Characterizing the I/O Behavior of Scientific Applications on the Cray XT

Philip C. Roth

Future Technologies Group
Computer Science and Mathematics Division
Oak Ridge National Laboratory
rothpc@ornl.gov

Managed by UT-Battelle
for the Department of Energy

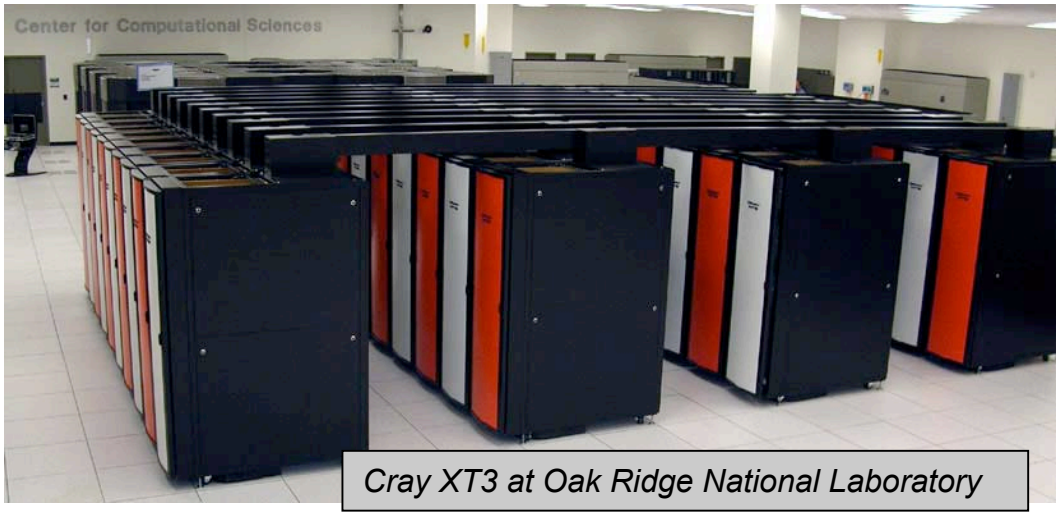
Challenges

- Many challenges to achieving high-performance I/O for scientific applications
  - Lack of information about I/O demands
  - Lack of information about achievable I/O performance
  - Performance portability
  - Increasing system scale

- Users need different types of information
  - Storage researchers: detailed I/O workload descriptions
  - Application developers: best practices
  - Systems designers: both!

Goals

- Understand I/O demands of scientific applications on leadership class systems
  - Focus on U.S. Department of Energy Office of Science applications
  - Initial focus on ORNL Cray XT system(s)
- Capture and share application I/O workload information for users with various needs



Cray XT3 at Oak Ridge National Laboratory

The Cray XT Platform

- Massively parallel architecture
- Processing Elements (PEs) connected in 3D mesh or torus topology
- Compute PEs run application processes
- Service PEs for logins, running batch scripts, and servicing I/O requests
- Catamount or Compute Node Linux (CNL) on compute PEs, full Linux on service Pes
- Lustre

