ハイパフォーマンス
コンピューティング 77－9
（1999．8． 2）

NX-MPIブリッジライブラリの開発

田上豊彦，坂上仁志

姫路工業大学工学部情報工学科

概要

並列計算機上でプログラムを開発するためには一般にアーキテクチャに依存した専用ライブラリを用いなければならず，プログラムの可搬性を大きく損う．このため，アーキテクチャによらない汎用ライブラリとしてMPI(Message Passing Interface)が提案されている．MPIは多くの並列計算機上に実装されているので，プログラムをMPIで記述さえしておけば，アーキテクチャの異なる多くの並列計算機で実行することが可能である．本論文では，並列計算機Paragonの専用ライブラリであるNXライブラリを用いて書かれたプログラムを修正することなくMPI上で実行できるようにするNX-MPIブリッジライブラリの作成について述べる．

NX to MPI Bridge Library

Toyohiko TAGAMI, Hitoshi SAKAGAMI

Computer Engineering,
Himeji Institute of Technology

abstract

In order to develop programs on parallel computer systems, we must use the specific library that depends on their own architectures. This dependencies lead to less portability of the program. The MPI(Message Passing Interface) has been proposed as a general independent library and implemented on many parallel computer systems. Thus it is possible to execute the MPI program on the different architectures without modifying the source codes.

In this paper, we have developed the NX-MPI bridge library and made it possible to run the NX programs, which can be performed only on the Paragon system, with the MPI.

1 序論

近年，高速に計算するために並列計算機が用いられている．しかし，並列計算機上でプログラムを開発するためには一般にハードウェアアーキテクチャに依存した専用ライブラリを用いなければならない．これは，プログラムのポータビリティを大きく損ない，老朽化や演算速度の向上のため，新たな並列計算機へ移行する際の大きなボトルネックとなっている．このため，専用ライブラリから汎用ライブラリへ移行する必要性が大きくなってきている．そこで，最近になってMPI（Message Passing Interface）と呼ばれる並列プログラミングインターフェースが提案され，多くの並列計算機に実装されている．これにより，MPIで記述さえしておけば，アーキテクチャが異なる多くの並列計算機で実行することができ，ポータビリティを格段に向上させることができる．

本論文では，並列計算機Paragonの専用ライブラリであるNXライブラリからMPIライブラリへのブ

リッジライブラリの作成について述べる．このブリッジライブラリにより，NX を使って書かれたプラズマ粒子コードを修正することなく MPI 上で実行できるようにすることを目指す．

2 MPI

MPI とは，メッセージパッシングによる並列処理を記述するためのインターフェースである．メッセージパッシングとは，分散メモリ型並列処理環境において，並列に動作しているプロセス間でデータをやり取りする処理方式である．

MPI によるプログラミングは，通常 SPMD(Single Program Multiple Data) モデルで作成され，プログラム中に MPI のルーチンを加えることでプログラミングを行なう．並列プログラミングは，従来のシングルプロセッサによるプログラミングとは異なり，プロセッサ間でのデータ通信，実行の同期や I/O アクセスについて配慮しながらプログラミングを行う必要がある．

MPI-1 は既に多くの並列計算機上に実装されており，現在，新たな拡張機能を組み込んだ MPI-2 の実装が進められている．MPI-2 では，MPI-1 と比較して次のような機能が追加されている．

- 並列 I/O — 多くの並列計算機上で実装されている並列 I/O の MPI によるサポート
- C++，Fortran90 への対応 — 使用可能言語の拡大
- プロセスの生成と制御 — 新たにプロセスを生成しそれらを制御する
- 片側通信 — 送信側と受信側での対応した操作を行わず，片側だけのプロセスで直接リモートホストのメモリにアクセスする

現在，MPI-2 を実装しているソフトはまだ存在していない．しかし，MPI-2 の機能を少しだけ取り込んだ MPI-1.2 と呼ばれるバージョンの MPI が既に開発されている．これは，MPI-2 の機能のうち並列 I/O 機能と C++，Fortran90 への対応の機能をほぼ行っているものである．そこで，本論文ではこの MPI-1.2 を実装している mpich1.2 を利用して並列 I/O の動作確認を行う．

