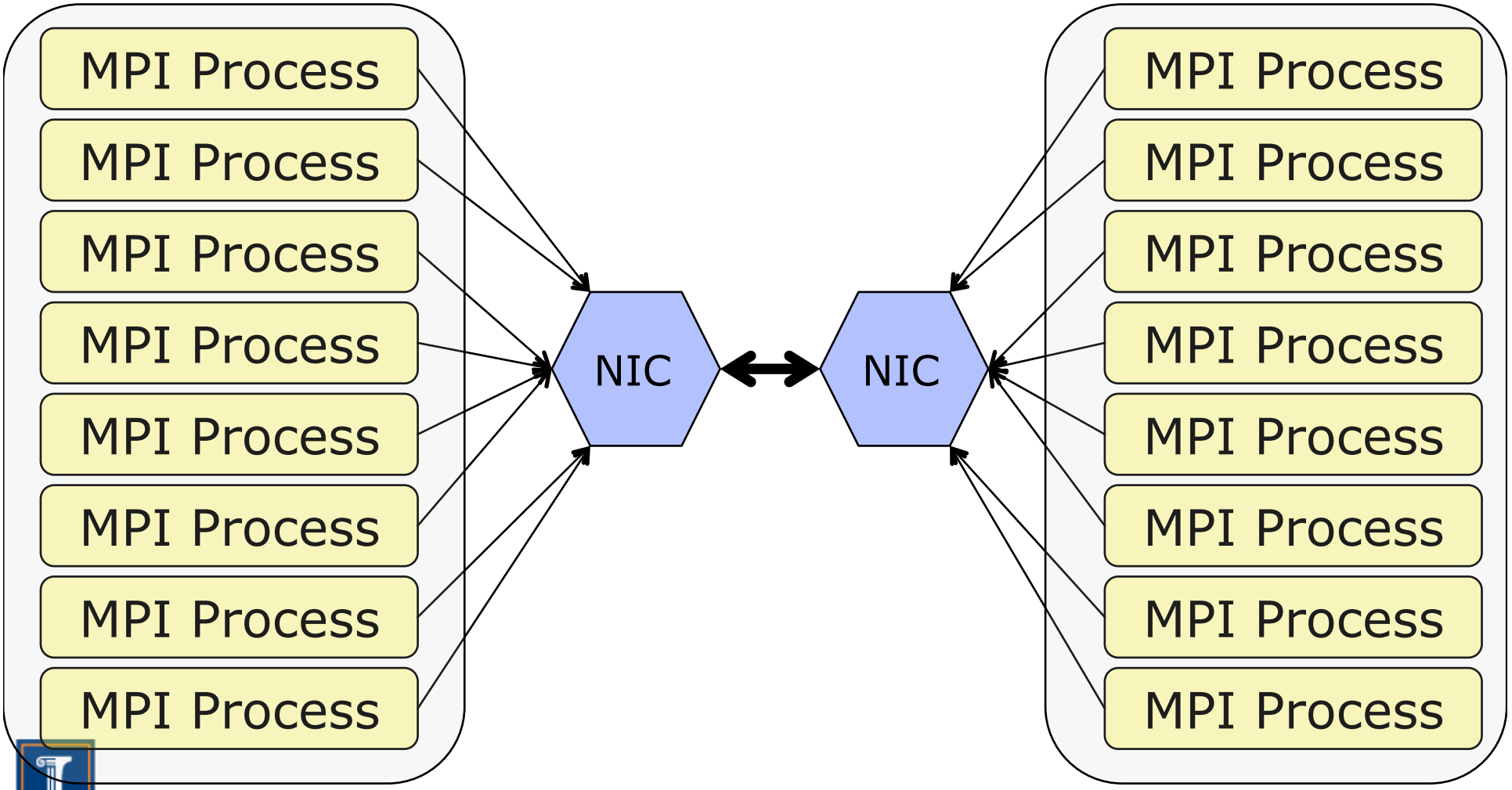
Modeling MPI Communication Performance on SMP Nodes: Is it Time to Retire the Ping Pong Test

William Gropp, Luke Olson
and Philipp Samfass

SMP Nodes: One Model

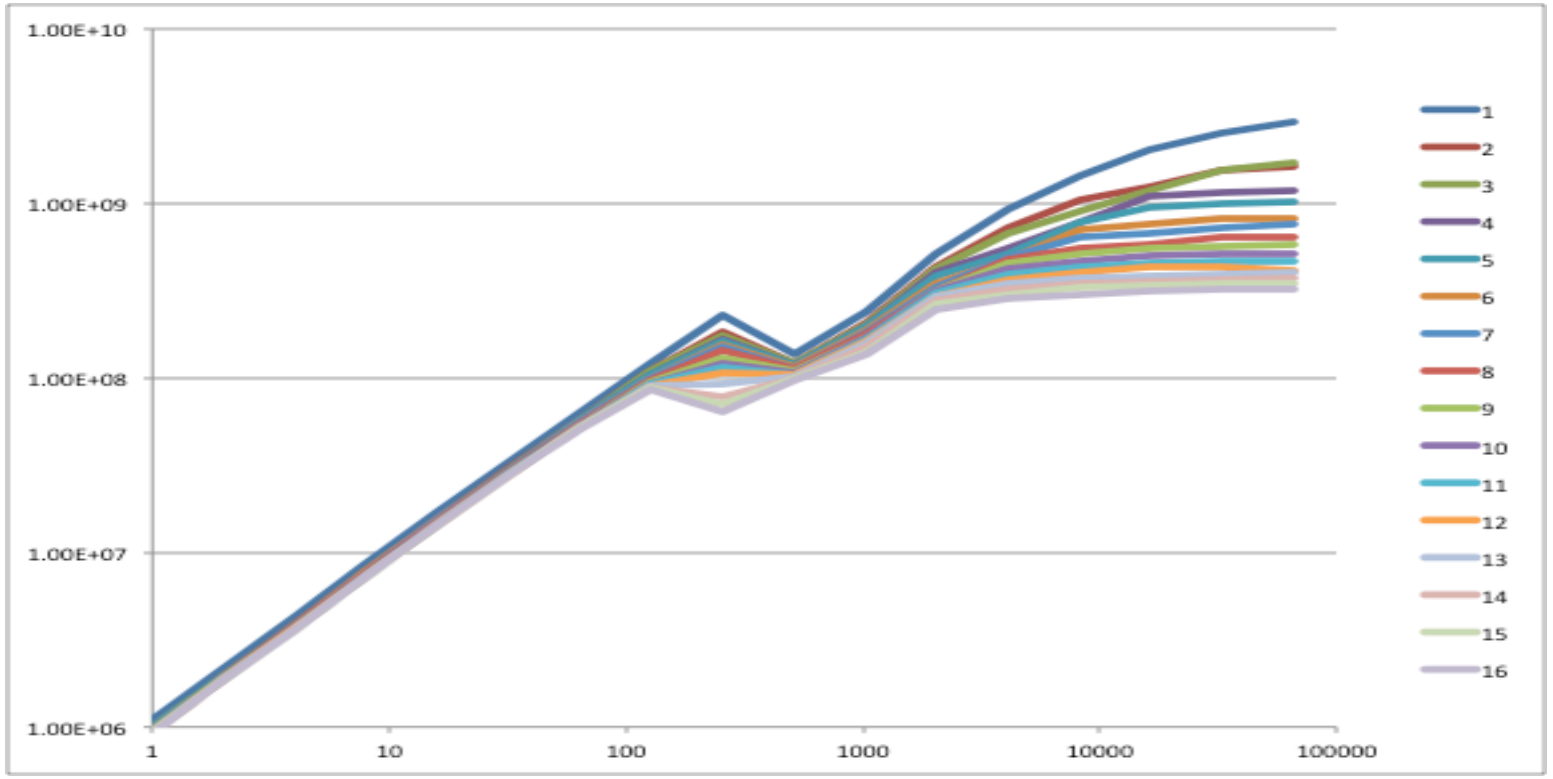


Classic Performance Model

- s + rn
- Model combines overhead and network latency (s) and a single communication rate 1/r for n bytes of data
- Good fit to machines when it was introduced
- But does it match modern SMP-based machines?
  - Let's look at the the communication rate per process with processes communicating between two nodes

