JAERI-Data/Code
98-034

異機種並列計算機間通信ライブラリ：Stampi ー利用手引書

1998年11月

今村俊幸・小出　洋・武宮　博

日本原子力研究所
Japan Atomic Energy Research Institute

本レポートは，日本原子力研究所が不定期に公刊している研究報告書です。
入手の問合わせは，日本原子力研究所研究情報部研究情報課（〒319-1195　茨城県那珂郡東海村）あて，お申し越しください。なお，このほかに財団法人原子力弘済会資料センター（〒319-1195　茨城県那珂郡東海村日本原子力研究所内）で複写による実費頒布をおこなっております。

This report is issued irregularly.
Inquiries about availability of the reports should be addressed to Research Information Division, Department of Intellectual Resources, Japan Atomic Energy Research Institute, Tokai-mura, Naka-gun, Ibaraki-ken, 319-1195, Japan.

編集兼発行　　日本原子力研究所

JAERI-Data/Code 98-034

異機種並列計算機間通信ライブラリ：Ｓｔａｍｐｉ ― 利用手引書

日本原子力研究所計算科学技術推進センター
今村　俊幸・小出　　洋・武宮　　博

（1998年10月２日受理）

異なる並列計算機を任意に組み合わせ計算を行うために、ＭＰＩを通信のための単一のインターフェイスとする異機種並列計算機間通信ライブラリ Stampi を開発した。 Stampi はＭＰＩ２のインターフェイス仕様に基づき、異なる計算機への動的なプロセス生成とＭＰＩのセマンティクスを保ったまま外部との通信を可能としている。従来、ベンダー提供のライブラリでは提供されていなかった外部へのプロセス生成と通信の両機能を実現し、これにより異機種の並列計算機を用いた計算が可能となった。現在 Stampi は計算科学技術推進センター所有の並列計算機群ＣＯＭＰＡＣＳの５台の並列機とグラフィックサーバの計６機に実装され、それら計算機間での任意の通信を可能としている。

日本原子力研究所：〒153－0061　東京都目黒区中目黒２－２－54

JAERI-Data/Code 98-034

Stampi: A Message Passing Library for Distributed Parallel Computing — User's Guide

Toshiyuki IMAMURA, Hiroshi KOIDE and Hiroshi TAKEMIYA

Center for Promotion of Computational Science and Engineering
Japan Atomic Energy Research Institute
Nakameguro, Meguro-ku, Tokyo

(Received October 2, 1998)

A new message passing library, Stampi, has been developed to realize a computation with different kind of parallel computers arbitrarily and making MPI (Message Passing Interface) as an unique interface for communication. Stampi is based on MPI2 specification. It realizes dynamic process creation to different machines and communication between spawned one within the scope of MPI semantics. Vender implemented MPI as s closed system in one parallel machine and did not support both functions; process creation and communication to external machines. Stampi supports both functions and enables us distributed parallel computing. Currently Stampi has been implemented on COMPACS (COMplex PArallel Computer System) introduced in CCSE, five parallel computers and one graphic workstation, and any communication on them can be processed on.

Keywords: Distributed Parallel Computing, Communication Library, MPI,
Dynamic Process Creation, External Communication

目　　次

1．はじめに …… 1
2．異機種並列計算機間通信ライブラリ Stampi …… 3
2.1 Stampiシステム構成 …… 3
2.2 Stampiが対象とする並列計算機 …… 4
3. Stampiの利用方法 …… 5
3.1 Stampiを利用する前に（利用者セットアップ） …… 5
3.2 Stampiを用いたプログラム開発 …… 6
3.2.1 Fortran での開発: jmpif90 …… 6
3.2.2 C言語での開発: jmpicc …… 7
3.3 Stampiでのプログラム実行: jmpirun …… 7
4. Stampiにおける基本的なプログラミング手法 …… 8
4.1 動的プロセス生成と通信 …… 8
4.2 Infoオブジェクトの取り扱い …… 12
4.3 Infoオブジェクトを使った外部プロセス生成 …… 13
5. おわりに …… 15
参考文献 …… 16
付録A Stampiライブラリインターフェイス …… 18
付録B ＣＯＭＰＡＣＳ各機種でのコンパイル・リンク オプション …… 20
B.1 日立SR2201でのコンパイル・リンク手順 …… 20
B.2 富士通VPP300でのコンパイル・リンク手順 …… 21
B.3 IBM SP/2でのコンパイル・リンク手順 …… 22
B.4 CRAY T94でのコンパイル・リンク手順 …… 23
B.5 NEC SX4でのコンパイル・リンク手順 …… 24
B.6 SGI Onyxでのコンパイル・リンク手順 …… 25
付録C ＣＯＭＰＡＣＳ各機での.rhostsの設定例 …… 26
C.1 hi00011（日立 SR2201）の設定 …… 26
C.2 fu00011（富士通 VPP300）の設定 …… 26
C.3 ibmsp50（IBM SP2）の設定 …… 27
C.4 cr00011（CRAY T94）の設定 …… 27
C.5 ne00011（NEC SX4）の設定 …… 28
C.6 ccsemga（SGI Onyx）の設定 …… 28

Contents

1. Introduction 1
2. Stampi: A Message Passing Library for Distributed Parallel Computing 3
2.1 System Configuration 3
2.2 Target Parallel Computers 4
3. Usages of Stampi 5
3.1 Starting-Up for Stampi(User's Setup) 5
3.2 Programming using Stampi 6
3.2.1 Fortran: jmpif90 6
3.2.2 C: jmpicc 7
3.3 Invoking: jmpirun 7
4. Basic Programming with Stampi 8
4.1 Dynamic Process Creation and Communications 8
4.2 Info Object and its Treatment 12
4.3 Process Creation using Info Object 13
5. Summary 15
Reference 16
Appendix A Stampi Library Interfaces 18
Appendix B Compile and Link Options for each Machines on COMPACS 20
B.1 Hitachi SR2201 20
B.2 Fujitsu VPP300 21
B.3 IBM SP/2 22
B.4 CRAY T94 23
B.5 NEC SX4 24
B.6 SGI Onyx 25
Appendix C Examples of .rhosts on COMPACS 26
C.1 hi00011 (Hitachi SR2201) 26
C.2 fu00011 (Fujitsu VPP300) 26
C.3 ibmsp50 (IBM SP/2) 27
C.4 cr00011 (CRAY T94) 27
C.5 ne00011 (NEC SX4) 28
C.6 ccsemga (SGI Onyx) 28

1 はじめに

計算機ならびにネットワーク技術の発展に伴い, ベクトル並列計算機, スカラ並列計算機等種々のアーキテクチャの並列計算機を同時に利用し高速で大規模な科学技術計算が可能となってきている. 原研においても計算科学技術推進センターに 5 台の並列計算機, 東海研, 那加研, 関西研にそれぞれ高性能の並列計算機を計 10 数台保有しており, それら並列機は計算物理分野を中心として所内外の研究者によって利用されている (図 1.1 参照).

これら各種並列計算機にはアーキテクチャに応じて有効・効率的に働く問題対象がある. 流体などの大規模な問題で連続アドレスへのアクセスが主要部分を占める場合にはベクトル計算機が効率的である. また有限要素法などに多く見られる間接参照や局所的なメモリ領域へのアクセスが多い場合にはスカラ計算機が有効である. 原研が開発中のプラズマシミュレーション [1], 翼とその回りの流体の連成計算 [2] ではプログラムの一部がベクトル計算機上で効率がよく別の部分ではスカラ計算機上で効率がよいという特色を持つ. そのため単一の並列計算機上で実行した場合には必ず非効率な部分が生じ, 全体の実効性能を下げてしまう. この様なプログラムの各部分を効率のよい並列計算機上で計算することにより, 単一並列計算機以上の高速な計算が可能となる. またグラフィックスワークステーションを用いたデータの可視化を行う場合, 数値実験には高速な並列計算機を使用し可視化にグラフィックスワークステーションを用いることが普通であろう. この様に複数の計算機資源の中からハードウェア的に効率的な組合わせで一つの計算を簡単に行うことをメタコンピューティングなどと呼び近年非常に注目されている計算パラダイムである. 複数の計算機を連携させて計算を行うためには相互にデータをやり取りするための通信手段を確保しなくてはならない. そして通信のためのインターフェイスが汎用的な通信ライブラリとして使用する全ての機種で利用可能でなくてはならない.

現在単一並列計算機上で動作する通信ライブラリは NX[3], MPL[4], p4[5], PVM[6] など多数提供されているが, 事実上業界標準といえる MPI(Message Passing Interface)[7, 8] が並列計算のための最も一般的な通信ライブラリであろう. MPI は多くの並列計算機上で実装されており MPI を用いて開発されたプログラムは移植性が非常に高いことが知られている. しかしながら異なる並列計算機を連携させて計算を行う場合, それら異機種並列計算機間での通信が既存 (ベンダー提供の意味で) の MPI には備わっておらず, それを支援するための通信ライブラリを新たに準備し, MPI とは別の通信手順をプログラム中に記述する必要が生じる. 既存の並列プログラムも多くが MPI で記述されており, それらプログラムを改編することなく MPI 単一の通信セマンティクスを保った形で異機種並列計算機を用いた計算が行えることが望ましい.

