COMP 322: Fundamentals of Parallel Programming

Lecture 33: Introduction to the Message Passing Interface (MPI) cont.

Mack Joyner
mjoyner@rice.edu

http://comp322.rice.edu

Collective Communications

- A popular feature of MPI is its family of collective communication operations.
- Each collective operation is defined over a communicator (most often, MPI.COMM_WORLD)
  — Each collective operation contains an *implicit barrier*. The operation completes and execution continues when all processes in the communicator perform the *same* collective operation.
  —A mismatch in operations results in *deadlock* e.g.,

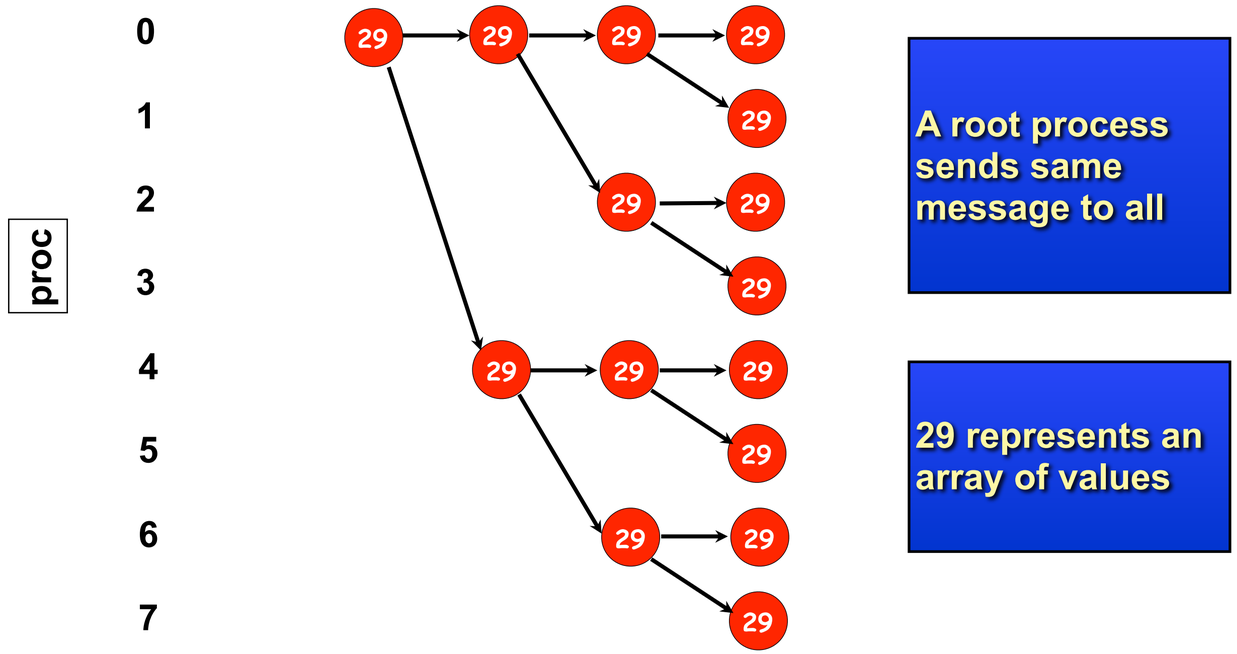
  ```
  Process 0: .... MPI.Bcast(...) ....
  Process 1: .... MPI.Bcast(...) ....
  Process 2: .... MPI.Gather(...) ….
  ```

- A simple example is the broadcast operation: all processes invoke the operation, all agreeing on one root process. Data is broadcast from that root.

  ```
  void Bcast(Object buf, int offset, int count, Datatype type, int root)
  ```


MPI Bcast

```
buf = new int[1]; if (rank==0) buf[0] = 29;
void Bcast(buf, 0, 1, MPI.INT, 0); // Executed by all processes
```



More Examples of Collective Operations

void Gather(Object sendbuf, int sendoffset, int sendcount, Datatype sendtype, Object recvbuf, int recvoffset, int recvcount, Datatype recvtype, int root)

- Each process sends the contents of its send buffer to the root process.

void Scatter(Object sendbuf, int sendoffset, int sendcount, Datatype sendtype, Object recvbuf, int recvoffset, int recvcount, Datatype recvtype, int root)

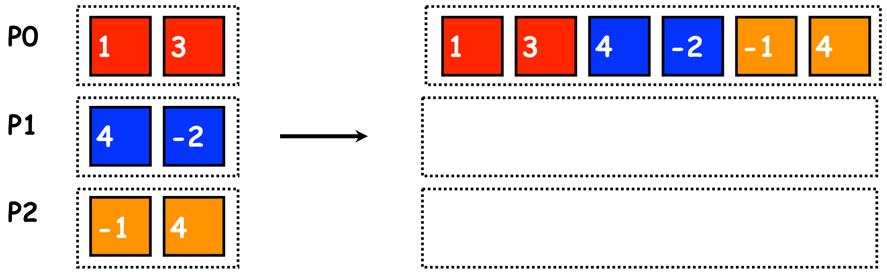
- Inverse of the operation Gather.

void Reduce(Object sendbuf, int sendoffset, Object recvbuf, int recvoffset, int count, Datatype datatype, Op op, int root)

- Combine elements in send buffer of each process using the reduce operation, and return the combined value in the receive buffer of the root process.

