Санкт-Петербургский государственный университет
кафедра вычислительной физики

С.А.Немнюгин

Лабораторная работа 0.1
Знакомство со структурой MPI-программы и процедурами блокирующего двухточечного обмена MPI

Методические материалы к курсу «Средства программирования для многопроцессорных вычислительных систем»

Intel Multicore Curriculum Initiative

Санкт-Петербург
2007

Двухточечные блокирующие обмены

Участниками двухточечного обмена являются два процесса: процесс-отправитель и процесс-получатель. Блокирующие операции двухточечного обмена приостанавливают выполнение вызывающего процесса. Он переходит в состояние ожидания завершения передачи данных. Блокировка гарантирует выполнение действий в заданном порядке, обеспечивая предсказуемость поведения программы. С другой стороны, она создает условия для возникновения тупиковых ситуаций, когда оба процесса-участника обмена блокируются одновременно.

Трансляция MPI-программ

Утилиты трансляции и сборки находятся в каталоге **/каталог_установки_MPI/bin**. Его следует включить в путь поиска исполняемых файлов. Утилиты трансляции запускают трансляторы обычных языков C/C++ и Fortran. В командную строку их запуска, при необходимости, подставляют ссылки на необходимые библиотечные и заголовочные файлы.

Для трансляции и компоновки программ на языке C++ используется команда `mpiCC`, для трансляции программ на языке C — команда `mpicc`. Для трансляции и компоновки программ на языке Fortran 77 и Fortran 90 используются команды `mpif77` и `mpif90`.

Информацию о ключах можно найти на справочных страницах MPICH (для их просмотра применяют команду man). Пример применения команды:

```
mpicc –o fft fft.c
```

Назначить другой транслятор можно, определив значения переменных окружения `MPICH_CC`, `MPICH_F77`, `MPICH_CCC` или `MPICH_F90`. Изменить программу-компоновщик можно с помощью переменных окружения `MPICH_CLINKER`, `MPICH_F77LINKER`, `MPICH_CCLINKER` и `MPICH_F90LINKER`.

Выполнение MPI-программ

Для выполнения MPI-программ в MPICH используется загрузчик приложений `mpirun`. Он запускает указанное количество копий программы. Команда запуска:

```
mpirun –np n [ключи MPI] программа [ключи и аргументы программы]
```

где *n* — число запускаемых процессов. Некоторые ключи MPI (они указываются перед именем исполняемого файла программы) приведены в таблице:

Ключ	**Описание**
`–h`	Краткая информация о команде
`–machinefile файл`	Для запуска программы использовать список компьютеров из указанного файла

Лабораторная работа

В заданиях лабораторной работы 0.1 предлагается дописать или исправить предлагаемые фрагменты программ на языках Fortran и C, написанные с использованием процедур MPICH 1.2.7, в том числе, процедуры блокирующего двухточечного обмена. Пропущенные фрагменты обозначены многоточием.

Необходимый для выполнения данной лабораторной работы справочный материал можно найти на стр. 27 – 29 *методического пособия* «Средства программирования для многопроцессорных вычислительных систем».

Задание 1

В исходном тексте программы на языке Fortran пропущены вызовы процедур подключения к MPI, определения количества процессов и ранга процесса. Добавить эти вызовы, откомпилировать и запустить программу.

```
program main_mpi
include 'mpif.h'
integer myid, numprocs, ierr
 ....
print *, "process ", myid, " of ", numprocs
call mpi_finalize(ierr)
stop
end
```

Задание 2

В исходном тексте программы на языке С пропущены вызовы процедур подключения к MPI, определения количества процессов и ранга процесса. Добавить эти вызовы, откомпилировать и запустить программу.

```
#include "mpi.h"
#include <stdio.h>
int main(int argc,char *argv[])
{
 int myid, numprocs;
 ....
 fprintf(stdout,"Process %d of %d\n", myid, numprocs);
 MPI_Finalize();
 return 0;
}
```

Задание 3

В исходном тексте программы на языке С пропущены вызовы процедур стандартного блокирующего двухточечного обмена. Предполагается, что при запуске двух процессов один из них отправляет сообщение другому. Добавить эти вызовы, откомпилировать и запустить программу.

```
#include "mpi.h"
#include <stdio.h>
int main(int argc,char *argv[])
{
 int myid, numprocs;
 char message[20];
 int myrank;
 MPI_Status status;
 int TAG = 0;
 MPI_Init(&argc, &argv);
 MPI_Comm_rank(MPI_COMM_WORLD, &myrank);
 if (myrank == 0)
{
 strcpy(message, "Hi, Second Processor!");
 MPI_Send(...);
}
else
{
 MPI_Recv(...);
 printf("received: %s\n", message);
}
 MPI_Finalize();
 return 0;
}
```

Задание 4

В исходном тексте программы на языке С пропущены вызовы процедур стандартного блокирующего двухточечного обмена. Предполагается, что при запуске четного числа процессов, те из них, которые имеют четный ранг, отправляют сообщение следующим по величине ранга процессам. Добавить эти вызовы, откомпилировать и запустить программу.

```
#include "mpi.h"
#include <stdio.h>
int main(int argc,char *argv[])
{
int myrank, size, message;
int TAG = 0;
MPI_Status status;
MPI_Init(&argc, &argv);
MPI_Comm_rank(MPI_COMM_WORLD, &myrank);
MPI_Comm_size(MPI_COMM_WORLD, &size);
message = myrank;
if((myrank % 2) == 0)
{
 if((myrank + 1) != size)
 MPI_Send(...);
}
else
{
 if(myrank != 0)
 MPI_Recv(...);
 printf("received :%i\n", message);
}
MPI_Finalize();
return 0;
}
```

Задание 5

Следующая программа на языке Fortran написана некорректно. Найдите ошибку и исправьте ее.

```
program main_mpi
include 'mpif.h'
integer rank, tag, cnt, ierr, status(mpi_status_size)
real sndbuf, rcvbuf
tag = 0
sndbuf = 3.14159
cnt = 1
call mpi_init(ierr)
call mpi_comm_rank(mpi_comm_world, rank, ierr)
if (rank.eq.0) then
call  mpi_recv(rcvbuf,  cnt,  mpi_real,  1,  tag,  mpi_comm_world,
status, ierr)
call  mpi_send(sndbuf,  cnt,  mpi_real,  1,  tag,  mpi_comm_world,
ierr)
else
call  mpi_recv(rcvbuf,  cnt,  mpi_real,  0,  tag,  mpi_comm_world,
status, ierr)
call  mpi_send(sndbuf,  cnt,  mpi_real,  0,  tag,  mpi_comm_world,
ierr)
end if
call mpi_finalize(ierr)
stop
end
```