2D－2

# 汎用超並列オペレーティングシステム SSS-CORE — 高速 MPI の実装と評価 —

森本 健司　松本 尚　平木 敬
東京大学大学院理学系研究科情報科学専攻

## 1 はじめに

分散メモリ型並列計算機環境でのプログラミングパラダイムとして、共有メモリモデルとメッセージパッシングモデルとが挙げられる。これら 2 つのモデルは一方を用いて他方を実現可能であるが、ライブラリとして提供され実現が容易であるメッセージパッシングモデルがこれまでのところ多く用いられている。しかし、プログラミングモデルとしてではなくシステムが提供する機能として考えた場合、通信の最適化や効率、柔軟性の観点からは共有メモリモデルの方が優っているということが議論されている [8]。

当研究室において開発が進められている汎用超並列オペレーティングシステム SSS-CORE[6] では、低コストでリモートのメモリにアクセスすることのできるメモリベース通信機能 (MBCF: Memory-Based Communication Facilities)[7] が提供されており、共有メモリモデルでのプログラムの実行を高速に行うことができる。さらに、メッセージパッシングモデルに基づいて書かれたプログラムに関しても、MBCF を用いて適切な書換えを施すかあるいは適切なランタイムルーチンを加えることによって、直接的にメッセージパッシング機能を実現する場合と比べても遜色のない性能で実行できることが期待される。

本研究では、メッセージパッシングモデルでのプログラミングの標準的なインタフェースである MPI (Message Passing Interface) Ver. 1.1[1] の基本機能を SSS-CORE 上で MBCF を用いて実装し、その基本性能を評価した。さらに、量子色動力学シミュレーションプログラム QCDMPI[3] を実行し、性能評価を行った。

## 2 メモリベース通信 MBCF

MBCF は、ハードウェアによる遠隔メモリアクセス実現機構 Memory-Based Processor (MBP)[4] および高機能分散共有メモリシステム Strategic Memory System (SMS)[5] を基に考案された、ソフトウェアによる遠隔メモリアクセス機能である。MBCF の特徴を以下に挙げる。

1. 通信先メモリへの直接的な操作が可能
2. 特殊な通信ハードウェアを必要としない
3. オーバヘッドが少ない
4. 通信先での操作が高機能
5. 通信の到着保証、順序保証がなされている

ワークステーションを 100Base-TX の Hub で接続した環境での MBCF の性能は以下の通りである。計測に使用した機器は、Axil 320 model 8.1.1 (Sun SS20 互換機、85 MHz SuperSPARC CPU × 1) 2 台、Sun Microsystems Fast Ethernet SBus Adapter 2.0、および 3Com Hub8/TP100 である。OS として SSS-CORE Ver. 1.0 を使用した。2 ノード間での MBCF 遠隔書き込みの Round-trip time、Peak bandwidth を計測し、Round-trip time 51 $\mu$s (データサイズ 4 byte 時)、Peak bandwidth 11.24 MB/s (データサイズ 1408 byte 時) という結果が得られた。これらの値は 100Base-TX のハードウェア性能をほぼ使い切った値と言え、専用の内部相互結合網を持つ MPP とほぼ同等の性能を持つことが示される [9]。

## 3 実装方式

一般にメッセージパッシングにより送信を行う場合、通信対象として指定されるのはタスク (ないしはタスク間に設けられた通信路) のみであり、対象アドレス空間内での受信位置は指定されない。このためメッセージパッシングを基としたシステムでは、送信されるべきデータは一旦受信側において共通に使われるバッファに蓄えられ、対応する受信が発行される時に改めて受信用領域にコピーされることになる。しかし、送信より以前に受信が発行される場合は、送信側は受信側の受信用領域の位置を知ることが可能である。この場合、通信システムが共有メモリモデルに基づいたものであれば、送信操作において送信すべきデータを受信用領域へ直接書き込むことが可能となり、バッファリングを介する必要はない。

そこで本実装では、通信プリミティブとして MBCF の遠隔メモリ書込機能と遠隔 FIFO 書込機能とを用いて次のように MPI 標準モード送受信関数を実現した。すなわち、送信関数ではまず受信側からの送信要求の有無を調べ、対応する要求がある場合は遠隔メモリ書込機能により直接データを送る。対応する要求がない場合は遠隔 FIFO 書込機能により受信側のバッファにデータを書える。受信関数では送信側から既にデータが送られているかを調べ、対応するデータがある場合は受信領域にコピーする。対応するデータがない場合は遠隔 FIFO 書込機能により送信側に送信要求を発行する。

