

Коллективные операции передачи данных

Коллективные операции передачи данных...

Под *коллективными операциями* в MPI понимаются операции данных, в которых принимают участие все процессы используемого коммуникатора. Причем гарантировано, что эти операции будут выполняться гораздо эффективнее, поскольку MPI-функция реализована с использованием внутренних возможностей коммуникационной среды.

Коллективные операции передачи данных...

Обобщенная передача данных от одного процесса всем процессам...

- *Распределение данных* – ведущий процесс (*root*) передает процессам различающиеся данные

```
int MPI_Scatter(void *sbuf, int scount, MPI_Datatype stype,  
               void *rbuf, int rcount, MPI_Datatype rtype,  
               int root, MPI_Comm comm),
```

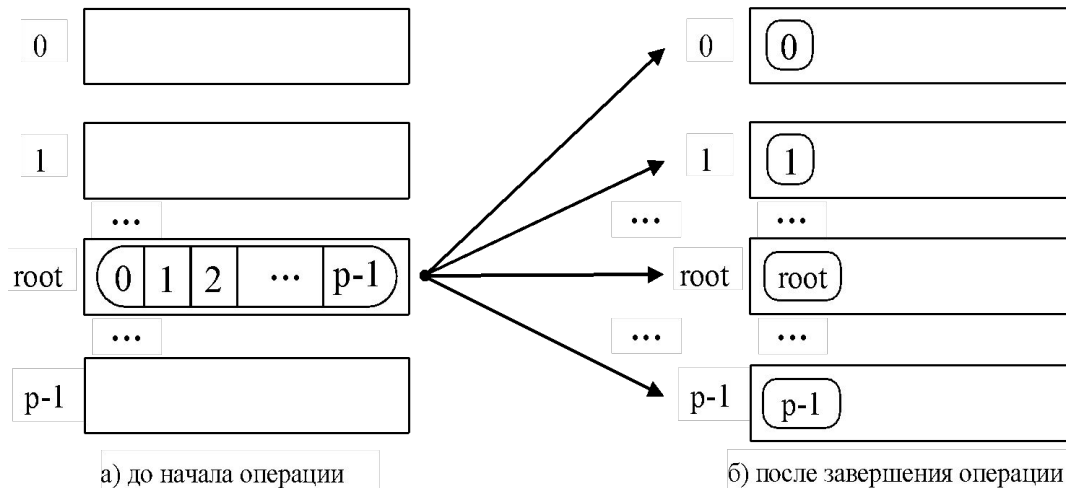
где

- **sbuf**, **scount**, **stype** – параметры передаваемого сообщения (**scount**,
– определяет количество элементов, передаваемых на каждый процесс),
- **rbuf**, **rcount**, **rtype** – параметры сообщения, принимаемого в процессах,
- **root** – ранг процесса, выполняющего рассылку данных,
- **comm** – коммутатор, в рамках которого выполняется передача данных.

Коллективные операции передачи данных...

Обобщенная передача данных от одного процесса всем процессам

- Вызов *MPI_Scatter* при выполнении рассылки данных должен быть обеспечен в каждом процессе коммутатора,
- *MPI_Scatter* передает всем процессам сообщения одинакового размера. Если размеры сообщений для процессов могут быть разными, следует использовать функцию *MPI_Scatterv*.

