社団法人 情報処理学会 研究報告
IPSJ SIG Technical Report

2007－HPC－111 (12)
2007／8／1

MatlabMPI-to-MPIトランスレータの開発

笹岡泰司† 川端英之† 北村俊明†

MatlabMPIはMATLABにMPIと同様な並列プログラミングモデルを提供するライブラリでMATLABのみで記述されているという特徴を持つ．MatlabMPIを用いれば，数値計算を記述しやすいMATLABで並列に処理を行うプログラムを作成することが可能になり，比較的容易に高速なプログラムを作成できる．しかし，MatlabMPIは通信にファイルI/Oを利用しているために通信性能について難がある．本稿では，この通信性能を改善してよりよいプログラミング開発環境を提供するために，MatlabMPIプログラムをMPI+C言語に変換するトランスレータを設計，開発した．開発したシステムでは，入力としてMatlabMPIで記述された並列処理プログラムを受け取り，MPIを用いて並列化されたC言語プログラムを出力する．行列積の実測では，変換したプログラムはMatlabMPIより約2倍程度高速であった．これにより，可読性・記述の容易さと性能を両立させるアプローチの有効性が確認できた．

Design and implementation of a MatlabMPI-to-MPI translator

Taiji Sasaoka,† Hideyuki Kawabata† and Kitamura Toshiaki†

MatlabMPI is a set of routines supplying MATLAB environment with an MPI-like programming model. Using MatlabMPI, you can construct efficient programs with relatively easily. However, MatlabMPI has an obvious disadvantage of low performance communication via file-I/O. In this article, we show the design and implementation of a MatlabMPI-to-MPI translator to provide a good programming environment for parallel computing. This improvement of parallel execution speed of translated codes is obtained mainly by the replacement of communication routines. Our translator receives a MatlabMPI M-file script as an input and outputs a C program parallelized with MPI. Experimental results show that translated matrix-matrix multiplicaton programs with this system can run about twice as fast as the original code with MatlabMPI. This confirms the effectiveness of our approach to construct a parallel programming environment to maintain readability, describability, and performance.

1. はじめに

近年，クラスタなどの並列計算機環境が発達してきており，これまで扱えなかったような大規模の問題が扱えるようになってきたり，これまで時間のかかっていた計算処理が比較的短時間で行えるようになってきた．一方で，並列計算機向けのプログラムは，スカラ計算機向けのプログラムより複雑で，効率的な並列処理を行うプログラムを作成することは容易ではない．このことから，CやFortran90などではなく，プログラムを比較的容易に開発できるMATLABのような実行環境において，並列計算可能な開発環境を提供するための研究が幅広く行われている．

MATLAB[1]はMathworks社が提供している行列言語および実行環境であり，行列演算を基本演算と定義していることから，数値計算アルゴリズムが記述しやすい言語である．また，ユーザインタフェースやグラフィック機能，数値計算アルゴリズムなどの豊富な組込み関数なども充実しており，ユーザがプログラムを開発しやすい環境である．しかし，MATLABにおいて変数は基本的に配列であり，MATLABが動的型付け言語であるために，複雑なアルゴリズム記述などの処理を行うと実行速度においてCやFortranで記述したプログラムとは比較にならない場面もある．そこで，MATLABの記述しやすさをそのまま並列計算機環境で利用できるようにすることで，プログラム記述の容易性と高速な実行速度を兼ね備えた環境を提供しようと検討がなされている．

MATLABのような行列計算言語を並列化する方法として，大きく分類して3つの手法がある[3]．1つ目として，MPIのようなプロセス間でのデータの通信関数を提供する方法が挙げられる．次に，並列計算を行うバックエンドのシステムを用意し，行列計算言語をインタフェースとして利用する方法がある．最後に，行

† 広島市立大学大学院情報科学研究科
Graduate School of Information Sciences, Hiroshima City University

