MPI Working Group

Point to Point Communication

Current Topics

- Streams
  - Active discussions in teleconferences
- Allocate receive
  - On hold until San Jose meeting
- Receive reduce
  - On hold until San Jose meeting

What is an MPI Stream?

- From single sender to single receiver only
  - Joined by an existing communicator
- Ordered and reliable
- Sender can send any amount(s) of data
- Receiver can receive any amount(s) of data
  - (up to what is available)

Discussion Issues with MPI Streams

- Data-types as the unit of transmission?
  - Should normal message boundaries be ignored?
- Flow-control/buffering?
  - e.g. receiver consistently slower than sender
- Allow buffer underrun or block receiver?
  - e.g. receiver wants 33 integers but only 16 available
- What benefit(s) does an MPI Stream have over normal point-to-point?

Allocate Receive

- MPI_Arecv: the receive buffer is an *output* argument instead of an input argument, and the implementation allocates that memory internally.
- Allows implementation to allocate memory for the size of the message, eliminates buffering overhead when message size is not known a priori.
- Allows copy-free implementation of unexpected messages using an eager-like protocol.

Receive Reduce/Accumulate

- MPI_Recv_{reduce,accumulate}: the incoming data is reduced/accumulated onto the receive buffer.
- Matches a common application pattern during boundary element exchange and allows implementation to minimize buffering in this case and potentially do more efficiently.
- Useful for creating user-defined, potentially dynamic reduction trees, without graph communicators.
- May allow for more efficient implementation of some forms of active-messages.

Meeting details

- Teleconference calls
  - Fortnightly on Monday at 11:00 central US
  - Next on 5th January 2015
- Email list:
  - mpiwg-p2p@lists.mpi-forum.org
- Face-to-face meetings
  - http://meetings.mpi-forum.org/Meeting_details.php
  - Next on 8th-11th December in San Jose, CA