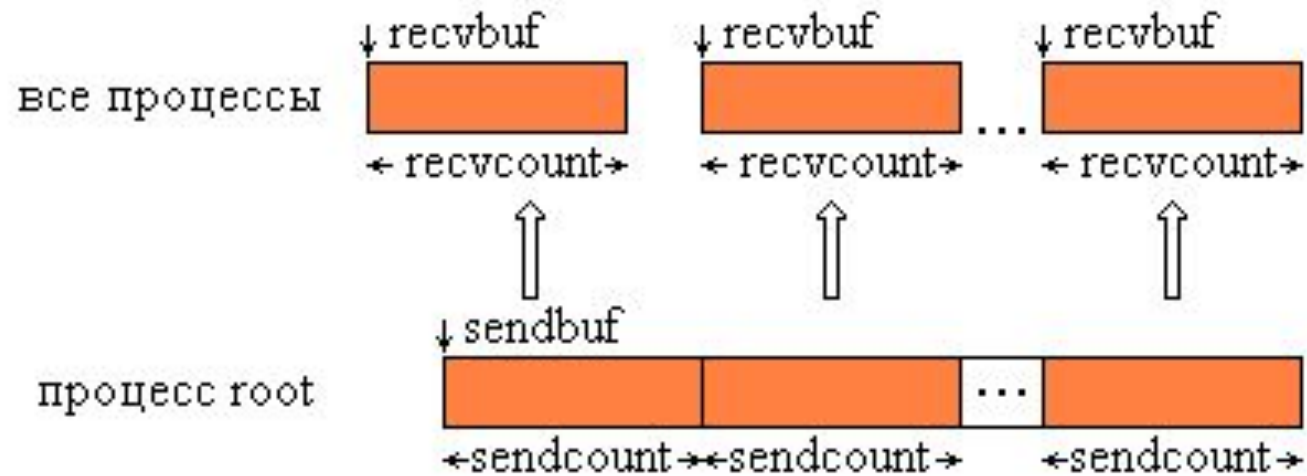


Основы параллельных вычислений:

*Моделирование и анализ
параллельных вычислений*

© Гергель В. П.

Графическая интерпретация операции Scatter



Рассылка различного количества данных

```
int MPI_Scatterv(void *sbuf, int *scounts, int *displs, MPI_Datatype stype,  
                void *rbuf, int rcount, MPI_Datatype rtype, int root, MPI_Comm comm)
```

sbuf - адрес рассылаемого массива данных.

Начало рассылаемых порций задает массив **displs**, количество элементов в порции задает массив **scounts**.

scounts – целочисленный массив, содержащий количество элементов, передаваемых каждому процессу (индекс равен рангу адресата, длина равна числу процессов в коммутаторе).

displs – целочисленный массив, содержащий смещения относительно начала массива sbuf (индекс равен рангу адресата, длина равна числу процессов в коммутаторе).

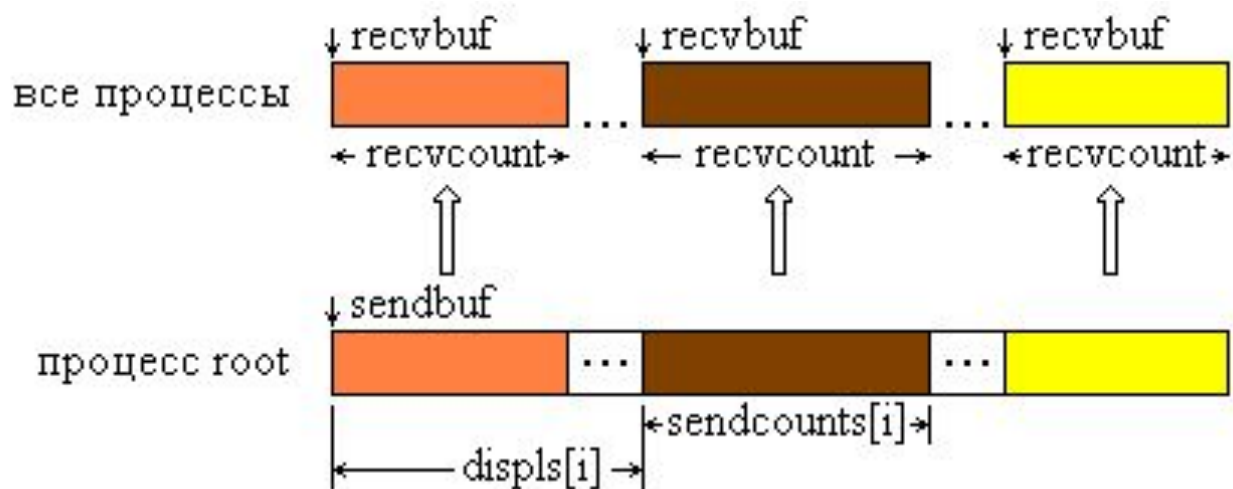
rbuf - адрес массива, принимающего порцию данных в i-ом процессе.

rcount - размер порции, принимаемой в ранге адресата .

root - ранг процесса, выполняющего рассылку данных.

stype, rtype – типы рассылаемых и принимаемых данных

Графическая интерпретация операции Scatterv



Определение частей массива в rank's

```
...
count=m / size;  ost=m % size;
/* Calculating parts of array for root rank */
if (rank==0) /* Process 0 - master */ {
/* Creation auxiliary arrays for data communication */
displs = (int *)malloc(size * sizeof(int));
rcounts = (int *)malloc(size * sizeof(int));
for(i=0;i < size;i++){
scol = i < ost ? count+1 : count;          rcounts[i] = scol;
nach = i*scol + (i >= ost ? ost : 0);      displs[i] = nach;
}
} /* End of work process 0 */
...
/* Calculating parts of vector vA for rank in others processes */
scol = rank < ost ? count+1 : count;
/* Offset (in strings) part for rank in vector vA */
nach = rank*scol + (rank >= ost ? ost : 0);
```


Коллективные операции передачи данных...

