EGS5-MPI user's manual

Morihito Shimizu
National Metrology Institute of Japan,
National Institute of Advanced Industrial Science and Technology

June 27, 2017

1 EGS5-MPI Overview

The EGS5-MPI is an extended package of the EGS5 (Electron Gamma Shower version 5) for parallelization by using MPI (Message Passing Interface) implementation. MPI is a language-independent communication protocol used to program parallel computers and MPI is the dominant model used in High-Performance Computing (HPC) today. Therefore, the EGS5-MPI can be used on Linux, Cygwin, Mac OSX and other UNIX OS and the program parallelized by using the EGS5-MPI on desktop PC, are executable on a HPC machine without major modification.

When the EGS5-MPI simulation starts, the multiple EGS5 simulations are started and the random seed of each EGS5 simulation is managed by random seed management function of the EGS5-MPI. Each EGS5 simulations can communicate between each other by using MPI communication functions. Thus, the timing-synchronization, concentration, distribution can be performed between each EGS5 simulation.

EGS5-MPI features

- distributed memory parallelization by using MPI implementation
- random seed management function
- PEGS timing-synchronization function
- MPI initialize and finalize function
- some common variables (mpi_rank, mpi_size, etc.)

2 System requirement

The EGS5-MPI can be used on UNIX like OS which has MPI libraries. The system environment which carried out operation check is shown as follows.

- Linux (GNU FORTRAN Compiler g77/gfortran + OpenMPI)
- Linux (Intel FORTRAN Compiler + OpenMPI)
- Linux on Intel Xeon Phi 7200 series (Intel FORTRAN Compiler + OpenMPI)

- Cygwin (GNU FORTRAN Compiler g77/gfortran + OpenMPI or MPICH2)
- MacOSX (GNU FORTRAN Compiler gfortran + OpenMPI, on Fink environment)

2.1 Recommended environment for a desktop computer for daily use

The EGS5-MPI simulation which uses all CPU cores exhausts CPU resource nearly completely, and may cause unexpected system failure (In many cases, it is hang-up or thermal run-away). Therefore, performing the EGS5-MPI which uses all CPU cores is not recommended for a desktop computer for daily use. Particularly, don 't use the EGS5-MPI on a laptop computer.

If you want to develop the EGS5-MPI simulation on a desktop computer for daily use, you should use the EGS5-MPI on the Linux with Virtual Machine (VM) software. VM software provides a completely virtualized set of hardware to the guest OS. The host OS and the guest OS are independent of each other, and the guest OS cannot occupy all the resource of a single physical PC. Therefore, the host OS is not subject to the influence of the critical system failure of guest OS. "VMware Workstation Player" is free VM software provided by VMware, Inc.[2]. Linux environment on VMware Workstation Player is recommended for the EGS5-MPI simulation development on a Windows PC for daily use.

3 EGS5-MPI Installation

Please set up the EGS5 and the MPI enviroment before setting up the EGS5-MPI.

1. EGS5-MPI installation
   The distribution file of the EGS5-MPI is a compressed (with bzip2) UNIX "tar"archive, egs5mpi.tar.bz2. Here, it decompressed to /home/egs5user as a home directory of user "egs5user".Please type,

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

   After these command executed, a directory named "egs5mpi" will be created on home directory.

2. egs5mpirun Configuration
   Sample script "egs5mpirun"under egs5mpi directory can be used to compile and run EGS5-MPI codes with minimal modification. Example modifications follow.

   ```
   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''
```

- BASKET: EGS5 directory
- MPI_BASKET: EGS5-MPI directory
- MY_MACHINE: operating system and MPI type
- OPT_FLAGS: additional option flags. The required option flags are already set up.
- MPI_SIZE: number of MPI processes. (Usually, the number of core)
- MPI_COMPILER: MPI compiler command name. (Usually, "mpif77")
- MPI_RUN: MPI run command name. (Usually, "mpirun" or "mpiexe")

The preamble to egs5mpirun contains some examples of how the variables described above might be set in egs5mpirun.

3. Running EGS5-MPI
There are five options for executing egs5mpirun, invoked by specifying a single command line argument, as shown below.

Command line	Action of egs5mpirun
egs5mpirun	Compile user code and execute.
egs5mpirun comp	Compile user code but do not execute.
egs5mpirun cl	Clear out files (and links) and exit egs5mpirun.

For using EGS5-MPI on HPC machine, you should use "egs5mpirun comp" command and execution command should be written in job script file.

4 Detail of EGS5-MPI

4.1 COMMON variables of EGS5-MPI

4.1.1 MAX_MPI_SIZE

MAX_MPI_SIZE is a number of maximum MPI processes. Default value is 1024 and the number to 10^7 can be set as this value. But, there is no HPC machine with 10^7 or more cores in the world.

4.1.2 mpi_mainseed

The random seed number of a EGS5-MPI program. The random seed number of each MPI process is generated from this variable.

4.1.3 mpi_inseed(MAX_MPI_SIZE)

The integer-type array in which the random seed number of each MPI process is stored. When subroutine egs5mpi_rluxinit is called, the random seed numbers of each MPI processes are generated and stored in this array and this array is distributed by MPI communication.

