A Performance Comparison of Linux and a Lightweight Kernel

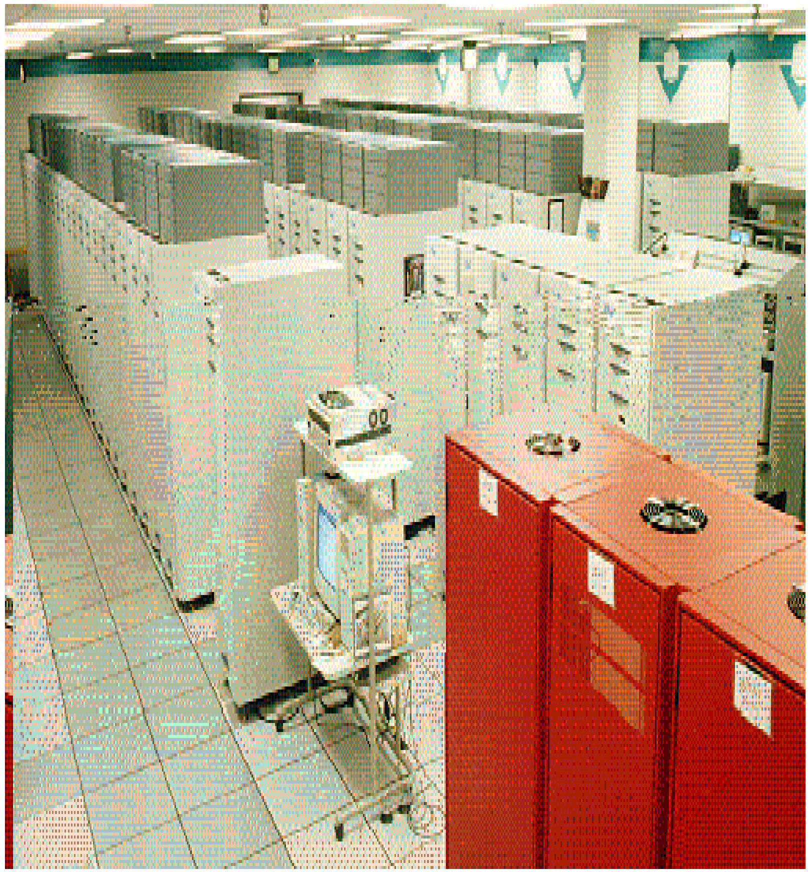
Keith D. Underwood

Sandia National Laboratories

Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company,
for the United States Department of Energy under contract DE-AC04-94AL85000.

ASCI Red Hardware

- 4640 compute nodes
  - Dual 333 MHz Pentium II Xeons
  - 256 MB RAM
- 800 MB/sec bi-directional network links
- 20 microsecond network latency
- 38x32x2 mesh topology
- Deployed in 1997



ASCI Red Storm

- 10,368 compute node processors (AMD Opterons @ 2.0 GHz)
- 10 TB of DDR memory @ 333MHz (1GB per node)
- Compute node topology:
  - 27 x 16 x 24 (x, y, z)
  - Mesh in x & y, torus in z
- 3 GB/s network bandwidth
- 5 microsecond network latency

A Lightweight Compute Node Operating System is a Fundamental Part of the Sandia Architecture

- It is essential for
  - Maximizing CPU resources
    - Reduce OS and runtime system overhead
  - Maximizing memory resources
    - Small memory footprint, large page support
  - Maximizing network resource
    - No virtual memory, physically contiguous address mapping
  - Increasing reliability
    - Small code base, reduced complexity
  - Deterministic performance
    - Repeatability
  - Scalability
    - OS resources should be independent of job size
- Others have realized these benefits
  - nCUBE (Vertex), Cray T3 (UNICOS/mk), IBM BG/L (HPK)

ASCI Red Compute Node Software

- Puma lightweight kernel
  - Follow-on to Sandia/UNM Operating System (SUNMOS)
  - Developed for 1024-node nCUBE-2 in 1993 by Sandia/UNM
  - Ported to 1800-node Intel Paragon in 1995 by Sandia/UNM
  - Ported to ASCI Red in 1996 by Intel and Sandia
  - Productized as “Cougar” by Intel