Обобщенная передача данных от всех процессов одному процессу...

- Передача данных от всех процессоров одному процессу (*сбор данных*) является обратной к операции распределения данных

```
int MPI_Gather(void *sbuf, int scount, MPI_Datatype stype,  
               void *rbuf, int rcount, MPI_Datatype rtype,  
               int root, MPI_Comm comm),
```

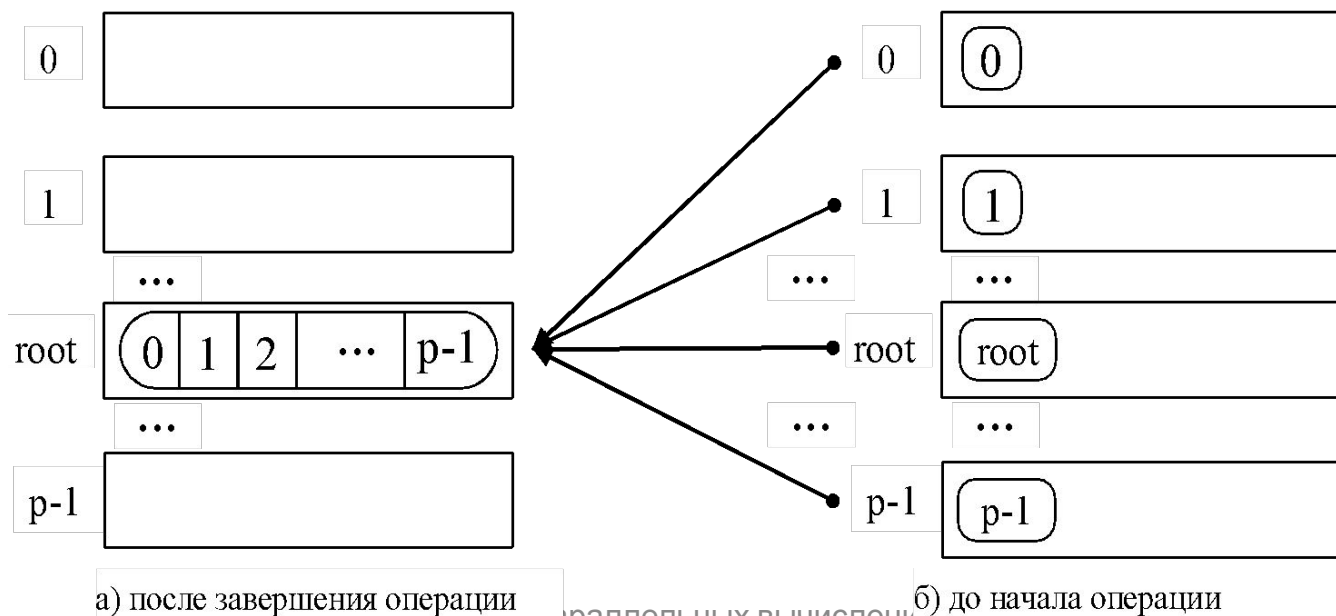
где

- **sbuf**, **scount**, **stype** - параметры передаваемого сообщения,
- **rbuf**, **rcount**, **rtype** - параметры принимаемого сообщения,
- **root** - ранг процесса, выполняющего сбор данных,
- **comm** - коммутатор, в рамках которого выполняется передача данных.

