

Упаковка-распаковка данных для передачи

Упаковка-распаковка данных для передачи

Для компактной пересылки разнородных данных можно использовать операции упаковки и распаковки данных. Разнородные или расположенные не в последовательных ячейках памяти данные помещаются в один непрерывный буфер. Буфер пересылается, а потом полученное сообщение снова распределяется по нужным ячейкам памяти.

Для операций упаковки и распаковки используются функции **MPI_Pack** и **MPI_Unpack**.

Упаковка-распаковка данных для передачи

Функция упаковки

```
int MPI_Pack ( void *data, int count, MPI_Datatype type,  
              void *buf, int buflen, int *bufpos, MPI_Comm comm),
```

где

- **data** – буфер памяти с элементами для упаковки,
- **count** – количество элементов в буфере,
- **type** – тип данных для упаковываемых элементов,
- **buf** – буфер памяти для упаковки,
- **buflen** – размер буфера в байтах,
- **bufpos** – позиция для начала записи в буфер (в байтах от начала буфера),
- **comm** – коммуникатор для упакованного сообщения.

Функция выполняет упаковку **count** элементов типа **type** из массива **data** в массив **buf** со сдвигом **bufpos**. После выполнения функции параметр **bufpos** увеличивается на число байт, равное размеру помещенных в буфер данных.

Основы параллельных вычислений:

*Моделирование и анализ
параллельных вычислений*

© Гергель В.П.

Упаковка-распаковка данных для передачи

Пересылка упакованных данных

Буфер для упаковки – `buf [buflen]` - массив типа `char` размера `buflen` . Он должен быть больше суммарной длины всех данных для упаковки в байтах. Вместо переменной `buflen` можно использовать числовое значение.

Пересылка буфера с упакованными данными производится любой функцией передачи (индивидуальной или коллективной). Должен использоваться тип данных `MPI_PACKED`.

Упаковка-распаковка данных для передачи

Функция распаковки

```
int MPI_Unpack (void *buf, int buflen, int *bufpos,  
               void *data, int count, MPI_Datatype type, MPI_Comm comm);
```

