5L-09

共役勾配法におけるダブルバッファリングを利用したRDMA通信の性能評価

出蔵英真† 藤井昭宏† 田中輝雄†
工学院大学†

1 はじめに

大規模科学技術計算のプログラムでは，MPI 通信による並列化が利用されることが一般的である．しかし，通信が繰り返し現れる場合，通信の速度に大きな影響を受ける．通信の遅延を軽減するために，名古屋大学のスーパーコンピュータ FX100 は RDMA（Remote Direct Memory Access）インタフェースが用意されている．RDMA は通信データを送信先ノードのメモリ（リモートメモリ）に直接書き込むことで高速に通信をすることが可能である．しかし，RDMA は非同期通信であり，データを参照している間にデータを上書きしてしまうことがある．そのため通信する前に同期が必要である．我々が提唱してきたダブルバッファリング[1]は 2 つの送受信バッファを用意することで同期処理を記述しなくてもデータの不整合を防ぐことができる．本研究では実アプリケーションへの応用としてダブルバッファリングを利用した共役勾配法（CG 法）を実装し実行速度の性能評価をした．

2 既存の疎行列ベクトル積における隣接通信

2.1 MPI_Isend, Irecv による隣接通信

並列計算において一般的には MPI による送信処理 Isend, 受信処理 Irecv が用いられる．どちらの処理も送受信が完了していなくても送受信とは関係のない処理を進めることができる．通信完了処理 MPI_Waitall を行うまで送受信は完了しない．

2.2 RDMA による隣接通信

RDMA による通信では，送信先ノードのプログラムを介在することなくデータの読み出しや書き込みを行うことができ，Isend，Irecv による MPI 通信と比較して，高速な通信を可能にしている．一方で送信先ノードは送信元ノードから送信完了通知もないため，送信完了を知るために送信先メモリ上に完了通知領域（マーク）を設けてその変更の完了をスピンウェイトで待機する．

3 ダブルバッファリング

通常の RDMA では送信先ノードの状況に関わらず送信元ノードがデータを書き込むので，読み出し中のデータが上書きされデータの不整合が発生する可能性がある．

Performance Evaluation of RDMA Communication Using Double Buffering in Conjugate Gradient Method
Hidemasa Degura†, Akihiro Fujii† and Teruo Tanaka†
†Kogakuin University

そのため，データの整合性を保つために同期をとる必要がある．本論で扱うダブルバッファリングを利用した RDMA 通信では送受信のバッファを 2 つ用意し，通信バッファを毎回切り替えて使用する．こうすることで受信データを読み込んでいる最中にデータが送信されたとしても，もうひとつのバッファに書かれることになる．そして送信先ノードのデータの読み込みが完了しないとき，送信先ノードは次のデータを送信しない．送信元ノードは送信先ノードが送るデータの受信を待機するので 2 回以上データの送受信がずれることはなく，読み込みの最中にそのデータが上書きされることはなくなる．これにより同期処理をしなくてもデータの整合性を維持することができ，高速化を実現することが可能である．動作イメージのフローチャートを図 1 に示す．

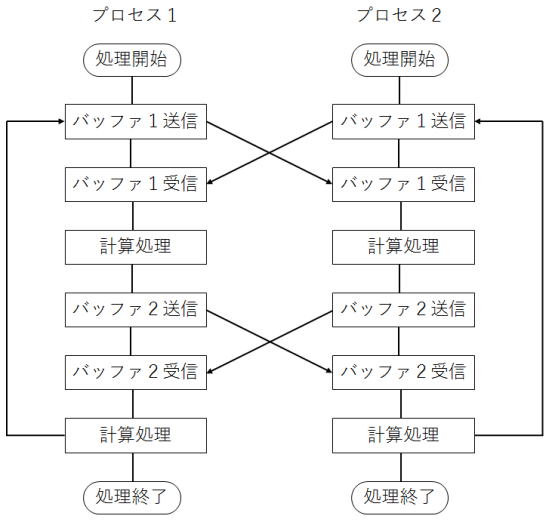

図 1 ダブルバッファリングの動作イメージ

4 実験

4.1 実験条件・環境

今回は実行速度に関して疎行列ベクトル積(SpMV)と CG 法について実験する．疎行列の分割方法はそれぞれ行ブロック分割と Metis[2]による分割を使用し，通信方法は MPI 通信，RDMA 通信，ダブルバッファリングを利用した RDMA 通信を使用した．

各実験では 100×100×100 の立方体領域をもつポアソン問題を対象とした．これは一行あたりおよそ 27 の非ゼロ要素を持つ疎行列に対応する．ノードあたりのプロセス数は 1 とする．実験環境[3]を表 1，計測名称を表 2 に示す．

表 1　実験環境

CPU	Fujitsu SPARC64 XIfx
コア数	32
総ノード数	2880
コンパイラ	mpifccpx
分割数	4-100

表 2　計測名称

	ダブルバッファリング無し	ダブルバッファリング有り
行ブロック分割	rdma-fix	dbf-fix
Metis 分割	rdma-metis	dbf-metis

4.2 実験結果・考察

行ブロック分割の項目は行ブロック分割を行った MPI 通信を，Metis 分割の項目は Metis 分割を行った MPI 通信を 100%として， SpMV の結果を表したものを図 2, CG 法の結果を図 3 に示す.

