Channel Interface

Takeshi Nanri (Kyushu Univ., Japan) and
Shinji Sumimoto (Fujitsu Ltd.)

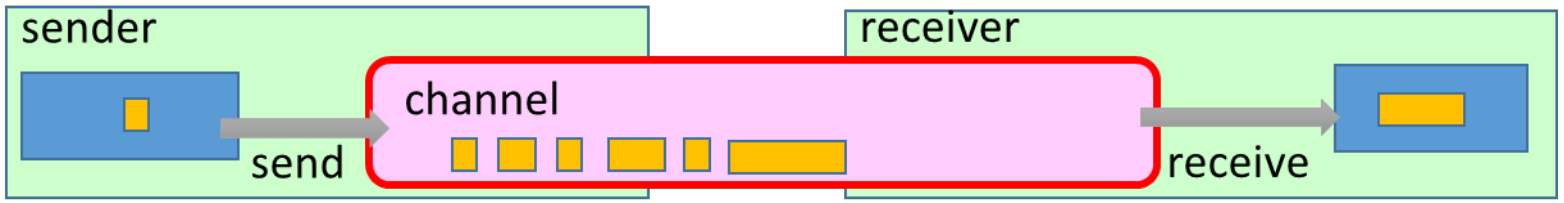
Dec. 2014

Advanced Communication for Exa (ACE) project

supported by JST, Japan.

Overview of the proposal in Kobe (Sep, 2014)

- Single-directional, in-order and on-demand channel
- Data type
  - `MPI_Ch` : a handle of a channel
- Routines for channel allocation / deallocation
  - `MPI_Channel_create(sender, receiver, &ch, comm)`
  - `MPI_Channel_free(ch)`
  - `MPI_Channel_ifree(ch, &req)`
- Routines for message passings on a channel
  - `MPI_Channel_send(ch, addr, count, type, comm)`
  - `MPI_Channel_isend(ch, addr, count, type, comm, &req)`
  - `MPI_Channel_recv(ch, addr, count, type, comm)`
  - `MPI_Channel_irecv(ch, addr, count, type, comm, &req)`



Background and motivation of the proposal

- Background:
  Memory requirement for communication.
  - For buffers and/or control structures
    (e.g. remote addresses, flags and counters)
  - Requirement depends on the pattern of producers and consumers.
    - High speed communication tends to require larger memory.
- Motivation:
  Enable memory-efficient implementation of the library.
  - lower latency, higher bandwidth
  - just enough memory consumption
- Approach:
  Explicit specification of the duration and the pattern of the communication.

Comments so far

- Need use-cases to show its advantage.
- There were similar approaches.
- Can't be memory efficient.
  - MPI_COMM_WORLD requires O(N) memory consumption.
  - Even proposed channels require O(N) memory to keep information for establishing connections.
- 'On demand' techniques already exists.
  - Lazy creation of connection.
  - Heuristic de-allocation.
- Effect from 'in order' is not promising.
  - 'no-wildcards' on INFO (Ticket 381) will solve the problem.
  - Data transfers on networks can be 'unordered'.

Comments so far (cont.)

- Should consider other patterns.
  - P2P is too primitive
    - Dedicated buffers for every P2P are not scalable.
    - Require dumping information for creating channels.
  - 'Topology' may be more efficient way to express the change of the communication pattern.
    - Collectives, Graphs, etc.
- Should consider MPI_Freeze / MPI_Thaw also.
- 'Count' should be MPI_Count.
- How to decide the internal buffer size?
- Functions for querying the address of the buffer?
  - Helps to use zero-copy data transfer.

Chances of advantages of the current proposal

- Still, memory consumption can be small enough.
  - Choose appropriate options to minimize the memory consumption at MPI_Init.
  - Information for connection is usually less than 100byte / procs.
    ... less than 100MB for 1M procs.
  - In case the number of connections per process is few enough.
- Different from persistent communication.
  - Address and size of the message are not fixed.
- More precise de-allocation of buffers than heuristics.
- Hints can be provided per channel, rather than per communicator.
  - e.g. size of the internal buffer, etc.

but ...

Problems in the current proposal.

- No practical use-cases
  - Persistent communication seems to be sufficient in most of the cases.
    - ex)
      Adaptive Mesh Refinement, Stencil, ...
      - Messages with different addresses and sizes may be packed into one buffer.

- May conflict with other similar approaches.
  - Op_notify / Sync_notify
  - Stream(?)

Other ways

- Topology: Multiple producers and consumers
  - Persistent collectives
    (proposed 7 years ago in Collectives WG).
    - https://svn.mpi-forum.org/trac/mpi-forum-web/wiki/PersColl
    - ex)

```
MPI_Request *persreq;
 ...
 do_init();
 MPI_Bcast_init(buf, count, datatype, root, comm, persreq);
 while (!finished) {
    fill_buffer(buf, count);
    MPI_Start(persreq);
    finished = do_computation();
    MPI_Wait(persreq, status);
 }
 MPI_Request_free(persreq);
```

  - Persistent neighbors
    - Graph comms with fixed address and size of the buffer to be used.

Conclusions

- Proposal of Channel Interface.
- At least, there are some differences over other ones.
- Practical use case is the critical issue.
  - We continue to look for it.
- Persistent collectives and neighbors are other options.