Trends in High-Performance Networking: the Good, the Bad and the Very Ugly

Patrick Geoffray, Ph.D
Senior Software Architect

Myricom®

www.myri.com

Trade-off in numbers

	I/O bus	System call	Memory Copy	PIO Copy	Memory Registration	Copy/Reg Threshold
Agony Dual P3 1 GHz	PCI 64/66	340 ns	250 MB/s	88 MB/s*	3.3 us + 0.26 us/pg	1.3 KB
Fog Dual P4 2.4 GHz	PCI-X 64/133	460 ns	1.7 GB/s	213 MB/s	5.8 us + 0.28 us/pg	23.3 KB
Rain Dual Opteron 2.2 GHz	PCI-E 8x	77ns	2.1 GB/s	1100 MB/s	3.0 us + 0.23 us/pg	11.7 KB
Shower Dual 2-core Opteron 2.8 GHz	PCI-E 8x	71 ns	2.2 GB/s	1100 MB/s	2.9 us + 0.19 us/pg	14.6 KB
Flood Dual 2-core Woodcrest 3 GHz	PCI-E 8x	86 ns	3.2 GB/s	1000 MB/s	0.6 us + 0.09 us/pg	3.6 KB

Kernel-bypass

- Past:
  - System call overhead was expensive, relative to the network latency.
  - Number of processes/thread/cores per node was small.
- Future:
  - System call overhead negligible, relative to network latency.
    - latency bounded by laws of physics and IO bus.
  - Number of processes/thread/cores per node increasing.
    - CPU can't get faster, so they get bigger.
- Prediction:
  - **High-performance communication libraries will go back in the kernel**.
    - Robustness: isolation from the applications.
    - Scalability: resources sharing, multiplexing.
    - Performance: trust, hardware support (asynchronous copy IOAT).

Zero-copy

- Past:
  - Memory copy slower than I/O bus throughput, expensive in CPU cycles.
  - Memory registration cost cheap, relative to the CPU speed.
- Future:
  - Memory copy faster than I/O bus throughput, cheap/free in CPU cycles.
  - Memory registration cost expensive, even much worse with virtualization.
- Prediction:
  - **The range of messages sent/received with a memory copy (Eager protocol) will increase, limited only by size of unexpected buffer**.
    - No synchronization between sender and receiver in MPI.
    - PIO Write used instead of Copy + DMA on send side.
    - Asynchronous Copy engine used instead of PIO Write on send side.
    - No need for evil RDMA.

RDMA vs Send/Recv

- Scalability:
  - Polling time is O(1) for Send/Recv, O(n) for RDMA (one buffer per sender).
    - RDMA: last-byte-written-last constraint.
  - Memory footprint is O(1) for Send/Recv, O(n) for RDMA.
    - SRQ is only for slow fallback.
  - All RDMA interfaces today are connection-oriented, all Send/Recv interfaces today are connection-less.
- Performance:
  - All problems that apply to zero-copy (synchronization, etc).
  - Send/Recv with matching can optimize unexpected messages (frequent).
  - Send/Recv with matching can provide efficient asynchronous progress.
- Prediction:
  - **The Send/Recv model (Portals, MX, QSnet, Ipath) will take over the RDMA model (IB, Iwarp, DAPL). Threats to MPI will finally cease.**

Offload

- Pros:
  - Latency and bandwidth are reaching physical constraints (distance, cabling, I/O bus) and network vendors need added value for differentiation.
  - Moore's law has lot of head room with NIC processing.
- Cons:
  - CPU clock and IPC are reaching physical constraints (power, cooling) and processor vendors need added value for differentiation.
  - General purpose compute cycle is cheap, specific hardware support possible.
- Trade-off:
  - Do not offload what is cheap /infrequent: reliability, flow control,
  - Do not offload processing constrained by NIC resources:
    - Does not scale with multi-core, security/fairness problem with virtualization.
- Prediction:
  - **All interconnects will offload something, as little as possible**.

TCP Offload ?

- TCP Offload Engine (TOE) is a bad idea:
  - TCP state machine is huge, complex/unwise to implement in silicon.
  - Connections require resources on the NIC.
  - Does not remove the need for memory copy or interrupts.
  - Good performance with state-less offload and TCP in the host.
- Prediction:
  - **TOE will run out of VC money and go away. (Various high-performance interfaces on top of TCP like iWarp and iSCSI will share the same fate).**

Netperf Test	MTU	Myri-10G Throughput	Myri-10G CPU	Chelsio TOE Throughput	Chelsio TOE CPU
TCP_STREAM	9000	9.7 Gb/s	50% / 50%	?	?
TCP_SENDFILE	9000	9.9 Gb/s	18% / 50%	?	?
TCP_STREAM	1500	8.2 Gb/s	43% / 70%	7.9 Gb/s	43% / 62%
TCP_SENDFILE	1500	8.2 Gb/s	11% / 70%	7.8 Gb/s	12% / 62%

News from Melmak

- What about Grid Computing ?
  - The Next Big Thing for the last 5 years.
  - Slow academic adoption.
  - No real commercial success.

- Slight speed of light issue.
  - Something beyond embarrassingly parallel jobs ?
- Slight human interaction issue.
  - People never agree on anything, just watch CNN.
- Slight toy sharing issue.
  - Poor kid want to play but rich kid does not want to share.
- Slight hype issue.
  - Grid Computing will replace big machines, at a fraction of the cost…