SpMV について，行ブロック分割と Metis 分割のどちらもダブルバッファリングを利用した RDMA，通常の RDMA，MPI の順に速くなっていることが確認できた．分割数 100 では通常の RDMA とダブルバッファリングを比較すると行ブロック分割では 6%，Metis 分割では 14%速くなった．行ブロック分割と Metis 分割で分割した SpMV を比較すると，分割数が少ない場合は行ブロック分割のほうが速く，分割数が多くなるほど Metis 分割のほうが速くなった．これは Metis 分割と比べて行ブロック分割のほうが分割数を大きくすると通信の量が大きくなる分割方法だからだと思われる．

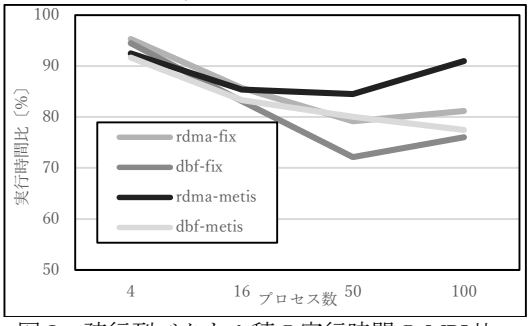

図 2　疎行列ベクトル積の実行時間の MPI 比

CG 法についてもどちらの分割法もダブルバッファリングを利用した RDMA，通常の RDMA，MPI の順に速くなった．行ブロック分割，Metis 分割による CG 法を比較すると，こちらでも分割数が少ない場合は行ブロック分割のほうが速く，分割数が多くなるほど Metis 分割のほうが速くなった．通常の RDMA とダブルバッファリングを利用したものを比較すると行ブロック分割では 3%，Metis 分割では 13%速くなった.

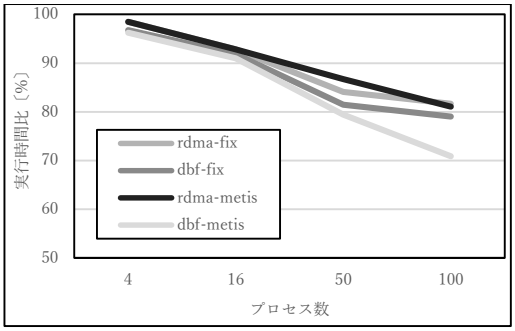

図 3　CG 法の実行時間の MPI 比

行ブロック分割を行った際について，通常の RDMA とダブルバッファリングを利用した RDMA の実行時間の比は，CG 法は SpMV で計測したときと比べて半分ほどとなった．原因として CG 法の計算では SpMV の直後にベクトルの内積計算を行っていることが考えられる．ベクトルの内積計算では，各プロセスに分割されたベクトルをすべて足し合わせる必要があるので，その時点で各プロセスは同期をとることになる．ダブルバッファリングで SpMV の同期処理を削減しても，直後の内積計算で同期するので大きな差が出なかったのだと予想した．しかし，Metis 分割では SpMV と変わらない速度差が出ている．これは，隣接通信時の隣接プロセス数が影響していると考える．通常の RDMA は隣接通信の直前に送信完了を通知する領域にマークをリセットする処理が隣接プロセスの数だけ必要である．そして Metis 分割では行ブロック分割と比べ隣接プロセス数が多くなる傾向がある．ダブルバッファリングを利用した RDMA は通信完了を通知するマークを通信ごとに変化させているため，次回通信時にマークをリセットする処理がないので差が出たのだと考えられる．

5　おわりに

ダブルバッファリングを利用した RDMA の実アプリケーションへの応用として CG 法を実装した．比較対象として通信の手法，MPI と RDMA を利用した CG 法と，それぞれの通信手法を利用した疎行列ベクトル積を実装し，比較・評価した．結果として SpMV においては、ダブルバッファリングを利用すると Metis による分割では 14%，行ブロック分割では 6%ほど実行時間が削減された．しかし，CG 法においては，行ブロック分割の場合，ダブルバッファリングを利用しても 3%ほどと SpMV と比べ十分な効果が得られなかった．しかし，Metis による分割では 13%ほど削減し SpMV と同様に十分な効果が得られた.

今後の課題として，CG 法の内部で発生する内積計算で行われる同期を削減する必要がある.

参考文献

[1] 熊井駿太，FX10 上における隣接通信の高速化，2018 年度工学院大学卒業論文

[2] METIS - Serial Graph Partitioning and Fill-reducing Matrix Ordering
http://glaros.dtc.umn.edu/gkhome/metis/metis/overview
アクセス日時（2018 年 12 月 25 日）

[3] スーパーコンピュータシステム全体構成-名古屋大学情報連携統括本部
http://www.icts.nagoya-u.ac.jp/ja/sc/overview.html
アクセス日時（2018 年 12 月 25 日）

謝辞

本研究の一部は JSPS 科研費 JP18K19782, JP18K11340, JP15K15998 の助成を受けたものです.