1P08

PGAS モデルに基づく通信ライブラリを用いた OpenFMO プログラムの実装

○本田宏明[1,4], 稲富雄一[2,4], 眞木淳[3]
(九大情基センター[1], 九大シス情[2], 九州先端研[3], JST-CREST[4])

【はじめに】

現在，数万ノードからなる超高並列なスパコンが実現されており，研究開発が行われているエクサスケールクラスでは数千万ノードになると予想されている．一方，大規模並列量子化学計算プログラムの多くは，その実装として MPI 通信ライブラリを利用している．一部の NWChem [1] 等のプログラムは，Remote Memory Access (RMA) もしくは片側通信と呼ばれる方法を用いており，ARMCI や Global Arrays ライブラリ [2,3] を利用して実装を行なっているが，ミドルウエアレベルでは MPI ライブラリと共に動作するように実装されている．この標準ともいえる MPI ライブラリであるが，1 千万ノード環境にてライブラリのみのプログラムを実行した場合では，各ノードにて 300GB を越える要求メモリ量になるとの試算があり [4]，エクサスケールスパコンではノードメモリ量が数 GB に制限されるとの予想と比較するとプログラムの実行が困難になると予想される．そこで，ACE プロジェクト [5] の研究グループでは，Partitioned Global Address Space (PGAS) モデルならびに RMA に基づく，低遅延省メモリな Advanced Communication Primitives (ACP) ライブラリ [3] を新規に開発しており，OpenFMO [6] 等のアプリケーションに対して組込みを行なっている．本発表では，既存の MPI, ARMCI ライブラリや新規 ACP ライブラリを利用した OpenFMO の各実装や省メモリ化について，定性的な比較を行うことを目的とする．

【RMA と ACP ライブラリ】

現状の MPI ライブラリでは，並列プロセス数が増加することで送受信バッファや通信管理用バッファが増加してしまう．また，RMA が可能な ARMCI や Global Arrays, OpenSHMEM [7] 等のライブラリにおいても，グローバルデータアクセスのための共有データ領域の確保が同期的斉一的に行われるため，省メモリアルゴリズムを導入する余地は少ない．例えば ARMCI ライブラリでは， ARMCI_Malloc 関数の第一引数で与える共有メモリ情報に対応する void **ptr 変数について，各ノードが全てのノード分の情報を持つ必要があり，この変数ならびに RMA に関わる情報は 100Byte 弱程度であるため，1 千万ノードシステムでは，ノードあたり $100Byte \times 10^7 \sim 1GB$ 程度の要求メモリ量となってしまう．

これに対し ACP ライブラリでは，ランク毎のプログラムタスク毎に実際の通信に必要な最小限の Global Address(GA) と呼ばれる情報を持つことが出来，必要に応じユーザプログラムからこの GA の動的な確保ならびに廃棄が可能である．そのため，既存通信ライブラリに比較して省メモリな実装が可能であると考えている．ここで，ACP を利用した通信の実装では，ユーザプログラム中で異なるランク同士の送信元メモリアドレスと受信先メモリアドレスに対応した 2 つの GA によりデータの送受信が可能となるが，RMA を行なうランクに両方の GA を一旦集める必要がある．また，動的な確保が必要な GA 以外に，予めライブラリに登録されているランク毎の小さな starter 領域があり，この領域のデータのみは明示的な GA の交換なしに RMA が可能となっている．

図 1 に ACP ライブラリによる Gather の通信アルゴリズム例を示す．1. ルートランクにおいて，Sendbuf や Recvbuf に対応する GA_src ならびに GA_dst を取得し，ルート以外では Sendbuf のみの GA_src を取得する．2. ルートランクにおいて GA_dst を starter 領域にコピーする．3. ルー

