L6 - ECE 669 (not in textbook)

\[ \text{MPI} = \text{Message Passing Interface} \]

Processes: 0 1 2 3 4 ...
Message-Passing Programming Paradigm

- All variables are private.
- Processes communicate via subroutine calls.
- Typically written in a conventional sequential language (e.g., C, Fortran).
- Messages: "packets of data" moving between processes.

Encoded information:
- Sending process, source location, data type, length, size of buffers.
- Payload, header, bits.

Fig. A message
MPI - Important Concepts
- Communicators
- Point-to-Point Communication
- Collective Communication
- Virtual Topologies

Standard?

- Two years of proposal meeting
- Europe & US
- 60 people, 40 organizations
- First standard for message-passing
- "MPI document" produced
Point-to-Point

- from process to process
- different semantics for send & receive(s)
  - when the operation terminates?
  - when can resources (e.g., buffers) be reused?

Blocking Nodes \[\rightarrow\] Synchronous
Blocking Nodes \[\rightarrow\] Standard
Non-blocking Nodes \[\rightarrow\] Buffered

SEMANTICS?
Non-Blocking Communication

* Three phases
  1. initiate communication
  2. do some work
  3. wait for non-blocking communication to complete
Example

Non-blocking Synchronous Send

MPI_Issend (buf, count, datatype, dest, tag, comm, handle)

MPI_Wait (handle, status, ierror)

MPI_Recv (buf, count, datatype src, tag, comm, handle)

comm = Communicator = group of processes involved in communication.
Synchronous lends

Blocking

Non-blocking

"Beep"

Work could be done! Here fast or wait()
Asynchronous sends or buffered

Analogy - just put it in Mailbox
(- here is buffer!)

Blocking semantics

{ 'wait until buffer can be reused'

Non-blocking

initial: send

do other work

Note: independent of Receive!
Standard tends in MPI

- either synchronous or buffered....

- depends on message size and resources available....

- can mean different things, i.e., implemenation specific!

- if you expect certain behavior use synchronous or buffered modes!
Ready Mode
= immediate return!
from MPI

- Not recommended!
- Useful when no other message is sent to a processor, and no information about resources/completion time is needed!
Implementation Issues:
Point-to-Point Data Transfer Protocols on CRAY T3D

Transfer - Acknowledge (TA)
Request - Transfer - Acknowledge (RTA)
Request - Acknowledge - Transfer (RAT)
Derived Data Types in MPI

**Contiguous:** (address, size)

**Strided:** (address, numFragments, fragment size, stride size)

Example of MPI Datatypes:

- **MPI_INT:**
  - newtype: block0  block1
  - count = 2; array_of_block_lengths[0] = 1;
  - array_of_types[0] = MPI_INT;
  - array_of_types[1] = MPI_DOUBLE;

- **MPI_DOUBLE:**
Virtual Topologies

- Convenient process naming
- Simplifies writing of code
- Allows MPI to do optimizations

- Types
  - Cartesian
  - Virtual grid
    - Processes identified with Cartesian coordinates
  - Graph
    (not covered here)

- How to use it?
  - Create a topology -> new communicate
  - MPI mapping functions compute processor ranks
  - Partitioning?
Example: A cylinder
- MPI provides the mapping

Rank of $(2,1)$? $\overset{\text{mpi}}{\implies} 9$
Collective Communications

- Involving a group (a "communicator")
- Called by ALL in the communicator involved
- Types:
  - Barrier
  - Broadcast, scatter, gather
  - Global sum, etc

- All MPI collective communications are blocking!
Example API:

```c
int MPI_BARRIER ( MPI_Comm comm )

int MPI_Bcast ( void *buffer, int count, MPI_Datatype t, int root, MPI_Comm comm )
```
Scatter

to

\[ t_i \]

"root"

A, B, C

A, B, C

13

C

PL7.
Gather

rank 0

rank 1

rank 2

A

B

C

D

E

P. 18.
Global Reduction

MPI - Reduce

Operations:
- min
- max
- sum
- AND
- OR
- user-defined

Rank
0
- A, B, C, D
- "Root"
1
- E, F, G, H
2
- J, K, L
3
- M, N, O, P

P.19.
Variants (same example)

MPI - ALLREDUCE
"no root"

Rank
0
1
2
3

MPI - SCAN
"parallel prefix"

P. 20.
Problem for home

1. Rewrite the "equation solver" from textbook in MPI

2. Rewrite the "matrix multiply" with MPI

Note: you will need the MPI manual for this