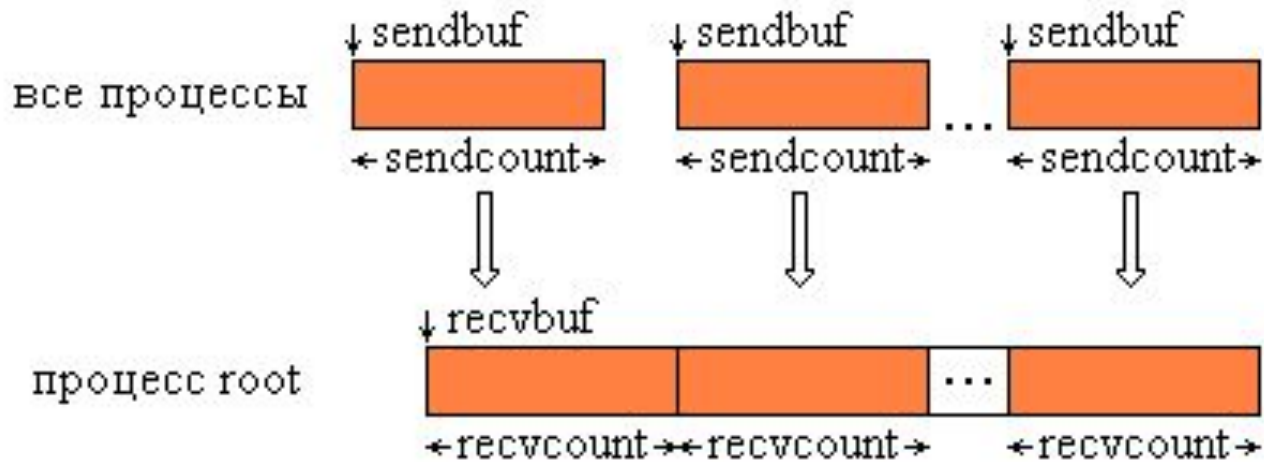
Коллективные операции передачи данных...

Обобщенная передача данных от всех процессов одному процессу...

- *MPI_Gather* определяет коллективную операцию, и ее вызов при выполнении сбора данных должен быть обеспечен в каждом процессе коммутатора



Графическая интерпретация операции Gather



Сборка различного количества данных со всех процессов в один процесс

```
int MPI_Gatherv(void *sbuf, int scount, MPI_Datatype stype,  
               void *rbuf, int *rcounts, int *displs, MPI_Datatype rtype, int root,  
               MPI_Comm comm)
```

sbuf – адрес рассылаемых данных в процессе отправителе.

scount – число рассылаемых данных.

rbuf – адрес буфера в принимающем процессе

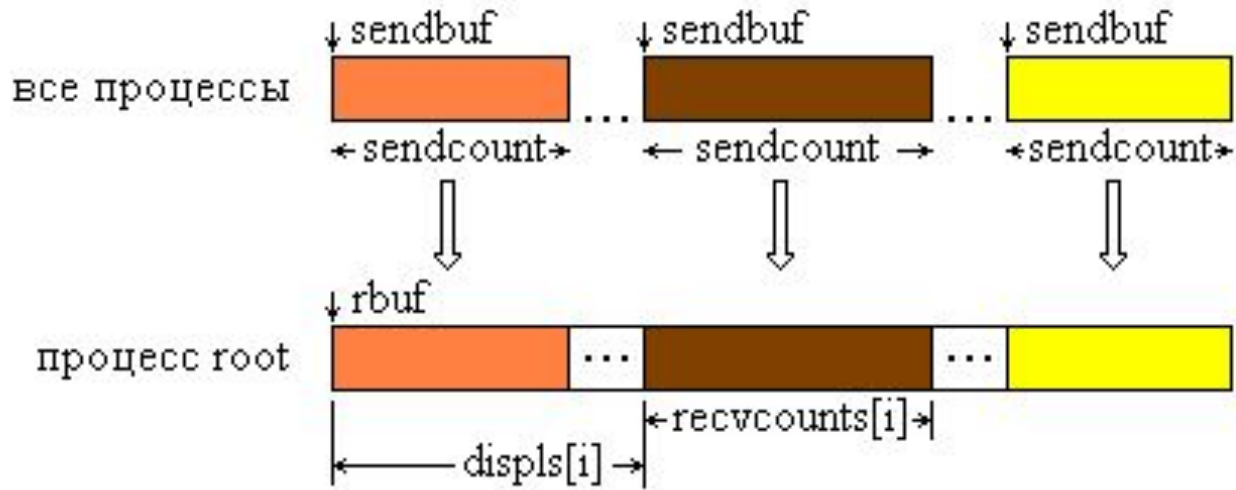
rcounts – целочисленный массив, содержащий количество элементов, принимаемых от каждого процесса (индекс равен рангу процесса, длина равна числу процессов в коммутаторе).

displs – целочисленный массив, смещений относительно начала массива rbuf (индекс равен рангу адресата, длина равна числу процессов в коммутаторе).

root – ранг принимающего процесса.

stype, rtype – типы рассылаемых и принимаемых данных.