Cray XE6

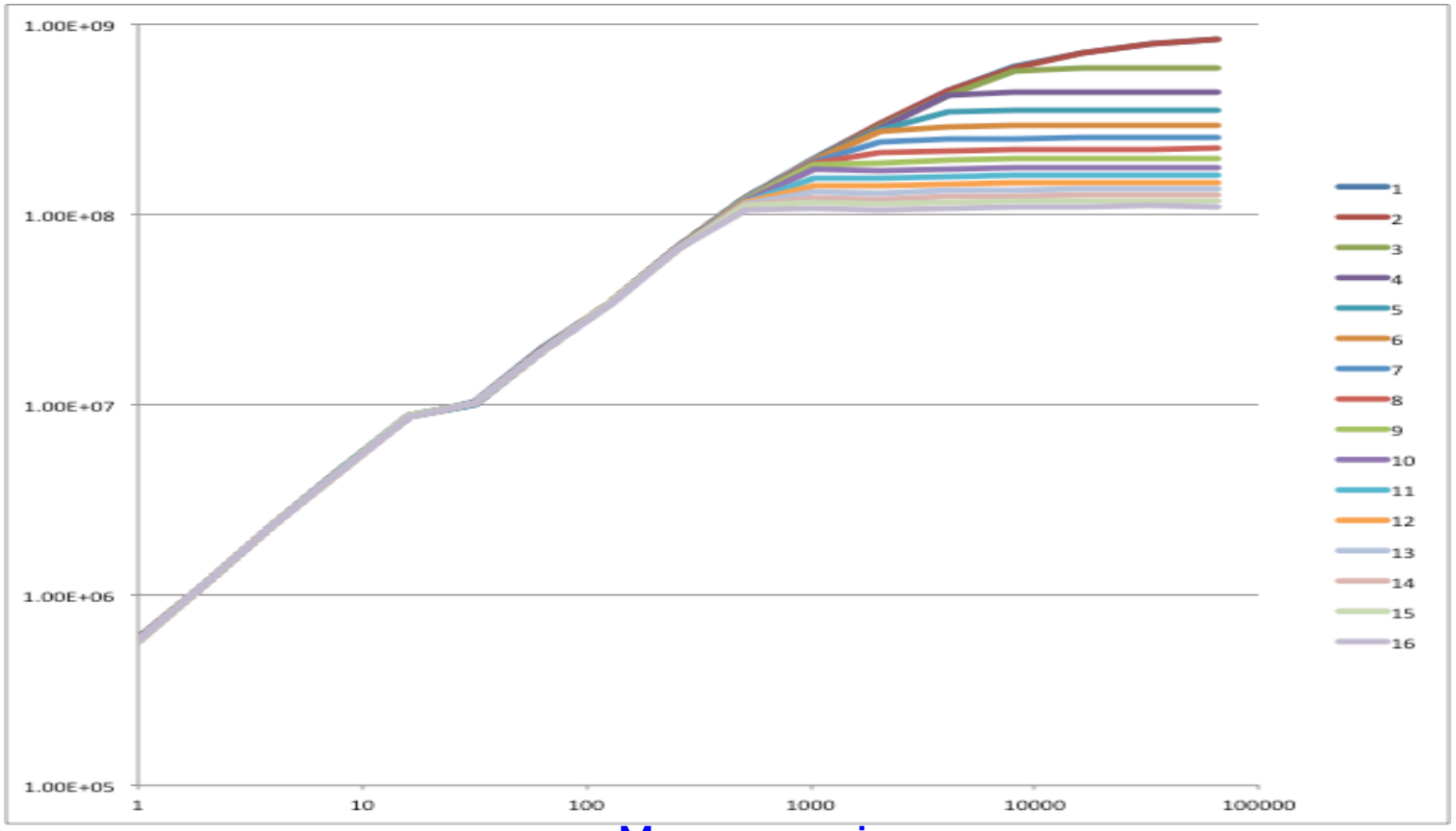
- Rate per MPI process



PARALLEL@ILLINOIS

Blue Gene/Q

- Rate per MPI process

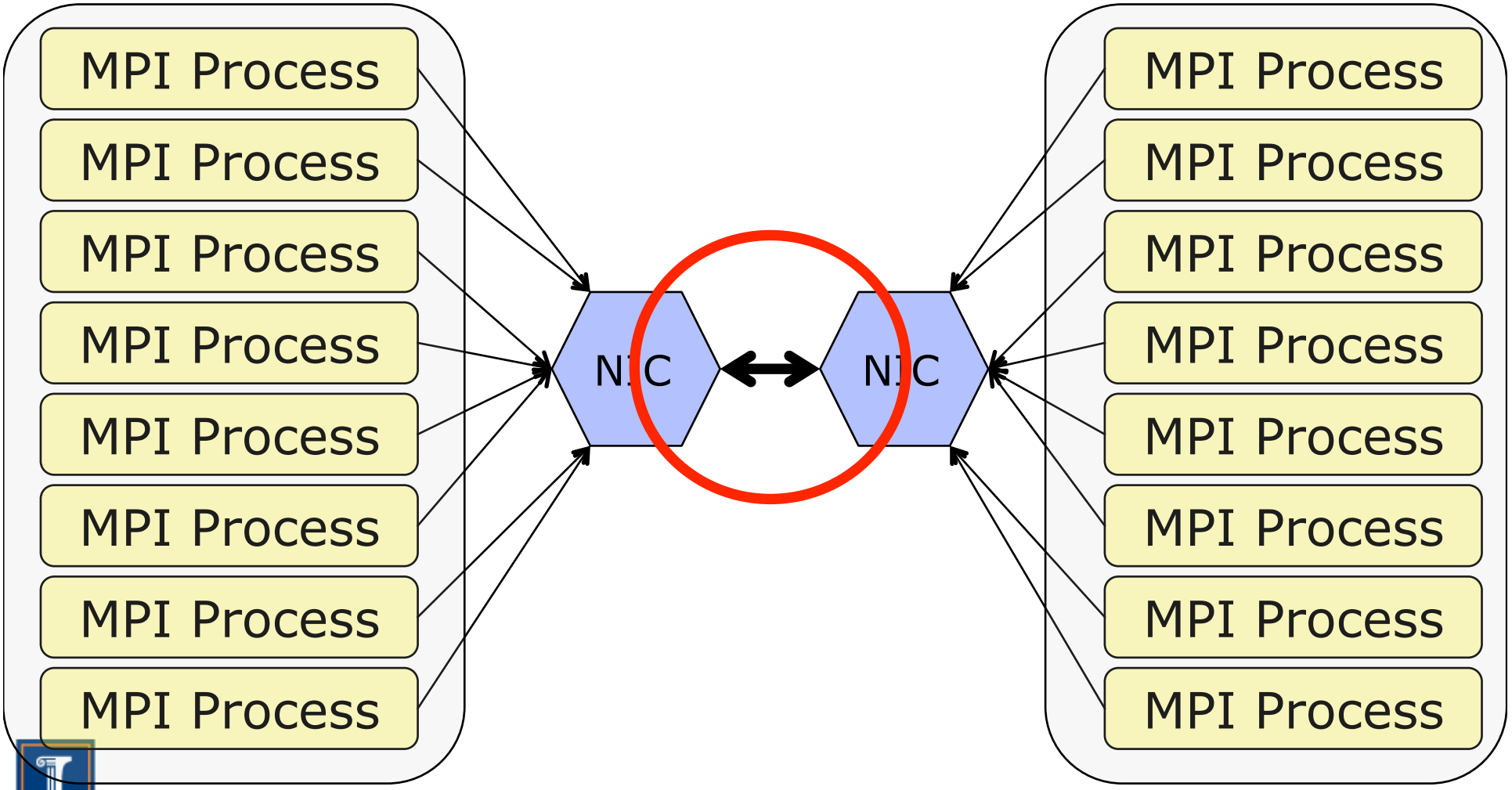


PARALLEL@ILLINOIS

Why this Behavior?

- The T = s + r n model predicts the *same* performance independent of the number of communicating processes
  - What is going on?
  - How should we model the time for communication?

SMP Nodes: One Model



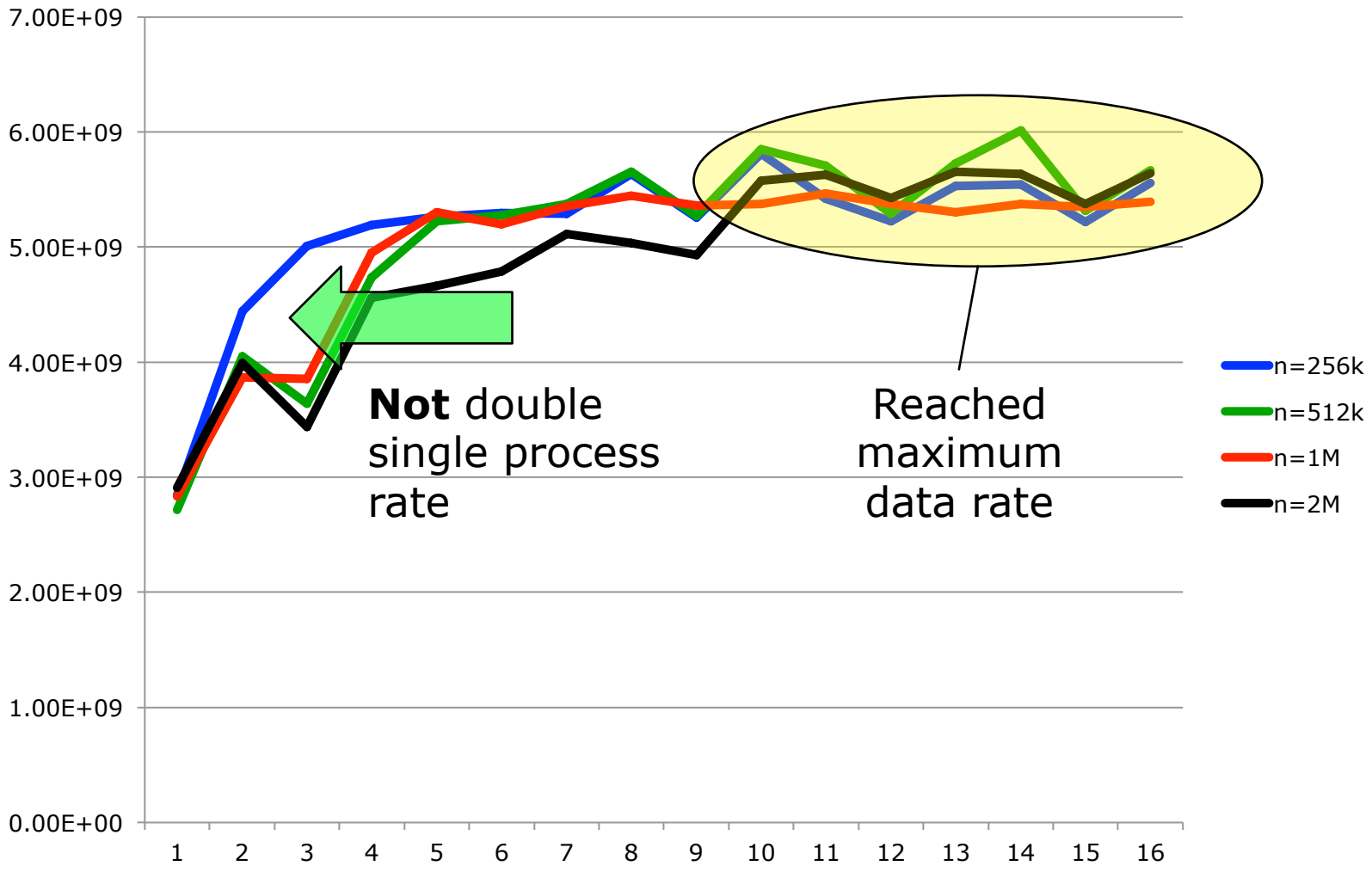
Modeling the Communication

- Each link can support a rate r_L of data
- Data is pipelined (Logp model)
  - ♦ Store and forward analysis is different
- Overhead is completely parallel
  - ♦ k processes sending one short message each takes the same time as one process sending one short message

Sending One Message From Each Process

- How do we model each process sending one message to another process on another node?
  - Classic "postal" model:
  - T = s+r n
  - Each process has no impact on the time that another process takes

Observed Rates for Large Messages



A Slightly Better Model

- Assume that the sustained communication rate is limited by
  - The maximum rate along any shared link
    - The link between NICs
  - The aggregate rate along parallel links
    - Each of the "links" from an MPI process to/from the NIC

A Slightly Better Model

- For k processes sending messages, the sustained rate is
  - $\min(R_{NIC\text{-}NIC}, k\ R_{CORE\text{-}NIC})$
- Thus
  - $T = s + k\ n/\min(R_{NIC\text{-}NIC}, k\ R_{CORE\text{-}NIC})$
- Note if $R_{NIC\text{-}NIC}$ is very large (very fast network), this reduces to
  - $T = s + k\ n/(k\ R_{CORE\text{-}NIC}) = s + n/R_{CORE\text{-}NIC}$