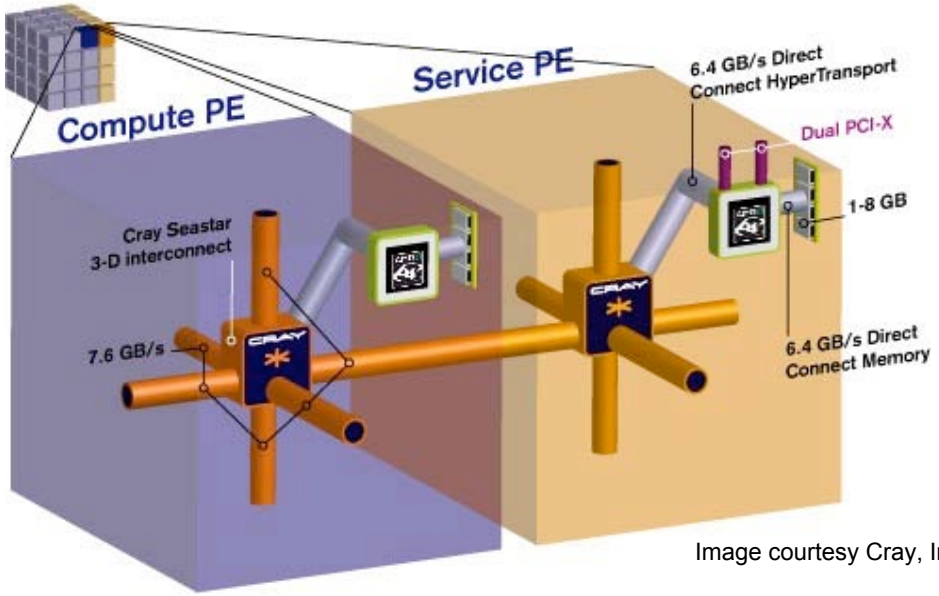
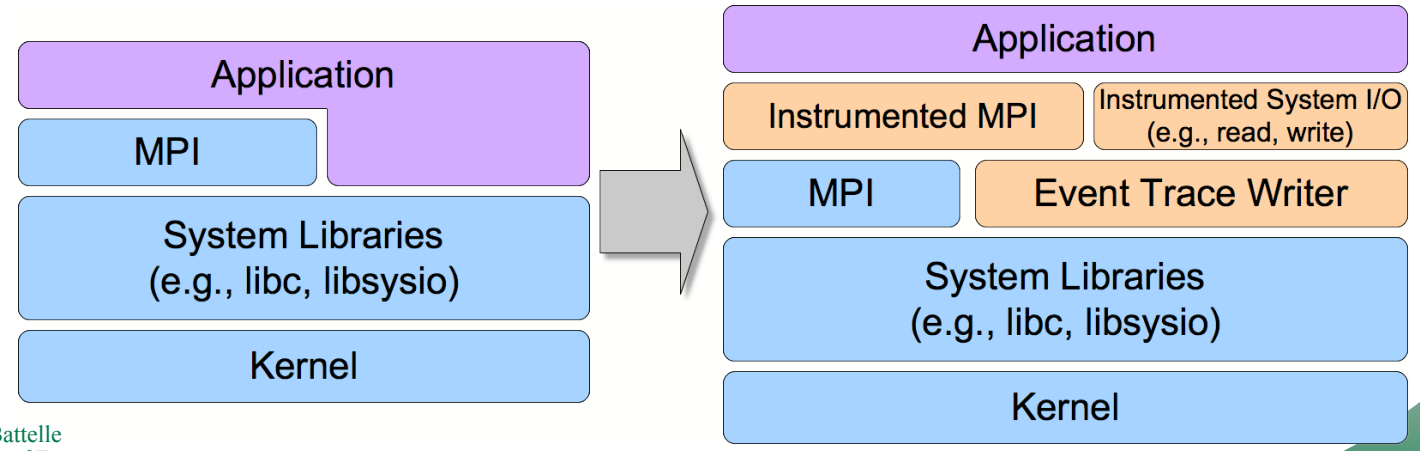


Image courtesy Cray, Inc.

Approach

- Interpose instrumented functions between application and interesting functions
  - MPI, especially MPI-IO
  - System I/O
- Custom compiler driver scripts for C, C++, Fortran (e.g., use *iot_ftn* instead of *ftn*)
- GNU linker generates wrappers for system I/O functions
- MPI functions instrumented at standard PMPI interface



Event Tracing

- Prototype implementation saves detailed event traces
- Currently using Open Trace Format (OTF)
  - Support for function enter/exit events, communication events, I/O operations
  - Readable long format, encoded short format, and compression support
  - Reader and writer libraries in C and Python
  - http://www.paratools.com/otf.php

Performance Data Analysis

- Custom tools using OTF reader libraries
- Existing tools like TAU, Vampir, SCALASCA with native support or trace file converters

Vampir event trace visualization of checkpoint I/O from S3D combustion simulation on ORNL Cray XT Jaguar

Case Study: Parallel Ocean Program

- Parallel Ocean Program (POP) climate simulation from Los Alamos National Laboratory
- Part of Community Climate System Model
- Fortran 90 with MPI
- Uses either netCDF or Fortran I/O
- Performs I/O for traditional reasons:
  - Read input files (topography grid, forcing data)
  - Write periodic checkpoint files (even-odd supported)
  - Write time-varying results (movie frames, calculation history)