このような現状の計算機環境を背景として, 我々は MPI を通信のための単一ユーザインターフェイスとして異なる並列計算機を同時利用可能とする異機種並列計算機間通信ライブラリ Stampi を開発した [9, 10]. Stampi は日本原子力研究所計算科学技術推進センター所有の並列計算機群 COMAPCS 上で実装され, 任意の計算機を組み合わせた計算が可能となっている.

本報告書は従来 MPI を利用し並列プログラムを作成してきた利用者を対象に, Stampi ユーザーズガイドとして Stampi の利用法を中心に編纂したものである. 2 章では Stampi の概要について説明する. 3 章では Stampi 利用方法として, 利用者セットアップならびに各種コマンド群について説明する. 4 章では異機種並列計算の基本プログラミングとしてマネージャー / ワーカープログラムの例を挙げて解説する. また基本的なライブラリ関数の用例についても例示する. 付録には Stampi の全ライブラリインターフェイス, COMPACS での各種設定例などを付記した.

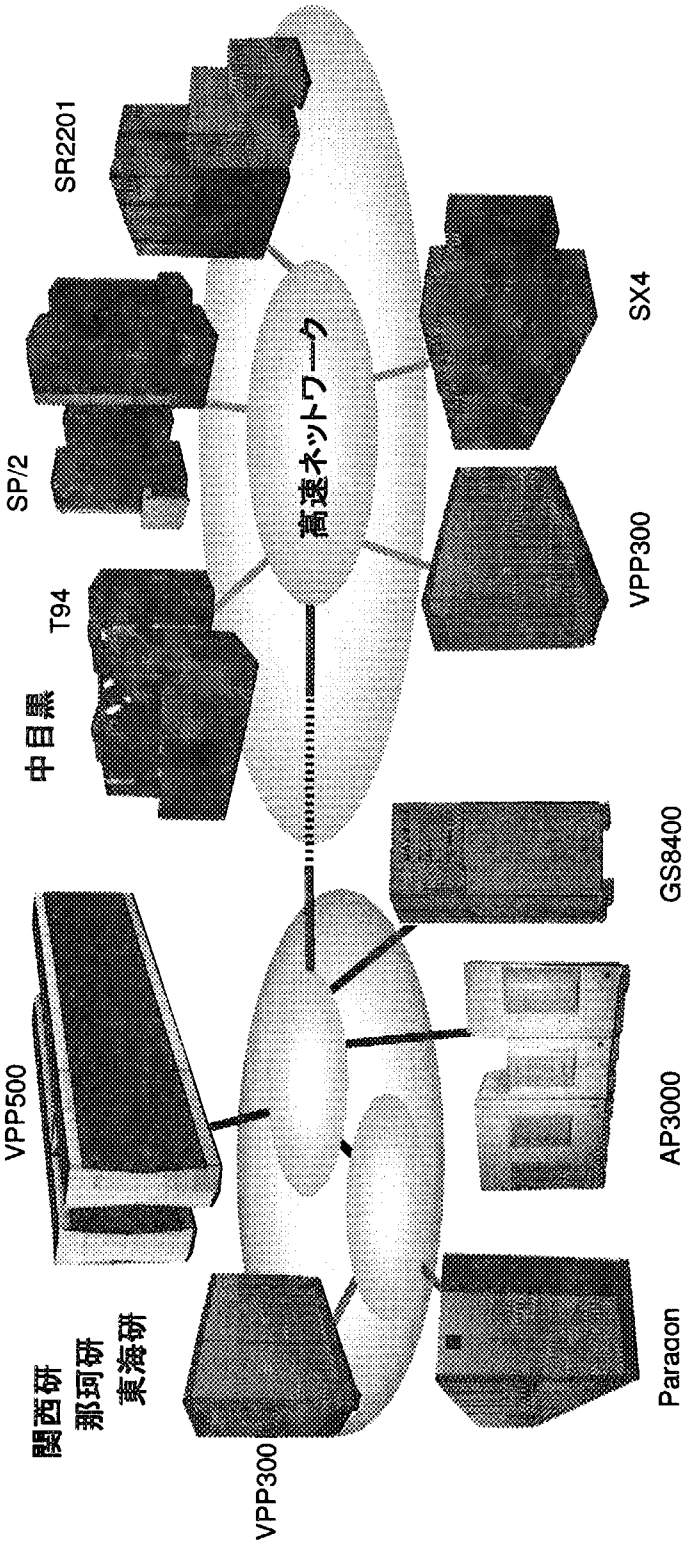


図 1.1: 日本原子力研究所 並列計算機の概要 (1998 年 9 月 現在)

2 異機種並列計算機間通信ライブラリ Stampi

2.1 Stampi システム構成

Stampi のシステム構成を, 図 2.1に示す. Stampi は, MPI (Message Passing Interface) のユーザインターフェイスを用いて異なる並列計算機で稼働する利用者プログラム間で通信を行う機能を提供する.

利用者は, Fortran(77/90) 言語または C 言語で記述した利用者プログラムをコンパイルし, 本システムが提供するライブラリとリンクすることで, 実行形式の利用者プログラムを作成する. 利用者プログラムは, プログラムを実行するそれぞれの計算機に用意しておく. その後, 本システムが提供する異種 MPI 起動コマンドを使用して利用者プログラムの実行を行うことにより, 複数の計算機間で MPI 通信を用いた計算が可能になる.

Stampi による単一 MPI プロセス内 (並列計算機内) の通信はベンダー提供の MPI ライブラリを利用し, 外部との通信には TCP/IP を利用する. 場合によっては機種にまたがる MPI プロセス同士の直接通信が不可能のため, ルーターを適時置きデータの中継をする.

また従来各並列計算機で微妙に異なっていたコンパイルコマンド, オプション, 実行コマンドの差異を統一した異機種共通コマンド群を提供し異機種環境下におけるプログラミングの支援を行う.

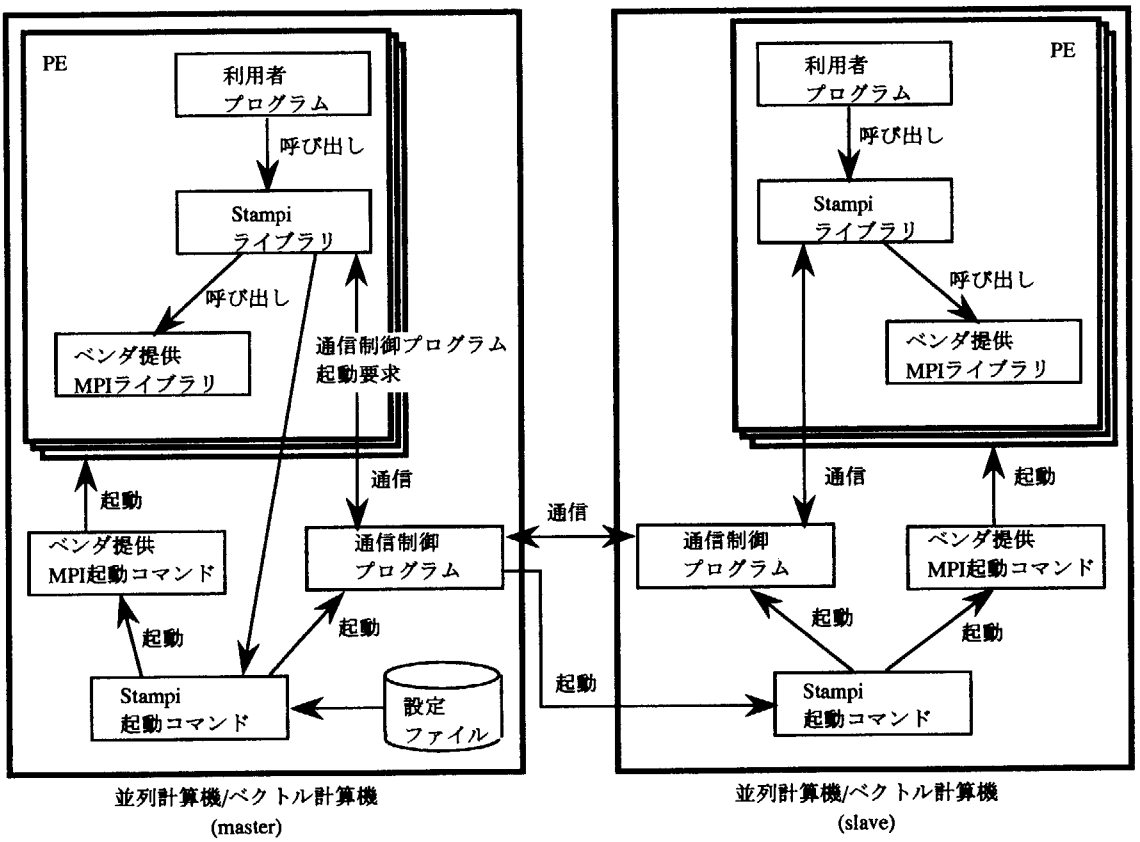


図 2.1: Stampi システムの構成

2.2 Stampiが対象とする並列計算機

Stampiは原研計算科学技術推進センター(以下CCSE)に設置された以下に示す5台の並列計算機とグラフィックスワークステーション(以下COMPACS)を対象に開発されている．ただし，それ以外の高性能計算機への移植を考慮してライブラリの内部構造は計算機に依存する部分と非依存の部分を明確に切り分けたつくりになっている．

