COL380

Introduction to Parallel & Distributed Programming

Agenda

- MPI

Starting and Ending

- MPI_Init(&argc, &argv);
  - Needed before any other MPI call

MPI_Init_thread

```
int nump, id;
MPI_Comm_size (MPI_COMM_WORLD, &nump);
MPI_Comm_rank (MPI_COMM_WORLD, &id);
```

- MPI_Finalize();
  - Required

Send/Receive

```
int MPI_Send(void* buf, int count, MPI_Datatype datatype, int dest,
  int tag, MPI_Comm comm)
```

message contents	block of memory
count	number of items in message
message type	MPI_Datatype of each item
destination	rank of recipient
tag	integer “message identifier”
communicator	

Send/Receive

int MPI_Send(void* buf, int count, MPI_Datatype datatype, int dest, int tag, MPI_Comm comm)

int MPI_Recv(void* buf, int count, MPI_Datatype datatype, int source, int tag, MPI_Comm comm, MPI_Status *status)

s	block of memory
	number of items in messag
	MPI_Datatype of each iten
	rank of recipient
	integer “message identifier

• message contents	memory buffer to store received message
• count	space in buffer, overflow error if too small
• message type	type of each item
• source	sender’s rank (or MPI_ANY_SOURCE)
• tag	message identifier (or MPI_ANY_TAG)
• communicator	
• status	information about message received

Blocking calls

```
int MPI_Send(void* buf, int count, MPI_Datatype datatype, int dest,
   int tag, MPI_Comm comm)
```

```
int MPI_Recv(void* buf, int count, MPI_Datatype datatype, int
   source, int tag, MPI_Comm comm, MPI_Status *status)
```

s block of memory
number of items in messag
MPI_Datatype of each iten
rank of recipient
integer “message identifier

• message contents	memory buffer to store received message
• count	space in buffer, overflow error if too small
• message type	type of each item
• source	sender’s rank (or MPI_ANY_SOURCE)
• tag	message identifier (or MPI_ANY_TAG)
• communicator	
• status	information about message received

Blocking calls

Send/Receive

int MPI_Send(void* buf, int count, MPI_Datatype datatype, int dest, int tag, MPI_Comm comm)

MATCHING (Per context)

int MPI_Recv(void* buf, int count, MPI_Datatype datatype, int source, int tag, MPI_Comm comm, MPI_Status *status)

s block of memory
number of items in messag
MPI_Datatype of each iten
rank of recipient
integer “message identifier

• message contents	memory buffer to store received message
• count	space in buffer, overflow error if too small
• message type	type of each item
• source	sender’s rank (or MPI_ANY_SOURCE)
• tag	message identifier (or MPI_ANY_TAG)
• communicator	
• status	information about message received

Eager vs Rendezvous

Eager

- Send-stub packetizes and transmits
  (May save a local message copy)
- Send-stub signals <u>Done</u>
- Recv-stub continuously accepts
- Delivered when Recv call matches

Rendezvous

- Send-stub transmits envelope info
  (May save local message copy)
- Recv-stub continuously accepts envelope info
- Recv-stub may signal <u>OK</u> (if it has space)
  Or, wait for matching Recv call to be made
- Recv-stub sets up “RDMA” with Send-stub
- Data transmitted
- Recv-stub signals <u>Done</u>
- Send-stub signals <u>Done</u>

Subodh Kumar

Send/Recv Synchronization

- Blocking
  - ➡ Send returns after some progress guarantee
    - ▶ Receive completed?
    - ▶ Synchronization (up to network delay)
- Immediate
  - ➡ Send returns with no progress guarantee
  - ➡ Receiver may also proceed immediately (message arrives later)

Send Semantics

- Standard mode:
  - ➡ implementation dependent
- Buffered mode
  - ➡ MPI saves a copy of message, Receiver can post later
  - ➡ User provided buffer
- Synchronous mode
  - ➡ Will complete only once a matching receive has started
- Ready mode
  - ➡ Send may start only if a matching receive has already been called
  - ➡ Helps performance

Send Semantics

- Standard mode:
  - ➡ implementation dependent
- Buffered mode
  - ➡ MPI saves a copy of message, Receiver can post later
  - ➡ User provided buffer
- Synchronous mode
  - ➡ Will complete only once a matching receive has started
- Ready mode
  - ➡ Send may start only if a matching receive has already been called
  - ➡ Helps performance

- MPI_Send/MPI_Recv are blocking
  - ➡ Recv blocks until output buffer is filled
  - ➡ Send blocks until some ‘progress’

Send Semantics

- Standard mode: MPI_Send
  - ➡ implementation dependent
- Buffered mode MPI_Bsend
  - ➡ MPI saves a copy of message, Receiver can post later
  - ➡ User provided buffer See MPI_Buffer_attach
- Synchronous mode MPI_Ssend
  - ➡ Will complete only once a matching receive has started
- Ready mode MPI_Rsend
  - ➡ Send may start only if a matching receive has already been called
  - ➡ Helps performance

- MPI_Send/MPI_Recv are blocking
  - ➡ Recv blocks until output buffer is filled
  - ➡ Send blocks until some 'progress'

Review

- MPI Blocking Send- Receive
  - ➡ Semantics
  - ➡ Matching