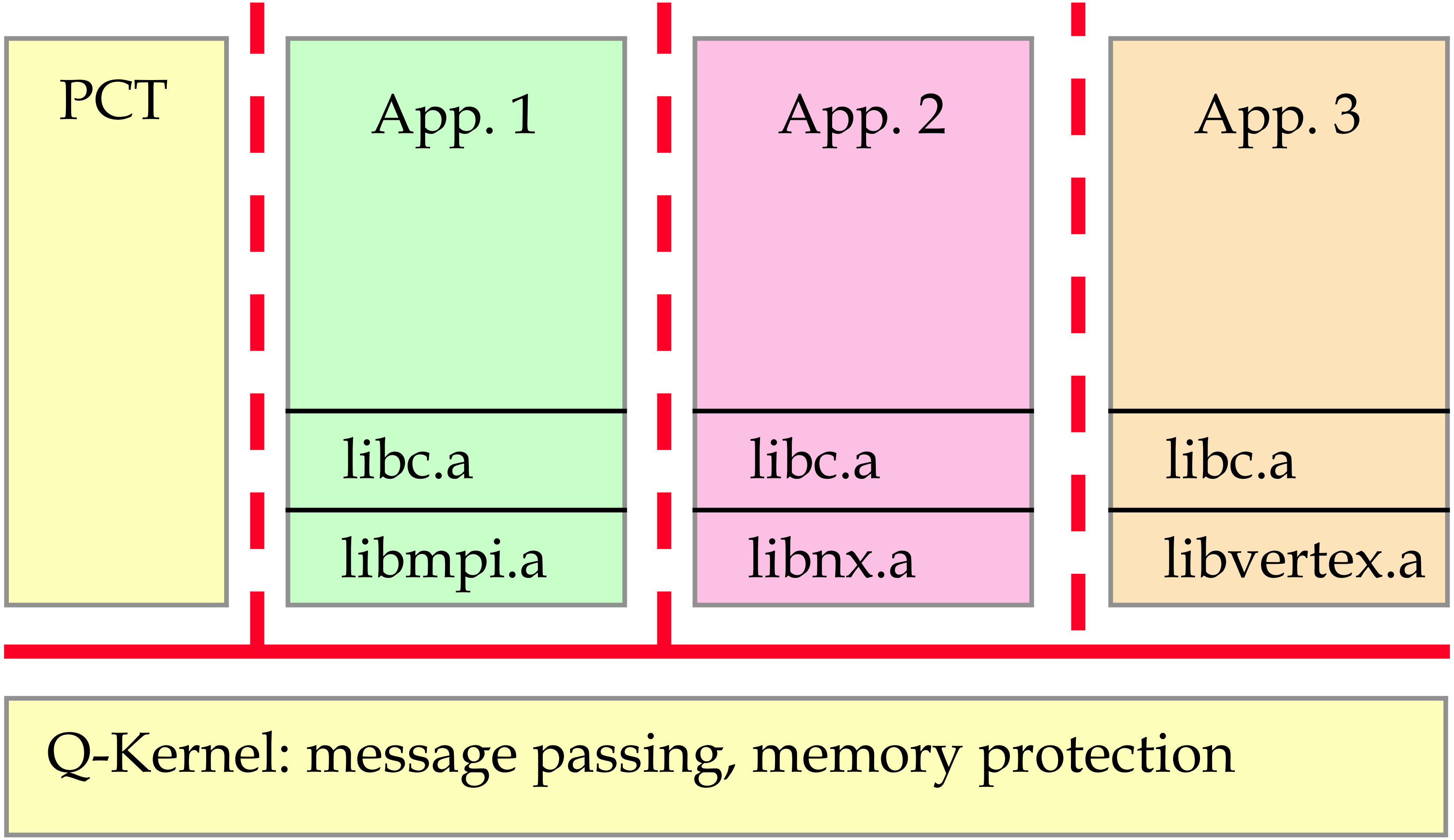
ASCI Red Software (cont'd)

- Cougar
  - Space-shared model (not time-shared)
  - Exposes all resources to applications
  - Consumes less than 1% of compute node memory
  - Four different execution modes for managing dual processors
  - Portals 2.0
    - High-performance message passing
    - Avoid buffering and memory copies
    - Supports multiple user-level libraries (MPI, Intel N/X, Vertex, etc.)