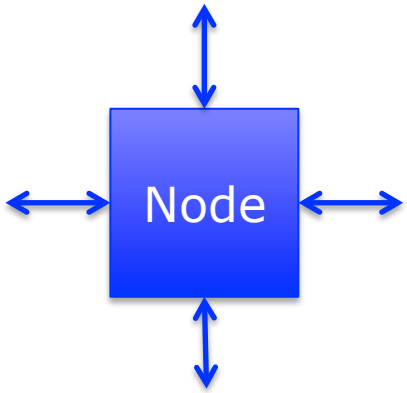
Another Refinement

- If communication by a second process can't achieve the same bandwidth as a single process, we can split the rate into two terms:
  - $T = s + k\, n/\min(R_N, R_{Cb} + (k-1)R_{Ci})$
- While slightly better than the 3 term formula (in the rendezvous regime), not enough better for the added complexity

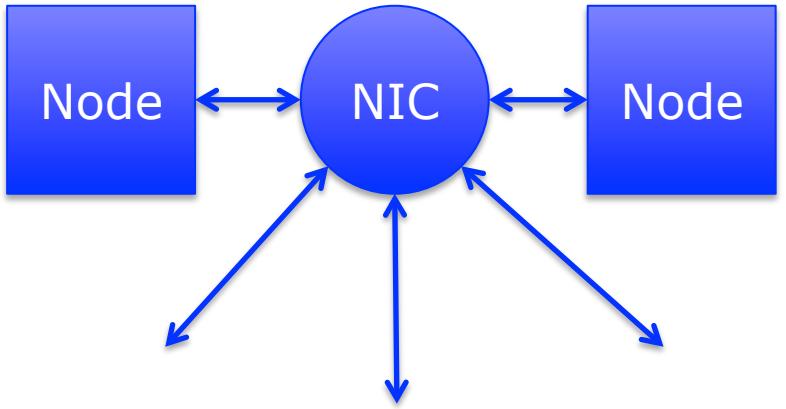
Two Examples

- Two simplified examples:

Blue Gene/Q



Cray XE6



- Note differences:
  - BG/Q : Multiple paths into the network
  - Cray XE6: Single path to NIC (shared by 2 nodes)
  - Multiple processes on a node sending can exceed the available bandwidth of the single path

The Test

- Nodecomm discovers the underlying physical topology
- Performs point-to-point communication (ping-pong) using 1 to # cores per node to another node (or another chip if a node has multiple chips)
- Outputs communication time for 1 to # cores along a single channel
  - Note that hardware may route some communication along a longer path to avoid contention.

Examples from Current Systems

- The following results use the code available soon at
  - https://bitbucket.org/william gropp/baseenv

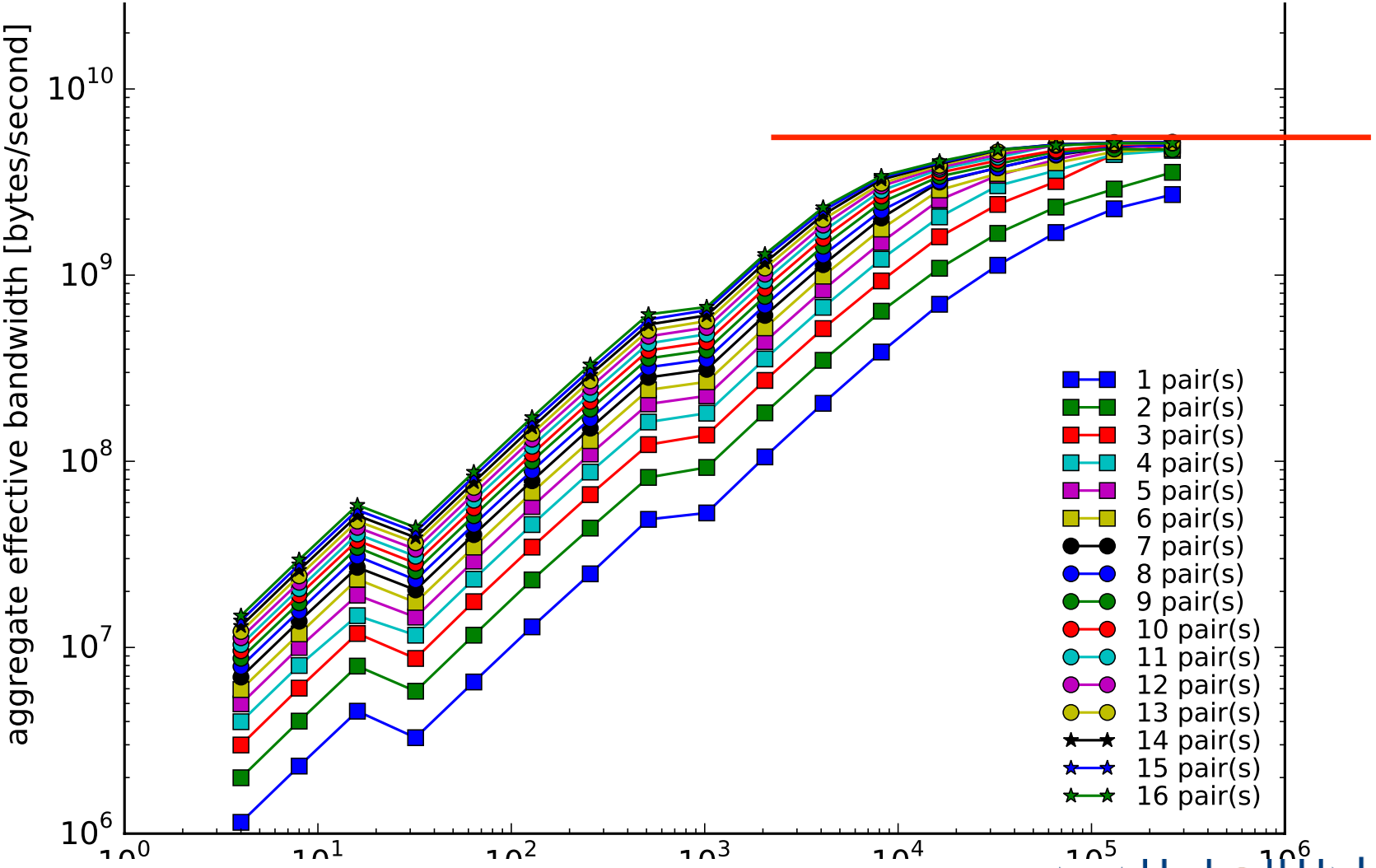
New Model
(Full PingPong Time, 4 parameter model)

- $R_N = R_{NIC}$; $R_C = R_{CORE\text{-}NIC}$
- Short regime
  - s = 4 usec, R_C = 0.63 GB/s, R_{Ci} = -0.18GB/s, $R_N = \infty$
- Eager regime
  - s = 11 usec, R_{Cb} = 1.7GB/s, R_{Ci} = 0.062GB/s, $R_N = \infty$
- Rendezvous regime
  - s = 20 usec, R_{Cb} = 3.6 GB/s, R_{Ci} = 0.61GB/s, R_N = 5.5 GB/s

How Well Does this Model Work?