上記の実装により、メッセージパッシングに伴う無駄なバッファリング・無駄なデータコピーを削減することができる。

## 4 実験および結果

3 章で述べた方式により実装した MPI 送受信関数の性能を評価するため、ワークステーションを 100Base-TX のスイッチで接続した環境で送受信の Round-trip time および Peak bandwidth を計測した。計測に使用した機器は以下の通り。

- Axil 320 model 8.1.1 (Sun SS20 互換機、85 MHz SuperSPARC CPU × 2 台
- Sun Microsystems Fast Ethernet SBus Adapter 2.0
- Bay Networks BayStack 350T

MBCF が提供されている OS としてワークステーション版 SSS-CORE Ver. 1.1 を用い、独自の計測プログラムにより測定を行った。比較のため、同一の機器の上で、OS として SunOS 4.1.4 を、MPI の実装として MPICH[2] Ver. 1.1 を用いた場合の値も計測した。MPICH は Argonne National Laboratory および Mississippi State University において開発された MPI の実装

---

The General-Purpose Scalable Operating System : SSS-CORE — Implementation and Evaluation of High Performance MPI —
Kenji MORIMOTO, Takashi MATSUMOTO, and Kei HIRAKI
Department of Information Science, Faculty of Science, University of Tokyo
morimoto@is.s.u-tokyo.ac.jp

で、ワークステーションクラスタに対する実装では TCP ソケットによる通信を基としている。

まず、メッセージサイズを変えながら Round-trip time を測った。2 つのプロセスが一方は送信・受信を、もう一方は受信・送信を繰り返す。前者において通信の開始直前から通信の終了直後までの時間を Round-trip time とした。表 1 はメッセージサイズを 4 バイトから 1 キロバイトまで変化させた時の 2 方式の Round-trip time である。表中、MPI/MBCF は MBCF を用いた本実装を、MPICH/TCP は TCP ソケットを用いた MPICH を表す。

表 1: 100Base-TX による MPI 送受信の Round-trip time

| Message size (byte) | 4 | 16 | 64 | 256 | 1024 |
|---|---|---|---|---|---|
| MPI/MBCF ($\mu$s) | 160 | 164 | 173 | 265 | 638 |
| MPICH/TCP ($\mu$s) | 1055 | 1123 | 1163 | 1131 | 1352 |

MBCF を用いた実装の通信遅延が、常に TCP 上の実装の遅延を下回っており、特にメッセージサイズが小さい時の比が著しい。さらに MPI/MBCF に関しては精度 0.5 $\mu$s のシステムクロックを用いてより詳細な測定を行い、メッセージサイズが 4 byte の時の Round-trip time が最小で 131 $\mu$s となることを確かめた。この値を MBCF 自体の 1 回の通信の遅延 51 $\mu$s と比べると、MPI を実現することによる追加的なオーバヘッドが極めて小さいことが分かる[1]。

次にメッセージサイズを変えながら Peak bandwidth を測定した。2 つのプロセスが一方は送信を、もう一方は受信を繰り返す。前者において送信の開始直前から最後に MPI_Barrier() で同期を取る直後までの時間を通信時間とし、総メッセージ転送量をこれで割ったものを Peak bandwidth とする。表 2 はメッセージサイズを 4 バイトから 1 キロバイトまで変化させた時の 2 方式の Peak bandwidth である。

表 2: 100Base-TX による MPI 送受信の Peak bandwidth

| Message size (byte) | 4 | 16 | 64 | 256 | 1024 |
|---|---|---|---|---|---|
| MPI/MBCF (KB/s) | 132 | 496 | 1724 | 4369 | 7384 |
| MPICH/TCP (KB/s) | 16 | 79 | 307 | 954 | 3261 |

一般にメッセージサイズが小さいと送受信操作のオーバヘッドのため充分な bandwidth を得られないが、MPI/MBCF はメッセージサイズに対する Peak bandwidth の立ち上がりが MPICH/TCP と比較して鋭く、メッセージサイズが小さくても bandwidth を得やすい。これは細粒度の通信が必要なアプリケーションや通信の統合が充分になされていないアプリケーションにおいても MPI/MBCF では効率を大きく損なうことなく通信を行えることを意味する。また、表からは洩れているが、メッセージサイズを大きくした時の MPI/MBCF の Peak bandwidth は 9.0 MB/s となっており、これは 100Base-TX のハードウェア性能の限界である 12.5 MB/s やこれを用いた際の MBCF の性能 11.2 MB/s に近い値といえる。