POP I/O Characterization

- X1 benchmark problem (one degree grid), strong scaling, on Cray XT running Catamount
- Four output tasks
- Artificial output frequency
  - Checkpoint every 10 timesteps
  - Movie file every 5 timesteps
  - No calculation history files
- Primary I/O demands
  - Input
    - ~6.9MB horizontal grid file, ~1KB vertical grid file
    - ~490KB topography file
  - Output
    - Checkpoint file: 10KB text metadata file, 346MB binary data
    - Movie file: 3.9MB netCDF file

POP I/O Characterization

- MPI rank 0
  - Checkpoint: 87 writes; header plus 80 ~983KB writes, each 1/np of an ocean layer
  - Movie: Two writes, each 3.9MB
  - 10 read/write pairs by netCDF, reads between 25KB and 50KB, writes of either 438 or 2117 bytes
- MPI rank != 0
  - Checkpoint: 80 ~983KB writes, each 1/np of an ocean layer, aligned
  - Reads:
    - 80 zero-byte reads per checkpoint
    - One non-zero read of 1267 bytes from “/etc/localtime” at startup

POP Trace File Characteristics

- One file per MPI task

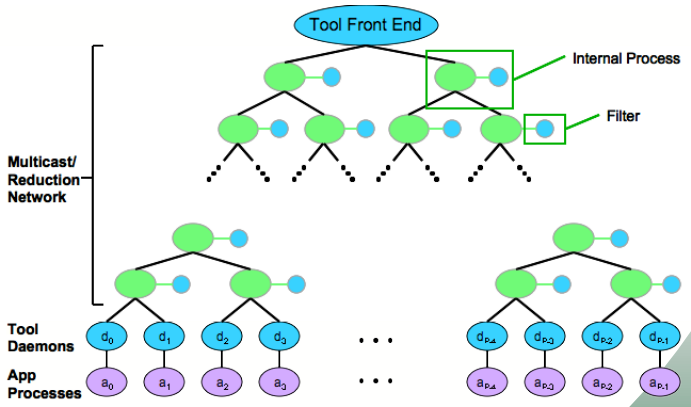
	Long			Short			Short-compressed		
	def	Fixed	Per-step	def	Fixed	Per-step	def	Fixed	Per-step
Rank 0	314	41877	8479.4	283	23873	4833.15	158	6101	720.25
Rank !=0	314	424	1287.4	283	256	737.3	158	2266	66.6

All values in bytes

- Performance data volume reflects:
  - Local OTF metadata files
  - Fortran I/O operations (open, read, write, close)
  - Number of bytes read/written
  - Operation timestamp and duration
- Data volume does not reflect:
  - Global metadata file
  - Complete function metadata
  - MPI communication in support of I/O
  - Number of bytes requested
  - Seeks

Ongoing and Future Work

- More sophisticated data collection and analysis
  - Data collection scalability improvements
    - Selective, dynamic instrumentation
    - Scalable data reduction using Tree-Based Overlay Networks like MRNet (http://www.paradyn.org/mrnet)
  - Online analysis
  - Overhead analysis
- Increased breadth and depth in characterizations
- Packaging for release
- Support for Blue Gene/P
- Integration with Sequoia tracing infrastructure



Acknowledgements

- This research is sponsored by the Office of Advanced Scientific Computing Research; U.S. Department of Energy. The work was performed at the Oak Ridge National Laboratory, which is managed by UT-Battelle, LLC under Contract No. De-AC05-00OR22725. Accordingly, the U.S. Government retains a non-exclusive, royalty-free license to publish or reproduce the published form of this contribution, or allow others to do so, for U.S. Government purposes.
- Thanks to Jeffrey Vetter, Weikuan Yu, and the rest of the ORNL Future Technologies Group
- Thanks to Pat Worley for facilitating Cray XT system access

Summary

- Want to understand the gap between application I/O demands and system I/O capabilities on leadership class systems
- IOT event tracing infrastructure represents initial steps toward that goal

- Project information:
  - http://ft.ornl.gov/projects/io/
  - rothpc@ornl.gov