Tag matching logic

The MPI standard defines a set of rules, known as tag-matching, for matching source send operations to destination receives. The following parameters must match the following source and destination parameters:

- Communicator
- User tag - wild card may be specified by the receiver
- Source rank â€“ wild car may be specified by the receiver
- Destination rank â€“ wild

The ordering rules require that when more than one pair of send and receive message envelopes may match, the pair that includes the earliest posted-send and the earliest posted-receive is the pair that must be used to satisfy the matching operation. However, this doesnâ€™t imply that tags are consumed in the order they are created, e.g., a later generated tag may be consumed, if earlier tags canâ€™t be used to satisfy the matching rules.

When a message is sent from the sender to the receiver, the communication library may attempt to process the operation either after or before the corresponding matching receive is posted. If a matching receive is posted, this is an expected message, otherwise it is called an unexpected message. Implementations frequently use different matching schemes for these two different matching instances.

To keep MPI library memory footprint down, MPI implementations typically use two different protocols for this purpose:

1. The Eager protocol- the complete message is sent when the send is processed by the sender. A completion send is received in the send_cq notifying that the buffer can be reused.

2. The Rendezvous Protocol - the sender sends the tag-matching header, and perhaps a portion of data when first notifying the receiver. When the corresponding buffer is posted, the responder will use the information from the header to initiate an RDMA READ operation directly to the matching buffer. A fin message needs to be received in order for the buffer to be reused.

Tag matching implementation

There are two types of matching objects used, the posted receive list and the unexpected message list. The application posts receive buffers through calls to the MPI receive routines in the posted receive list and posts send messages using the MPI send routines. The head of the posted receive list may be maintained by the hardware, with the software expected to shadow this list.

When send is initiated and arrives at the receive side, if there is no pre-posted receive for this arriving message, it is passed to the software and placed in the unexpected message list. Otherwise the match is processed, including rendezvous processing, if appropriate, delivering the data to the specified receive buffer. This allows overlapping receive-side MPI tag matching with computation.

When a receive-message is posted, the communication library will first check the software unexpected message list for a matching receive. If a match is found, data is delivered to the user buffer, using a software controlled protocol. The UCX implementation uses either an eager or rendezvous protocol, depending on data size. If no match is found, the entire pre-posted receive list is maintained by the hardware, and there is space to add one more pre-posted receive to this list, this receive is passed to the hardware. Software is expected to shadow this list, to help with processing MPI cancel operations. In addition, because hardware and software are not expected to be tightly synchronized with respect to the tag-matching operation, this shadow list is used to detect the case that a pre-posted receive is passed to the hardware, as the matching unexpected message is being passed from the hardware to the software.