Cougar

- Goals
  - Target scientific and engineering applications on tightly coupled distributed memory architectures
  - Scalable to tens of thousands of processors
  - Fast message passing and execution
  - Small memory footprint
- Approach
  - Separate policy decision from policy enforcement
  - Protect applications from each other
  - Let user processes manage resources
  - Get out of the way

Cougar General Structure



Cougar Quintessential Kernel (QK)

- Policy enforcer
- Initializes hardware
- Handles interrupts and exceptions
- Maintains hardware virtual addressing (but no virtual memory)
- Small, static size
- Few, well defined entry points

Cougar Process Control Thread (PCT)

- **Runs in user space (but more privileged than apps)**
- **Customizable**
  - Single-tasking or multi-tasking
  - Round robin or priority scheduling
  - High performance, debugging, or profiling version
- **Changes behavior of OS without changing the kernel**
- **Policy Maker**
  - Process loading and scheduling
  - Virtual address space management
  - Name Server
  - Fault handling

Cougar Processor Modes

- Chosen at job launch time
- Heater mode (proc 0)
  - QK/PCT and application process on system CPU
- Message co-processor mode (proc 1)
  - QK/PCT on system CPU
  - Application process on second CPU
- Compute co-processor mode (proc 2)
  - QK/PCT and application process on system CPU
  - Application co-routines on on second CPU
- Virtual node mode (proc 3)
  - QK/PCT and application process on system CPU
  - Second application process on second CPU

Research Goals

- Assess the performance and reliability of a lightweight kernel versus a traditional monolithic kernel
- Determine how to bring lightweight kernel advantages to general platforms

Current Approach

- Short-term
  - Compare Cougar and Linux on ASCI/Red hardware
- Beyond that
  - Figure out how best to leverage Linux or other open-source operating systems to achieve important characteristics of previous LWKs
  - Provide a basis for future OS research activities

Motivation for Linux/LWK Comparison

- No direct comparison of LWK versus full-service OS since SUNMOS versus OSF1/AD nearly ten years ago
- Much has changed (improved?) since
- A direct comparison between a LWK and Linux is important for providing insight into what is important
- Platform balance is important
- Need real numbers to show people like: <insert favorite skeptic here>

