Ateneo de Manila University

Parallel Programming with MPI: Persistent Communications

Ateneo High Performance Computing Group
1st Semester
http://www.math.admu.edu.ph/ahpc/
william.s.yu@ieee.org
Persistent Communications
Persistent Communication Requests

- Save arguments of a communication call
- Take overhead out of subsequent calls (e.g., in a loop)
- `MPI_SEND_INIT` creates a communication request that completely specifies a standard send operation
- `MPI_RECV_INIT` creates a communication request that completely specifies a standard recv operation
- Similar routines for ready, synchronous, and buffered send modes
MPI_SEND_INIT

MPI_SEND_INIT(buf, count, datatype, dest, tag, comm, request)

IN       buf     initial address of send buffer
IN       count   number of elements sent
IN       datatype type of each element
IN       dest    rank of destination
IN       tag     message tag
IN       comm    communicator
OUT      request communication request
MPI_SEND_INIT bindings

```c
int MPI_Send_init(void* buf, int count,
    MPI_Datatype datatype, int dest,
    int tag, MPI_Comm comm,
    MPI_Request *request)

Prequest Comm::Send_init(const void* buf, int count,
    const Datatype& datatype, int dest,
    int tag) const

MPI_SEND_INIT(BUF, COUNT, DATATYPE, DEST, TAG,
    COMM, REQUEST, IERROR)

<typename> BUF(*)
INTEGER REQUEST, COUNT, DATATYPE, DEST, TAG,
    COMM, REQUEST, IERROR
```
MPI_RECV_INIT

MPI_RECV_INIT(buf, count, datatype, source, tag, comm, request)

<table>
<thead>
<tr>
<th>Type</th>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OUT</td>
<td>buf</td>
<td>initial address of receive buffer</td>
</tr>
<tr>
<td>IN</td>
<td>count</td>
<td>number of elements received</td>
</tr>
<tr>
<td>IN</td>
<td>datatype</td>
<td>type of each element</td>
</tr>
<tr>
<td>IN</td>
<td>source</td>
<td>rank of source or MPI_ANY_SOURCE</td>
</tr>
<tr>
<td>IN</td>
<td>tag</td>
<td>message tag or MPI_ANY_TAG</td>
</tr>
<tr>
<td>IN</td>
<td>comm</td>
<td>communicator</td>
</tr>
<tr>
<td>OUT</td>
<td>request</td>
<td>communication request</td>
</tr>
</tbody>
</table>
MPI_RECV_INIT bindings

```c
int MPI_Recv_init(void* buf, int count,
                  MPI_Datatype datatype,
                  int source,
                  int tag, MPI_Comm comm,
                  MPI_Request *request)
```

```cpp
Prequest Comm::Recv_init(void* buf, int count,
                         const Datatype& datatype,
                         int source,
                         int tag) const
```
MPI_RECV_INIT bindings (cont.)

MPI_RECV_INIT(BUF, COUNT, DATATYPE, SOURCE, TAG,
    COMM, REQUEST, IERROR)

<type> BUF(*)
INTEGER COUNT, DATATYPE, SOURCE, TAG, COMM,
REQUEST, IERROR
Persistent Communication Requests

- To start a send or receive:
  
  ```
  MPI_START (REQUEST, IERR)
  MPI_START_ALL (COUNT, REQUESTARRAY, IERR)
  ```

- The wait and test routines can be used to block until completion, or to check on status
MPI_START

MPI_START(request)

INOUT request communication request

```
int MPI_Start(MPI_Request *request)

void Request::Start()

MPI_START(REQUEST, IERROR)
INTEGER REQUEST, IERROR
```
MPI\_START\_ALL

MPI\_START\_ALL(count, array\_of\_requests)

IN count list length
INOUT array\_of\_requests array of requests
MPI_START_ALL bindings

int MPI_Startall(int count,
                  MPI_Request *array_of_requests)

static void Prequest::Startall(int count,
                                Prequest array_of_requests[])

MPI_STARTALL(COUNT, ARRAY_OF_REQUESTS, IERROR)
INTEGER COUNT, ARRAY_OF_REQUESTS(*), IERROR
Copyright © 2000-2001 by William Emmanuel S. Yu and Jeff Squires. This material may be distributed only subject to the terms and conditions set forth in the Open Content License, v1.0 or later (the latest version is presently available at http://opencontent.org/opl.shtml).