



MPI and the Exascale

Jeffrey M. Squyres, Cisco Systems

Fabian Tillier, Microsoft

16 March 2010

Our scope: MPI



- Leave hardware, power, runtime systems, filesystems, etc. to others
 - We're MPI + software wonks
- Assume: we know little about what a lottaflops system will look like (yet)
 - So let's make further assumptions
 - If you're looking at these slides in 5 years, please try not to laugh if we ended up being dreadfully wrong

Assumptions



- Assume: MPI will be used in some way
 - Otherwise this panel would be meaningless 😊
 - Either:
 - Directly in applications as today
 - Underlying transport for something else (PGAS, etc.)
 - MPI has spent 15+ years optimizing parallel communications; it would be silly to throw it away
- Assume: system will be a hierarchy of some kind
 - Memory, processors, network
 - MPI will need to understand the topology – particularly for collective operations (broadcast, reductions, etc.)

Assumptions



- Assume: limited resources for each MPI process
 - Memory, network buffers, etc.
 - Cannot store $O(N)$ information
 - Network may therefore need to be “smarter”
- Assume: there will be failures
 - MPI needs to get at least as reliable as sockets
 - Meaning: MPI implementation has to survive network failures
 - MPI-3 standard effort is examining such issues
 - To include what this means to MPI applications

“Thin” MPI



- Network must be reliable and connectionless
 - MPI should not handle tracking and retransmits
- Runtime system must support MPI:
 - Locally tell MPI location/topology, peer, and network info
 - Route stdin/out/err, stage files, etc.
 - Some better systems today behave like this
- Resources dedicated to MPI collective support
 - Maybe: network hardware, cores, memory, ...?
 - Asynchronous progress seems critical
 - More than just multicast: think about MPI_Alltoall

Will it run OFED (verbs)?



- We don't know what the network hardware will be
- If it runs verbs, there is much work to be done
 - Performant reliable connectionless will be necessary
 - Therefore: connection setup complexity disappears
 - Memory registration must disappear
 - Or get much better (no software reg cache, no dereg intercept)
 - Some form of MPI collective assist must be available
 - Learn from Intel, Cray, Quadrics, Voltaire, Mellanox, etc.
 - Export network / processor / memory / etc. topology info
 - Topology-aware algorithms will be critical
 - Separate header and data on completions

Thank You



Microsoft[®]