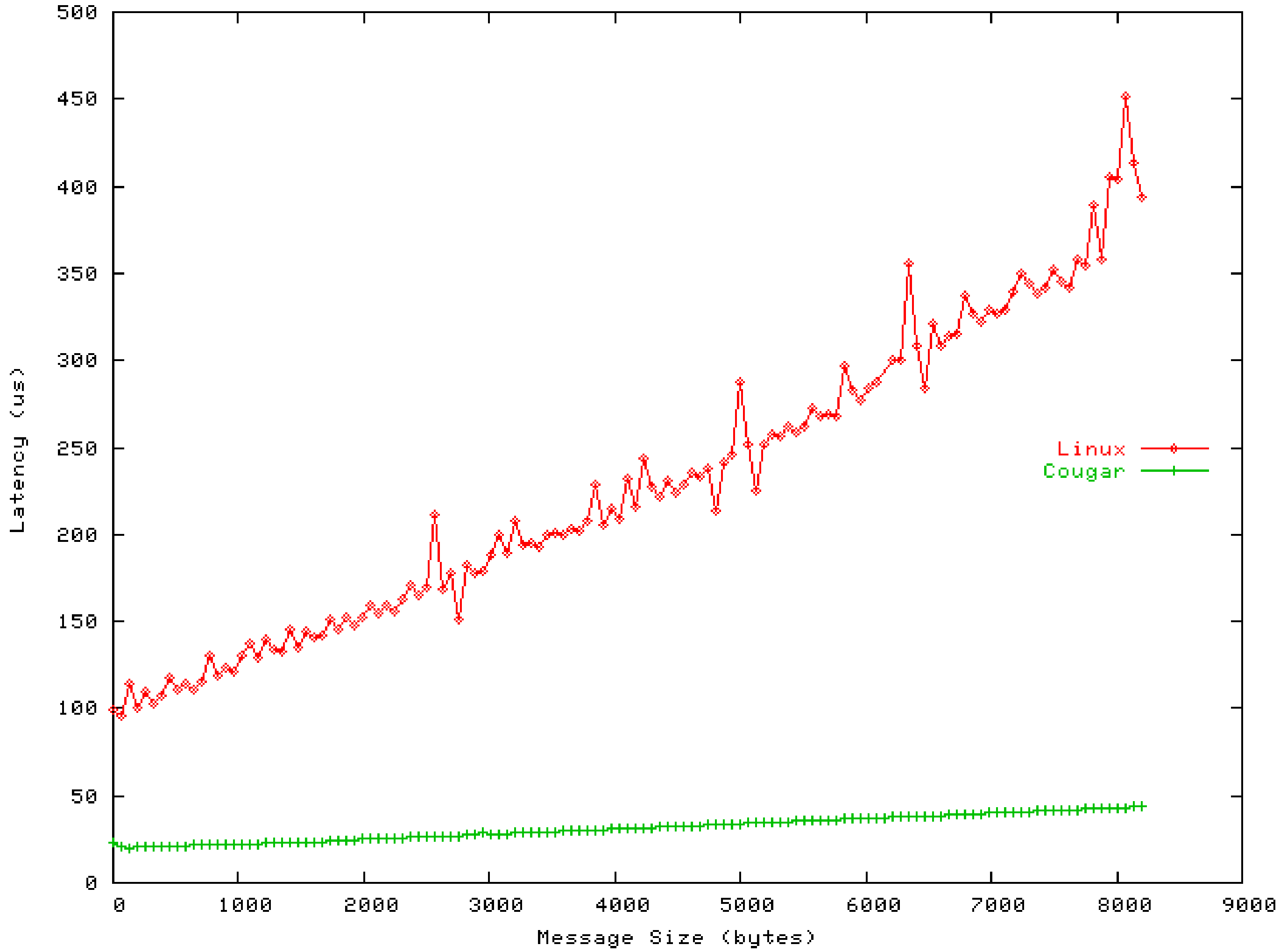
Linux on ASCI Red

- RedHat 7.2 - Linux 2.4.18
- Adapted Linux bootloader and startup code to work with bootmesh protocol
- Service node receives Linux kernel via bootmesh and root filesystem from attached SCSI disk
- Compute nodes mount root filesystem from service node
- Sparse compute node services
  - sshd for remote access
  - Enough libraries for MPI jobs to run

Linux IP Implementation for ASCI Red

- Implemented a Linux network driver for CNIC
  - Interrupt-driven ring buffer
  - Based on `isa-skeleton.c`
- Varying IP MTU from 4 KB (1 page) to 16 KB (4 pages) showed no noticeable difference in bandwidth
- Bandwidth is CPU limited
  - 45 MB/s for 333 Mhz processors
  - 32 MB/s for 200 MHz processors
- Custom raw device achieved 310 MB/s