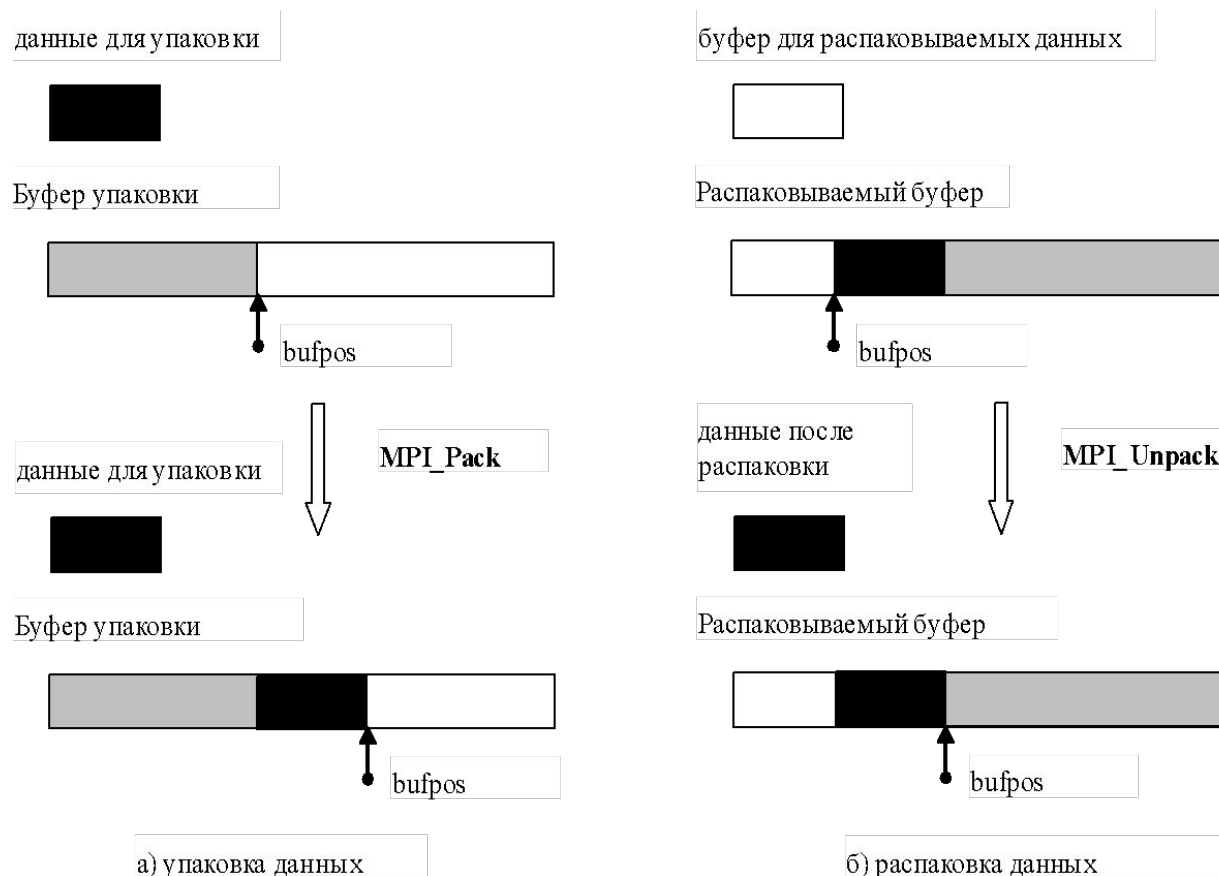
где

- **data** - буфер памяти с элементами для распаковки,
- **count** - количество элементов в этом буфере,
- **type** - тип данных для распаковываемых элементов,
- **buf** - буфер памяти для распаковки,
- **buflen** - размер буфера в байтах,
- **bufpos** - позиция для начала записи в буфер (в байтах от начала буфера),
- **comm** - коммуникатор для упакованного сообщения.

Функция выполняет распаковку **count** элементов типа **type** из массива **buf** со сдвигом **bufpos** байт от начала массива в массив **data**. После выполнения функции параметр **bufpos** увеличивается на число байт, равное размеру считанных данных.

Упаковка-распаковка данных для передачи.

Пояснение



Упаковка-распаковка данных для передачи

- **Формирование сообщений при помощи упаковки и распаковки данных...**

- Вызов функции *MPI_Pack* осуществляется последовательно для упаковки всех необходимых данных. Если в сообщение должны входить данные *a,b,c*, то для их упаковки необходимо выполнить:

```
bufpos = 0;
MPI_Pack(a, 1, MPI_DOUBLE, buf, buflen, &bufpos, comm);
MPI_Pack(b, 1, MPI_DOUBLE, buf, buflen, &bufpos, comm);
MPI_Pack(n, 1, MPI_INT, buf, buflen, &bufpos, comm);
```

- Для распаковки упакованных данных необходимо выполнить:

```
bufpos = 0;
MPI_Unpack(buf, buflen, &bufpos, a, 1, MPI_DOUBLE, comm);
MPI_Unpack(buf, buflen, &bufpos, b, 1, MPI_DOUBLE, comm);
MPI_Unpack(buf, buflen, &bufpos, n, 1, MPI_INT, comm);
```

Пример формирования сообщений при помощи упаковки и распаковки данных

```
char buf[100];  int pos;
float a;  int n;  int mas[6];
...
if (rank == 0){
    printf("Enter a, and n\n");
    scanf("%f %i", &a, &n);
    printf("Enter 6 elements of array \n");
    for(i=0;i<6;i++)
        scanf("%i",&mas[i]);
    /* упаковка a, n и массива mas из 6 элементов */
    pos = 0;
    MPI_Pack(&a, 1, MPI_FLOAT, &buf, 100, &pos, MPI_COMM_WORLD);
    MPI_Pack(&n, 1, MPI_INT, &buf, 100, &pos, MPI_COMM_WORLD);
    MPI_Pack(&mas, 6, MPI_INT, &buf, 100, &pos, MPI_COMM_WORLD);
    MPI_Bcast(&buf, 100, MPI_PACKED, 0, MPI_COMM_WORLD);
} else {
    MPI_Bcast(&buf, 100, MPI_PACKED, 0, MPI_COMM_WORLD);
    pos = 0;
    /* распаковка a, n и массива mas из 6 элементов */
    MPI_Unpack(&buf, 100, &pos, &a, 1, MPI_FLOAT, MPI_COMM_WORLD);
    MPI_Unpack(&buf, 100, &pos, &n, 1, MPI_INT, MPI_COMM_WORLD);
```


Упаковка-распаковка данных для передачи

Формирование сообщений при помощи упаковки и распаковки данных:

- данный подход приводит к необходимости дополнительных действий по упаковке и распаковке данных,
- он может быть оправдан при сравнительно небольших размерах сообщений и при малом количестве повторений,
- упаковка и распаковка может оказаться полезной при явном использовании буферов для буферизованного способа передачи данных.