MPI Gather

- Use to copy an array of data from each process into a single array on a single process.
- Graphically:



- Note: only process 0 (P0) needs to supply storage for the output

void Gather(Object sendbuf, int sendoffset, int sendcount, Datatype sendtype, Object recvbuf, int recvoffset, int recvcount, Datatype recvtype, int root)

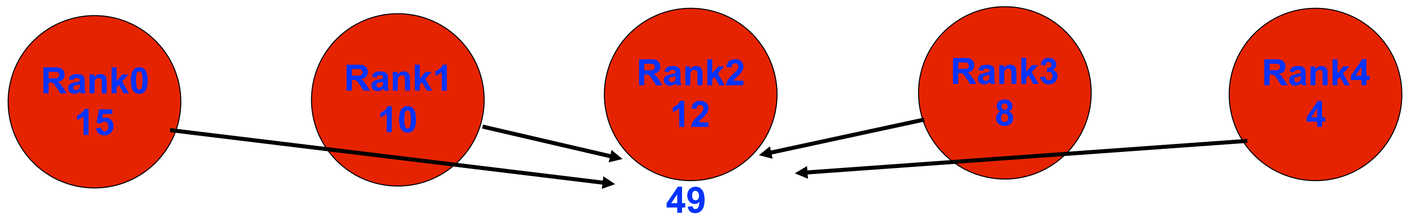
  - Each process sends the contents of its send buffer to the root process.

Predefined Reduction Operations

Operation	Meaning	Datatypes
`MPI_MAX`	Maximum	int, long, float, double
`MPI_MIN`	Minimum	int, long, float, double
`MPI_SUM`	Sum	int, long, float, double
`MPI_PROD`	Product	int, long, float, double
`MPI_LAND`	Logical AND	int, long
`MPI_BAND`	Bit-wise AND	byte, int, long
`MPI_LOR`	Logical OR	int, long
`MPI_BOR`	Bit-wise OR	byte, int, long
`MPI_LXOR`	Logical XOR	int, long
`MPI_BXOR`	Bit-wise XOR	byte, int, long
`MPI_MAXLOC`	max-min value-location	Data-pairs
`MPI_MINLOC`	min-min value-location	Data-pairs

Exercise: MPI Reduce

```
void MPI.COMM_WORLD.Reduce(
                Object    sendbuf  /* in */,
                int       sendoffset     /* in */,
                Object    recvbuf   /* out */,
                int       recvoffset     /* in */,
                int       count          /* in */,
                MPI.Datatype      datatype /* in */,
                MPI.Op    operator /* in */,
                int       root       /* in */ )
```



How would you write this using MPI Reduce?

More Collective Communication Operations

- If the result of the reduction operation is needed by all processes, MPI provides:

```
void AllReduce(Object sendbuf, int sendoffset, Object recvbuf, int recvoffset, int count, Datatype datatype,
               Op op)
```

- MPI also provides the MPI_AllGather function in which the data are gathered at all the processes.

```
void AllGather(Object sendbuf, int sendoffset, int sendcount, Datatype sendtype, Object recvbuf, i
               int recvoffset, int recvcount, Datatype recvtype)
```


Announcements & Reminders

- Quiz for Unit 7 is due Friday, April 17th at 11:59pm

- The entire written + programming (Checkpoint #2) is due by Wednesday, April 22nd at 11:59pm

Worksheet #33: MPI_Gather

In the space below, indicate what value should be provided instead of ??? in line 6, and how it should depend on myrank.

```
2.  MPI.Init(args) ;
3.   int myrank = MPI.COMM_WORLD.Rank() ;
4.   int numProcs = MPI.COMM_WORLD.Size() ;
5.   int size = ...;
6.   int[] sendbuf = new int[size];
7.   int[] recvbuf = new int[???];
8.   . . . // Each process initializes sendbuf
9.   MPI.COMM_WORLD.Gather(sendbuf, 0, size, MPI.INT,
10.                         recvbuf, 0, size, MPI.INT,
11.                        0/*root*/);
12.  . . .
13.  MPI.Finalize();
```