列計算言語向けのプログラムをコンパイルしてC言語やFortranに変換し，そのうえでMPIや並列数値計算ライブラリを使用して並列処理を行う方法が挙げられる．それぞれの方法に特徴が存在するが，その中でMatlabMPI[2]はMATLABにプロセス間でのデータ通信関数を提供する方法の1つであり，MPIのような通信関数を純粋にMATLABスクリプトのみで実装している．MatlabMPIを使用してMATLABプログラムを並列動作するように記述することで，MATLABの記述の容易性と並列処理による高速化が期待できる．しかし，MatlabMPIでは各プロセス間のデータの通信にNFSなどのファイルシステムを利用しており，その通信性能の低さから，適用可能範囲の制約が大きいと考えられる．

そこで，我々はMatlabMPIをMPIを伴うC言語のプログラムに変換するトランスレータの開発を行なった．これによりMatlabMPIの並列処理の記述しやすさと，一般的に広く扱われているMPIのプログラムの高速性を両立させたプログラム開発環境の提供を目指した．

本稿ではそのMatlabMPIプログラムをC言語でMPIを用いたプログラムに変換するトランスレータの開発について以下2章でMatlabの並列化について行われている研究や方法と，プログラム変換対象であるMatlabMPIの概要と特徴について説明し，3章でトランスレータの概要や変換方法などについて説明し，5章でまとめる．

2. MATLABプログラムの並列処理手法とMatlabMPI

2.1 MATLABプログラムの並列処理手法

MATLABで並列処理を実現するために様々な研究が行われている．これらの研究は大きく分けて3つのタイプに分類することができる．それはメッセージパッシング型，バックエンドサポート型，コンパイル型である．以下でそれぞれについて説明する．

2.1.1 メッセージパッシング型

メッセージパッシング型はMATLABにMPIのようなメッセージパッシングな通信関数を用意し，その通信関数を利用して，処理を並列化する方法である．代表的なものとして，MatlabMPI[2]，MultiMATLAB[4]などがある．

メッセージパッシング型を用いる利点としては，自由度の高いプログラミングができるということが挙げられる．メッセージパッシング型においてはユーザがどのデータをどのプロセスに渡すか明記するので，ユーザの思い通りの並列プログラムを記述することができる．

一方で，プログラムが複雑になりやすいという欠点も挙げられる．つまり，ユーザの自由度が高く，データを自由にプロセスに分散することができるが，そのための処理をユーザが記述しなければならず，並列計算するための手間がかかる．効率の良い並列計算をしようとすれば並列処理に関する知識も必要となり，ユーザの負担が大きくなる．

2.1.2 バックエンドサポート型

MATLABで並列計算するための方法としてバックエンドに並列計算サーバを用意して，そのサーバ上で計算する方法がある．この方法においてMATLABはインタフェースとしての役割をし，MATLABは主にサーバとのデータ通信のために用いられる．この方法を採用しているプロジェクトとしてStar-P[5]がある．

バックエンドサポート型の利点として，ユーザの負担が少ないことが挙げられる．バックエンドサポート型では，MATLABに処理を入力すると，MATLABはデータと処理内容をサーバに転送する．サーバはそれらを受け取り，処理内容にあった並列演算処理を行う．この方法で，並列計算はバックエンドで自動的に行われるので，ユーザは並列計算の知識を必要としなくてすむ．また，プログラムに並列処理のための記述が必要ないので，ユーザが手軽に利用できる．

2.1.3 コンパイル型

並列計算の最後の方法としてMATLABプログラムをコンパイルして，C言語やFortranに変換し，そこでMPIやScaLAPACKを使い，並列に計算する方法がある．この方法を採用しているプロジェクトとしてはOtter[6]などがある．

この方法を用いる利点として，MATLABならではの動的処理を軽減できるので高速なプログラムが期待できることや，C言語やFortranの既存のプログラムとの連携がとりやすいことが挙げられる．

一方で，動的な言語から静的な言語に変換するためには，プログラムの特殊化の指示などをユーザが与える必要がある．また，MATLAB実行環境を用いないことを前提にしているために，行列言語独自で用意している豊富な関数との連携が他の手法より困難だという面もある．