3 NX ライブラリと MPI

3.1 通信ルーチン

NX とは並列計算機 Paragon 専用の並列処理記述ライブラリである．NX もメッセージパッシングにより並列処理を行っており，MPI とは非常に類似した構成となっている．そのため，ほとんどの場合，ルーチンにおける引数の数や順番が異なる程度の違いしかないため，それぞれの引数の対応を考慮することにより NX ルーチンと MPI ルーチンを 1 対 1 に対応づけることができる．また，ブロードキャストやリダクション操作など，引数の対応だけではうまくいかない場合でも，単一ルーチンの機能を複数のルーチンの組み合わせで実現することにより，NX ルーチンを MPI ルーチンに置き換えることが可能となる．例えば，ブロードキャストの場合は，NX では送信動作の特別な場合として定義され，受信側は通常の送受信と同じ受信ルーチンを用いているが，MPI では集団通信機能として送受信両方を含んだものとなっている．そのため，NX のブロードキャスト送信に相応する受信についてのみ MPI の集団通信ルーチンに対応づけなければならない．

表 1: 主な通信ルーチンにおける NX と MPI の対応関係

処理	NX	MPI
プロセッサ数	np=numnodes()	mpi_comm_size(comm,np,err)
自プロセッサ ID	iam=mynode	mpi_comm_rank(comm,iam,err)
送信	csend(tag,buf,len,node,ptype)	mpi_send(buf,len,mpi_byte, node,tag,comm,err)
受信	crecv(tag,buf,len)	mpi_recv(buf,len,mpi_byte, mpi_any_source,tag,comm,err)
ブロードキャスト	csend(tag, buf, len, -1, ptype) crecv(tag, buf, len,ptype)	mpi_bcast(buf,len,mpi_byte, root, comm, err)
時間の取得	dclock()	mpi_wtime()
同期	gsync()	mpi_barrier(comm,err)
集計加算	gdsum(dvar, size, tmp)	mpi_allreduce(dvar,tmp,size,mpi_byte, mpi_sum, iam,comm,err) dvar=tmp
集計最大値	gdhigh(dvar, size,tmp)	mpi_allreduce(dvar,tmp,size,mpi_byte,mpi_max, iam,comm,err) dvar=tmp
開始	無	mpi_init(err)
終了	無	mpi_finalize(err)

3.2 並列 I/O

Paragon には Parallel File System (pfs) という並列ファイルシステムが実装されており，NX ライブラリの並列 I/O ルーチンから pfs 上のファイルにアクセスが可能となっている．ファイルのオープンには gopen というルーチンを用い，様々な入出力モードでファイルにアクセスできる．今回使用しているプラズマ粒子コード中では以下のモードが用いられている．

M_SYNC 共有ファイルポインタを用い，シーケンシャルにアクセスを行う．プロセッサごとで異なるデータ長を扱うことができる．

M_RECORD プロセッサごとに個別のファイルポインタを持っており，各ノードが並列に I/O アクセスを行う．データ長はプロセッサごとで一定となるが，高速に処理を行うことができる．

MPI-2 の並列 I/O の機能を用いて pfs を実現することを考える．MPI-2 の並列 I/O 機能は各 PE ごとにダイナミカルなファイルアクセスを可能としている．M_SYNC モードは，ファイルポインタを共通に持つことで容易に実現できる．M_RECORD モードは，ファイルのアクセス範囲を指定する view と呼ばれる属性を変更することにより実現できる．それらのモードごとの MPI-2 による対応関係を表 2 に示す．