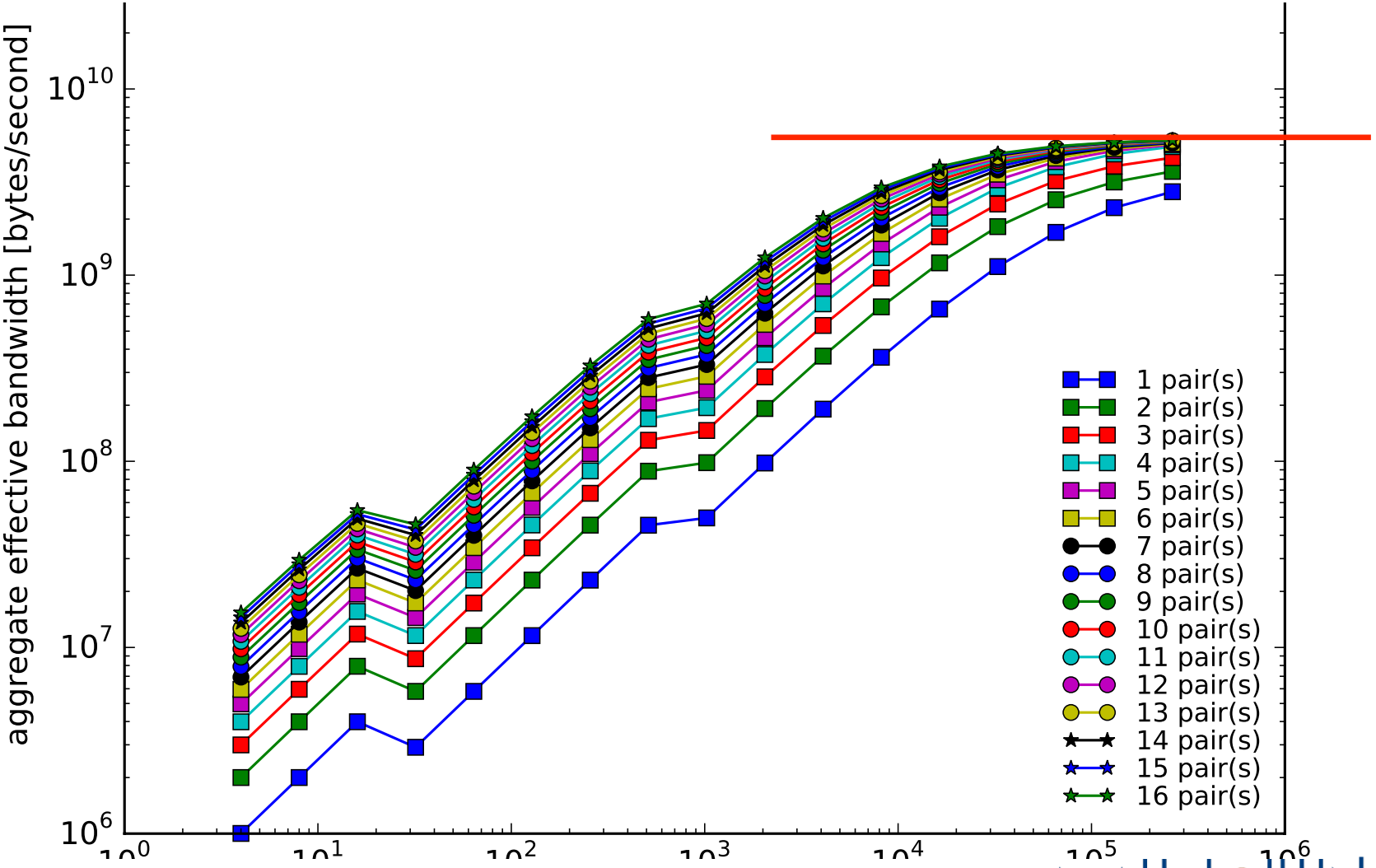
- We show results on a wide range of systems:
  - Cray XE6 with Gemini network
  - IBM BG/Q
  - Cluster with InfiniBand
  - Cluster with another network
- Results show nodecomm performance, model predictions, and relative error