2.2 MatlabMPI

この節では，本研究で開発したトランスレータによる変換対象であるMatlabMPIについての説明を行う．

MatlabMPIはMATLAB上にMPIに類似のインタフェースを持つ通信ライブラリを実装したもので，純粋にMATLABスクリプトだけを用いて作られている．MatlabMPIの関数はMPIの関数と比較すると次のような相違点がある．

- データの送受信などプロセス間の通信はファイル経由で行われる．
- 送受信する変数に関してサイズ・型などの情報が必要ない．
- 複数の変数を一度に送受信することが可能である．

これらの特徴からMatlabMPIではMPIより並列

表 1　MatlabMPI の通信関数

関数名	関数の仮引数と返り値
MPI_Bcast	[dout] = MPI_Bcast(source, tag, comm, din)
MPI_Send	MPI_Send(dest, tag, comm, data)
MPI_Recv	[data] = MPI_Recv(source, tag, comm)

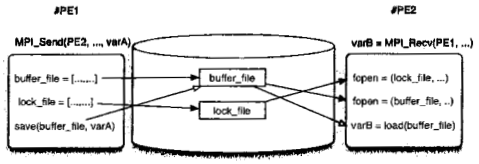

図 1　MatlabMPI によるデータ通信

処理を記述し易いといえる．なぜなら，MatlabMPI での通信には MPI で必要なサイズ情報・型情報が必要ないため，ユーザが把握しなければならない情報が MPI より少ないからである．また，一度に複数のデータを送受信できることにより，同じ相手との通信関数を記述する回数が減ると考えられる．一方で、データの送受信をファイル経由で行なっていることから，MatlabMPI では通信関数が頻出するプログラムには向いてないと言える．

2.2.1　MatlabMPI による実行の流れ

MatlabMPI を利用した並列処理は次のような流れで行なわれる．

まずユーザは並列処理を記述した MATLAB プログラムである M-file を用意して，MATLAB を起動する．そして次のコマンドを入力する．

`eval(MPI_Run('filename', np, {machine}));`

ここで filename は並列処理を記述したプログラムファイルである M-file の名前，np はプロセスを走らせる台数，{machine} は MatlabMPI で並列処理をするために使うマシン名である．このマシン名は複数記述することもでき，その場合は {machine1 machine2} のように列挙する．

このコマンドを入力すると，各マシン上で MATLAB が起動し，プログラムファイルの処理内容に沿った並列処理が行われる．各マシンには rsh/ssh を用いて接続する．接続時にそのマシンが実行する処理内容が入ったシェルスクリプトが実行コマンドとして渡される．この時に，コミュニケータ (MPI における MPI_Comm に相当) が作られ，各プロセスは固有のランクを割り当てられる．

2.2.2　MatlabMPI 通信関数の動作

MatlabMPI に用意されている通信関数は表 **1** のように 3 種類あり，データを送受信する方法として MPI_Send, MPI_Recv を使う方法と MPI_Bcast を使う方法の二通りがある．

まず MPI_Send, MPI_Recv を使って送受信する方法について説明する．送信側のプロセスが MPI_Send を呼ぶと，そのプロセスは 2 つのファイルを作成する．このファイルの名前は自分のプロセスのランク，送信先のプロセスのランク，データの内容を識別するためのタグの 3 つの値によって決まる．そして，その名前のバッファファイル，ロックファイルを作り，バッファファイルにデータを保存する．

```
...
tag = 1;
...
if my_rank == 0
  MPI_Send(1, tag, comm, data1, data2);
elseif my_rank == 1
  [data1 data2] = MPI_Recv(0, tag, comm);
end
...
...
```

図 2　複数のデータを一度に送受信するプログラム例

受信側のプロセスは，送信元のプロセスのランク，自分のプロセスのランク，タグの 3 つの値によって送信元が作ったバッファファイルとロックファイルのファイル名を特定する．特定したロックファイルが存在すれば，バッファファイルからデータを読み込み，返り値として data を返す．ロックファイルが存在しなければ送信元のプロセスがロックファイルを作るまで，待機する．