表 2: 入出力ルーチンの対応関係

(a) M_SYNC モード

処理	NX	MPI
ファイルオープン	gopen(unit,path,iomode)	mpi_file_open(mpi_file_open(comm, path, MPI_MODE_SEQUENTIAL, info, fh, err)
書き出し	cwrite(unit,buf,nbytes)	mpi_file_shared_write(fh, buf, nbytes, type, status, err)
読み込み	cread(unit,buf,nbytes)	mpi_file_shared_read(fh, buf, nbytes, type, status, err)
ファイルクローズ	close(unit)	mpi_file_close(fh)

(b) M_RECORD モード

処理	NX	MPI
ファイルオープン	gopen(unit,path,iomode)	mpi_file_open(comm, path, mode, info, fh, err)
書き出し	cwrite(unit,buf,nbytes)	mpi_type_create_subarray(1, nbytes*np, nbytes, nbytes*iam, order, type, newtype, err) mpi_file_set_view(ifh, 0, type, newtype, datarep, info, err) mpi_file_write(fh, buf, nbytes, type, status, err)
読み込み	cread(unit,buf,nbyte)	mpi_type_create_subarray(1, bytes*np, nbytes, nbytes*iam, order, type, newtype, err) mpi_file_set_view(ifh, 0, type, newtype, datarep, info, err) mpi_file_read(fh, buf, nbytes, type, status, err)
ファイルクローズ	close(unit)	mpi_file_close(fh)

4 NX-MPI ブリッジライブラリ

前節で述べたように NX ルーチンは機能的に MPI ルーチンで置換することができる．そこで，NX ルーチンを MPI ルーチンに橋渡しするブリッジライブラリを作成することで，プログラムの MPI 化を容易にする．

4.1 通信ルーチン

表 1 の対応関係をもとに，MPI ルーチンによる NX ルーチンのインターフェースルーチンを作成する．ただし，MPI には必須であるが NX には存在しないルーチン，すなわち，開始ルーチンと終了ルーチンに関しては，NX で書かれたプログラムを修正してそれらのルーチンの呼びだしを追加する必要がある．開始ルーチン，終了ルーチンをそれぞれ NX_INIT，NX_TERM と定義し，これらのルーチンを用いて対応する MPI のルーチンを呼ぶ．また，受信インターフェースルーチンは，ブロードキャスト送信に相応する受信と通常の送信に相応する受信を区別する必要があるが，これは引数の内容を判断するだけではできない．そこで，ブロードキャスト送信の動作を全 PE へのユニキャスト送信として実現する．

4.2 並列I/O

通信ルーチンと同様に，並列I/Oも表2の対応関係を用いることによって，ライブラリを開発することができる．それぞれのモードのによって，異なる動作を行うようにする．しかし，表2の並列I/Oルーチンだけをライブラリ化しても，Paragonの並列I/Oのファイルオープン用サブルーチンであるgopenに対応するファイルクローズがFORTRANの通常のclose文であるため，gopenに対応するファイルクローズのルーチンを呼びだすことができないという問題が発生する．そこで，この問題を解決するためにgopenに対応するclose文は，gcloseという独自のルーチンを作成し，ファイルのオープンとクローズを対応させる．

以上のことから，プログラムに開始ルーチンおよび終了ルーチンを追加し，gopenに対応するclose文をgcloseの呼びだしに変更することだけで，このブリッジライブラリを介して，NXで書かれたプログラムをMPI上で実行することができる．

5 性能評価

まず，表1の対応関係を元に，NXで書かれたプラズマ粒子コードを直接修正する．このとき，実験を行うParagonではバージョン1.2以上のMPIが実装されていないため，MPIの並列I/Oルーチンを利用することはできない．そのため，並列I/Oルーチンについてはそのまま NXライブラリを用いる．

次に，通信ルーチンのブリッジライブラリを作成し，データ通信をMPI化する．このときも，並列I/OルーチンはNXライブラリを用いている．

また，本論文では，何も修正していないNXライブラリをそのまま用いたものをNX版，直接修正を行ったものをMPI版，通信ルーチンライブラリのみを用いて作成したものを通信ライブラリ版と呼ぶ．