Графическая интерпретация операции Gatherv

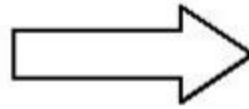


данные

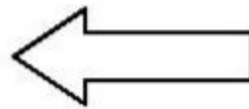
процессы

A_0	A_1	A_2	A_3	A_4	A_5

scatter



gather



A_0					
A_1					
A_2					
A_3					
A_4					
A_5					

Коллективные операции передачи

данных...

Обобщенная передача данных от всех процессов одному процессу

- *MPI_Gather* собирает данные на одном процессе. Для получения всех собираемых данных на каждом процессе нужно использовать *функцию сбора и*

```
int MPI_Allgather(void *sbuf, int scount, MPI_Datatype
stype,
    void *rbuf, int rcount, MPI_Datatype rtype, MPI_Comm
comm) .
```

- В случае, когда размеры передаваемых процессами сообщений могут быть различны, для передачи данных необходимо использовать функции *MPI_Gatherv* и *MPI_Allgatherv*.

Основы параллельных вычислений:

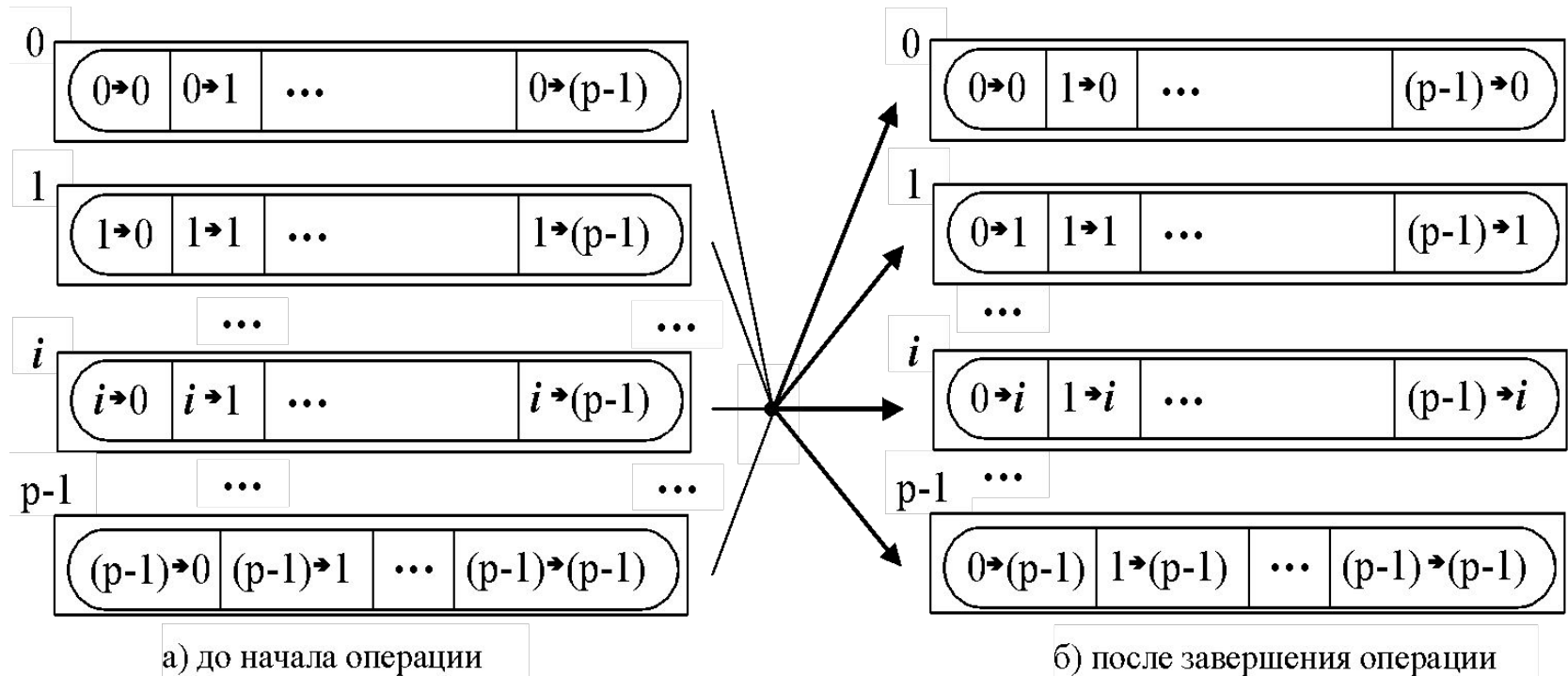
*Моделирование и анализ
параллельных вычислений*

© Гергель В.П.