実装並びに運用面からStampiの利用可能性を言及するならば，StampiはMPIでの中核となる部分は既存のMPIライブラリを利用することが前提になっており，また異機種間の認証はUNIXでのrshコマンドを利用する．従って一般的にUNIXと呼ばれるOSで運用されている並列計算機でベンダーもしくはフリーのMPIが実装されておりrshなどの遠隔利用コマンドを制限しないポリシーの元で運用されている計算機(間)で利用可能といえる．

移植に関して特記すべきことは，SGIのOnyx上ではベンダー提供のMPIではなくアルゴンヌ研究所(米国)が開発したフリーソフトMPICH[11]を利用している点である．つまりMPICHが利用可能なワークステーションやSMP，パーソナルコンピュータ(クラスタなども含む)へのStampi移植は基本的に困難ではない．従って並列計算機のみならず適用範囲は広いと開発者らは考えている．

3 Stampiの利用方法

異機種並列計算機間ライブラリ StampiはCCSEに設置されたCOMPACS上で実装されたものであるが, 他機種への移植等も考慮して本節の利用方法については特にCOMPACSに限定しないで一般的な説明を行う. COMPACSの6台の並列計算機個々に限定した設定方法などは巻末の付録B,Cに掲載する.

3.1 Stampiを利用する前に (利用者セットアップ)

Stampiは通信ライブラリのみならず, これまで各並列計算機で異なっていたコンパイルコマンド, コンパイラのオプション指定やライブラリの指定方法, 並列プログラム実行コマンドを統一したStampi用異機種共通コマンド群を用意しており, 複数の並列計算機間で殆んど機種依存のないプログラム開発が可能となっている. これらStampi用コマンド群を利用するためには各並列計算機上で適切なpathを設定しなくてはならない. インストール時の設定によって異なるが, デフォルトでは/usr/local/stampi/bin (現在のCCSEでは/home01/g0186/j3559/JMPI3/bin) に実行コマンド類が格納されているので, .cshrcなどにpathを追加する設定を行うとよい[1].

- Cシェルを利用する場合 (.cshrc等に)
  `set␣path=(Stampi実行ファイル群が格納されているディレクトリ␣$path)`

- Bornシェルを利用する場合 (.profile等に)
  `PATH=Stampi実行ファイル群が格納されているディレクトリ:$PATH;export␣PATH`

/usr/local/stampiディレクトリの下には図3.1に示すディレクトリが作成されている.

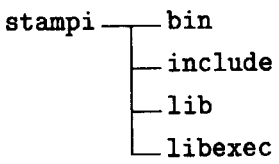


図 3.1: Stampiシステムディレクトリの構成

各ディレクトリの説明を表3.1に示す.

表 3.1: Stampiシステムディレクトリの説明

#	ディレクトリ	説明
1	stampi	Stampiシステムディレクトリ
2	stampi/bin	利用者が実行するコマンドを格納する
3	stampi/include	コンパイルに必要なインクルードファイルを格納する
4	stampi/lib	リンク時に使用するライブラリを格納する
5	stampi/libexec	システムが起動するコマンドを格納する

[1] .cshrcを変更した場合にはsource ~/.cshrc;rehashを実行するか, 再度loginし直す必要がある.

さらにStampiでは内部処理で rsh コマンドを使用するため, 並列計算機間で rsh が有効になるよう各自ホームディレクトリ上.rhosts の設定が不可欠である. 付録 C に CCSE での .rhosts の設定例を掲載する. .rhosts による設定が有効になっているかどうかは, 以下のコマンドを実行することによって確認できる (機種, OS によっては rsh の代わりに remsh を使用する). コマンドを実行して "Permission denied." などのエラーが発生する場合は, .rhosts を見直すこと[2].

```
% rsh␣ホスト名␣-l␣ユーザ ID␣ls
```

3.2 Stampiを用いたプログラム開発

3.2.1 Fortranでの開発: jmpif90

Stampiを用いたFortran(本書では基本的にFortran 90を想定している)プログラムには, 通常のMPIでのプログラミング同様に手続きの先頭 (IMPLICIT文, USE文を使用する場合はその直後) に以下のINCLUDE文を記述しなくてはならない.

```
    INCLUDE "mpif.h"
```

またコンパイル・リンクを行うために異機種共通コマンドjmpif90を用いる. jmpif90はMPIをコンパイルするために必要だったオプションやインクルードパス, ライブラリパスの設定を全てコマンド内で吸収しているため, マシン個々で大きく異なっていたMPIに関する指定を省略できかつ同一のコマンドでMPIプログラムの開発が行えるようになった. MPI以外のコンパイルオプション指定 (最適化など) は従来通りに行うことができる. Makefileを用いる場合には, マクロの再定義 (FC=jmpif90) を行うとよい.

- コンパイルの例

```
% jmpif90␣-c␣foo.f
```

- リンクの例

```
% jmpif90␣-o␣foo␣foo.o␣bar.o
```

FORTRAN 77コンパイラを用いる場合にはjmpif90の様な異機種共通コマンドを用意していないが, 従来MPIのコンパイルに必要だったオプションにStampiのインクルードパス, ライブラリパス等を追加することで可能である. 注意として所謂FORTRAN 77でStampiを使用するためには, INCLUDE文が有効であり, かつ, 変数名の命名法の制限 (英数文字で6文字以下, アンダースコアが利用できないなど) がFortran90並みに緩和されていなくてはならない. COMPACS各機種で必要なオプションを付録Bに掲載しておく.

[2] .rhosts の記述が正しいにも関わらずrsh が受け付けられない場合には, まず .rhosts の実行属性が600(-rw-------)かを確認すること. それでも正しく行かない場合には telnet でそのホストにログインし who コマンドで表示されるログイン元のアドレスと .rhosts の記述が一致しているかを確認すること. しばしば経路によって使用するネットワークデバイスが違うため, ホスト名が通常のホスト名と異なって識別される場合ある.

3.2.2 C言語での開発: jmpicc

Stampiを用いたCプログラムには, 通常のMPIでのプログラミング同様にファイルの先頭で以下のヘッダファイルをincludeしなくてはならない.

```
#include "mpi.h"
```

C言語についてもFortran90と同様に, 異機種共通コマンドjmpiccを用いてコンパイル・リンクを行う. MPI以外のコンパイルオプション指定(最適化など)は従来通りに行うことができる. Makefileを用いる場合には, マクロの再定義(CC=jmpicc)を行うとよい.

```
• コンパイルの例
% jmpicc␣-c␣foo.c

• リンクの例
% jmpicc␣-o␣foo␣foo.o␣bar.o
```

尚, C++言語のサポートはしない.

3.3 Stampiでのプログラム実行: jmpirun

Stampiを用いて作成したプログラムの実行は, jmpirunコマンドを使用して行う[3].
jmpirunコマンドの構文は以下の通り([]は省略可能を表す).

```
jmpirun␣-np␣ノード数␣[-part␣パーティション]␣コマンド及びパラメタ
```

ノード数は, 起動するMPIプロセスの数を表す[4]. コマンド及びパラメータには, 利用者プログラムと(もしあれば)パラメータを指定する.

jmpirunの使用例を以下に示す. この例では, fooコマンドをノード数3で起動する.

```
% jmpirun -np 3 foo
```

尚, この例はコマンドラインからの起動であるが, NQS (SP2ではLoadLeveler)でバッチ実行する場合は, jmpirunコマンドをバッチキューに投入するシェルスクリプトに記述する. 特にVPPシリーズでは, shellからMPIコマンドを実行することができないため注意が必要である.
バッチジョブの作成方法, 実行方法については, 各ベンダのNQSマニュアル(SP2ではLoadLevelerのマニュアル)を参照のこと.

[3] jmpirunを用いずにプログラムを実行した場合の動作は保証しない.
[4] パーティションはSR2201のパーティションを表す. SR2201以外の機種で指定した場合は, 無視される. SR2201で指定を省略した場合は, DEFPART環境変数の値が用いられる.

4 Stampiにおける基本的なプログラミング手法

Stampiの基本的なプログラミングは, 動的プロセス生成とそれによって生成されたプロセスとの間の通信で構成される. 本節ではStampiの基本的なプログラミング手法をFortranを中心に説明する.

4.1 動的プロセス生成と通信

動的プロセス生成とは実行中のMPIプロセスから新たに別のMPIプロセスを実行しグループ間コミュニケータを確立することをいう. このとき新たなMPIプロセスは同一並列計算機に限らず, 異なる並列計算機上に生成することができる. そしてMPI2のセマンティックスの範囲内で相互通信が可能である. ベンダー提供のMPI2ライブラリではプロセス生成は自社計算機内に制限されており, 他社の計算機へのプロセス生成及び相互通信はStampiで実現可能な機能である.

