Лабораторная работа №5

Управление группами процессов и коммуникаторами

Цель: изучить основные принципы управления группами процессов и коммуникаторами в технологии MPI на примере использования в рамках языка C++.

Для получения **теоретических сведений** настоятельно рекомендуется при подготовке изучить материалы, представленные в списке литературы в конце разработки, а также прочие материалы по тематике лабораторной работы, представленные в открытых источниках.

Далее следует краткий конспект материала, приведенного в данных источниках, в конце включающий короткие примеры фрагментов программ.

1. Управление группами процессов и коммуникаторами. Общие сведения

Под **коммуникатором** в MPI понимается специально создаваемый служебный объект, объединяющий в своем составе группу процессов и ряд дополнительных параметров (**контекст**), используемых при выполнении операций передачи данных. Как правило, парные операции передачи данных выполняются для процессов, принадлежащих одному и тому же коммуникатору. Коллективные операции применяются одновременно для всех процессов коммуникатора.

Все имеющиеся в параллельной программе процессы входят в состав создаваемого по умолчанию коммуникатора с идентификатором MPI_COMM_WORLD.

При необходимости передачи данных между процессами из разных групп необходимо создавать глобальный коммуникатор.

2. Управление группами

Группы процессов могут быть созданы только из уже существующих групп. В качестве исходной группы может быть использована группа, связанная с предопределенным коммуникатором MPI_COMM_WORLD.

Для получения группы, связанной с существующим коммуникатором, используется функция:

```
int MPI_Comm_group(MPI_Comm comm, MPI_Group *group)
```

Далее, на основе существующих групп, могут быть созданы новые группы:

- создание новой группы **newgroup** из существующей группы **oldgroup**, которая будет включать в себя n процессов, ранги которых перечисляются в массиве **ranks**:

```
int MPI_Group_incl(MPI_Group oldgroup,int n, int *ranks,MPI_Group *newgroup)
```

- создание новой группы **newgroup** из группы **oldgroup**, которая будет включать в себя n процессов, ранги которых не совпадают с рангами, перечисленными в массиве **ranks**:

```
int MPI_Group_excl(MPI_Group oldgroup,int n, int *ranks,MPI_Group *newgroup)
```

Для получения новых групп над имеющимися группами процессов могут быть выполнены операции объединения, пересечения и разности:

- создание новой группы **newgroup** как объединения групп **group1** и **group2**:

```
int MPI_Group_union(MPI_Group group1, MPI_Group group2, MPI_Group *newgroup)
```

- создание новой группы **newgroup** как пересечения групп **group1** и **group2**:

```
int MPI_Group_intersection (MPI_Group group1, MPI_Group group2,
   MPI_Group *newgroup)
```

- создание новой группы **newgroup** как разности групп **group1** и **group2**:

```
int MPI_Group_difference (MPI_Group group1, MPI_Group group2,
       MPI_Group *newgroup)
```

При конструировании групп может оказаться полезной специальная пустая группа MPI_COMM_EMPTY.

Ряд функций MPI обеспечивает получение информации о группе процессов:

- получение количества процессов в группе:

```
int MPI_Group_size ( MPI_Group group, int *size )
```

- получение ранга текущего процесса в группе:

```
int MPI_Group_rank ( MPI_Group group, int *rank )
```

После завершения использования группа должна быть удалена:

int MPI_Group_free (MPI_Group *group)

при этом, выполнение данной операции не затрагивает коммуникаторы, в которых используется удаляемая группа.

3. Управление коммуникаторами

В данном пункте рассматривается управление **интракоммуникаторами**, используемыми для операций передачи данных внутри одной группы процессов. Для создания новых коммуникаторов применимы два основных способа их получения:

- дублирование уже существующего коммуникатора:

```
int MPI_Comm_dup (MPI_Comm oldcom, MPI_comm *newcomm);
```

- создание нового коммуникатора из подмножества процессов существующего коммуникатора:

```
int MPI_comm_create (MPI_Comm oldcom, MPI_Group group, MPI_Comm *newcomm)
```

Быстрый способ одновременного создания нескольких коммуникаторов обеспечивает функция:

```
int MPI_Comm_split ( MPI_Comm oldcomm, int split, int key, MPI_Comm *newcomm)
```

где

oldcomm – исходный коммуникатор,
split – номер коммуникатора, которому должен принадлежать процесс,
key – порядок ранга процесса в создаваемом коммуникаторе,
newcomm – создаваемый коммуникатор.

Создание коммуникаторов относится к коллективным операциям и, тем самым, вызов функции **MPI_Comm_split** должен быть выполнен в каждом процессе коммуникатора **oldcomm**.

После завершения использования коммуникатор должен быть удален:

int MPI_Comm_free (MPI_Comm *comm).

Задание. Реализуйте на основе технологии MPI многопоточную программу с использованием **групп процессов и коммуникаторов** в соответствии с вариантом задания**.** Проверьте корректность работы программы. **Результаты занесите в отчет.**

Вариант	**Задание**
0	Реализуйте тип «длинное целое». Напишите программу, которая осуществляет умножение A целых чисел заданной длины методом Карацубы.
1	Реализуйте тип «комплексная матрица». Напишите программу, которая осуществляет умножение A матриц размером NxN методом Штрассена.
2	Реализуйте тип «длинное целое». Напишите программу, которая осуществляет умножение A целых чисел заданной длины методом Тоома-Кука.

Требования к сдаче работы

1. При подготовке изучить теоретический материал по тематике лабораторной работы, представленный в списке литературы ниже, выполнить представленные примеры, занести в отчёт результаты выполнения.
2. Продемонстрировать программный код для лабораторного задания.
3. Продемонстрировать выполнение лабораторных заданий (можно в виде скриншотов).
4. Ответить на контрольные вопросы.
5. Показать преподавателю отчет.

Литература

1. Спецификации стандарта Open MPI (версия 1.6, на английском языке): http://www.open-mpi.org/doc/v1.6/
2. Материалы, представленные на сайте intuit.ru в рамках курса «Intel Parallel Programming Professional (Introduction)»: http://old.intuit.ru/department/supercomputing/ppinteltt/5/
3. С.А. Лупин, М.А. Посыпкин Технологии параллельного программирования. – М.: ИД «ФОРУМ»: ИНФРА-М, 2011. – С. 12-96. *(Глава, посвященная MPI)*
4. Отладка приложений MPI в кластере HPC http://msdn.microsoft.com/ru-ru/library/dd560808(v=vs.100).aspx
5. http://www.parallel.ru/tech/tech_dev/mpi.html
6. http://msdn.microsoft.com/ru-ru/library/ee441265(v=vs.100).aspx#BKMK_debugMany