

EGS5-MPI マニュアル

国立研究開発法人 産業技術総合研究所
計量標準総合センター 分析計測標準研究部門
清水森人

2017 年 6 月 27 日

1 EGS5-MPI の概要

EGS5-MPI は EGS5[1] を並列化する拡張パッケージであり、EGS5 を並列計算の通信規格である MPI(Message Passing Interface) を用いて並列化する。MPI は多くの大型計算機施設で採用されているとともに、規格化されているため、各環境間の互換性が非常に高い。そのため、EGS5-MPI を用いて並列化した EGS5 コードはデスクトップ環境上で作成、動作テストさえ行っておけば、多くの場合、ほとんどの大型計算機施設でそのまま実行することができる。

EGS5-MPI を用いて作成したシミュレーション計算を実行すると、指定した MPI プロセス数だけ EGS5 シミュレーションが実行され、各シミュレーション間の乱数の管理やタイミング同期が行われる。各シミュレーション間には MPI 通信による通信が可能であり、各シミュレーション間で計算結果などの集約、配信ができる。

以下に EGS5-MPI の主な特徴、機能を示す。

- MPI 通信規格を用いた分散メモリ型並列計算化
- 各プロセス間の乱数管理機能
- 各プロセス間の同期処理機能
- MPI プロセスの初期化および終了処理
- MPI サイズ、ランクなどの COMMON 変数の提供

2 動作環境

EGS5 が実行可能であり、MPI 規格に準拠した並列計算環境であれば、多くの場合、実行可能である。動作確認をした環境を以下に示す。

- Linux (g77 + OpenMPI)
- Linux (Intel Fortran Compiler + OpenMPI)
- Linux on Intel Xeon Phi 7200 series (Intel Fortran Compiler + OpenMPI)
- Cygwin (gfortran + OpenMPI)
- MacOSX (gfortran + OpenMPI)

2.1 推奨テスト環境

EGS5-MPI は並列計算を行うため、無限ループなどに陥るとシステムが操作不能になるなど、重大な障害が生じる可能性もある。特に、Cygwin 環境での使用は Windows 本体に影響を与える可能性があるので、推奨しない。

近年、仮想マシン技術の進歩が著しく、一般の Windows デスクトップ上でも VMware Player^[2] と呼ばれる無償の VMware 環境に Linux をインストールして使用する事ができる。前述のような重大な障害が生じたとしても、あくまでも仮想マシン上でのトラブルであるため、Windows 側に影響を及ぼすことは無く、比較的安全に開発、テストを行うことができる。また、大型計算機の多くは Linux で動作しており、実際の環境と似た条件で開発、テストが行えるのも魅力である。実際に開発者は仮想マシン上にセットアップした Scientific Linux 環境で開発を行っている。Scientific Linux では OpenMPI やジョブ管理スケジューラ torque なども用意されており、煩雑なセットアップ手続きをしなくても、すぐに EGS5-MPI を試すことができるので、強く推奨する。

3 セットアップ

EGS5 および MPI がセットアップされ、正常に動作している環境を前提に説明する。テスト環境としては、EGS5、MPI が設定済みの Linux 環境を VMWare などを使って用意する事を推奨する。

1. egs5mpi.tar.bz2 の展開

適当なディレクトリに egs5mpi.tar.bz2 を展開する。(ここでは、/home/egs5user/直下に展開したとする。)

```
$ pwd
/home/egs5user
$ tar jxf egs5mpi.tar.bz2
```