4.1.4 mpi_rank

The number of each MPI process's rank.

4.1.5 mpi_size

The total number of the MPI processes.

4.1.6 mpi_err

The error code of the MPI is stored in this variable.

4.2 Subroutines

4.2.1 egs5mpi_init

The subroutine "egs5mpi_init" initialize the MPI and get mpi_size and mpi_rank. This subroutine must be called at the beginning of the EGS5-MPI program.

4.2.2 egs5mpi_rluxinit

The random number is initialized by this subroutine. When the root process (rank number = 0) call "egs5mpi_rluxinit", the random seed numbers of each process are generated and stored in mpi_inseed. mpi_inseed is distributed each child process by using the MPI communication. Each process received mpi_inseed called rluxinit which is original EGS5 subroutine and initialize random number generator. Before calling "egs5mpi_rluxinit", you must set mpi_mainseed and luxlev.

4.2.3 egs5mpi_pegscall

This subroutine calls the PEGS5. When this subroutine is called, the root process (rank number = 0) call PEGS5. In order to prevent simultaneous file write, other processes wait until PEGS5 calculation finish.

4.2.4 egs5mpi_finalize

The subroutine "egs5mpi_finalize" finalize the MPI communicator. This subroutine must be called at the ending of the EGS5-MPI program.

5 Example program of EGS5-MPI: ucsampcg_mpi

Example program "ucsampcg_mpi" is the EGS5-MPI parallelization version of ucsamcg which is the sample program of the EGS5. In this section, how to parallelization existing program with the EGS5-MPI is explained. In this section, the parallelization procedure of the EGS5 existing program by the EGS5-MPI is explained.

5.1 Include EGS5-MPI header files

For using the MPI and the EGS5-MPI subroutines and common variables, “mpif.h” and ”egs5_mpif.h” must be included in egs5mpi program.

At 64th line,

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

5.2 Variables for MPI

Here, some variables are declared. ”totalcase” is the total number of calculation incident particle. ”rank_str” is the character string of rank number which is used for file name. Real-type array ”mpi_esum” is stored total of ”esum” in each process. Description of this part changes with the processing which you want to do. At 95th line,

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

5.3 Initialization egs5mpi

When variables declaration is finished, the EGS5-MPI must be initialized. At 99th line,

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

      call egs5mpi_init
```

5.4 File OPEN

In the EGS5-MPI calculation, many processes perform file input-output. Therefore, you have to prevent simultaneous file write. There are two solutions this problem. The 1st solution is the method of using MPI-IO. MPI-IO is input-output processing for MPI. It is necessary to expect a calculation result and this solution has a problem to which one file size becomes huge.

The 2nd solution that is the most reliable solution is the method of preparing a separate write-in file for each MPI process. If write-in processing is not frequently performed in about tens of MPI processes, scalability can fully be guaranteed by this method. In calculation which writes in frequently, if the number of MPI processes increases, the standby for write-in processing will increase and scalability will fall. In this case, you should use the file systems for parallel computing (Lustre file system etc.).

In ucsampcg_mpi, the character string which shows a rank number using mpi_rank was generated, and it has inputted between a file name and an extension.

At 116 th line,

```
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 Calling PEGS5

For calling pegs5, you should use egs5mpi_pegscall. At 155th line,

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

5.6 Random seed generator Initialization

Before random number generator initialization, you should input luxlev and mpi_mainseed. In the EGS5-MPI, Random number generator is initialized by calling egs5mpi_rluxinit.

At 213rd line,

```
      luxlev = 1
      mpi_mainseed=1

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

5.7 Calculation cases

In ucsampcg_mpi, the number of calculation incident particle of each process is calculated from the variable "totalcases", as follows,

$$ncases = totalcases/mpi\_size + 1 \tag{1}$$

ncases is the number of calculation incident particles of each process. One incident particle is added to ncases so that the actual number of calculation incident particle may not become smaller than totalcases. Upon calculation with many processes, you should be carefully about this.

At 233rd line,

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

5.8 Dummy file

The dummy file has also added the rank number between the file name and the extension for simultaneous write-in prevention.

At 269th line,

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

5.9 Calculation results

In the parallel computing of a Monte Carlo simulation, the random number management is the most important. If you can, the manual calculation is one of the realiable method for the analysis of calculation results. MPI provides the function which calculates the sum and difference of the calculation results of each MPI process. Here, the summation of the esum of each MPI process is calculated by mpi_allreduce. For the comparison, the calculation results of each MPI process and the total calculation results of all MPI process are outputted in the output file "egs5job.XXXXXXX.out".

At 369th line,

```
      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 Finalization of EGS5-MPI

The EGS5-MPI programs must be finalized at the ending of the program by calling "egs5mpi_finalize".

At 425th line,

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

More info

If you have any question about the EGS5-MPI, please contact us. Additional info about the EGS5-MPI is on our homepage (https://unit.aist.go.jp/rima/ioniz-rad/egs5mpi/).

References

[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 Japan, http://www.vmware.com/

[3] Scientific Linux, http://www.scientificlinux.org/