次に量子色動力学シミュレーションプログラム QCDMPI を実行し、実行時間とそのうちの通信に費やした時間とを測定した。問題サイズは $8 \times 8 \times 8 \times 8$ とし、プロセッサ数を 1 から 8 まで変化させて計測した。表 reftab:qcd に結果を示す。

今回のパラメータでのメッセージサイズは約 4 ～ 18 キロバイトであり比較的粗粒度の通信を行っているが、この場合でも MBCF を用いた MPI の実装による通信の方が MPICH/TCP より良い性能を出している。

[1]今回実験に用いたスイッチでは、MBCF の性能評価に用いた Hub に比べてパケットの通過に 20 $\mu$s 程度の遅延が生じる。

表 3: QCDMPI の実行時間

MPI/MBCF

| 台数 | 1 | 2 | 4 | 8 |
|---|---|---|---|---|
| 総実行時間 (秒) | 2.634 | 1.398 | 0.774 | 0.440 |
| 通信時間 (秒) | — | 0.084 | 0.100 | 0.096 |

MPICH/TCP

| 台数 | 1 | 2 | 4 | 8 |
|---|---|---|---|---|
| 総実行時間 (秒) | 2.550 | 1.428 | 0.803 | 0.526 |
| 通信時間 (秒) | — | 0.117 | 0.120 | 0.186 |

## 5 まとめ

低コスト・高機能な遠隔メモリアクセス機能であるメモリベース通信を用いて高速な MPI を実装した。メモリベース通信の機能である遠隔メモリ書込機能と遠隔 FIFO 書込機能とを用いることで、MPI においてシステムが用意すべきバッファを高速に実現し、さらにバッファを用いない通信も可能にした。その結果、メッセージ型通信である TCP を用いた MPI の実装に対して、MBCF を用いた実装の優位性が示された。

通信性能を 100Base-TX で接続されたワークステーションクラスタにおいて測定し、最小 Round-trip time 131 $\mu$s, Peak bandwidth 9.0 MB/s という値を得た。これらの値より、共有メモリを実現する機構であるメモリベース通信をメッセージパッシング方式の実現のベースとすることの有効性が示された。特に、メッセージサイズが小さい通信において、オーバヘッドの小さなメモリベース通信が有効であることが示された。

## 参考文献


[1] Message Passing Interface Forum. MPI: A Message-Passing Interface Standard. http://www.mcs.anl.gov/mpi/, May 1995.

[2] William Gropp, Ewing Lusk, Nathan Doss, and Anthony Skjellum. A High-Performance, Portable Implementation of the MPI Message-Passing Interface Standard. *Parallel Computing*, Vol. 22, No. 6, pp. 789–828, September 1996.

[3] 日置慎治. QCDMPI. http://insam.sci.hiroshima-u.ac.jp/QCDMPI/.

[4] 松本尚, 平木敬. Memory-Based Processor による分散共有メモリ. 並列処理シンポジウム JSPP '93 論文集, pp. 245–252, May 1993.

[5] 松本尚, 平木敬. キャッシュインジェクションとメモリベース同期機構の高速化. 計算機アーキテクチャ研究会報告 No.79-1, pp. 113–120. 情報処理学会, August 1993.

[6] 松本尚, 平木敬. 汎用並列オペレーティングシステム SSS-CORE の資源管理方式. 日本ソフトウェア科学会第 11 回大会論文集, pp. 13–16, October 1994.

[7] 松本尚, 平木敬. 汎用超並列オペレーティングシステム : SSS-CORE — ワークステーションクラスタにおける実現 —. 情報処理学会研究報告 96-OS-73, Vol. 96, No. 79, pp. 115–120. 情報処理学会, August 1996.

[8] 松本尚, 平木敬. 共有メモリ vs. メッシージパッシング. 情報処理学会研究報告 97-ARC-126, Vol. 97, No. 102, pp. 85–90. 情報処理学会, October 1997.

[9] 松本尚, 信国陽二郎, 満原茂, 竹岡尚三, 平木敬. 汎用超並列オペレーティングシステムカーネル SSS-CORE — 資源管理およびメモリベース通信の実現と評価 —. 第 16 回技術発表会論文集, pp. 283–296. 情報処理振興事業協会, October 1997.