Cray: Measured Data



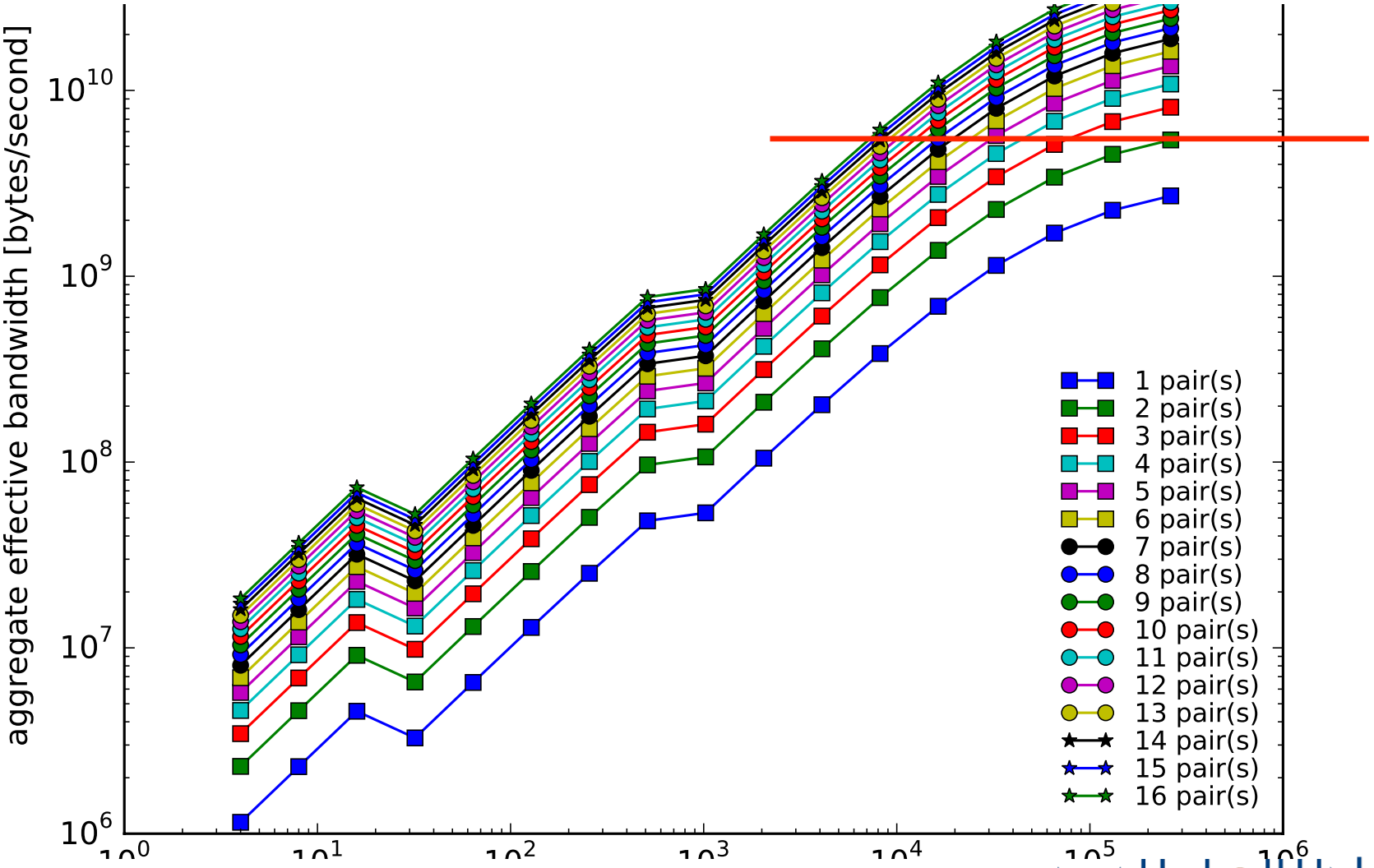
PARALLEL@ILLINOIS

Cray: 3 parameter model



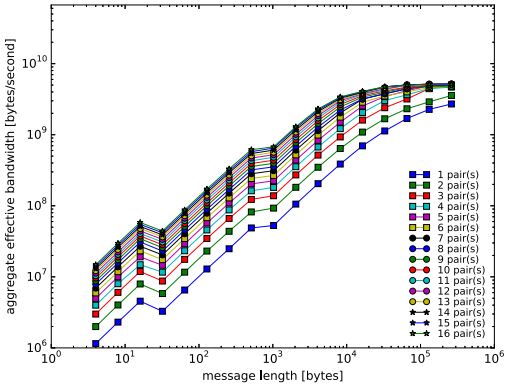
PARALLEL@ILLINOIS

Cray: 2 parameter model

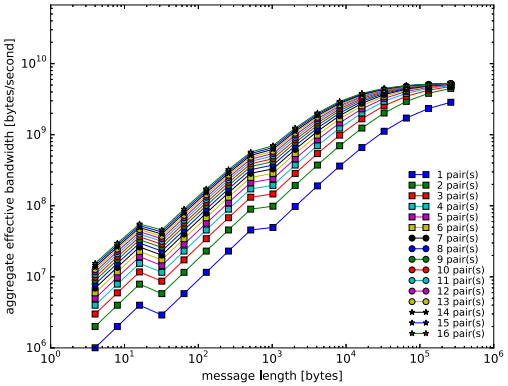


PARALLEL@ILLINOIS

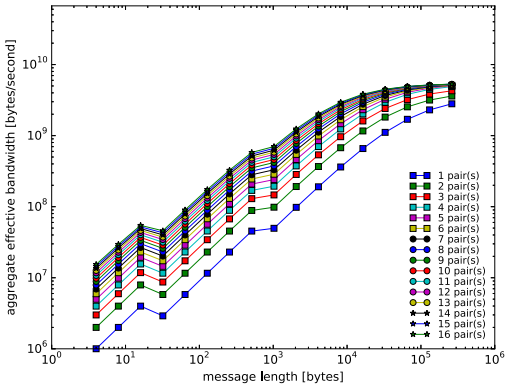
Cray XE6



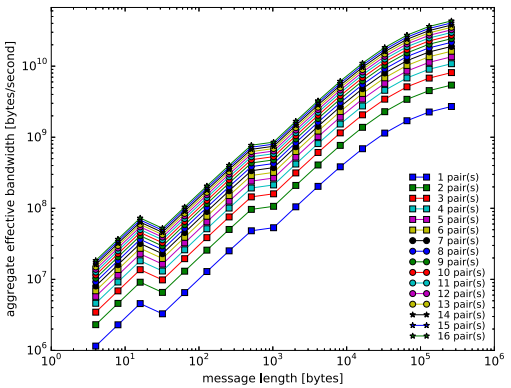
(a) Measured data.



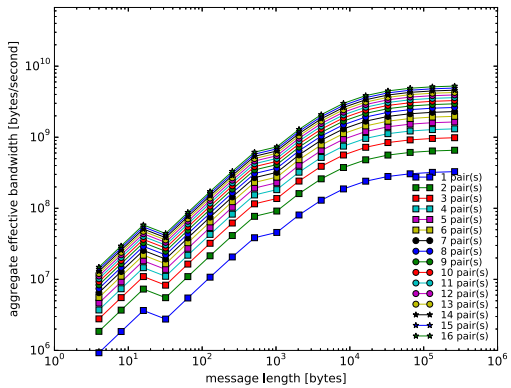
(b) Max-rate, three-parameter model.



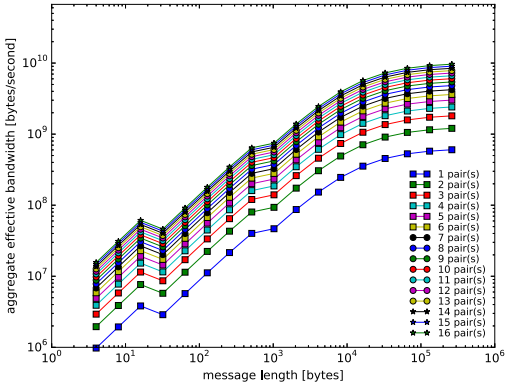
(c) Extended max-rate, four-parameter model.



(d) Postal, two-parameter model (1 pair).

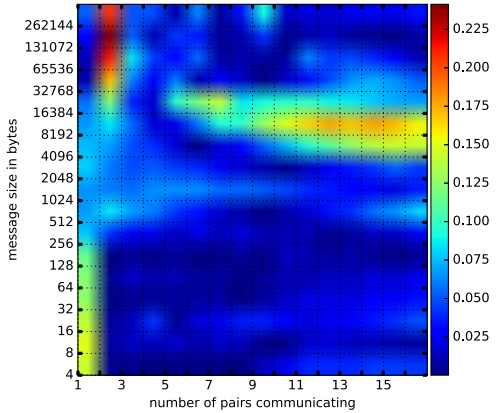


(e) Postal, two-parameter model (16 pairs).

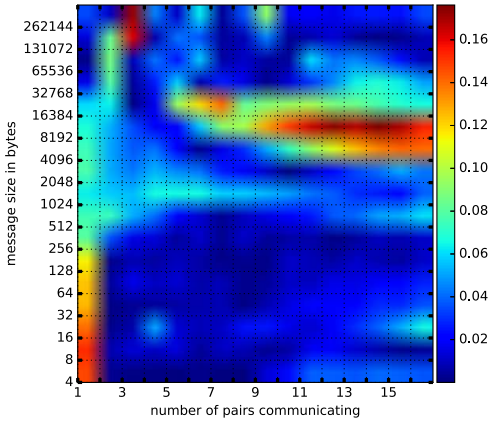


(f) Postal, two-parameter model (1–16 pairs).

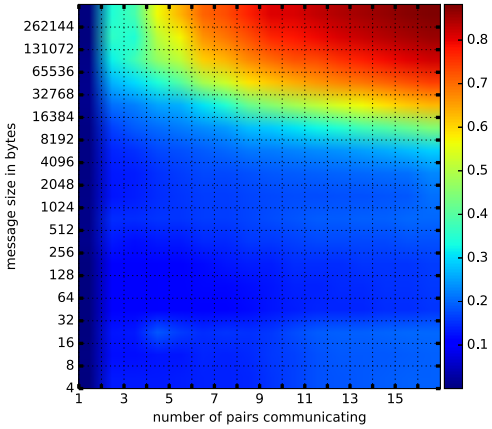
Cray XE6 – Relative Error



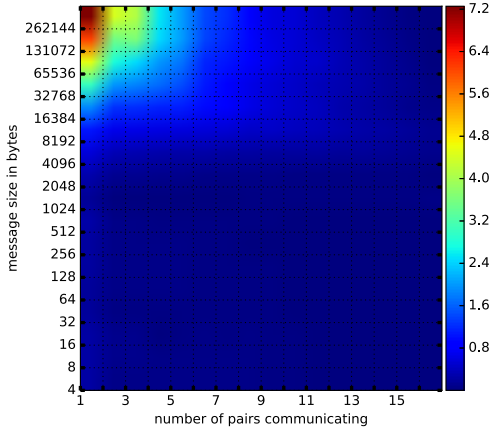
(a) Max-rate, three-parameter model.