2. egs5mpi ディレクトリ内の egs5mpirun を編集する

BASKET には egs5 本体のインストールディレクトリのパスを指定する。(ここでは、/home/egs5user/egs5 にあるとする。) MPLBASKET には egs5mpi のディレクトリのパスを指定する。MY_MACHINE には使用している OS および MPI のタイプを入力する。OPT_FLAGS にはオプションフラグを入力する。各 OS および MPI の組み合わせに最低限必要なフラグは既にセットされている。最適化レベルを指定するオプションフラグなどを入力すると良い。なお、最適化レベルは上げすぎるとコンパイルや実行に失敗するときがある。MPL_SIZE には実行する MPI プロセス数(同時に実行するシミュレーションの数)を指定する。MPL_COMPILER には MPI のコンパイルコマンドを入力する。MPL_RUN には MPI の実行コマンドを入力する。例に書かれている内容で通常問題無いが、大型計算機施設などでは独自のコマンドを用いていることがある。

```
BASKET=/home/egs5user/egs5
MPI_BASKET=/home/egs5user/egs5mpi
```

```

MY_MACHINE='Linux-OpenMPI'
OPT_FLAGS='-O2'
MPI_SIZE=2
MPI_COMPILER='mpif77'
MPI_RUN='mpirun'

```

以上でインストールは完了である．実行方法は `egs5` と全く同じであり，`egs5mpirun` をコンパイルしたい `egs5` シミュレーションのディレクトリにコピーし，実行するだけである．

```
$ ./egs5mpirun
```

後の指示は通常の EGS5 のシェルスクリプトと同じである．

大型計算機施設で使用する場合，コンパイル処理だけにとどめ，ジョブ管理システムなどで指定される環境変数を元に `mpirun` を実行するのが望ましい．ジョブスクリプトには次のように記述すると良い．

```

./egs5mpirun comp <'<プログラムファイル名>
<data ファイル名もしくは改行>
<inp ファイル名もしくは改行>
'>
mpirun <ジョブ管理システムから提供される環境変数> ./egs5job.exe

```

4 EGS5-MPI コードの詳細

ここでは EGS5-MPI 内部の COMMON 変数および，各処理を行うサブルーチンの説明を行う．

4.1 COMMON 変数

4.1.1 MAX_MPI_SIZE

`MAX_MPI_SIZE` は使用可能な最大の MPI プロセス数を指定する．デフォルトは 1024 としており，このプロセスを超える環境での動作実績は無い．また， 10^7 以上の値を指定した上で， 10^7 プロセスを超える計算を実行すると乱数などの出力に不具合が生じるが，プログラムの仕様上，問題は生じない．なお，当面の間，京コンピュータなどの PFLOPS 級計算機でも 10^7 を超えることは無い．

4.1.2 mpi_mainseed

シミュレーション全体の基準となる乱数種である．この値を元に各プロセスの乱数種が生成される．

4.1.3 mpi_inseed(MAX_MPI_SIZE)

各プロセスの乱数種が格納される整数型配列である．`egs5mpi_rluxinit` をコールすることで，`mpi_mainseed` を元に生成した各プロセスの乱数種が格納され，MPI 通信によって各プロセスに値が配信される．

4.1.4 mpi_rank

各プロセスの rank を示す整数型変数である．この値を参照することで，各プロセスは自身の rank を知ることができる．

4.1.5 mpi_size

全体の MPI プロセス数を示す整数型変数である．この値を参照することで，全体のプロセス数を知ることができる．

4.1.6 mpi_err

MPI 通信処理のエラーを格納する変数．

4.2 サブルーチン

4.2.1 egs5mpi_init

MPI の初期化，mpi_size，mpi_rank の取得を行うサブルーチンである．このサブルーチンをコールすると，MPI が初期化され，mpi_size には総プロセス数が，mpi_rank にはそのプロセスの rank が格納される．このサブルーチンはプログラムの最初にコールされなくてはならない．

4.2.2 egs5mpi_rluxinit

乱数の初期化を行うサブルーチンである．rank が 0 の主プロセスでこのルーチンがコールされると，mpi_mainseed の値を元に各プロセスの乱数種が主プロセスによって生成される．生成された乱数種は mpi_inseed に格納され MPI 通信によって各プロセスに配信される．各プロセスは配信された自身の乱数種を inseed に格納し，rluxinit をコールして乱数を初期化する．各プロセスの rank 及び乱数種の値が出力されて，処理が終了する．なお，このルーチンをコールする前に，mpi_mainseed および luxlev を指定しておく必要がある．並列計算では生成される乱数が膨大になるため，luxlev は高めに設定しておくが良い．

