Beyond Ping-Pong: Application-Centric Measurements of Network Performance

Scott Pakin
Performance and Architecture Lab
Los Alamos National Laboratory

12 November 2006
Outline

- Background
- Motivation
- Case studies
- Conclusions
Performance and Architecture Lab

- Performance analysis team at Los Alamos
  - Measurement
  - Modeling

- Large-scale:
  - Systems (1,000s to 100,000s of processors)
  - Applications

- Analyze existing systems (or near-to-market systems)

- Examine possible future systems
  - IBM PERCS (DARPA HPCS), RoadRunner (1.7PF peak), ...

- Recent work includes:
  - Optimization of ASCI Q, OS noise (SC2003 best paper)
  - Blue Gene/L (SC2004)
  - Large-scale optical circuit-switch network (SC2005)
  - System comparison: BG/L, Red Storm, ASC Purple (SC2006)
Background

- Overall goal is to improve application performance
- Important to measure/analyze network performance
  - Where is the time going in my application?
  - Which network would make my application run the fastest?
  - Is there a better way to make use of my existing network?

- Typical approach: Network microbenchmarks
  - SKaMPI, Intel (née Pallas) MPI Benchmarks, MPbench, MPIbench, Mpptest, …
  - Measure individual MPI primitives in isolation
  - Report peak communication performance
A Typical Microbenchmark (Throughput)

- Measures achievable bit rate
- Provides some understanding
- Unrealistic; What application uses this communication pattern?
Motivation

- Supplement application-oblivious microbenchmarks with application-centric ones
- Measure performance of a communication pattern in context
  - Multiple peers
  - Concurrent communication patterns
  - Interleaved computation
- Support experimentation
  - What if I used a 3-D domain decomposition instead of 2-D?
Case Study #1: Sweep3D

- $S_N$ transport kernel
  - Multiple overlapping wavefronts
- PAL team developed an accurate scalability model
  - Published in ICPP 2000
  - Inputs: single-CPU run time, network point-to-point performance, application inputs, architectural characteristics
  - Output: expected run time
- Model has been validated on many, many systems
Sweep3D Measurements vs. Model

- Pre-production version of ASC
  Red Storm
  - 350 nodes, Opteron, SeaStar interconnect, Portals-based MPI

- Model inputs
  - Typical problem size (5×5×400); weak-scaling mode
  - Red Storm network topology
  - Measured single-CPU run time
  - 11µs nearest-neighbor latency

- Model overpredicted run times at scale
  - First time ever model was wrong
  - Shape is correct but values are shifted upwards
  - What happened?

(Graph courtesy of Darren Kerbyson)
Measuring Network Latency

- Standard ping-pong test
- Repeatedly send A→B→A and divide by # of hops
  - A & B do not have synchronized clocks
- Utilized Sweep3D parameters
  - Blocking sends/receives
  - Appropriate message sizes (~2KB)
- Measured 11µs latency
A Sweep3D-specific Latency Benchmark

- **Observation**
  - Sweep3D performs computation between messages

- **Hypothesis**
  - Some amount of communication overlaps computation
  - Odd idea; all communication is blocking
  - Still, Red Storm does have an independent network processor
  - Overlapping sending and computing is valid by MPI specs

- **Revised benchmark ("Sweep1D")**
  - Include—then subtract off—computation
  - Not quite LogP overhead (blocking)
**Sweep1D Measurements**

- Red Storm *does* offload much of the communication cost (~75%)
  - Data shown for 0-byte message (extreme case)
  - Non-overlappable latency for 2KB message drops from 11µs to 6µs

- Other systems don’t
  - Ping-pong latency ≈ Sweep1D latency
  - Original latency benchmark is good enough
Sweep3D Model Revisited

- Custom benchmark proved quite useful
  - Helped test a performance hypothesis
  - Led to performance insight (computation can overlap a blocking send operation)
  - Increased model’s accuracy
Case Study #2: SAGE

- Large, hydrodynamics application
- Representative of ASC workload
- Slab-based domain decomposition
  - Not quite 1-D
  - Communicate with neighbors
  - # of neighbors varies with # of processors
Roadrunner Procurement

- Sustained 1 PF supercomputer
- Vendors were instructed to run various benchmarks and applications and submit results
- SAGE could not be included
  - Export controlled
  - Proprietary
- Solution was to provide a custom network benchmark that mimics communication in SAGE
A Custom SAGE Benchmark

- **Communication in SAGE**
  - Gather from all neighbors
  - Compute
  - Scatter to all neighbors

- **Communication properties**
  - All gathers and all scatters are concurrent
  - Implemented in terms of point-to-point primitives, not MPI collectives
  - Medium-sized messages (~50KB)

- **Simplified to a shift pattern**
  - Each process $p$ sends to $p-dist$ and $p+dist$ for varying $dist$
SAGE Benchmark Results

- Bandwidth can drop significantly with shift distance
  - Performance is susceptible to contention
  - Stock bandwidth tests show only peak performance
- Helps explain SAGE performance at scale
- Let us gauge how each of the Roadrunner proposals may perform on SAGE
Benchmark Development

- Why aren’t custom network benchmarks commonplace?
  - Require thinking; what do you want to measure?
  - Take too long to develop
- New: **coNCEPTual IDE**
  - Lets a programmer *draw* a network benchmark
  - Converts graphics to C+MPI via the coNCEPTual high-level language

(Going from scratch to a Sweep1D executable in 90 seconds)
Conclusions

● Typical communication benchmarks are necessary but not sufficient
  – Measure upper bound of performance
  – Not representative of how applications actually use the network

● Application-centric benchmarks provide additional insight
  – For Sweep3D, Sweep1D helped us determine that blocking sends are offloaded to the NIC in Red Storm/Portals
  – For SAGE, Shift enabled us to quantify the effect of network contention on application performance

● New tool, CONCEPTUAL IDE, facilitates rapid benchmark development
  – Useful benchmarks can be coded in mere minutes
For More Information

- Visit the Los Alamos booth
  - Kiosk display showcasing modeling work, CONCEPTual, and other PAL activities
  - Pick up a live CD with the CONCEPTual IDE preinstalled

- Download the CONCEPTual compiler

http://conceptual.sf.net/