MPI Ping-Pong Latency

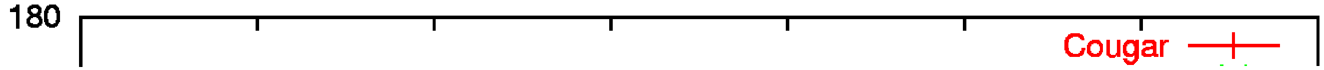


But to be fair…

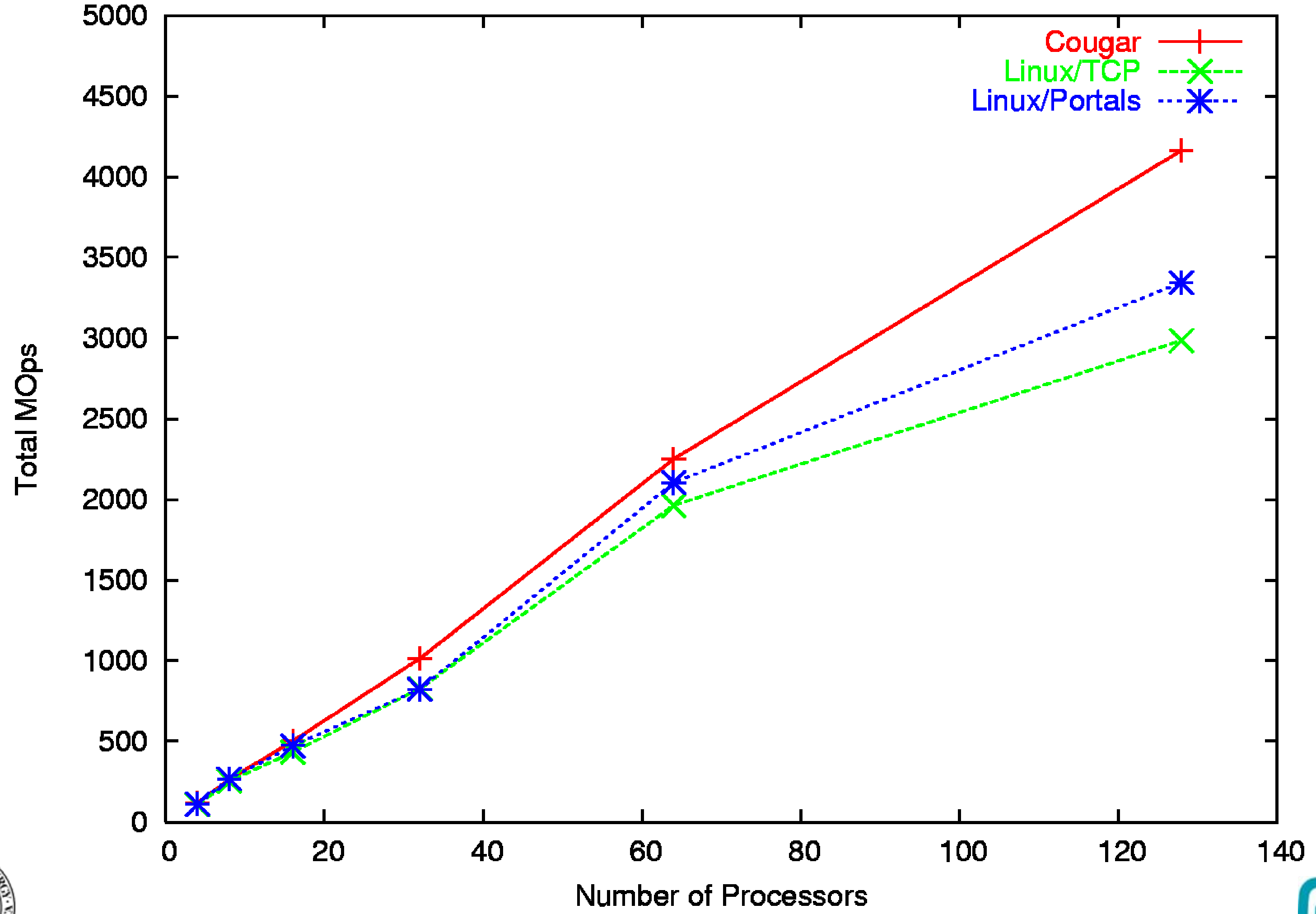
- Implemented a Portals 3.2 CNIC driver in Linux
  - 46 μs latency, 280 MB/s
- Not quite perfectly fair because Portals 3.2 vs. 2.0
- …but as close as we can get

NPB 2.4 - CG

NPB 2.4 - IS



NPB 2.4 - MG



Cougar
Linux/TCP
Linux/Portals
Total MOps
Number of Processors
0
500
1000
1500
2000
2500
3000
3500
4000
4500
5000
0
20
40
60
80
100
120
140

