

Взаимодействия типа точка-точка

НОЦ, МФТИ, 2011

Передача типа точка-точка

- Взаимодействуют два и только два процесса
- Один процесс отсылает данные, другой принимает
- Существует несколько типов пересылок для различных целей
- Любой тип передающей операции может быть принят любой принимающей операцией
- Утильные операции — сообщения о доставке и ожидание доставки

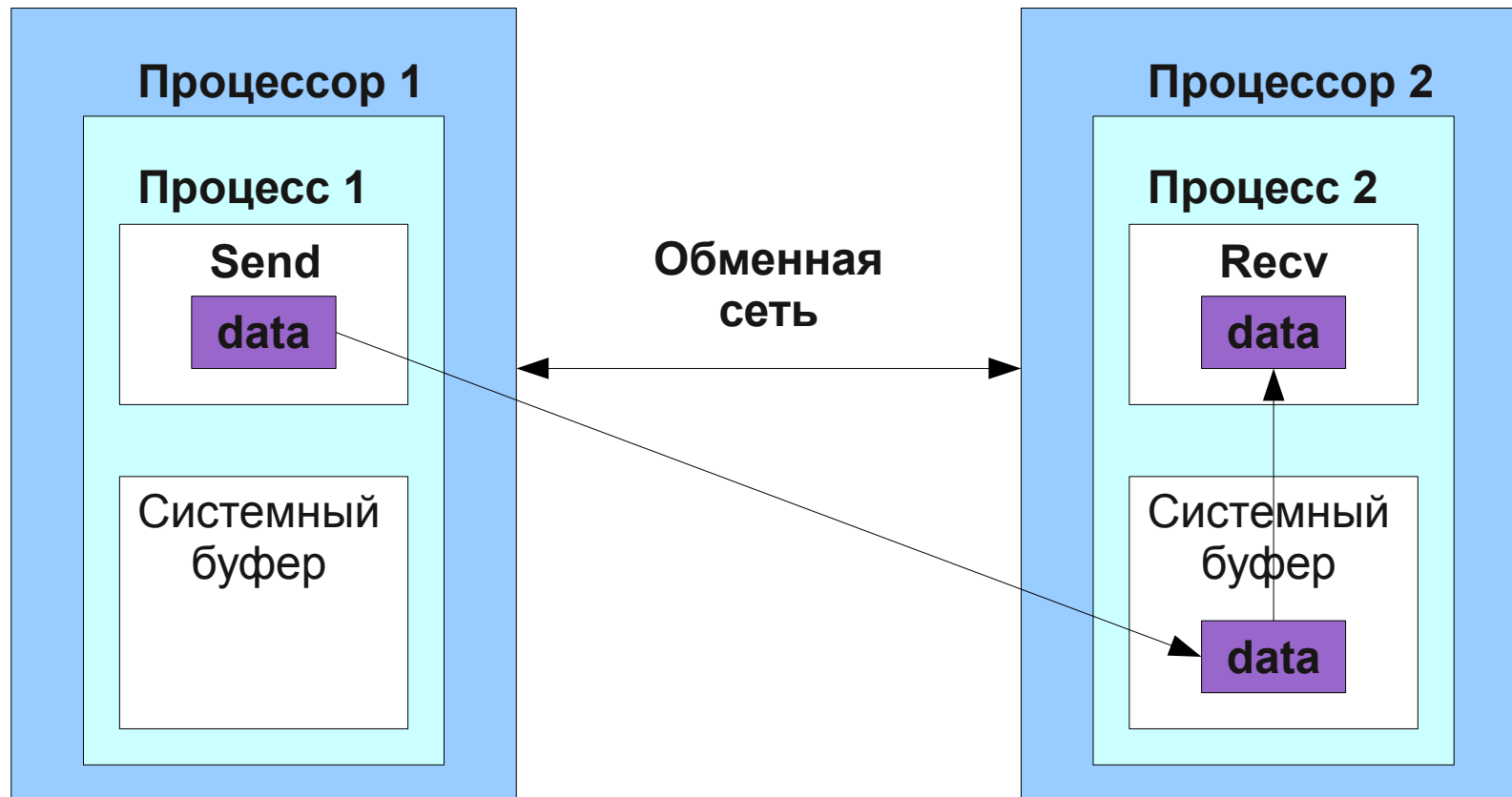
Типы пересылок

- Синхронная отсылка
- Блокирующая отсылка/прием
- Неблокирующая отсылка/прием
- Буферизованная отсылка
- Совместная отсылка-прием
- «Ready» отсылка

Буферизация в МРІ

- Буферизация:
 - Процесс не в состоянии принять сообщение в данный момент
 - Одновременный прием нескольких сообщений
- Спецификация не регламентирует реализацию системного буфера
- Каждая реализация имеет свою буферизацию
- Имеется возможность использования своего буфера (буферизированная отсылка)

