Введение в параллельные вычисления. Технология программирования МРІ (день второй)

Антонов Александр Сергеевич, к. ф.-м.н., н.с. лаборатории Параллельных информационных технологий НИВЦ МГУ

•MPI - Message Passing Interface, интерфейс передачи сообщений.

•Стандарт МРІ 1.1.

•Более 120 функций.

•SPMD-модель параллельного программирования.

•Префикс **мрі**_.

include 'mpif.h'(mpi.h для языка Си)

•Процессы, посылка сообщений.

•Группы процессов, коммуникаторы.

MPI COMM WORLD

Каждый процесс может одновременно входить в разные коммуникаторы.

Два основных атрибута процесса: коммуникатор (группа) и номер процесса в коммуникаторе (группе).

Если группа содержит n процессов, то номер любого процесса в данной группе лежит в пределах от 0 до n-1.

- •Сообщение набор данных некоторого типа.
- •Атрибуты сообщения: номер процессаотправителя, номер процесса-получателя, идентификатор сообщения и др.
- •Идентификатор сообщения целое неотрицательное число в диапазоне от 0 до 32767.
- •Для работы с атрибутами сообщений введен массив **MPI**_**STATUS**.

В последнем аргументе (в Си – возвращаемое значение функции) большинство функций MPI возвращают информацию об успешности завершения.

В случае успешного выполнения функция вернет значение **MPI_SUCCESS**, иначе — код ошибки.

Предопределенные значения, соответствующие различным ошибочным ситуациям, определены в файле **mpif**.h

MPI_INIT(IERR) INTEGER IERR

Инициализация параллельной части программы. Все другие функции МРІ могут быть вызваны только после вызова MPI INIT. Инициализация параллельной части для каждого приложения должна выполняться только один

MPI_FINALIZE (IERR)
INTEGER IERR

Завершение параллельной части приложения. Все последующие обращения к любым МРІ-функциям, в том числе к **МРІ ІПІТ**, запрещены. К моменту вызова MPI FINALIZE каждым процессом программы все действия, требующие его участия в обмене сообщениями, должны быть завершены.

Общая схема МРІ-программы:

```
PROGRAM EXAMPLE
INCLUDE 'mpif.h'
INTEGER IERR
CALL MPI INIT (IERR)
CALL MPI FINALIZE (IERR)
```

MPI_INITIALIZED (FLAG, IERR)
LOGICAL FLAG
INTEGER IERR

В аргументе **FLAG** возвращает . **TRUE** . , если вызвана из параллельной части приложения, и . **FALSE** . в противном случае. Единственная МРІ-функция, которую можно вызвать до вызова **MPI_INIT**.

MPI_COMM_SIZE (COMM, SIZE, IERR)
INTEGER COMM, SIZE, IERR
В аргументе SIZE возвращает число
параллельных процессов в коммуникаторе
СОММ.

MPI_COMM_RANK (COMM, RANK, IERR)
INTEGER COMM, RANK, IERR
В аргументе RANK возвращает номер
процесса в коммуникаторе COMM в
диапазоне от 0 до SIZE-1.

DOUBLE PRECISION MPI_WTIME (IERR) INTEGER IERR

Функция возвращает для каждого вызвавшего процесса астрономическое время в секундах (вещественное число), прошедшее с некоторого момента в прошлом. Момент времени, используемый в качестве точки отсчета, не будет изменен за время существования процесса.

DOUBLE PRECISION MPI_WTICK(IERR) INTEGER IERR

Функция возвращает разрешение таймера в секундах.

```
PROGRAM EXAMPLE
INCLUDE 'mpif.h'
INTEGER IERR, SIZE, RANK
CALL MPI INIT (IERR)
CALL MPI COMM SIZE (MPI COMM WORLD, SIZE,
       IERR)
CALL MPI COMM RANK (MPI COMM WORLD, RANK,
       IERR)
PRINT *, 'PROCESS', RANK, 'SIZE', SIZE
CALL MPI FINALIZE (IERR)
END
```