動的プロセス生成は次に示すMPI2の`MPI_Comm_spawn`関数を利用する.

- Fortranインターフェイス

```
MPI_COMM_SPAWN(command, argv, maxporcs, info, root, comm, intercomm,
               array_of_errcodes, ierror)
CHARACTER*(*) command, argv(*)
INTEGER maxprocs, info, root, comm, intercomm, array_of_errcodes(*), ierror
```

- C言語インターフェイス

```
int MPI_Comm_spawn(char *command, char *argv[], int maxprocs, MPI_Info info,
                   int root, MPI_Comm comm, MPI_Comm *intercomm,
                   int array_of_errcodes[]);
```

○ 引数

入力	*command*	文字型	生成する外部プロセスのプログラム名
入力	*argv*	文字型配列	*command*への引数
入力	*maxprocs*	整数型	生成するプロセスの数
入力	*info*	ハンドル	(key,value)組による情報の集合によって, プロセス生成に関する詳細を指示する
入力	*root*	整数型	プロセス生成に関する前述の引数が有効なランク
入力	*comm*	ハンドル	プロセス生成を含むグループ内コミュニケータ
出力	*intercomm*	ハンドル	旧グループと生成されたグループ間のコミュニケータ
出力	*array_of_errcodes*	整数型配列	プロセス毎の戻り値

次の3つの引数項についてはMPI予約定数を指定することができる.

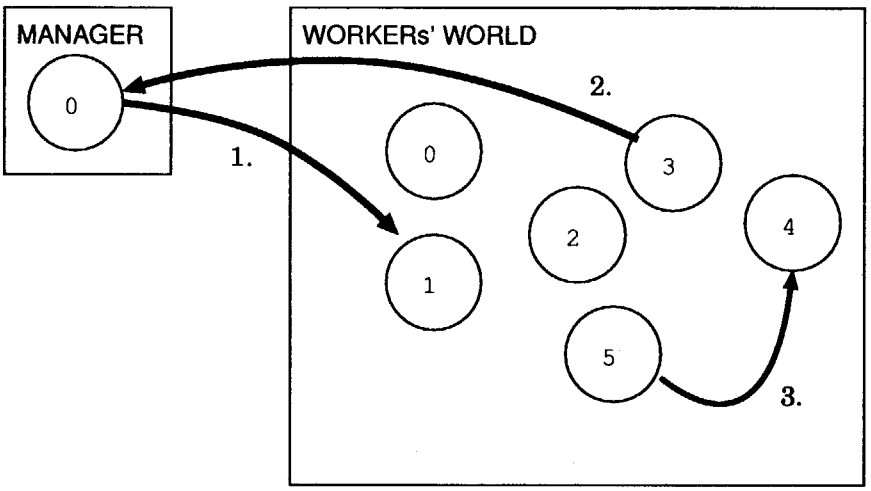
argv	`MPI_ARGV_NULL`	外部プロセスに引数なし
info	`MPI_INFO_NULL`	Infoオブジェクトによる詳細指定をしない
array_of_errcodes	`MPI_ERRCODES_IGNORE`	プロセス毎の戻り値(エラー)を返さない

*intercomm*には新たに生成された外部プロセスグループとのグループ間コミュニケータが格納される. このグループ間コミュニケータはMPIで規定されたグループ間コミュニケータと同等の機能を有するが, 現バージョンのStampiでは基本データ型に対する1対1通信以外の機能についてはサポートしない. 同様にして機種間を跨いだ形のグループの定義, コミュニケータの作成はできない.

MPI_Comm_spawn を用いた最も簡単な例として, マネージャによるワーカー制御プログラムを示す[5].

```
1  • マネージャープログラム
2        program manager
3        include 'mpif.h'
4        integer world_size, universe_size, universe_sizep, everyone
5        character*100 worker_program
6  c
7        call MPI_Init(ierr)
8        call MPI_Comm_size(MPI_COMM_WORLD, world_size, ierr)
9        if(world_size.ne.1) call errer("Top heavy with management")
10 c
11       write(6,*) 'How many processes total?'
12       read(5,*)   universe_size
13       if(universe_size.eq.1) call error("No room to start workers")
14 c
15       call choose_worker_program(worker_program)
16       call MPI_Comm_spawn(worker_program, MPI_ARGV_NULL, universe_size-1,
17      &        MPI_INFO_NULL, 0, MPI_COMM_SELF, everyone,
18      &        MPI_ERRCODES_IGNORE, ierr)
19 c
20 c  並列コードをここに記述する. コミュニケータ "everyone"は生成された
21 c  プロセス (グループ間コミュニケータ "everyone"におけるリモートグループ内で
22 c  ランク 0, ... universe_size-2) 間との通信に利用される.
23 c
24       call MPI_Finalize(ierr)
25       end
26 • ワーカープログラム
27       program worker
28       include 'mpif.h'
29       integer size, parent
30 c
31       call MPI_Init(ierr)
32       call MPI_Comm_get_parent(parent, ierr)
33       if(parent.eq.MPI_COMM_NULL) call error("No parent!")
34       call MPI_Comm_remote_size(parent, size, ierr)
35       if(size.ne.1) call error("Something's wrong with the parent")
36 c
37 c  並列コードをここに記述する.
38 c  マネージャーは (リモートグループの)parent のランク 0 で表される.
39 c  もしワーカー間で通信を行う必要があるときには, MPI_COMM_WORLD を利用する.
40 c
41       call MPI_Finalize(ierr)
42       end
```

[5]これは MPI2 仕様書 [8] の 92 頁に C 言語で書かれたソースコードをもとに主要部分のみを Fortran に翻訳したものである.



• 1の場合
マネージャー :: call MPI_Send(mes,len,MPI_INTEGER,1,tag,everyone,ierr)
ワーカー [1] :: call MPI_Recv(mes,len,MPI_INTEGER,0,tag,parent,status,ierr)

• 2の場合
ワーカー [3] :: call MPI_Send(mes,len,MPI_INTEGER,0,tag,parent,ierr)
マネージャー :: call MPI_Recv(mes,len,MPI_INTEGER,3,tag,everyone,status,ierr)

• 3の場合
ワーカー [5] :: call MPI_Send(mes,len,MPI_INTEGER,4,tag,MPI_COMM_WORLD,ierr)
ワーカー [4] :: call MPI_Recv(mes,len,MPI_INTEGER,5,tag,MPI_COMM_WORLD,status,ierr)

図 4.1: マネージャー / ワーカー間の通信

本プログラムはマネージャー, ワーカーがそれぞれ別の実行オブジェクトからなる MPMD の処理方式をとるものである. マネージャーがワーカーのプロセス数の入力を受け, 起動すべきプログラムを実行中に決定し (15 行目 choose_worker_program) 起動をかける. この例の場合ワーカーはマネージャーと同じ計算機上で起動される.

通信相手の指定は '(コミュニケータ, グループ内ランク)' で行われる. プログラム中のコメントにもあるように, マネージャー側からワーカーへの通信は MPI_Comm_spawn を呼んだ際に, 格納されたグループ間コミュニケータeveryone を用いて行う. 逆にワーカー側からマネージャーへの通信を行うためには, 32 行目にある MPI_Comm_get_parent を呼び出し, 自分自身を spawn した親のグループとのグループ間コミュニケータ (parent) を取得しそれを用いて通信を行わなくてはならない. ワーカー同士の通信にはワーカー世界でのMPI_COMM_WORLD を用いればよい (図 4.1 参照). ここで, マネージャー側とワーカー側のMPI_COMM_WORLD はそれぞれのグループ全体を表現するコミュニケータとして存在する. プログラムの字面上は同一のMPI_COMM_WORLD が別の実体として複数存在することに注意されたい.

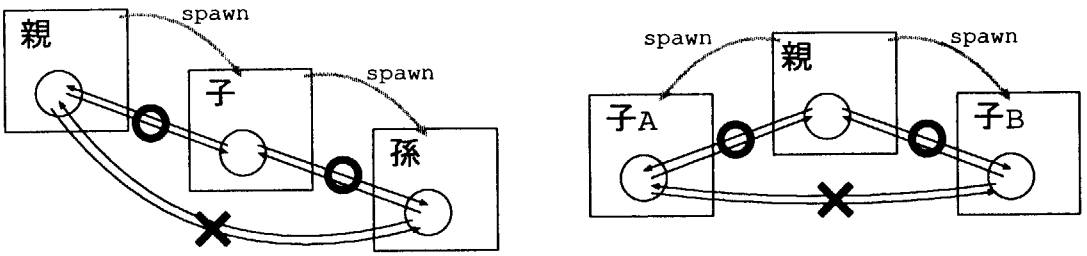


図 4.2: グループ間通信の制約 (左. 親 – 子 – 孫, 右. 親 – 子 A& 子 B)

また, 親が外部プロセス (子 A) を生成した後, 更に別の外部プロセス (子 B) を生成したり, 子の外部プロセスから更に孫の外部プロセスを生成することも可能である. ただし, 子 A から子 B へ直接通信したり親から孫へ直接通信することは MPI のセマンティクス上出来ない. 必ず間にある MPI プロセスを経由しなくてはならない (図 4.2 参照).