図 **1** はこの手順を図示したものである．このように MatlabMPI ではデータの送受信はファイル経由で行われる．ここで，それぞれのプロセスは通信を行うたびに個別のファイルを作成して用いる．また，データの送受信の時，data は送信側と受信側で数が揃っていれば複数並べても構わない．つまり，記述図 **2** のようにデータを送受信することもできる．

MPI_Bcast での通信もほぼ同様である．送信側が MPI_Bcast を呼び出すとそのデータを全プロセスに対して MPI_Send を使って送信する．受信側は送信側がファイルを作成するのを待機し，自分向けのファイルが作成されたらそのデータを受信する．

3.　MatlabMPI-to-MPI トランスレータ

MatlabMPI プログラムを MPI を用いた C 言語のプログラムに変換するトランスレータについて説明する．プログラム変換は基本的には MatlabMPI の通信関数呼び出しは MPI の通信関数呼び出しに変換する．残りの演算部分は変数の型を考慮しながら C 言語での演算記述に変換する．

MatlabMPI プログラムを変換する際に特に考慮しなければならない部分は以下の 3 点である．

- MPI にとって MatlabMPI の通信関数だけでは必要な情報が不足していること
- 受信側のプロセスにおいて，データ受信時に送信されてくるデータの型が不明なこと

```
typedef struct {
  double *array; データ本体を格納するポインタ
  int size;          データ全体のサイズ
  int size1;         データの行方向のサイズ
  int size2;         データの列方向のサイズ
} DArray
```

図 3　送受信する変数のデータ構造

- 通信関数以外は MATLAB の構文規則に則ったプログラムであること

3.1 MATLAB 記述の解析

MatlabMPI は MATLAB をベースにしているので，並列処理が記述されている M-file は MATLAB の構文に従っている．この構文を解析するものとして我々が開発中の処理系である CMC[7] を用いる．

MatlabMPI プログラムを変換する部分を主に 2 つに分けて考える．1 つは MPI のような送受信などを行う MatlabMPI の通信関数の変換で，もう 1 つは通信以外の演算部分である．それぞれ次節以降で説明する．

3.2 MatlabMPI 関数の変換

3.2.1 通信関数の変換

ユーザに利用される MatlabMPI のプログラムの通信関数は表 1 の通りである．これらの MatlabMPI の関数呼び出しをそれぞれ対応する MPI ライブラリに変換させる．MPI の通信関数と比較して MatlabMPI の通信関数の大きな特徴は以下の 3 つであり，これらを考慮した変換を行わなければならない．

(1) データ属性（型・サイズ）に関する情報が不要
(2) 複数のデータを一度に送受信可能
(3) MPI_Bcast で送信するデータを MPI_Recv で受信可能

MatlabMPI の送受信関数を MPI の送受信関数に変換する手順を説明する．C 言語での MPI の送信関数に必要な情報は (送信データ・送信データサイズ・送信データの型・送信先のプロセスのランク・タグ・コミュニケータ) の 6 つである．この中で，MatlabMPI の送信関数の引数に含まれる情報は (送信データ・送信先のプロセスのランク・タグ・コミュニケータ) の 4 つである．残りの送信データサイズとデータの型は MatlabMPI の送信関数から読み込むことができない．

C 言語での MPI の受信関数に必要な情報は (受信データ名・受信データサイズ・受信するデータの型・送信元のプロセスのランク・タグ・コミュニケータ・MPI_Status) の 7 つである．この中で MatlabMPI の受信関数に含まれる情報は (受信データ・受信先のプロセスのランク・タグ・コミュニケータ) の 4 つである．よって送信関数の場合と同じように，MatlabMPI の受信関数からデータサイズとデータの型に関する情報を読み取ることができない．ちなみに MPI_Status は考慮しない．