実際にParagonで実行させNX版とMPI版，通信ライブラリ版との比較を行う．まず，粒子数を131,072，メッシュサイズを64 × 256に固定し，プロセッサ数の変化させて並列性能を計測する．NX版の時間を基準とした速度比を図1に示す．

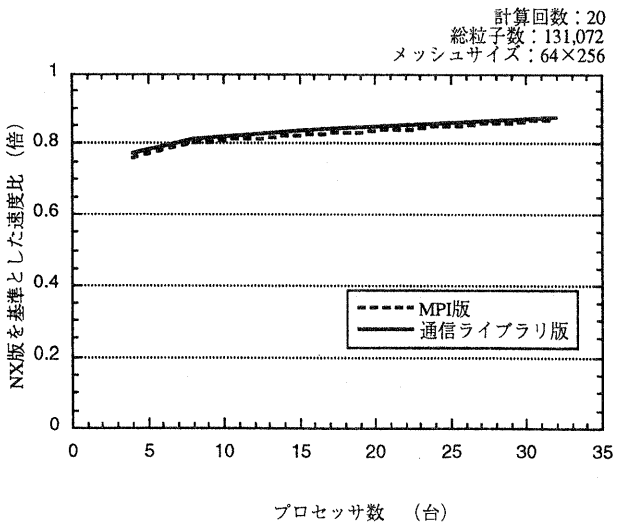

図 1: プロセッサ数の変化に伴う性能評価

図1から明らかなように，通信ライブラリ版はMPI版とほぼ同等の性能が出ている．つまり，このライブラリを用いてMPI化を行うだけでよく，ソースコードを直接修正する必要はないことが分かる．また，通信ライブラリ版はNX版に比べ約80%の性能しか得られていない．この原因を詳細に調べたところ，NXライブラリに比べてMPIでは，リダクション演算の性能が劣っていることが分かった．これは，NXはParagon専用でハードウェア性能を最大限に引きだすように作られているが，MPIはハードウェアに対する最適化が不十分であるためだと考えられる．

更に，並列I/Oを含むブリッジライブラリを作成して，すべてのNXルーチンをMPI化し，NXで書かれたプラズマ粒子コードを本学UNIX室で実行させ，完全にMPI化されたことを確認した．これを並列I/O版と呼ぶ．

並列I/O版は，M_RECORDモードにおいて，本学のUNIX室で正常に動作している．このためバージョン1.2以上のMPIが実装されれば多くの並列計算機上で動作するものと考えられる．また，M_SYNCモードについては，バージョン1.2のMPIでは動作しないので，動作確認を行うことができなかった．

6 結論

本研究では，NXルーチンからMPIルーチンへのNX-MPIブリッジライブラリを作成し，すでにNXライブラリを用いて記述されたプログラムに数行の修正を行うだけでMPI上で実行することを可能にした．

プラズマ粒子コードにおいて，直接修正したMPI版と通信ブリッジライブラリを介した通信ライブラリ版では，同程度の性能を出すことができ，NX-MPIブリッジライブラリの有効性が確認できた．また，両者ともNX版に比べて80%以上の性能で動作し，MPIを利用することによる大幅な性能低下は見られなかった．

並列計算機Paragon上でしか動作しなかった粒子コードが，並列I/Oを含めてMPIを実装した本学UNIX実習室で動作するようになった．

参考文献

[1] “MPI:メッセージ通信インターフェース標準（日本語訳ドラフト）”，MPIフォーラムMPI日本語訳プロジェクト (1996)

[2] “メッセージ通信インターフェースに対する拡張機能（日本語訳ドラフト）”，メッセージ通信インターフェースフォーラム (1999)

[3] “MPI-2 : Extensions to the Message-Passing Interface” ,Message Passing Interface Forum(1997)

[4] “関西研究所 並列計算機 (Paragon) 利用手引き (第3版)”，日本原子力研究所計算機科学技術センター (1998)

[5] 上島豊・荒川拓也・佐々木明・横田恒：“Paragon上でのスカラー超並列プログラム開発ガイド”，日本原子力研究所 (1998)