親グループとのグループ間コミュニケータの取得

- Fortran インターフェイス
  `MPI_COMM_GET_PARENET(`*parent, ierror*`)`
  `INTEGER` *parent, ierror*
- C 言語インターフェイス
  `int MPI_Info_create(MPI_Comm *`*parent*`)`

○ 引数

出力	*parent*	ハンドル	親グループ間のコミュニケータ

もし spawn されてないプロセスがこの関数を呼び出した場合には, *parent* には`MPI_COMM_NULL` が返却される. MPMD でなく SPMD で異機種の制御を行う場合には`MPI_Comm_get_parent` の戻り値をもとに if 文などで分岐をかける方法が有効である.

また, 親グループのプロセス数は 34 行目にある様に `MPI_Comm_remote_size` を利用して得ることができる.

親グループのプロセス数取得

- Fortran インターフェイス
  `MPI_COMM_REMOTE_SIZE(`*parent, size, ierror*`)`
  `INTEGER` *parent, size, ierror*
- C 言語インターフェイス
  `int MPI_Comm_remote_size(MPI_Comm *`*parent*`, int *`*size*`)`

○ 引数

入力	*parent*	ハンドル	親グループ間のコミュニケータ
出力	*size*	整数型	親グループ間のプロセス数

4.2 Info オブジェクトの取り扱い

次に Info オブジェクトを使った外部プロセスの詳細情報の指定方法を説明するが, その前に Info オブジェクトの取り扱いについて解説する. Info オブジェクトは (key,value) 組の情報を格納するためのリスト構造体であり, MPI2 の幾つかの関数で情報の引用に用いられる. 利用者は Info オブジェクトにアクセスするための以下に示すような特別な関数を呼び出さなくてはならない.

Info オブジェクトの生成

- Fortran インターフェイス
  `MPI_INFO_CREATE(`*info*`, `*ierror*`)`
  `INTEGER` *info*`, `*ierror*
- C 言語インターフェイス
  `int MPI_Info_create(MPI_Info *`*info*`)`

○ 引数

出力	*info*	ハンドル	生成された Info オブジェクト

Info オブジェクトへの情報の登録

- Fortran インターフェイス
  `MPI_INFO_SET(`*info*`, `*key*`, `*value*`, `*ierror*`)`
  `INTEGER` *info*`, `*ierror*
  `CHARACTER*(*)` *key*`, `*value*
- C 言語インターフェイス
  `int MPI_Info_set(MPI_Info` *info*`, char *`*key*`, char *`*value*`)`

○ 引数

入出力	*info*	ハンドル	Info オブジェクト
入力	*key*	文字型	登録する情報の key
入力	*value*	文字型	key に対する value

Info オブジェクトの情報参照

- Fortran インターフェイス
  `MPI_INFO_GET(`*info*`, `*key*`, `*valuelen*`, `*value*`, `*flag*`, `*ierror*`)`
  `INTEGER` *info*`, `*valuelen*`, `*ierror*
  `CHARACTER*(*)` *key*`, `*value*
  `LOGICAL` *flag*
- C 言語インターフェイス
  `int MPI_Info_get(MPI_Info` *info*`, char *`*key*`, int` *valuelen*`, char *`*value*`, int *`*flag*`)`

○ 引数

入出力	*info*	ハンドル	Info オブジェクト
入力	*key*	文字型	参照する情報の key
入力	*valuelen*	整数型	value に格納可能な文字列長
出力	*value*	文字型	key に対する value
出力	*flag*	論理型	key が定義されているかのチェック

4.3 Info オブジェクトを使った外部プロセス生成

Info オブジェクトを用いて次の条件でプログラムを生成する例を示す.

- hi00011 という計算機上に
- /home001/g0188/j0000/test ディレクトリ上の
- sample01 というプログラムを
- /home001/g0188/j0000/run というディレクトリにおいて
- NQS のqueX というクラス, partX というパーティションで
- プロセス数 3 個分

• Info オブジェクトを使った例

```
      call MPI_INFO_CREATE(info, ier)
      call MPI_INFO_SET(info, "host", "hi00011",ier)
      call MPI_INFO_SET(info, "path", "/home001/g0188/j0000/test", ier)
      call MPI_INFO_SET(info, "wdir", "/home001/g0188/j0000/run", ier)
      call MPI_INFO_SET(info, "nqsq", "queX", ier)
      call MPI_INFO_SET(info, "part", "partX", ier)
      call MPI_COMM_SPAWN("sample01", MPI_ARGV_NULL, 3, info,
     1         0, MPI_COMM_WORLD, icomm, MPI_ERRORCODES_IGNORE, ier)
      if(ier.ne.MPI_SUCCESS)then
            write(*,*) "Spawn failed"
            stop 1
      endif
       ......
```

この例ではプログラムに対してパラメータ群をハードコーディングしてしまうのでそれらの変更が必要になった場合には, ソースの再コンパイルをしなくてはならない. パラメータを頻繁に変更する場合には, それらを file を介してシステムに通知する仕組みが MPI2 では用意されている. Info オブジェクトのfile キーに対してファイル名を指定することでなされる.

• info ファイルを使った例

```
      call MPI_INFO_SET(info, "file", "info",ier)
      call MPI_COMM_SPAWN("", MPI_ARGV_NULL, 0, info,
     1         0, MPI_COMM_WORLD, icomm, MPI_ERRORCODES_IGNORE, ier)
```

• info ファイルの内容

```
      host: hi00011
      path: /home001/g0188/j0000/test
      -cmd: sample01
      wdir: /home001/g0188/j0000/run
      nqsq: queX
      part: partX
      -np:  3
```

info ファイルの書式は 1 行単位で

key : *value*

と Stampi では決めている[6]. 行頭に# が付くものはコメントと見なされる.

外部プロセス起動に有効な Info オブジェクトのキー, 並びに info ファイルに指定可能な情報は表 4.1, 表 4.2に示す通りである. 尚これら以外の情報が記述されている場合にはその情報は無視される.

また外部プロセス起動での同一キーに対する情報の優先順位は

info ファイル > Info オブジェクト > MPI_Comm_spawn への引数 > デフォルト

である.

表 4.1: 外部プロセス起動を制御する情報

#	キー	説明
1	host	プロセス生成先ホスト
2	user	プロセスを実行するユーザ
3	wdir	プロセスを実行するディレクトリ
4	path	コマンドサーチパス
5	part	プロセスを実行するパーティション (SR2201 のみ)
6	nqsq	NQS のキュー (指定しない場合, TSS で実行)
7	node	NQS で使用するプロセッサノード数 (省略時はプロセス数)
8	file	プロセス起動制御ファイル

表 4.2: info ファイルに記述できる情報

#	キー	説明
1	host	プロセス生成先ホスト
2	user	プロセスを実行するユーザ
3	wdir	プロセスを実行するディレクトリ
4	path	コマンドサーチパス
5	part	プロセスを実行するパーティション (SR2201 のみ)
6	nqsq	NQS のキュー (指定しない場合, TSS で実行)
7	node	NQS で使用するプロセッサノード数 (省略時はプロセス数)
8	-cmd	実行するコマンド及びパラメタ (MPI_Comm_spawn の指定を置き換える)
9	-np	MPI プロセスのプロセス数 (MPI_Comm_spawn の指定を置き換える)

[6]MPI2 では info ファイルの書式は実装依存にしているため他の MPI2 実装とは互換性がない

5 おわりに

異機種並列計算機間通信ライブラリ Stampi は, 従来のベンダー提供の MPI ライブラリでは実現されていなかった異機種間の通信を MPI インターフェイスのみで可能とする世界的にも初めての MPI2 実装系である. 本報告書はユーザズガイドとして, Stampi の利用法, 異機種共通コマンドならびに関数仕様について説明した.

すでに Stampi を使用した異機種並列プログラムの成果が出始めており, 新たな計算機の利用法が展開されつつあるといえよう. 今後 COMPACS 以外にも移植を進めていく予定であり、各種ワークステーション (クラスタ), SMP(マルチプロセッサ) 機等での利用が可能になるであろう. 本報告書を通じて多くの研究者に Stampi を利用していただければ幸いである.

参考文献

[1] 徳田, 小出, 今村, 松本 ほか： トカマク・プラズマにおけるハイブリッド・シミュレーション, 日本物理学会 53 回年会, 1pP5 (1998).

[2] Kimura T. and Takemiya H. : Local Area Metacomputing for Multidiciplinary Problems: A Case Study for Fluid/Structure Coupled Similation, *Proc. of Int. Conf. on Supercomputing*, pp. 149–156 (1998).

[3] Intel Corporation : *Paragon System Fortran Calls Reference Manual* (1995).

[4] 日本IBM : *Parallel Environment for AIX, MPL Programming and Subroutine Reference*, Version 2 Release 1, IBM manual GS23-3893-00 (1995).