CTH Family of Codes

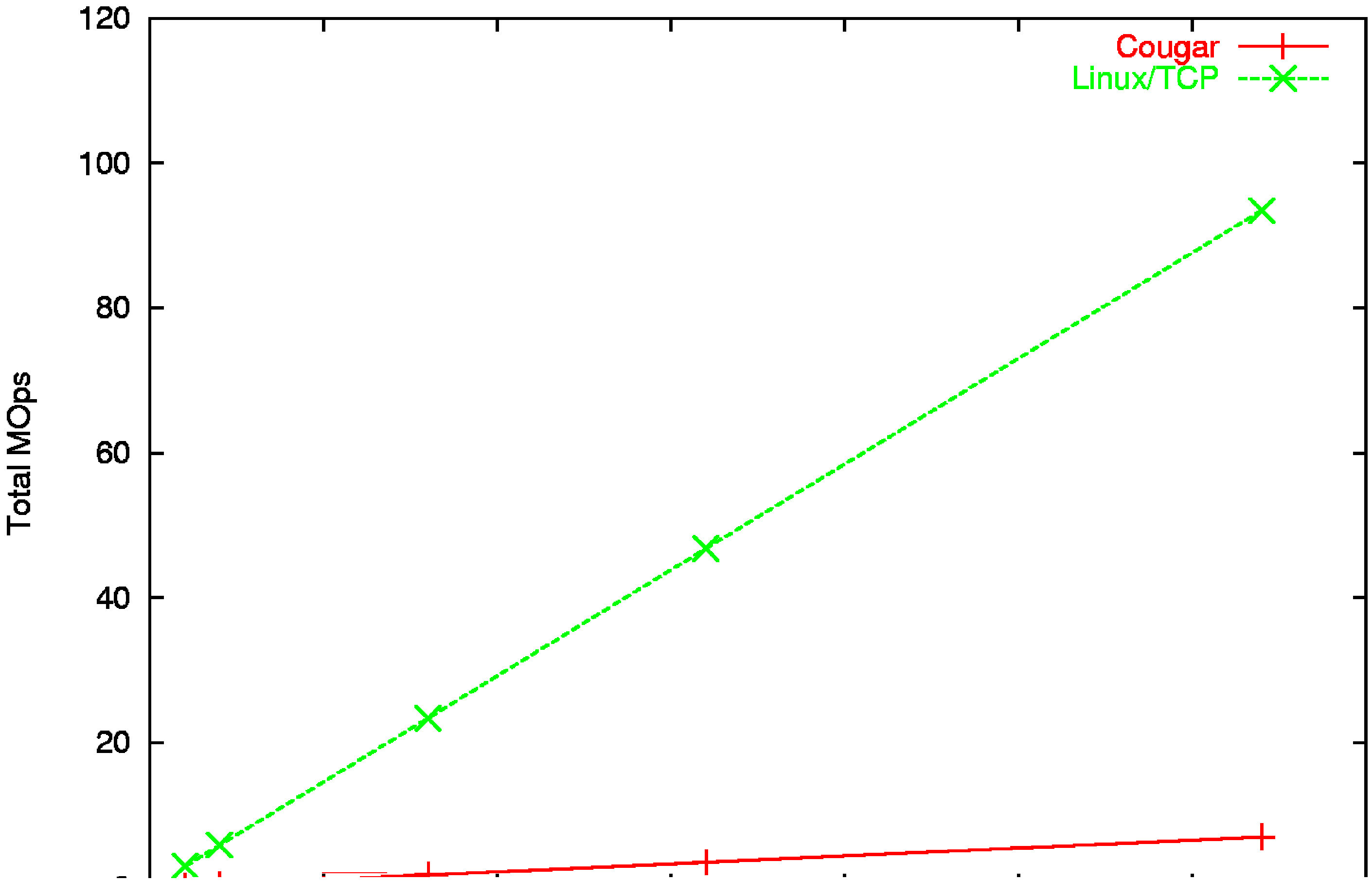
- Models complex multi-dimensional, multi-material problems characterized by large deformations and/or strong shocks
- Uses two-step, second-order accurate finite-difference Eulerian solution
- Material models for equations of state, strength, fracture, porosity, and high explosives
- Impact, penetration, perforation, shock compression, high explosive initiation and detonation problems

CTH Steps

- Read initial restart file, one file per node
- Simulate shock wave physics
  - Many nearest-neighbor communications, a few global reductions per time step
- Write results to restart, history, and viz files
- Performance measured in grind time
  - Time to compute all calculations on a single cell for a single time step

CTH Performance

NPB 2.4 - EP



Issues

- Compilers and runtime
  - Cougar numbers are from (old) PGI compilers
  - Linux numbers are from (new) Intel compilers
- Determinism
  - No variability in Cougar execution times
    - Even on a loaded machine
  - Significant (>5%) variability in Linux execution times
- Level of effort
  - Maintaining LWK may be equivalent to maintaining a Linux driver

Conclusions

- Finally have a real apples-to-apples comparison (albeit granny smith to red delicious)
- Numerous issues make fair comparison hard (should be perfectly fair on Red Storm, but that is still a long time away)
- Definite evidence of limitation of Linux at scale (still investigating)
- Definite advantages of LWK for some apps (IS)

Future Work

- Linux 2.6
  - Large page support
- Cougar
  - Provide a modern set of compilers/libraries
- Broader range of applications

Acknowledgments

- Project team
  - Ron Brightwell (who gave me most of the slides), Marcus Epperson, Mike Levenhagen, Rolf Riesen, Zhaofang Wen (Sandia)
  - Trammell Hudson (OS Research, Inc.)
  - Patrick Bridges, Kurt Ferreira, Barney Maccabe (UNM)