MPI SEND (BUF, COUNT, DATATYPE, DEST, MSGTAG, COMM, IERR) <type> BUF(*) INTEGER COUNT, DATATYPE, DEST, MSGTAG, COMM, IERR Блокирующая посылка массива **BUF** с идентификатором MSGTAG, состоящего из **COUNT** элементов типа **DATATYPE**, процессу с

номером DEST в коммуникаторе COMM.

```
Типы данных:
```

```
MPI INTEGER - INTEGER
```

```
MPI REAL - REAL
```

```
MPI DOUBLE PRECISION - DOUBLE
```

PRECISION

```
MPI COMPLES - COMPLEX
```

Модификации функции **MPI_SEND**:

MPI_BSEND — передача сообщения с буферизацией.

MPI_SSEND — передача сообщения с синхронизацией.

MPI_RSEND — передача сообщения по готовности.

```
MPI_BUFFER_ATTACH(BUF, SIZE,
IERR)
```

<type> BUF(*)

INTEGER SIZE, IERR

Назначение массива **BUF** размера **SIZE** для использования при посылке сообщений с буферизацией. В каждом процессе может быть только один такой буфер.

```
MPI_BUFFER_DETACH(BUF, SIZE,
IERR)
```

<type> BUF(*)

INTEGER SIZE, IERR

Освобождение массива **BUF** для других целей. Процесс блокируется до того момента, когда все сообщения уйдут из данного буфера.

```
MPI_RECV(BUF, COUNT, DATATYPE,
SOURCE, MSGTAG, COMM, STATUS,
IERR)
```

<type> BUF(*)

INTEGER COUNT, DATATYPE, SOURCE,
MSGTAG, COMM, IERR,
STATUS (MPI_STATUS_SIZE)

Блокирующий прием сообщения длины не более **COUNT** от процесса с номером **SOURCE** с заполнением массива **STATUS**.

Вместо аргументов **SOURCE** и **MSGTAG** можно использовать константы:

MPI_ANY_SOURCE — признак того, что подходит сообщение от любого процесса **MPI_ANY_TAG** — признак того, что подходит сообщение с любым идентификатором.

Параметры принятого сообщения всегда можно определить по соответствующим элементам массива **STATUS**:

- **STATUS (MPI_SOURCE)** номер процессаотправителя.
- **STATUS (MPI_TAG)** идентификатор сообщения.
- STATUS (MPI_ERROR) код ошибки.

Если один процесс последовательно посылает два сообщения, соответствующие одному и тому же вызову **MPI_RECV**, другому процессу, то первым будет принято сообщение, которое было отправлено раньше.

Если два сообщения были одновременно отправлены разными процессами, то порядок их получения принимающим процессом заранее не определен.

MPI_GET_COUNT(STATUS, DATATYPE,
COUNT, IERR)

INTEGER COUNT, DATATYPE, IERR, STATUS (MPI STATUS SIZE)

По значению параметра **STATUS**функция определяет число **COUNT**уже принятых (после обращения к **MPI_RECV**) или принимаемых (после обращения к **MPI_PROBE** или **MPI IPROBE**) элементов сообщения

MPI PROBE (SOURCE, MSGTAG, COMM, STATUS, IERR) INTEGER SOURCE, MSGTAG, COMM, IERR, STATUS (MPI STATUS SIZE) Получение в массиве STATUS информации о структуре ожидаемого сообщения с блокировкой. Возврата не произойдет, пока сообщение с подходящим идентификатором и номером процесса-отправителя не будет доступно для получения.

```
CALL MPI COMM SIZE (MPI COMM WORLD, SIZE,
        IERR)
CALL MPI COMM RANK (MPI COMM WORLD, RANK,
       IERR)
IF (MOD (RANK, 2) . EQ. 0) THEN
  IF (RANK+1 .LT. SIZE) THEN
     CALL MPI SEND(..., RANK+1, TAG,
                MPI COMM WORLD, IERR)
  ENDIF
ELSE
  CALL MPI RECV(..., RANK-1, TAG,
             MPI COMM WORLD, STATUS, IERR)
 &
ENDIF
```