[5] Butler R. and Lusk E. : Monitors, messages and clusters: The p4 parallel programming system. Tech. Rep. MCS-P362-0493, Argonne National Laboratory (1993).

[6] Geist A., Beguelin A., Dongarra J., Jiang W., Manchek R. and Sunderam V. : *PVM: A User's Guide and Tutorial for Networked Parallel Computing*, The MIT Press (1994). `http://www.netlib.org/pvm3/book/pvm-book.html`

[7] Message Passing Interface Forum : *MPI: Message passing interface standard* (1996). `http://www.mpi-forum.org/`
同日本語訳は `http://www.ppc.nec.co.jp/mpi-j/mpi-report-j.ps.Z` より入手可能.

[8] Message Passing Interface Forum : *MPI-2: Extensions to the message-passing interface* (1997). `http://www.mpi-forum.org/`

[9] 小出, 今村, 太田, 川崎 ほか：並列分散科学技術計算環境 STA(3) — 異機種並列計算機間ライブラリの構築, 計算工学会講演論文集, Vol. 3, No. 1, pp. 81–84 (1998).

[10] 小出, 今村, 武宮, 平山：異機種並列計算機間の通信を支援する並列分散通信ライブラリ: Stampi, 日本流体力学会年会'98 講演論文集, pp. 413–414 (1998).

[11] Gropp W. and Lusk E. : *User's Guide for mpich, a Portable Implementation of MPI* (1998). `http://www-c.mcs.anl.gov/mpi/mpich/`

[12] Gropp W., Lusk E. and Skjellum A.: *Using MPI, Portable Parallel Programming with the Message-Passing Interface*, The MIT Press (1996).

[13] 日立製作所: *HI-UX/MPP MPI·PVM* 使用の手引き, 日立ソフトウェアマニュアル 6A20-3-026-10 (1998).

[14] 富士通株式会社: *UXP/V MPI* 使用手引書 *V11* 用, 富士通マニュアル J2U5-0271-01 (1998).

[15] 日本IBM: *Parallel Environment for AIX: MPI Programming and Subroutine Reference*, Ver. 2, Rel. 1, IBM manual GC23-3894-00 (1995).

[16] CRAY Research Inc.: *Message Passing Toolkit: Release Overview*, CRAY manual RO-5290 1.1 (1996).

[17] CRAY Research Inc.: *Message Passing Toolkit: MPI Programmer's Manual*, CRAY manual SR-2197 1.1 (1996).

[18] 日本電機株式会社: *SUPER-UX MPI/SX*利用の手引き, 日本電気マニュアル G1AF09-1 (1996).

付録 A Stampi ライブラリインターフェイス

Stampi がサポートする MPI 関数を表 1.1に示す。一番右のカラムは外部通信への対応範囲を表している。○印が付いたものは外部通信に対応している (内部通信にも対応している). △印は現在外部通信はサポート対象外であるが今後拡張の対象となっているものである. ×印のものは内部通信のみサポートする. ここで外部通信に対応するとは, 外部に生成したグループとのグループ間コミュニケータならびにそれから派生する通信のリクエストや派生データ型, Info オブジェクトなどを通信関数の引数に指定できることをいう.

また, 本表にないものであっても MPI の関数群はグループ内通信の範囲内で利用可能である. MPI または MPI2 の仕様ならびに使用方法については参考文献 [7, 8, 12], 各社利用マニュアル [13, 14, 15, 17, 18, 11] を参考にされたい.

表 1.1: Stampi 関数一覧

#	関数	説明	外部通信対応
1	MPI_Abort	コミュニケータ内のプロセスの実行を中断する.	○
2	MPI_Allreduce	コミュニケータ内の前プロセスで大域的なリダクション操作 (総和, 最大値, 最小値, 論理演算など) を行い, 結果を全プロセスに返却する.	△
3	MPI_Barrier	コミュニケータ内の全プロセスでバリア同期を行う.	○
4	MPI_Bcast	コミュニケータ内の 1 つのプロセスから他の全プロセスに対してデータを送信 (または受信) する.	△
5	MPI_Cart_coords	コミュニケータ内のプロセスのランクを座標に変換する.	×
6	MPI_Cart_create	コミュニケータにカルテシアン (直積構造のトポロジ) 情報を付加した新しいコミュニケータを返却する.	×
7	MPI_Cart_shift	カルテシアン構造のトポロジにおいて, 座標方向へのデータシフト動作を支援する. 具体的には MPI_Sendrecv 関数に指定するランク (通信相手のコミュニケータ内のランク) を求める.	×
8	MPI_Comm_dup	コミュニケータの複製を返却する.	×
9	MPI_Comm_free	コミュニケータを解放する.	×
10	MPI_Comm_get_parent	親プロセスのコミュニケータを返却する. 親プロセスがない場合 MPI_COMM_NULL を返却する.	○
11	MPI_Comm_rank	コミュニケータ内での自プロセスのランクを返却する.	○
12	MPI_Comm_remote_size	グループ間コミュニケータでの他グループ内のプロセス数を返却する.	○
13	MPI_Comm_size	コミュニケータ内のプロセス数を返却する.	○
14	MPI_Comm_spawn	MPI プロセスを生成する.	○
15	MPI_Comm_split	コミュニケータ内のプロセスをサブグループに分割することによって, 新しいコミュニケータを定義する.	△
16	MPI_Finalize	MPI 終了関数.	○

17	MPI_Gather	コミュニケータ内の全プロセスから1つのプロセスにデータを収集し, ランクの順にデータを格納する.	△
18	MPI_Gatherv	コミュニケータ内の全プロセスから1つのプロセスに可変データを収集し, ランクの順にデータを格納する.	△
19	MPI_Get_count	MPI_Recv や MPI_Irecv で受信したデータ数を返却する.	○
20	MPI_Info_create	Info オブジェクトを生成する.	○
21	MPI_Info_delete	Info オブジェクトから情報を削除する.	○
22	MPI_Info_dup	Info オブジェクトの複製を作成する.	○
23	MPI_Info_free	Info オブジェクトを解放する.	○
24	MPI_Info_get	Info オブジェクトから key に対応する情報を返却する.	○
25	MPI_Info_get_nkeys	Info オブジェクトに登録されている情報の個数を返却する.	○
26	MPI_Info_get_nthkey	Info オブジェクトに登録された n 番目の情報を返却する.	○
27	MPI_Info_get_valuelen	Info オブジェクトに登録されてた key に対応する情報の大きさを返却する.	○
28	MPI_Info_set	Info オブジェクトに key に対応する情報を登録する.	○
29	MPI_Init	MPI 初期化関数.	○
30	MPI_Iprobe	ノンブロッキングでの受信確認をする.	○
31	MPI_Irecv	ノンブロッキング通信によって, 他プロセスからデータを受信する.	○
32	MPI_Isend	ノンブロッキング通信によって, 他プロセスにデータを送信する.	○
33	MPI_Probe	ブロッキングでの受信確認をする.	○
34	MPI_Recv	ブロッキング通信によって, 他プロセスからデータを受信する.	○
35	MPI_Reduce	コミュニケータ内の前プロセスで大域的なリダクション操作 (総和, 最大値, 最小値, 論理演算など) を行い, 結果を特定のプロセスに返却する.	△
36	MPI_Send	ブロッキング通信によって, 他プロセスにデータを送信する.	○
37	MPI_Sendrecv	ブロッキング通信によるデータの送受信を行う.	×
38	MPI_Type_commit	派生データ型の記憶操作を行う. 通信バッファの記述 (通信バッファの内容ではない) を記憶する.	×
39	MPI_Type_extent	派生データ型の大きさ (バイト数) を求める.	×
40	MPI_Type_hvector	等間隔に並んだデータ型のコピーから, 新しい派生データ型を生成する. データの間隔はバイト数で指定する.	×
41	MPI_Type_vector	等間隔に並んだデータ型のコピーから, 新しい派生データ型を生成する. データの間隔はデータの数で指定する.	×
42	MPI_Wait	ノンブロッキング通信の終了待ちをおこなう.	○
43	MPI_Wtime	ある時刻からの経過時間を返却する. ある時刻は利用者プログラムの開始から変更されない. 返却する時刻は MPI_Wtime を呼び出したノードにローカルである.	○

付録 B　COMPACS 各機種でのコンパイル・リンク オプション

日立 SR2201、富士通 VPP300、IBM SP、CRAY T94、NEC SX4、及び SGI Onyx の各機種について、C 言語 並びに Fortran 言語でのコンパイル・リンク手順を説明する。

B.1　日立 SR2201 でのコンパイル・リンク手順

コンパイル・リンクは、C コンパイルコマンド (cc)、Fortran コンパイルコマンド (f90) に以下のオプションを指定することによって行う。

- コンパイルオプション (C、Fortran 共通)

```
-I/usr/local/jmpi/include
```

- C 言語リンクオプション

```
-L/usr/local/jmpi/lib -ljmpi -L/usr/local/mpi/lib/hmpp2/cml -lpmpi -lmpi
```