4.2.3 egs5mpi_pegscall

PEGS5 のコールを行うサブルーチンである．PEGS5 によるファイルへの同時書き込みを防ぐため，主プロセスのみが PEGS5 をコールし，処理が完了するまでそれ以外のプロセスは待機する．

4.2.4 egs5mpi_finalize

MPI の終了処理を行うサブルーチンである．プログラムの終わりには必ずこれをコールしなくてはならない．

5 ucsampcg_mpi の解説

ucsampcg_mpi は EGS5 のサンプルコードである ucsampcg を EGS5-MPI を用いて並列化したものである．EGS5-MPI による既存の EGS5 プログラムの並列化手続きについて説明する．なお，ucsampg 自体の動作については EGS5 のマニュアルを参照すること．

5.1 EGS5-MPI のインクルード

EGS5-MPI 使用するには、まず、MPI のインクルードファイル `mpif.h` および EGS5-MPI のインクルードファイルをインクルードする必要がある。64 行目の処理がこれにあたる。

```
!      -----
!      EGS5MPI COMMONs
!      -----
      include "mpif.h"
      include 'mpi_include/egs5mpi_h.f'
```

5.2 MPI 処理用の変数の宣言

ここでは全体の計算粒子数を格納する `totalcase` および、ファイル名に rank 番号をつけるための文字列 `rank_str`、各プロセスでの `esum` の総和を取る `mpi_esum` を宣言している。この処理は求めたい内容によって変化する部分である。95 行目の処理がこれにあたる。

```
      integer totalcases
      character*6 rank_str
      real*8 mpi_esum(MXREG)
```

5.3 egs5mpi の初期化

変数の宣言が終わったら、直ちに EGS5-MPI を初期化する。99 行目の処理がこれにあたる。

```
!-----
!      EGS5-MPI Init
!-----

      call egs5mpi_init
```

5.4 ファイルの OPEN 処理

MPI による並列計算では複数のプロセスが一斉にファイル入出力を行うため、ファイルへの同時書き込みが生じないように、気をつける必要がある。MPI 用のファイル入出力処理として `MPHIO` などもあるが、ある程度計算結果の目算をつけておく必要があり、また処理によっては 1 つのファイルサイズが膨大になってしまうなどの問題もある。

最も確実な方法は、各プロセス毎に別々の書き込みファイルを用意する方法であり、数十プロセス程度で頻繁に書き込み処理を行わないのであれば、この方法で十分にスケーラビリティ（プロセス数に比例して計算速度が向上すること）を確保できる。頻繁に書き込みを行うような計算の場合、プロセス数が増加すると、書き

込み処理の待機が増え、スケーラビリティが低下する。この場合は大型計算機施設などで利用されている並列計算用のファイルシステム (Lustre ファイルシステムなど) を利用する必要がある。なお、読み込みしか行わないファイルについては名前を振り分ける必要は無い。

ucsampeg_mpi では、mpi_rank を用いて rank 番号を示す文字列を生成し、ファイル名と拡張子の間に入力している。116 行目の処理がこれにあたる。

```
write (rank_str, '(I6.6)') mpi_rank

open(UNIT=6,FILE='egs5job.'//rank_str//'.out',STATUS='unknown')
open(UNIT=39,FILE='egs5job.'//rank_str//'.pic',STATUS='unknown')
```

5.5 PEGS5 のコール

PEGS5 のコールは各プロセス間での同期が必要なため、egs5mpi_pegscall を用いてコールする。155 行目の処理がこれにあたる。

```
!      =====
      call egs5mpi_pegscall
!      =====
```

5.6 乱数の初期化

乱数の初期化を egs5mpi_rluxinit を用いて行う。まず、luxlev、mpi_mainseed を指定し、egs5mpi_rluxinit をコールする。これにより、各プロセスの乱数が初期化される。213 行目の処理がこれに当たる。

```
luxlev = 1
mpi_mainseed=1

!      =====
      call egs5mpi_rluxinit  ! Initialize the Ranlux random-number generator
!      =====
```