Коллективные операции передачи данных...

данных...

Общая передача данных от всех процессов всем процессам...

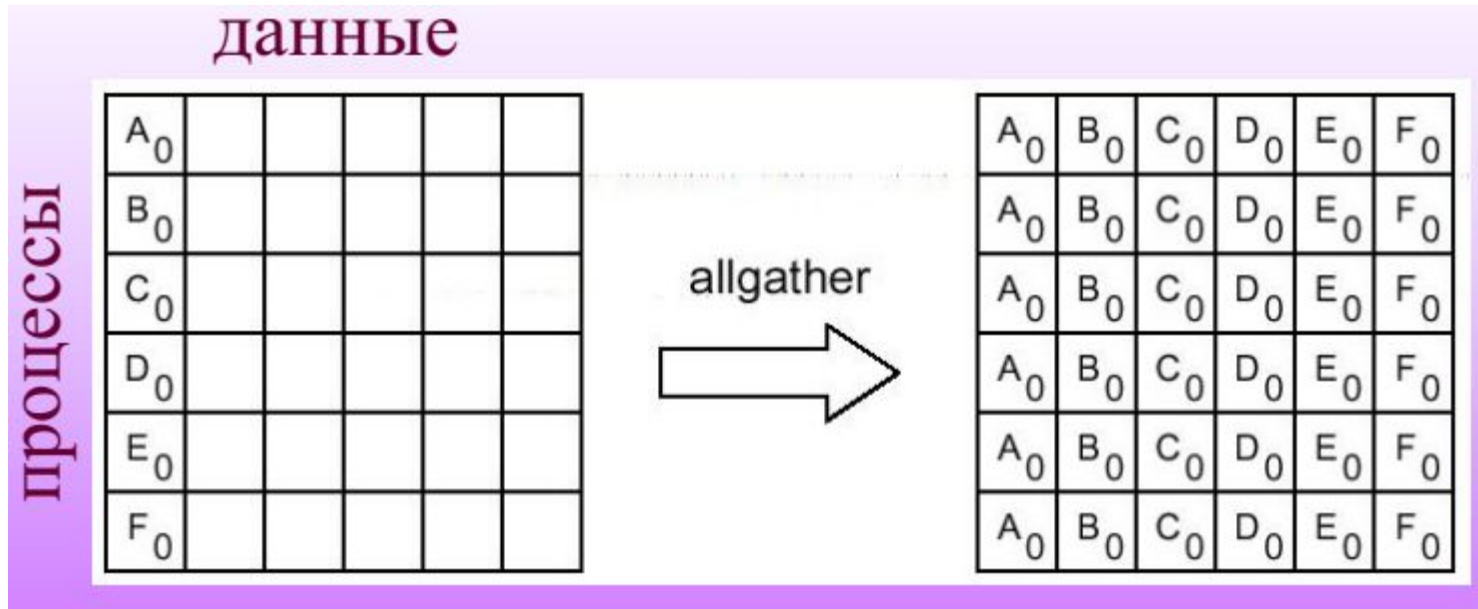


Основы параллельных вычислений:

*Моделирование и анализ
параллельных вычислений*

© Гергель В. П.

Коллективные операции передачи данных...



Коллективные операции передачи данных...

```
int MPI_Alltoall(void *sbuf, int scount, MPI_Datatype stype,  
void *rbuf, int rcount, MPI_Datatype rtype, MPI_Comm comm)
```

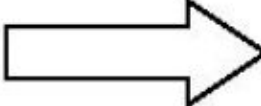
Рассылка каждым процессом коммуникатора **comm** различных порций данных всем другим процессам. **j**-й блок массива **sbuf** процесса **i** попадает в **i**-й блок массива **rbuf** процесса **j**.