Системный буфер в MPI



Системный буфер в MPI

- Полностью скрыт от программиста и вся работа осуществляется средствами библиотеки
- Ограниченный ресурс, за размеры которого легко вылезти
- Может быть на отсылающей, на принимающей или обеих сторонах
- Иногда позволяет увеличить скорость работы программы за счет асинхронных взаимодействий

Блокирующие пересылки

- Возвращают только когда отсылающий буфер может дальше использоваться в программе
- Синхронная отсылка — возвращает только когда получатель подтвердил прием
- Асинхронная отсылка — данные помещен в системный буфер и отсылающий буфер свободен
- Блокирующий прием — когда сообщение принято и данные готовы к использованию

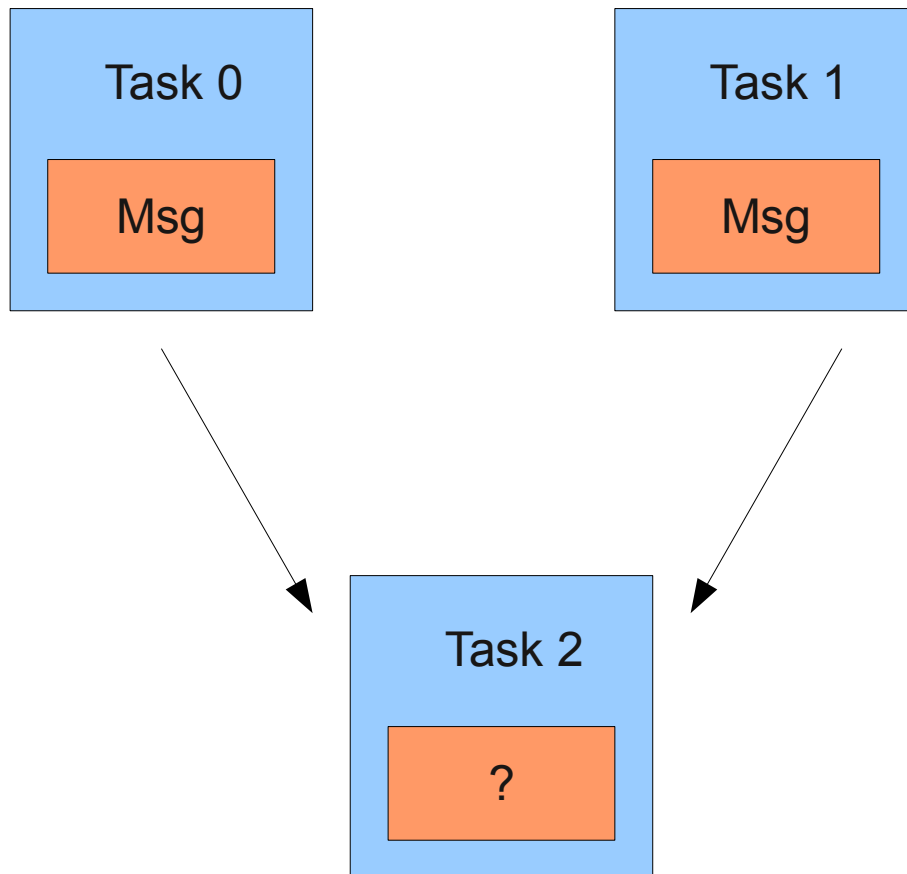
Неблокирующие пересылки

- Возвращают сразу
- Операции просто запрашивают библиотеку MPI отправить сообщение когда это будет возможно
- Небезопасно менять данные в отсылаемом буфере
- Можно подождать отсылки сообщений
- Используются для избежания блокировок и возможных улучшений производительности

Порядок отсылки

- MPI гарантирует, что сообщения не обгоняют друг-друга
- Если процесс послал два сообщения другому процессу и они прошли без ошибок, то они придут в той же последовательности
- Порядок не учитывается в многопоточных приложениях

Четность



Порядок прихода сообщений не гарантируется и полностью ложится на программиста

Синтаксис функций

Блокирующая отсылка

```
MPI_Send(buffer,count,type,dest,tag,comm)
```

Неблокирующая отсылка

```
MPI_Isend(buffer,count,type,dest,tag,comm,request)
```

Блокирующий прием

```
MPI_Recv(buffer,count,type,source,tag,comm,status)
```

Неблокирующий прием

```
MPI_Irecv(buffer,count,type,source,tag,comm,request)
```

Параметры

- **Buffer** — указатель на область памяти, откуда читаются/пишутся данные
- **Data Count** — число элементов конкретного типа, который будет переслан
- **Data Type** — тип данных
- **Destination** — процесс, который должен принять данные
- **Source** — процесс, от которого принимаем данные
- **Tag** — уникальный идентификатор сообщения
- **Communicator** — коммуникатор
- **Status** — структура, в которую заносятся данные о принятом сообщении
- **Request** — идентификатор сообщения при неблокирующих пересылках

Типы данных

- Необходимо указывать чтобы MPI знало о размерах типа и особенностях его расположения в памяти
- Есть набор стандартных типов для основных типов языка C и Fortran
- Программист может создавать свои новые типы данных