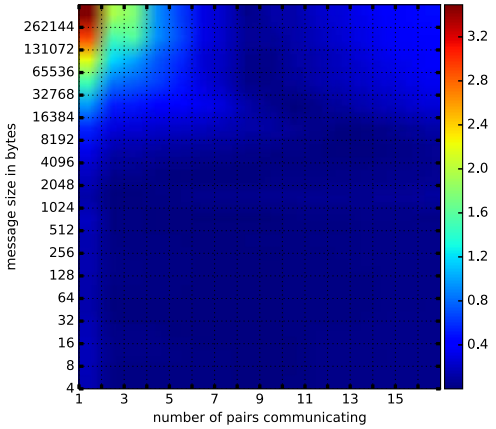
(b) Extended max-rate, four-parameter model.



(c) Postal, two-parameter model (1 pair).

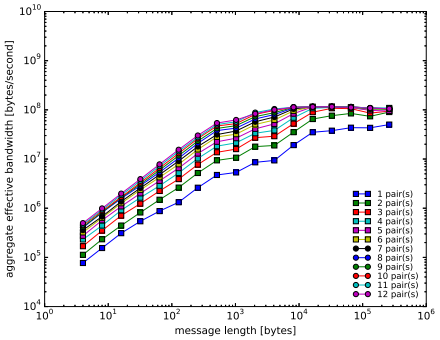


(d) Postal, two-parameter model (16 pairs).

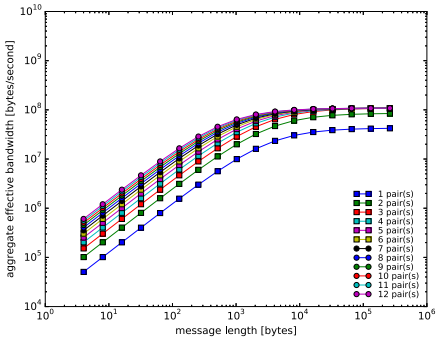


(e) Postal, two-parameter model (1–16 pairs).

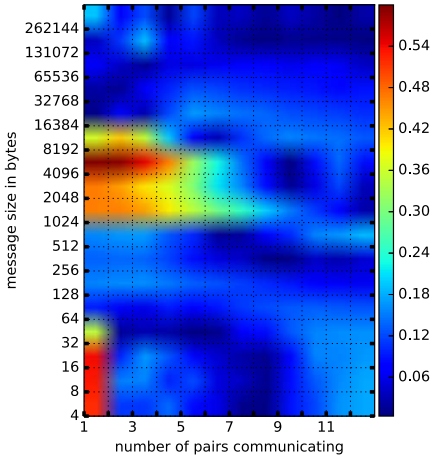
InfiniBand Cluster (Taub at Illinois)



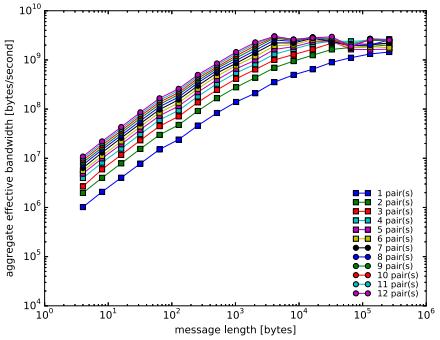
(a) Measured data (TCP).



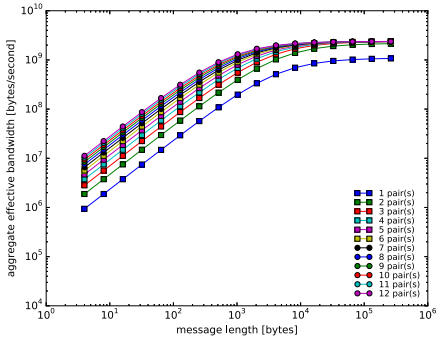
(b) Max-rate, three-parameter model (TCP).



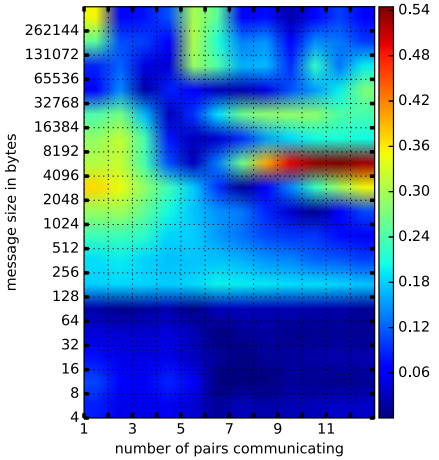
(c) Relative error (TCP).



(d) Measured data (IB).



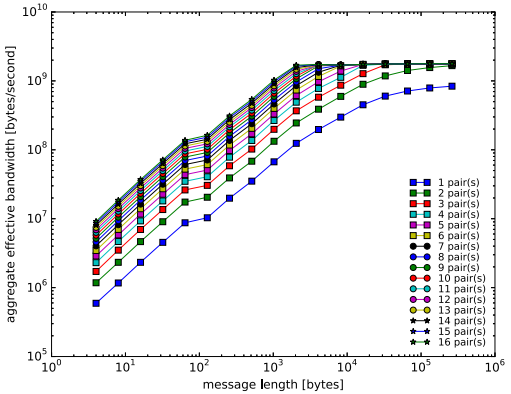
(e) Max-rate, three-parameter model (IB).



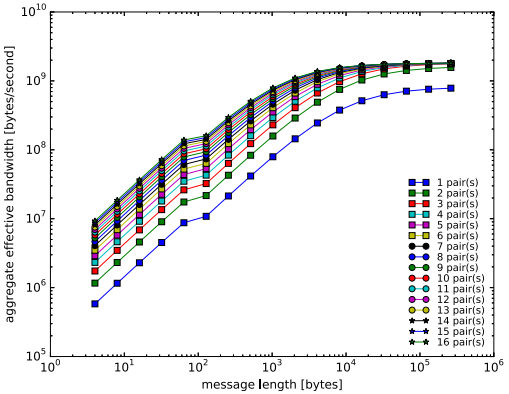
(f) Relative error (IB).

