PARALLEL PROGRAMMING III

Scuola Internazionale Superiore
di Studi Avanzati

The Abdus Salam
International Centre
for Theoretical Physics

SendRecv

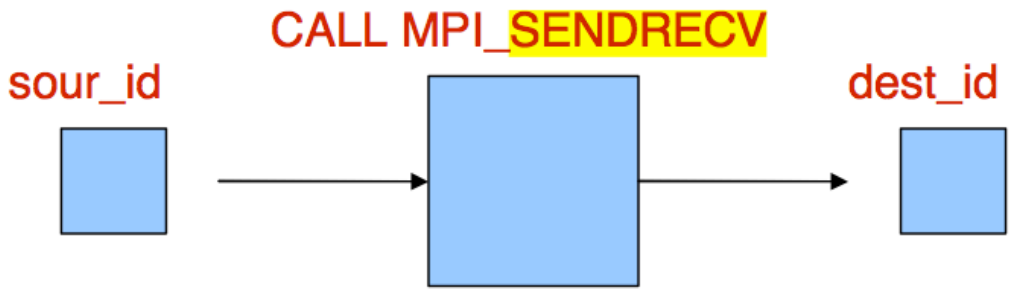
The easiest way to send and receive data without warring about deadlocks

Sender side

Fortran:

```
CALL MPI_SENDRECV(sndbuf, snd_size, snd_type, dest_id, tag,
rcvbuf, rcv_size, rcv_type, sour_id, tag, comm, status, ierr)
```

Receiver side

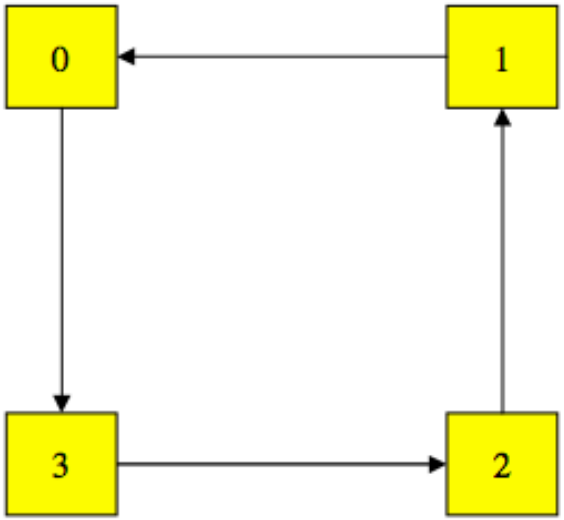


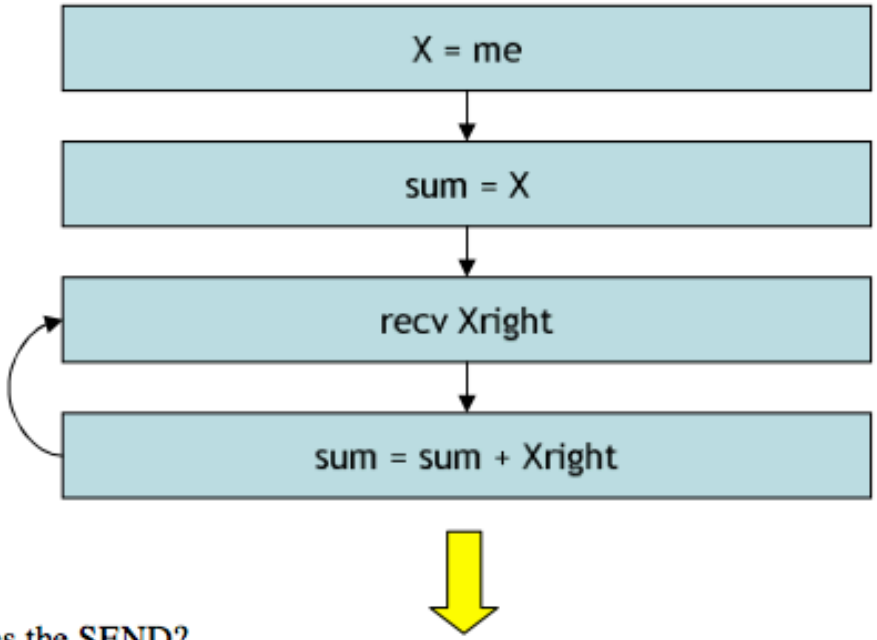
```
PROGRAM send_recv
INCLUDE 'mpif.h'
INTEGER ierr, myid, nproc
INTEGER status(MPI_STATUS_SIZE)
REAL A(2), B(2)
CALL MPI_INIT(ierr)
CALL MPI_COMM_SIZE(MPI_COMM_WORLD, nproc, ierr)
CALL MPI_COMM_RANK(MPI_COMM_WORLD, myid, ierr)
IF( myid .EQ. 0 ) THEN
  a(1) = 2.0
  a(2) = 4.0
  CALL MPI_SENDRECV(a, 2, MPI_REAL, 1, 10, b, 2, MPI_REAL, 1, 11, MPI_COMM_WORLD, status, ierr)
ELSE IF( myid .EQ. 1 ) THEN
  a(1) = 3.0
  a(2) = 5.0
  CALL MPI_SENDRECV(a, 2, MPI_REAL, 0, 11, b, 2, MPI_REAL, 0, 10, MPI_COMM_WORLD, status, ierr)
END IF
WRITE(6,*) myid, ': b(1)=', b(1), ' b(2)=', b(2)
CALL MPI_FINALIZE(ierr)
END
```

Communication Cycle

```
! right to left !
call MPI_SENDRECV(snd_buffer, ibuf, MPI_REAL, right, 1, &
     rcv_buffer, ibuf, MPI_REAL, left , 1, &
     comm_cart, istatus, ierr)

! left to right !
call MPI_SENDRECV (snd_buffer, ibuf, MPI_REAL, left, 1, &
     rcv_buffer, ibuf, MPI_REAL, right , 1, &
     comm_cart, istatus, ierr)
```





Who/when does the SEND?

What does it send?

How many times?

Exercise

Implement the proposed exercise, first exchanging one single element (mype) among processes as illustrated in class as well as on the previous slide. Try to optimize the code for sending in the ring a large set of data and overlapping the computation (Σ) and the communication (send-recv). In case of a dataset larger than one element the local sum is considered a vector sum (element by element).