ト以外のランクにおいて，ルートの starter 領域にある Recvbuf に対応する GA_dst をローカルメモリ領域にコピーする．4. ルートランクにおいて，Sendbuf から Recvbuf へ直接にデータコピーを行ない，ルート以外では GA_srt と GA_dst の情報によりデータ転送を行なう．5. GA_src, GA_dst の廃棄を行なう．以上の手続で全てのランクの Sendbuf のデータを Recvbuf に集めることが可能である．

【ACP ライブラリによる OpenFMO の実装】

図 1 の Gather はその手続の関数化が可能であり，プログラム中の利用 MPI 関数名のみ今回実装する関数名に変更することで，ACP を使用した省メモリな通信が可能になるともいえる．しかしながら，Remote Direct Memory Access をサポートする Infiniband や Tofu のハードウエア環境にて GA_src, GA_dst の取得ならびに廃棄を行なう場合では，OS の割込みを含めたハードウエアへのアクセスが必要となるため，図 1 の 1 や 5 のオーバーヘッド部分に無視出来ない遅延が発生してしまうと予想されている．この問題の回避のためには，送受信データアドレスが共通なリモートメモリ間で何度も通信を行なう場合に，一旦交換した GA の再利用を行なう必要がある．そのため，通信ライブラリのみの開発では無理があり，ユーザーアプリケーションのアルゴリズムを考慮した実装が必要となる．

現在，通常の RHF 法プログラムや OpenFMO にて実装しているグローバルカウンタを使用した動的負荷分散機構や共有データライブラリを対象に，低遅延かつ省メモリで見通しの良い ACP ライブラリの利用法について検討中であり，当日はその進捗を報告する．

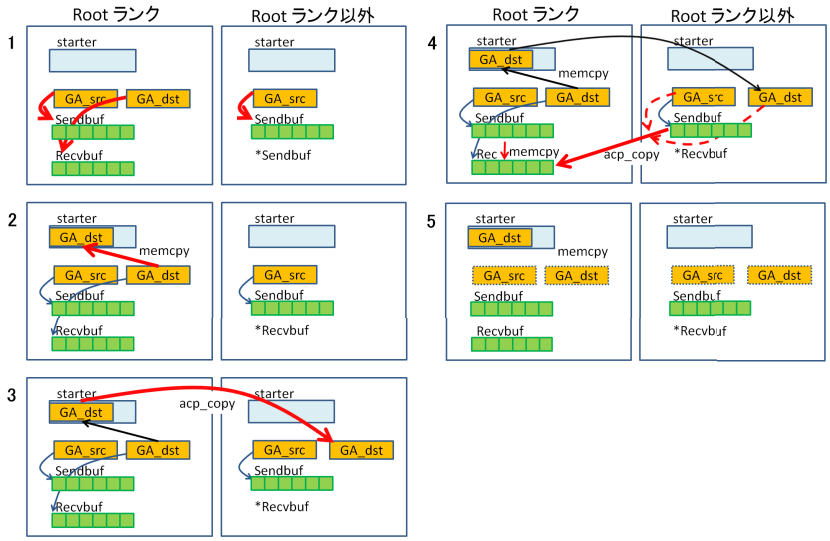

図 1: ACP ライブラリによる Gather アルゴリズム

【References】

[1] NWChem (online) available from ⟨http://www.nwchem-sw.org/index.php/Main_Page⟩.

[2] Aggregate Remote Memory Copy Interface (online) available from ⟨http://hpc.pnl.gov/armci/⟩.

[3] Global Arrays Toolkit (online) available from ⟨http://hpc.pnl.gov/globalarrays/⟩.

[4] 住元 他, 情報処理学会研究報告, Vol.2014-HPC-143, No.8, 2014., 安島 他, 同研究報告, No.9.

[5] ACE project (online) available from ⟨http://ace-project.kyushu-u.ac.jp/index.html⟩.

[6] OpenFMO Project (online) available from ⟨http://www.openfmo.org/⟩.

[7] The OpenSHMEM Project (online) available from ⟨http://openshmem.org/⟩.