5.7 計算粒子数の指定

ucsampeg_mpi では全体の計算粒子数の指定を totalcases 変数で行っている。これを mpi_size で商を取り、1 を加えたものを各プロセスの計算粒子数としている。これは、商を取った結果、実際の計算粒子数の和が最初に指定した計算粒子数の値よりも小さくなることを防ぐためである。大規模な並列計算を行う場合、気をつける必要がある。233 行目の処理がこれにあたる。

```
totalcases = 1000
ncases= totalcases / mpi_size + 1
```

5.8 dummy ファイル処理

同時書き込み防止のため，dummy ファイルもファイル名と拡張子の間に rank 番号を付け加えている．269 行目の処理がこれにあたる．

```
open(UNIT=KMP0,FILE='egs5job.'//rank_str//'.dummy',  
&      STATUS='unknown')
```

5.9 計算結果の集約

モンテカルロシミュレーションによる並列計算において最も重要なのは乱数の管理であり，計算結果の集約は手作業でも構わないし，最も確実な手段でもある．とはいえ，MPI 通信には計算結果を集約するとともに和や差を取ったりすることのできる処理が用意されている．ここでは，mpi_allreduce を用いて，各プロセスの esum の値を合計し，mpi_esum に格納している．比較のため，各プロセスでの出力と，MPI 全体で集約した場合の計算結果を出力するようにしている．369 行目の処理がこれにあたる．

```
call mpi_allreduce(esum,mpi_esum,nreg,  
&      MPI_DOUBLE_PRECISION,MPI_SUM,  
&      MPI_COMM_WORLD,mpi_err)  
  
write(6,'(//,a)') "This process result:"  
  
totke=ncases*ekin  
write(6,220) ei,ncases  
220  format(/,' Incident total energy of electron=',  
&      F12.1,' MeV',/,  
&      ' Number of cases in run=',I7,  
&      '//,' Energy deposition summary:',/)  
  
etot=0.D0  
do i=1,nreg  
    etot=etot+esum(i)  
    esum(i)=esum(i)/totke  
    write(6,230) i, esum(i)  
230  format(' Fraction in region ',I3,'=',F10.7)  
end do  
  
etot=etot/totke  
write(6,240) etot
```

```

240  FORMAT(//,' Total energy fraction in run=',G15.7,/,
      *'   Which should be close to unity')

      write(6,'(//,a)') "Total result:"

      totke = ncases * ekin * mpi_size
      totalcases = ncases * mpi_size
      write(6,221) ei,totalcases
221  format(/,' Incident total energy of electron=',
      &      F12.1,' MeV',/,
      &      ' Number of cases in run=',I7,
      &      '//,' Energy deposition summary:',/)

      etot=0.D0
      do i=1,nreg
          etot=etot+mpi_esum(i)
          mpi_esum(i)=mpi_esum(i)/totke
          write(6,231) i, mpi_esum(i)
231  format(' Fraction in region ',I3,'=',F10.7)
      end do

      etot=etot/totke
      write(6,241) etot
241  FORMAT(//,' Total energy fraction in run=',G15.7,/,
      &'   Which should be close to unity')

```

5.10 EGS5-MPI の終了処理

MPI プログラムは終了時に終了処理を行う必要がある．EGS5-MPI では `egs5mpi_finalize` をコールすることでの処理が行われる．425 行目の処理がこれにあたる．

```

!-----
!      EGS5-MPI FINALIZE
!-----
      call egs5mpi_finalize

```

問い合わせ先

EGS5-MPI に関する問い合わせ先は EGS5-MPI ホームページ, <https://unit.aist.go.jp/rima/ioniz-rad/egs5mpi/> に掲載しております．

参考文献

- [1] H. Hirayama, Y. Namito, A.F. Bielajew, S.J. Wilderman and W.R. Nelson, SLAC-R-730 (2005) and KEK Report 2005-8 (2005).
- [2] VMware 社ホームページ, <http://www.vmware.com/>
- [3] Scientific Linux, <http://www.scientificlinux.org/>