、同期の直前まで遅延させて一度に行う（non-blocking 通信）。これは一種の最適化であり、リモート側のデータ依存関係を壊さない範囲で可能である。

図 2 にこれらの改善の効果を示す。ping-ponf プログラムは PUT 版、MPI 版とも 8 変数 ping した後 8 変数 pong するように修正した。PUT の blocking と non-blocking は、プログラムは同じで、実行時の環境変数で処理系の動作をスイッチしている。MPI の blocking と non-blocking は、それぞれ MPI_Send/Recv と MPI_Isend/Irecv を使って記述した。

同図(a)はレイテンシの測定結果を示す。PUT について、多変数化だけでは改善の効果は見られなかったが、non-blocking 通信と併用することで、レイテンシを大きく下げられることが分かる。図 1(b)のレイテンシ値の 8 倍と比較して、データ量 8B から 8kB までの平均で、26%に縮小できた。一方で MPI では、blocking でも non-blocking でも多変数化の効果は薄かった。結果として、8B と 16B のプロットを除けば、non-blocking PUT が最も低いレイテンシを示している。



(a) バンド幅



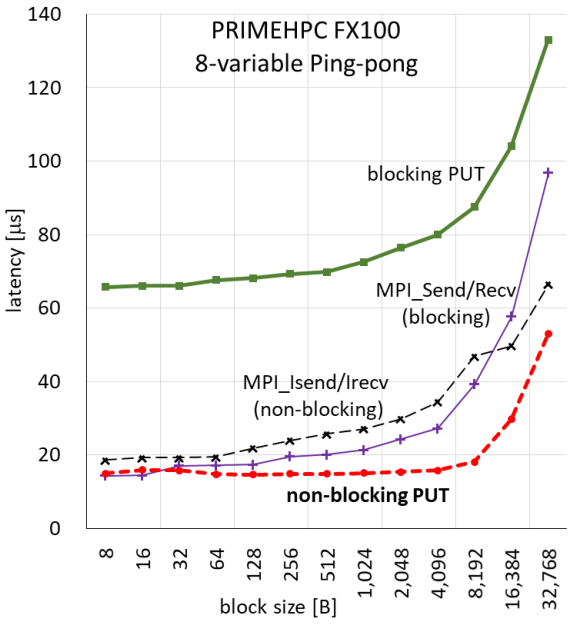
(b) レイテンシ

図 1　CAF ベンチ Ping-pong 性能の比較

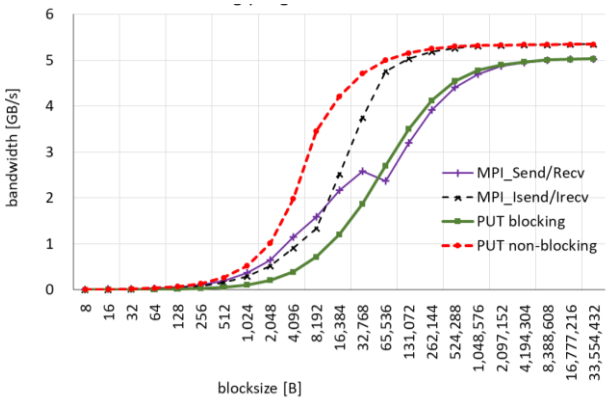
加えて、多変数 non-blocking 化は、バンド幅に対して予期せぬ良い効果があることが分かった。同図(b)に示すように、特に数 kB～数 MB の中規模サイズのデータに対して、non-blocking PUT のバンド幅が非常に高くなっている（MPI_Isend/Irecv に対して 8kB で 2.60 倍の性能）。

4．　まとめ

Coarray の PUT/GET 通信は MPI のメッセージパッシングと同等の高いバンド幅を持つが、小さなデータに対してはレイテンシが大きいことが分かった。レイテンシ時間の大部分は、通信の完了待ちと同期の時間であった。これらのコストは、多変数 non-blocking 通信によって大きく削減できることが分かった。



(a) レイテンシ



(b) バンド幅

図 2　多変数 non-blocking の効果

結論として、Coarray で MPI のメッセージパッシングと同等以上の性能を出すには、以下の 2 点が重要であると言える。

- Coarray 処理系は、最適化機能として non-blocking 通信をサポートする必要がある。
- Coarray プログラミングでは、多変数 non-blocking 通信のパターンに落とし込む性能チューニングが必要である。

５．　今後の計画・展望

CAF プログラミング手法について、より実アプリケーションに近いベンチマークを使って検証を続ける。既に Himeno ベンチマークでは、PUT 通信で MPI よりも高い性能が出ることを京コンピュータ上で確認したが、他のベンチマークについても同様な手法が有効かどうか試していく必要がある。

処理系については、最適化の実装を進める。

平成 29 年度　利用研究成果リスト

【国際会議などの予稿集、proceeding】

Hidetoshi Iwashita, Masahiro Nakao, Hitoshi Murai and Mitsuhisa Sato.
A Source-to-Source Translation of Coarray Fortran with MPI for High Performance
Proceedings of the International Conference on High Performance Computing in Asia-Pacific Region (HPC Asia 2018). Pages 86-97. January, 2018
ACM ISBN: 978-1-4503-5372-4/18/01 https://doi.org/10.1145/3149457.3155888

【国際会議、学会などでの口頭発表】

Hidetoshi Iwashita（発表者）, Masahiro Nakao, Hitoshi Murai and Mitsuhisa Sato.
A Source-to-Source Translation of Coarray Fortran with MPI for High Performance
International Conference on High Performance Computing in Asia-Pacific Region (HPC Asia 2018).
January 28, 2018, Chiyoda, Tokyo, Japan