このように Matlab の送受信関数から MPI の送受信関数に変換するにはデータのサイズ情報と型情報についてあらかじめ把握しておかなければならない．しかし，C 言語では動的にサイズ情報を把握することができない．そこで送信するデータはすべて行列扱いとし，サイズ情報をデータ自身に把握させておく方法をとる．そのために構造体図 **3** を用意する．送受信に関わるデータはすべて図 3 の形式にする．この構造体図 3 はデータ本体を格納するポインタ，データの行方向のサイズ，データの列方向のサイズ，データ全体のサイズの情報を格納でき，MPI でデータを通信する場合に必要不可欠なサイズ情報をこの構造体から取り出すことで送受信のサイズ情報を把握する．データの型情報については MATLAB は基本的にはデータを double 型として扱うのですべて double とする．

```
...
if my_rank == 0
  A = ones(size1, size2);
end
...
if my_rank == 0
  MPI_Send(dest, tag, comm, A);
elseif my_rank == 1
  A = MPI_Recv(source, tag, comm);
end
...
...
```

図 4　MatlabMPI によるデータの送受信

これによって MPI でデータを送受信する場合に必要なデータが確保された．実際に送受信関数を変換する手順は次の通りである．

(1) データを送受信するためのサイズ情報を送信側から受信側に送信する．サイズ情報は行方向と列方向と分けて送信する．
(2) 受信側は受け取ったサイズ情報を元に必要なデータ領域を確保する．
(3) データ本体の送受信を行う．

このように段階的な送受信に変更することで，MatlabMPI の送受信関数を MPI の送受信関数に変換する．MatlabMPI のプログラム図 **4** を変換すると図 **5** のようになる．

次に MatlabMPI の特徴の (2) の用に複数のデータを同時に送受信する場合の変換について述べる．これは先ほど述べた MatlabMPI の関数呼び出しの変換部分の送受信するデータが複数個になった場合である．MPI の関数呼び出しでは複数のデータを一度に送受信することができないので，存在しているデータを順に通信するような関数呼び出しに変換する．

Bcast の場合もほぼ同様の処理を行う．データ構造は図 3 とし，データを送信するプロセスはデータ本体を送信する前にデータのサイズに関する情報を各プロセスに送信する．データを受信するプロセスはサイズの情報を受信すると，そのサイズの領域を確保しその領域にデータを受信する．

3.2.2 その他の MatlabMPI 関数

MatlabMPI では送受信の関数以外にも MPI に関

```
...
DArray *A;
...
if (my_rank == 0) {
  A = mallocDArray(size1, size2);
...
  A->darray[...]  = 1.0;
...
}
...
if (my_rank == 0) {
  MPI_Send(&A->size1, 1, MPI_INTEGER,
       dest, tag+111, comm);
  MPI_Send(&A->size2+222, 1, MPI_INTEGER,
       dest, tag, comm);
  MPI_Send(A->array, A->size, MPI_DOUBLE,
       dest, tag, comm);
}
else if (my_rank == 1) {
  MPI_Recv(&recvsize1, 1, MPI_INTEGER,
       source, tag+111, comm, &status);
  MPI_Recv(&recvsize2, 1, MPI_INTEGER,
       source, tag+222, comm, &status);
  A = mallocDArray(recvsize1, recvsize2);
  A = MPI_Recv(A->array, A->size, MPI_DOUBLE,
        source, tag, comm &status);
}
...
...
```

図 5　変換後の C 言語プログラム例

表 2　MatlabMPI のその他の関数の変換

MatlabMPI 関数	変換後
MPI_Init	MPI_init(&argc,&argv)
s=MPI_Comm_size(comm)	MPI_Comm_size(comm,&s)
r=MPI_Comm_rank(comm)	MPI_Comm_rank(comm,&r)
MPI_Finalize	MPI_Finalize()

連する関数がいくつか用意されている．ここで用意されている関数はランク情報の取得など一定の形式で扱える関数である．そのため表 **2** のようにそのまま変換する．