Встроенные типы MPI

C		Fortran	
MPI_CHAR	char	MPI_CHARACTER	character(1)
MPI_SHORT	short		
MPI_INT	int	MPI_INTEGER	integer
MPI_LONG	long		
MPI_UNSIGNED_CHAR	unsigned char		
MPI_UNSIGNED_SHORT	unsigned short		
MPI_UNSIGNED	unsigned int		
MPI_UNSIGNED_LONG	unsigned long int		
MPI_FLOAT	float	MPI_REAL	real
MPI_DOUBLE	double	MPI_DOUBLE_PRECISION	double precision
MPI_LONG_DOUBLE	long double		
		MPI_COMPLEX	complex
		MPI_DOUBLE_COMPLEX	double complex
		MPI_LOGICAL	logical
MPI_BYTE	8 binary digits	MPI_BYTE	8 binary digits

Блокирующие операции

- `MPI_Send(&buf,count,datatype,dest,tag,comm)`
 - Основная блокирующая операция отсылки
 - Возвращает только когда буфер безопасен для использования
 - Может быть реализована по разному в различных реализациях (в частности через `Ssend`)
- `MPI_Recv(&buf,count,datatype,source,tag,comm,&status)`
 - Основная операция блокирующего приема
 - Возвращает как только в буфере находится все сообщение

Блокирующие операции

- `MPI_Ssend(&buf, count, datatype, dest, tag, comm)`
 - Синхронная блокирующая отсылка
 - Возвращает когда буфер безопасен для использования и принимающая сторона уже начала прием сообщения
- `MPI_Bsend(&buf, count, datatype, dest, tag, comm)`
 - Буферизированная блокирующая отсылка
 - Позволяет копировать данные во временный буфер выделенный программистом
 - Возвращает как только данные скопированы в буфер приложения
 - Работает вместе с `MPI_Buffer_attach`

Блокирующие операции

- **MPI_Buffer_attach(&buffer,size)**
 - Выделяет буфер для использования операциями буферизированных отсылок
 - Размер показывает число байт
 - Только один буфер может быть подключен к процессу в один момент времени
- **MPI_Buffer_detach(&buffer,size)**
 - Отключает буфер для буферизированных отсылок

Блокирующие операции

- `MPI_Rsend(&buf, count, datatype, dest, tag, comm)`
 - Блокирующая «ready» отсылка
 - Используется только если программист уверен что прием уже начался
- `MPI_Sendrecv(&sendbuf, sendcount, sendtype, dest, sendtag, &recvbuf, recvcount, recvtype, source, recvtag, comm, &status)`
 - Начинает прием перед блокирующей отсылкой
 - Возвращает когда оба буфера безопасны для использования

Неблокирующие операции

- `MPI_Isend(&buf, count, datatype, dest, tag, comm, &request)`
 - Неблокирующая асинхронная отсылка
 - Возвращает сразу, не дожидаясь копирования данных в системный буфер
 - Возвращает `request` для управления отсылкой
 - Перед использованием буфера необходимо вызвать `MPI_Wait` или `MPI_Test`
- `MPI_Irecv(&buf, count, datatype, source, tag, comm, &request)`
 - Неблокирующий асинхронный прием
 - Возвращает сразу, не дожидаясь копирования данных
 - Возвращает `request` для управления приемом

Неблокирующие операции

- `MPI_Issend(&buf, count, datatype, dest, tag, comm, &request)`
 - Аналогиче `MPI_Isend`, только `MPI_Wait` и `MPI_Test` показывают когда сообщение уже началось приниматься
- `MPI_Ibsend(&buf, count, datatype, dest, tag, comm, &request)`
 - Неблокирующая буферизированная отсылка
 - Использует программный буфер для хранения временных данных
 - Используется вместе с `MPI_Buffer_attach`
- `MPI_Irsend(&buf, count, datatype, dest, tag, comm, &request)`
 - «Ready» неблокирующая отсылка
 - Должна вызываться когда уже начался прием

MPI_Wait

- MPI_Wait (&request,&status)
 - MPI_Waitany (count,&array_of_requests,&index,&status)
 - MPI_Waitall (count,&array_of_requests,&array_of_statuses)
 - MPI_Waitsome(incount,&array_of_requests,&outcount,
&array_of_offsets, &array_of_statuses)
-
- Ждет завершения блокирующих операций
 - Можно ждать несколько отсылок/приемов
 - Можно ждать только одного из нескольких

MPI_Test

- Аналогично MPI_Wait, но только проверяет состояние
 - Результат записывает в flag: 1 — операция завершена, 0 — в противном случае
-
- MPI_Test (&request, &flag, &status)
 - MPI_Testany
(count, &array_of_requests, &index, &flag, &status)
 - MPI_Testall
(count, &array_of_requests, &flag, &array_of_statuses)
 - MPI_Testsome(incount, &array_of_requests, &outcount, &array_of_offsets, &array_of_statuses)