- Fortran 言語リンクオプション

```
-L/usr/local/jmpi/lib -ljmpif -ljmpi
-L/usr/local/mpi/lib/hmpp2/cml -lpmpi -lfmpi -lmpi
```

以下にコンパイルの例を示す。以下の例では、foo.f をコンパイルし、foo.o を作成する。

```
% f90 -c -I/usr/local/jmpi/include foo.f
```

次に Fortran でのリンクの例を示す。この例では、foo.o と bar.o をリンクし、foo コマンドを作成する。

```
% f90 -o foo foo.o bar.o -L/usr/local/jmpi/lib -ljmpif -ljmpi
      -L/usr/local/mpi/lib/hmpp2/cml -lpmpi -lfmpi -lmpi
```

B.2 富士通 VPP300 でのコンパイル・リンク手順

コンパイル・リンクは、C コンパイルコマンド (cc)、Fortran コンパイルコマンド (frt) に以下のオプションを指定することによって行う。

- C 言語コンパイルオプション

```
-I/usr/local/jmpi/include
```

- Fortran 言語コンパイルオプション

```
-I/usr/local/jmpi/include -X9
```

- C 言語リンクオプション

```
-Wl,-P,-J,-dy -L/usr/local/jmpi/lib -ljmpi -L/usr/lang/mpi/lib -lpmpi
-lmpi -lmp -lelf -lpx -lsocket
```

- Fortran 言語リンクオプション

```
-Wl,-P,-J,-dy -L/usr/local/jmpi/lib -ljmpif -ljmpi -L/usr/lang/mpi/lib
-lpmpi -lfmpi -lmpi -lmp -lelf -lpx -lsocket
```

以下に Fortran でのコンパイルの例を示す。以下の例では、foo.f をコンパイルし、foo.o を作成する。

```
% frt -c -I/usr/local/jmpi/include -X9 foo.f
```

次に Fortran でのリンクの例を示す。この例では、foo.o と bar.o をリンクし、foo コマンドを作成する。

```
% frt -o foo foo.o bar.o -Wl,-P,-J,-dy -L/usr/local/jmpi/lib -ljmpif
      -ljmpi -L/usr/lang/mpi/lib -lpmpi -lfmpi -lmpi -lmp -lelf -lpx -lsocket
```

VPP の場合 -o オプションが最初に指定されないと正しいオブジェクトがリンクされないことあるので注意が必要である.

B.3 IBM SP/2 でのコンパイル・リンク手順

コンパイル・リンクは、C コンパイルコマンド (mpcc)、Fortran コンパイルコマンド (mpxlf) に以下のオプションを指定することによって行う。

- C 言語コンパイルオプション

```
-I/usr/local/jmpi/include
```

- Fortran 言語コンパイルオプション

```
-I/usr/local/jmpi/include -qintsize=4
```

- C 言語リンクオプション

```
-L/usr/local/jmpi/lib -ljmpi -L/usr/lpp/ppe.poe/lib -lmpi
```

- Fortran 言語リンクオプション

```
-L/usr/local/jmpi/lib -ljmpif -ljmpi -L/usr/lpp/ppe.poe/lib
-lmpi
```

以下に Fortran でのコンパイルの例を示す。以下の例では、foo.f をコンパイルし、foo.o を作成する。

```
% mpxlf -c -I/usr/local/jmpi/include -qintsize=4 foo.f
```

次に Fortran でのリンクの例を示す。この例では、foo.o と bar.o をリンクし、foo コマンドを作成する。

```
% mpxlf -qintsize=4 -o foo foo.o bar.o -L/usr/local/jmpi/lib -ljmpif
      -ljmpi -L/usr/lpp/ppe.poe/lib -lmpi
```

B.4 CRAY T94 でのコンパイル・リンク手順

CTAY T94 の場合、コンパイル前に mpt (Message Passing Toolkit[16]) を使用する準備として以下のコマンドを実行する。本操作は、一回のログインについて一回だけ行えばよい。(jmpicc、jmpif90 コマンドを使用する場合は不要)

```
module load mpt
```

コンパイル・リンクは、C コンパイルコマンド (cc)、Fortran コンパイルコマンド (f90) に以下のオプションを指定することによって行う。

- C 言語コンパイルオプション

```
-I/usr/local/jmpi/include
```

- Fortran 言語コンパイルオプション

```
-I/usr/local/jmpi/include -a taskcommon
```

- C 言語リンクオプション

```
-L/usr/local/jmpi/lib -ljmpi -L/opt/ctl/mpt/mpt/lib -lpmpi -lmpi
```

- Fortran 言語リンクオプション

```
-L/usr/local/jmpi/lib -ljmpif -ljmpi -L/opt/ctl/mpt/mpt/lib
-lpmpi -lmpi
```

以下に Fortran でのコンパイルの例を示す。以下の例では、foo.f をコンパイルし、foo.o を作成する。

```
% f90 -c -I/usr/local/jmpi/include -a taskcommon foo.f
```

次に Fortran でのリンクの例を示す。この例では、foo.o と bar.o をリンクし、foo コマンドを作成する。

```
% f90 -a taskcommon -o foo foo.o bar.o -L/usr/local/jmpi/lib -ljmpif
      -ljmpi -L/opt/ctl/mpt/mpt/lib -lpmpi -lmpi
```

B.5 NEC SX4 でのコンパイル・リンク手順

コンパイル・リンクは、C コンパイルコマンド (cc)、Fortran コンパイルコマンド (f90) に以下のオプションを指定することによって行う。

- C 言語コンパイルオプション

```
-I/usr/local/jmpi/include -h float0
```

- Fortran 言語コンパイルオプション

```
-I/usr/local/jmpi/include -P multi -float0
```

- C 言語リンクオプション

```
-L/usr/local/jmpi/lib -ljmpi -lmpi -lpthread
```

- Fortran 言語リンクオプション

```
-L/usr/local/jmpi/lib -ljmpif -ljmpi -lmpi -lpthread
```

以下に Fortran でのコンパイルの例を示す。以下の例では、foo.f をコンパイルし、foo.o を作成する。

```
% f90 -c -I/usr/local/jmpi/include -P multi -float0  foo.f
```

次に Fortran でのリンクの例を示す。この例では、foo.o と bar.o をリンクし、foo コマンドを作成する。

```
% f90 -P multi -float0 -o foo foo.o bar.o -L/usr/local/jmpi/lib -ljmpif
      -ljmpi -lmpi -lpthread
```

B.6 SGI Onyx でのコンパイル・リンク手順

コンパイル・リンクは、C コンパイルコマンド (cc)、Fortran コンパイルコマンド (f90) に以下のオプションを指定することによって行う。

- C 言語コンパイルオプション

```
-I/usr/local/jmpi/include
```

- Fortran 言語コンパイルオプション

```
-I/usr/local/jmpi/include
```

- C 言語リンクオプション

```
-L/usr/local/jmpi/lib -ljmpi -L/usr/local/mpi/lib/IRIX64/ch_p4 -lpmpi -lmpi
```

- Fortran 言語リンクオプション

```
-L/usr/local/jmpi/lib -ljmpif -ljmpi -L/usr/local/mpi/lib/IRIX64/ch_p4
-lpmpi -lfmpi -lmpi
```

以下に Fortran でのコンパイルの例を示す。以下の例では、foo.f をコンパイルし、foo.o を作成する。

```
% f90 -c -I/usr/local/jmpi/include foo.f
```

次に Fortran でのリンクの例を示す。この例では、foo.o と bar.o をリンクし、foo コマンドを作成する。

```
% f90 -o foo foo.o bar.o -L/usr/local/jmpi/lib -ljmpif
      -ljmpi -L/usr/local/mpi/lib/IRIX64/ch_p4 -lpmpi -lfmpi -lmpi
```

付録 C　COMPACS 各機での.rhosts の設定例

C.1　hi00011 (日立 SR2201) の設定

各自のユーザ ID でログインし, エディタで .rhosts ファイルを編集する. 内容は, 以下の通り.

```
hi00011 ユーザ ID
fu00011 ユーザ ID
ne00011 ユーザ ID
cr00011 ユーザ ID
ibmsp50a ユーザ ID
ccsemga ユーザ ID
desr01 ユーザ ID
localhost ユーザ ID
```

MPI プロセス起動先のホストとして localhost を指定しない場合, localhost のエントリは不要である.

尚, ユーザ ID は, 各自のユーザ ID に置き換えること.

C.2　fu00011 (富士通 VPP300) の設定

各自のユーザ ID でログインし, エディタで .rhosts ファイルを編集する. VPP シリーズの場合は, 各ノードプロセッサの設定も必要である. 内容は, 以下の通り.

```
fu00011 ユーザ ID
fu00112 ユーザ ID
fu00212 ユーザ ID
fu00312 ユーザ ID
fu00412 ユーザ ID
fu00512 ユーザ ID
fu00612 ユーザ ID
fu00712 ユーザ ID
fu00812 ユーザ ID
fu00912 ユーザ ID
fu01012 ユーザ ID
fu01112 ユーザ ID
fu01212 ユーザ ID
fu01312 ユーザ ID
fu01412 ユーザ ID
fu01512 ユーザ ID
hi00011 ユーザ ID
ne00011 ユーザ ID
cr00011 ユーザ ID
ibmsp50a ユーザ ID
ccsemga ユーザ ID
desr01 ユーザ ID
localhost ユーザ ID
```

MPI プロセス起動先のホストとして localhost を指定しない場合, localhost のエントリは不要である.