3.3　演算部分の変換

MatlabMPI プログラムにおける通信関数以外の変換についての説明を行う．演算部分の変換は CMC[7] において行う．CMC ではプログラムに現れる変数についてデータフロー解析により型や形状（行ベクトル，行列，疎行列など）を決定する．データフロー解析によって各変数の型が定まると，その変数の形状や型，演算内容によって出力する内容が決まる．

4.　実測および評価

MatlabMPI-to-MPI トランスレータによって生成されるプログラムを評価するための実測を行なった．本章では，実測結果とその評価について説明する．

実測は以下の 2 種類のプログラムに対して行った．

- ping-pong ベンチマーク
- 行列積プログラム

ping-pong ベンチマークでは MatlabMPI と MPI と通信性能にどの程度差があるかを測定するために用いた．行列積プログラムでは MatlabMPI プログラムとそれを変換して得られたプログラムの実行時間を測定し，プログラム変換による影響を調べた．

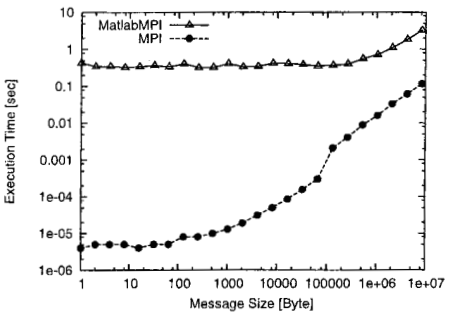

図 6　ping-pong ベンチマークによる実行時間

実装は，京都大学学術情報メディアセンターの以下の環境で行なった．

- FUJITSU PRIMEPOWER HPC2500(SMP)
- CPU : SPARC64V 2.08GHz
- メモリ : 512GB
- C コンパイラ : fcc (Fujitsu C Compiler Driver Version 5.6)
- fcc コンパイルオプション : KSPARC64_GP2,V8PLUS -Kfast_GP2=3 -KSSL2
- MATLAB Version 7.1.0.183 (R14) SP3
- MatlabMPI v1.2
- BLAS V5.3

4.1　実 測 結 果

4.1.1　ping-pong ベンチマーク

ping-pong ベンチマークプログラムでは，MatlabMPI と C 言語での MPI それぞれにおいて 2 台のプロセスでデータを往復させるのにどのくらい時間がかかるかを計測した．往復させるデータは 1B から 8MB まで変化させた．ping-pong ベンチマークの実測結果を図 **6** に示す．

図 6 を見ると，送受信するデータのサイズが小さい場合，MatlabMPI は MPI と比べるとデータを送受信するのに非常に時間を要している．これは MatlabMPI がデータの通信をファイルシステム経由で行なうからであり，そのためにデータサイズが小さい場合でもデータの送受信に時間を要す．また，MatlabMPI ではデータの転送単位が 1B から 512KB までで通信にかかる時間はほぼ同じである．このことからファイルを作成する時間やファイルへの書き出し，読み込みに一定の時間がかかっており，かつ，その時間は通信に対して大きな影響を持っていることがわかる．

4.1.2　行　列　積

次に，行列積プログラムにおける実行時間を台数を 1,2,4,8,16,32 と変化させて測定した．行列は 4000x4000 の正方行列で，$C = A * B$ を計算する

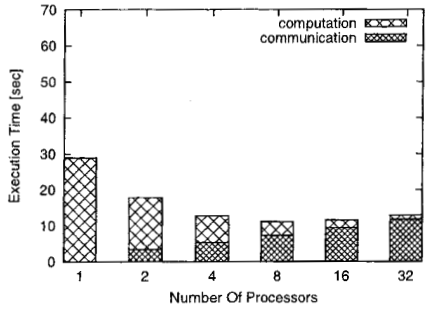

図 7　MPI による 4000x4000 の行列積実行時間

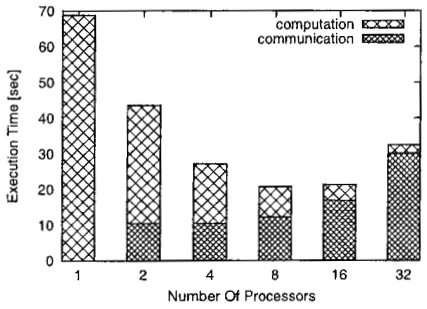

図 8　MatlabMPI による 4000x4000 の行列積実行時間

プログラムである．並列処理は単純な列方向のブロック分割で，行列 B と行列 C を分割して計算した．なお MPI において行列積の計算部分は BLAS の DGEMM ルーチンを用いた．