данные

процессы

A ₀	A ₁	A ₂	A ₃	A ₄	A ₅
B ₀	B ₁	B ₂	B ₃	B ₄	B ₅
C ₀	C ₁	C ₂	C ₃	C ₄	C ₅
D ₀	D ₁	D ₂	D ₃	D ₄	D ₅
E ₀	E ₁	E ₂	E ₃	E ₄	E ₅
F ₀	F ₁	F ₂	F ₃	F ₄	F ₅

alltoall



A ₀	B ₀	C ₀	D ₀	E ₀	F ₀
A ₁	B ₁	C ₁	D ₁	E ₁	F ₁
A ₂	B ₂	C ₂	D ₂	E ₂	F ₂
A ₃	B ₃	C ₃	D ₃	E ₃	F ₃
A ₄	B ₄	C ₄	D ₄	E ₄	F ₄
A ₅	B ₅	C ₅	D ₅	E ₅	F ₅

Коллективные операции передачи

данных...

Дополнительные операции редукции данных...

- *MPI_Reduce* обеспечивает получение результатов редукции данных только на одном процессе,
- Функция *MPI_Allreduce* редукции и рассылки выполняет рассылку между процессами всех результатов операции редукции:

```
int MPI_Allreduce(void *sendbuf, void *recvbuf, int count,  
                 MPI_Datatype type, MPI_Op op, MPI_Comm comm)
```

- Возможность управления распределением этих данных между процессами предоставляется функций *MPI_Reduce_scatter*,
- Функция *MPI_Scan* производит операцию сбора и обработки данных, при которой обеспечивается получение и всех частичных результатов редуцирования

```
int MPI_Scan(void *sendbuf, void *recvbuf, int count,  
            MPI_Datatype type, MPI_Op op, MPI_Comm comm)
```

Основы параллельных вычислений:

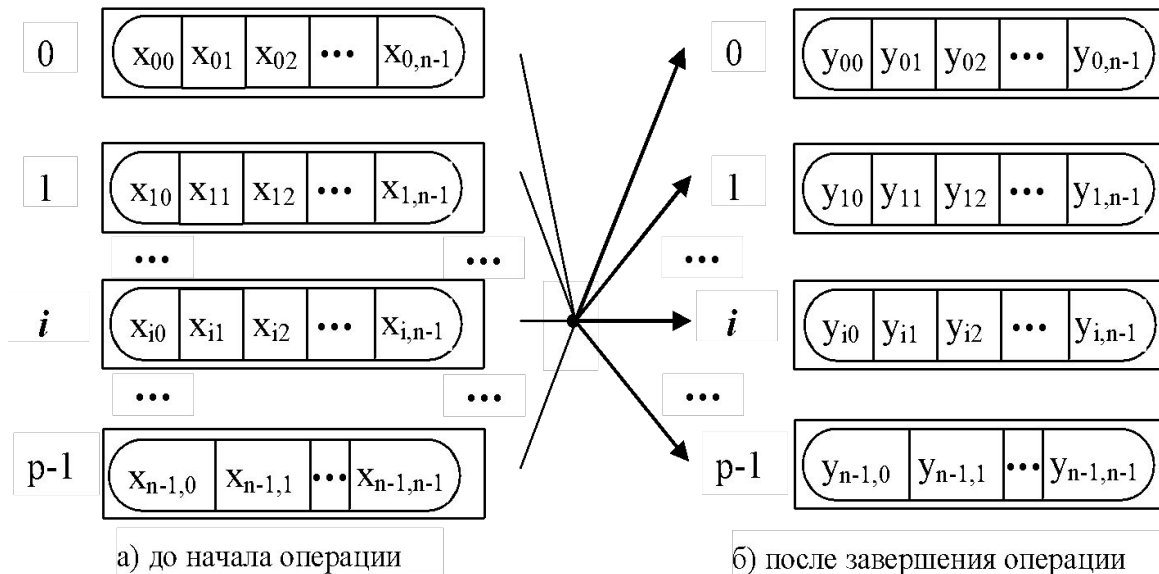
*Моделирование и анализ
параллельных вычислений*

© Гергель В.П.

Коллективные операции передачи данных

Дополнительные операции редукции данных

- При выполнении функции *MPI_Scan* элементы получаемых сообщений представляют собой результаты обработки соответствующих элементов передаваемых процессами сообщений, при этом для получения результатов на процессе с рангом i , $0 \leq i < n$, используются данные от процессов, ранг которых меньше или равен i



Основы параллельных вычислений:

*Моделирование и анализ
параллельных вычислений*

© Гергель В. П.

Графическая интерпретация операции Scan