PARALLEL@ILLINOIS

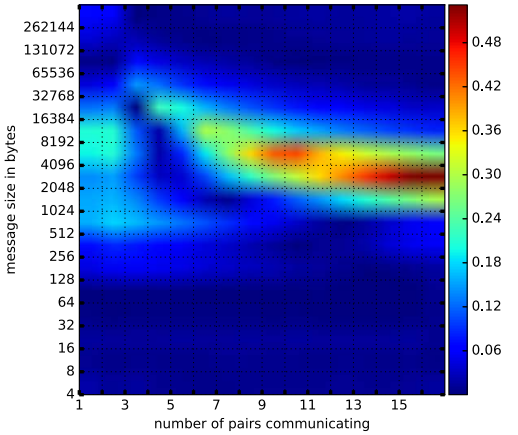
IBM BG/Q



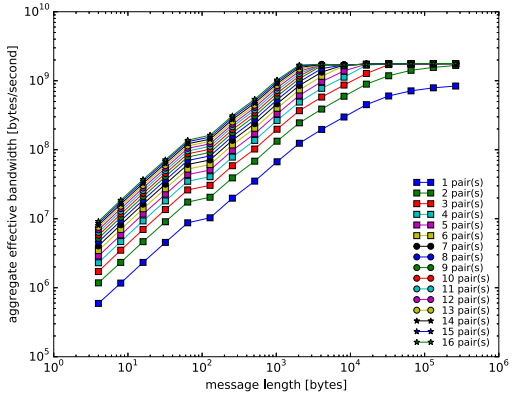
(a) Measured data.



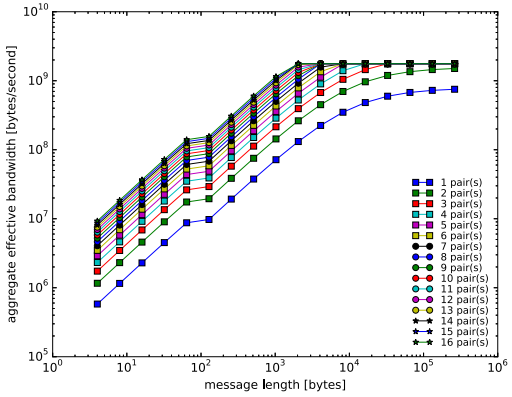
(b) Max-rate, three-parameter model.



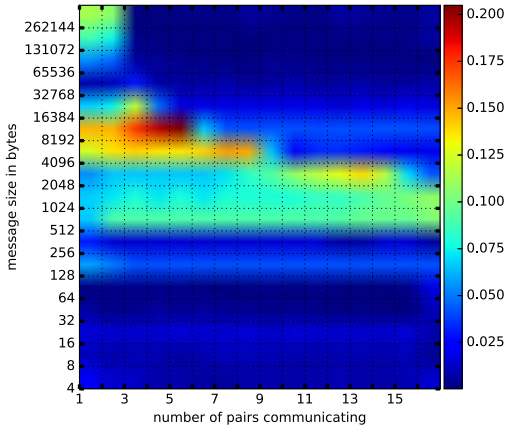
(c) Relative error.



(a) Measured data.



(b) Modified max-rate model.

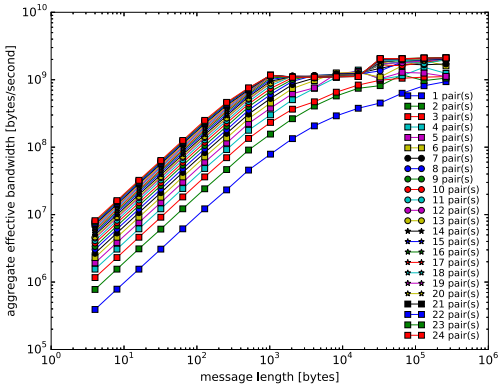


(c) Relative error of the model fit.

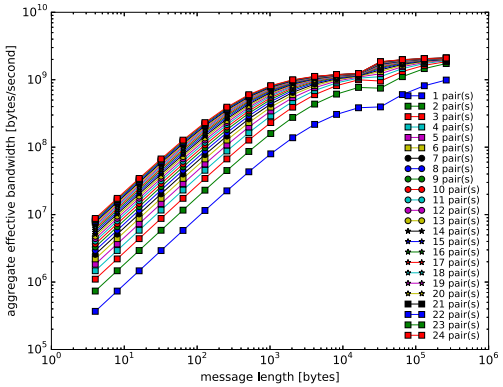
$T=kn/\min(R_N, kn/(s+n/R_C))$

PARALLEL@ILLINOIS

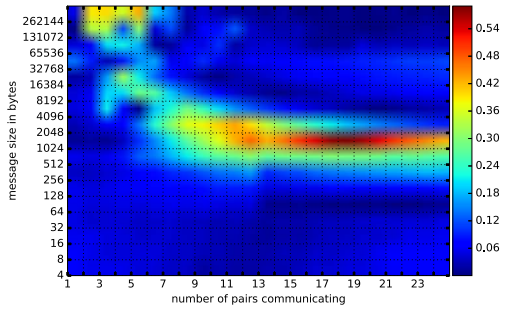
Cisco Cluster



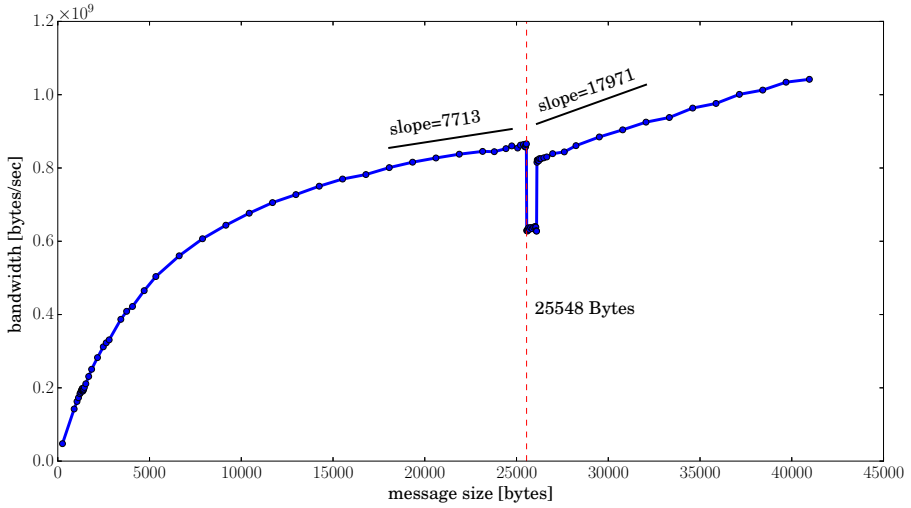
(a) Measured data.



(b) Max-rate, three-parameter model.



(c) Relative error.



PARALLEL@ILLINOIS

Notes

- Both Cray XE6 and IBM BG/Q have inadequate bandwidth to support each core sending data along the same link
  - But BG/Q has more independent links, so it is able to sustain a higher effective "halo exchange"

Modeling Communication

- For k processes sending messages concurrently from the same node, the correct (more precisely, a much better) time model is
  - $T = s + k\, n/\min(R_{NIC\text{-}NIC}, k\, R_{CORE\text{-}NIC})$
- Further terms improve this model, but this one is sufficient for many uses

Conclusion

- Yes, it is time to retire (or at least augment) the pingpong test
- Fortunately, a single additional parameter significantly improves the value of the communication performance model
- For algorithm and code designers, an additional message
  - Distribute communication in time so that off-node communication is less of a bottleneck

Thanks!

- ExxonMobile Upstream Research
- Blue Waters Sustained Petascale Project, supported by the National Science Foundation (award number OCI 07–25070) and the state of Illinois.
- Cisco Systems for access to the Arcetri UCS Balanced Technical Computing Cluster