このプログラムで行う通信は以下の 3 通りである．

- プロセス 0 で生成した 4000x4000 の正方行列 A を全プロセスに Broadcast する
- プロセス 0 で生成した 4000x4000 の正方行列 B を列方向に分割し，プロセス 0 から各プロセスに担当範囲の部分を送信する．
- 各プロセスで計算した部分行列 C を各プロセスからプロセス 0 に送信する．

このプログラムの通信と計算にかかる時間をそれぞれ計測した．**図 7**，**図 8**，はそれぞれ MatlabMPI でそのまま実行した場合と，トランスレータによって MatlabMPI プログラムを MPI を用いた C 言語プログラムに変換した場合の測定結果である．

図を見比べると，MPI に変換したプログラムでは通信に要する時間 (図中の communication の部分) は MatlabMPI より少ない．MPI での通信時間は台数が 2 台の場合は MatlabMPI の約 32%であり，16 台の場合で約 56%である．MPI において計算に要した時間 (図中の computation の部分) は 1 台の時は MatlabMPI の約 42%である．16 台の場合で約 43%である．

4.2 考　　察

図 8 から MatlabMPI を利用すれば比較的楽に MATLAB プログラムを高速化できることが分かる．MatlabMPI はファイルシステムを利用することで通信にかかる時間を無視することはできないが，あまり通信が発生しないプログラムで有効に利用できると考えられる．

今回設計した MatlabMPI-to-MPI トランスレータを用いれば，MatlabMPI プログラムをより高速に実行できるようになる．また，変換したプログラムは MPI を用いて記述された C 言語プログラムなので，MatlabMPI があまり得意でないデータサイズの小さな通信が頻発するようなプログラムにおいても，並列処理による高速化の効果が期待できると考えられる．

5. ま と め

今回，MatlabMPI プログラムを MPI を使った C 言語プログラムに変換するトランスレータについて述べた．本システムは MatlabMPI の通信関数を MPI の通信関数に変換し，MATLAB の演算処理部分を CMC によって C 言語に変換することによって，MatlabMPI プログラムを高速化することができた．これにより，可読性・記述の容易さと性能を両立させるアプローチの有効性が確認できた．

多様なプログラムへの適用と評価，MATLAB の豊富な組込み関数への対応などが今後の課題である．

謝辞　本研究の一部は広島市立大学特定研究費（一般研究費，課題番号 4111）の助成による．

参 考 文 献

1) http://www.mathworks.com/
2) Jeremy, K.: Parallel Programming with matlabMPI. Accepted by High Peformance Embedded Computing(HPEC 2001) Workshop, 2001.
3) Ron Choy,and Alan Edelman: Parallel MATLAB: Doing it right. Proceedings of the IEEE, 93(2) 2005.
4) Vijay Menon, Anne E. Trefethen. :MultiMATLAB integrating MATLAB with high-performance parallel computing Conference on High Performance Networking and Computing. pp.1–18 1997
5) http://www.interactivesupercomputing.com/
6) Quinn, M.J., et al.: Otter: Bridging the Gap between MATLAB and ScaLAPACK, Proc. 7th IEEE Intl. Symp. High Performance Distributed Computing, 1998.
7) 川端英之，鈴木睦: 疎行列に対応した行列言語コンパイラ CMC の開発，情報処理学会論文誌：コンピューティングシステム，Vol.45, No.SIG11(ACS7), pp.378–392, 2004.