尚, ユーザ ID は, 各自のユーザ ID に置き換えること.

C.3 ibmsp50 (IBM SP2) の設定

各自のユーザ ID でログインし, エディタで .rhosts ファイルを編集する. 内容は, 以下の通り.

```
hi00011 ユーザ ID
fu00011 ユーザ ID
ne00011 ユーザ ID
cr00011 ユーザ ID
ibmsp50a ユーザ ID
ccsemga ユーザ ID
desr01 ユーザ ID
localhost ユーザ ID
```

MPI プロセス起動先のホストとして localhost を指定しない場合, localhost のエントリは不要である.

さらにインタラクティブジョブを実行する場合には次の環境変数の設定が必要となる.

```
setenv      MP_RMPOOL    1
```

尚, ユーザ ID は, 各自のユーザ ID に置き換えること.

C.4 cr00011 (CRAY T94) の設定

各自のユーザ ID でログインし, エディタで .rhosts ファイルを編集する. 内容は, 以下の通り.

```
hi00011 ユーザ ID
fu00011 ユーザ ID
ne00011 ユーザ ID
cr00011 ユーザ ID
ibmsp50a ユーザ ID
ccsemga ユーザ ID
desr01 ユーザ ID
localhost ユーザ ID
```

MPI プロセス起動先のホストとして localhost を指定しない場合, localhost のエントリは不要である.

尚, ユーザ ID は, 各自のユーザ ID に置き換えること.

CRAY T94 の場合 mpt (Message Passing Toolkit) を使用する準備として以下のコマンドを実行する必要がある. 本操作は, .login に記述しておけばよい.

```
module load mpt
```

C.5 ne00011 (NEC SX4) の設定

各自のユーザ ID でログインし, エディタで .rhosts ファイルを編集する. 内容は, 以下の通り.

```
hi00011 ユーザ ID
fu00011 ユーザ ID
ne00011 ユーザ ID
cr00011 ユーザ ID
ibmsp50a ユーザ ID
ccsemga ユーザ ID
desr01 ユーザ ID
localhost ユーザ ID
```

MPI プロセス起動先のホストとして localhost を指定しない場合, localhost のエントリは不要である.

尚, ユーザ ID は, 各自のユーザ ID に置き換えること.

C.6 ccsemga (SGI Onyx) の設定

各自のユーザ ID でログインし, エディタで .rhosts ファイルを編集する. 内容は, 以下の通り.

```
hi00011 ユーザ ID
fu00011 ユーザ ID
ne00011 ユーザ ID
cr00011 ユーザ ID
ibmsp50a ユーザ ID
ccsemga ユーザ ID
desr01 ユーザ ID
localhost ユーザ ID
```

MPI プロセス起動先のホストとして localhost を指定しない場合, localhost のエントリは不要である.

尚, ユーザ ID は, 各自のユーザ ID に置き換えること.

国際単位系（SI）と換算表

表1　SI基本単位および補助単位

量	名称	記号
長さ	メートル	m
質量	キログラム	kg
時間	秒	s
電流	アンペア	A
熱力学温度	ケルビン	K
物質量	モル	mol
光度	カンデラ	cd
平面角	ラジアン	rad
立体角	ステラジアン	sr

表2　SIと併用される単位

名称	記号
分，時，日	min，h，d
度，分，秒	°，′，″
リットル	l，L
トン	t
電子ボルト	eV
原子質量単位	u

1 eV=1.60218×10^{-19}J

1 u=1.66054×10^{-27}kg

表3　固有の名称をもつSI組立単位

量	名称	記号	他のSI単位による表現
周波数	ヘルツ	Hz	s^{-1}
力	ニュートン	N	m·kg/s^2
圧力，応力	パスカル	Pa	N/m^2
エネルギー，仕事，熱量	ジュール	J	N·m
工率，放射束	ワット	W	J/s
電気量，電荷	クーロン	C	A·s
電位，電圧，起電力	ボルト	V	W/A
静電容量	ファラド	F	C/V
電気抵抗	オーム	Ω	V/A
コンダクタンス	ジーメンス	S	A/V
磁束	ウェーバ	Wb	V·s
磁束密度	テスラ	T	Wb/m^2
インダクタンス	ヘンリー	H	Wb/A
セルシウス温度	セルシウス度	℃	
光束	ルーメン	lm	cd·sr
照度	ルクス	lx	lm/m^2
放射能	ベクレル	Bq	s^{-1}
吸収線量	グレイ	Gy	J/kg
線量等量	シーベルト	Sv	J/kg

表4　SIと共に暫定的に維持される単位

名称	記号
オングストローム	Å
バーン	b
バール	bar
ガル	Gal
キュリー	Ci
レントゲン	R
ラド	rad
レム	rem

1 Å=0.1nm=10^{-10}m

1 b=100fm^2=10^{-28}m^2

1 bar=0.1MPa=10^{5}Pa

1 Gal=1cm/s^2=10^{-2}m/s^2

1 Ci=3.7×10^{10}Bq

1 R=2.58×10^{-4}C/kg

1 rad=1cGy=10^{-2}Gy

1 rem=1cSv=10^{-2}Sv

表5　SI接頭語

倍数	接頭語	記号
10^{18}	エクサ	E
10^{15}	ペタ	P
10^{12}	テラ	T
10^{9}	ギガ	G
10^{6}	メガ	M
10^{3}	キロ	k
10^{2}	ヘクト	h
10^{1}	デカ	da
10^{-1}	デシ	d
10^{-2}	センチ	c
10^{-3}	ミリ	m
10^{-6}	マイクロ	μ
10^{-9}	ナノ	n
10^{-12}	ピコ	p
10^{-15}	フェムト	f
10^{-18}	アト	a

(注)

1．表1－5は「国際単位系」第5版，国際度量衡局1985年刊行による。ただし，1eVおよび1uの値はCODATAの1986年推奨値によった。

2．表4には海里，ノット，アール，ヘクタールも含まれているが日常の単位なのでここでは省略した。

3．barは，JISでは流体の圧力を表わす場合に限り表2のカテゴリーに分類されている。

4．EC閣僚理事会指令ではbar，barnおよび「血圧の単位」mmHgを表2のカテゴリーに入れている。

換算表

力	N(=10^{5}dyn)	kgf	lbf
	1	0.101972	0.224809
	9.80665	1	2.20462
	4.44822	0.453592	1

粘度　1Pa·s(N·s/m^2)=10P(ポアズ)(g/(cm·s))

動粘度　1m^2/s=10^{4}St(ストークス)(cm^2/s)

圧力	MPa(=10bar)	kgf/cm^2	atm	mmHg(Torr)	lbf/in^2(psi)
	1	10.1972	9.86923	7.50062×10^3	145.038
	0.0980665	1	0.967841	735.559	14.2233
	0.101325	1.03323	1	760	14.6959
	1.33322×10^{-4}	1.35951×10^{-3}	1.31579×10^{-3}	1	1.93368×10^{-2}
	6.89476×10^{-3}	7.03070×10^{-2}	6.80460×10^{-2}	51.7149	1

エネルギー・仕事・熱量	J(=10^7 erg)	kgf·m	kW·h	cal(計量法)	Btu	ft·lbf	eV
	1	0.101972	2.77778×10^{-7}	0.238889	9.47813×10^{-4}	0.737562	6.24150×10^{18}
	9.80665	1	2.72407×10^{-6}	2.34270	9.29487×10^{-3}	7.23301	6.12082×10^{19}
	3.6×10^6	3.67098×10^5	1	8.59999×10^5	3412.13	2.65522×10^6	2.24694×10^{25}
	4.18605	0.426858	1.16279×10^{-6}	1	3.96759×10^{-3}	3.08747	2.61272×10^{19}
	1055.06	107.586	2.93072×10^{-4}	252.042	1	778.172	6.58515×10^{21}
	1.35582	0.138255	3.76616×10^{-7}	0.323890	1.28506×10^{-3}	1	8.46233×10^{18}
	1.60218×10^{-19}	1.63377×10^{-20}	4.45050×10^{-26}	3.82743×10^{-20}	1.51857×10^{-22}	1.18171×10^{-19}	1

1 cal= 4.18605J（計量法）

= 4.184J（熱化学）

= 4.1855J（15℃）

= 4.1868J（国際蒸気表）

仕事率　1 PS(仏馬力)

= 75 kgf·m/s

= 735.499W

放射能	Bq	Ci
	1	2.70270×10^{-11}
	3.7×10^{10}	1

吸収線量	Gy	rad
	1	100
	0.01	1

照射線量	C/kg	R
	1	3876
	2.58×10^{-4}	1

線量当量	Sv	rem
	1	100
	0.01	1

(86年12月26日現在)

異機種並列計算機間通信ライブラリ:Stampi —利用手引書