Debugging scalable hybrid and accelerated applications on the Cray XC30 and CS300 with TotalView

Agenda

- Introduction
- Rogue Wave Update
  - OpenLogic
  - Klocwork
- Totalview Overview
  - NVIDIA and Xeon Phi
  - Memory Debugging
  - Reverse Debugging
- Current work and future plans
  - Improved Performance at Scale
  - Next release

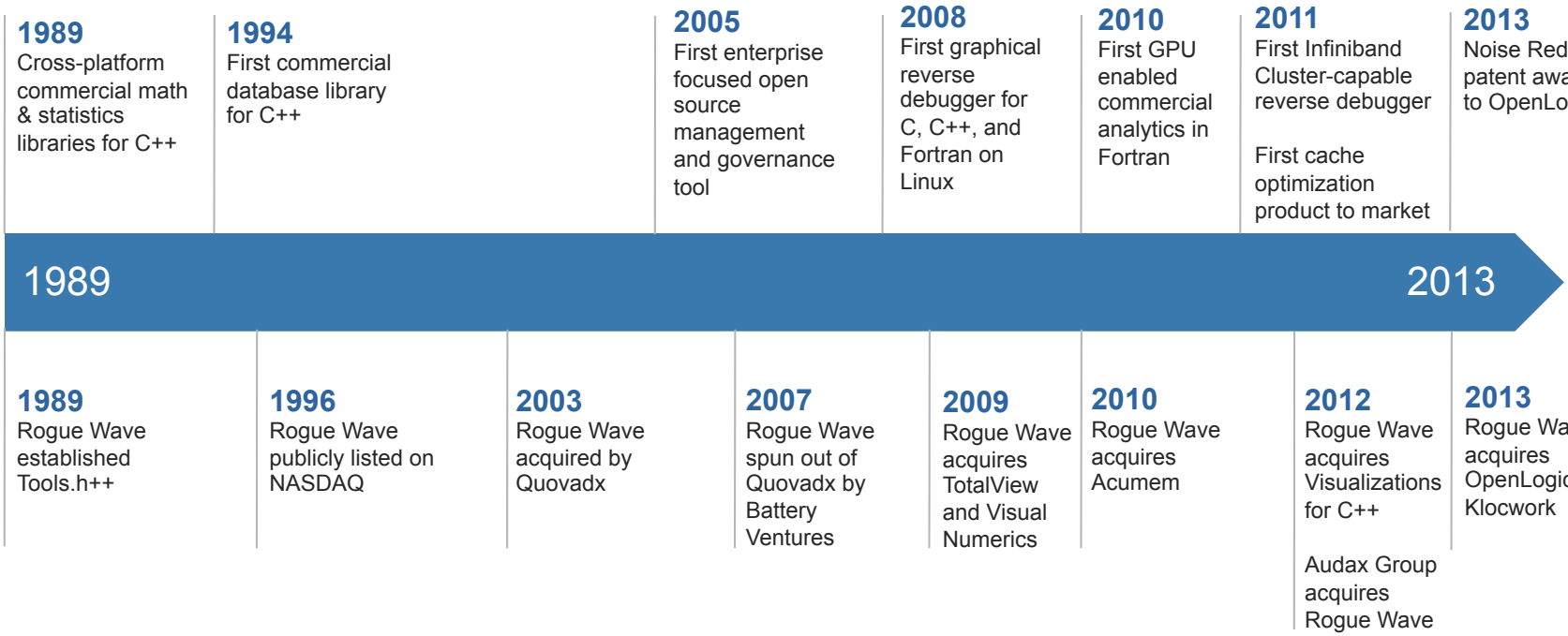
Hybrid and Accelerated Applications

- What do we see
  - NVIDIA Tesla GP-GPU computational accelerators
  - Intel Xeon Phi Coprocessors
  - Complex memory hierarchies (numa, device vs host, etc)
  - Custom languages such as CUDA and OpenCL
  - Directive based programming such as OpenACC and OpenMP
  - Core and thread counts going up
- A lot of complexity to deal with if you want performance
  - C or Fortran with MPI starts to look “simple”
  - Everything is Multiple Languages / Parallel Paradigms
  - Up to 4 “kinds” of parallelism (cluster, thread, heterogeneous, vector)
  - Data movement and load balancing

Rogue Wave Update

Company timeline

Technology Timeline



Corporate Timeline

OpenLogic workflow and governance

Challenge: For a large financial services organization, when security information is available in open source – who and what does this impact? How can they systematically address?

Results:

- Clear request/approval process to track which teams are using which OSS packages
- API-based access to new security information and affected packages
- Notifications to requestors informing them of the new security information

:: Users are automatically kept up-to-date when security notices may affect them

OpenLogic vulnerability scanning

Challenge: Identify new OSS in ISV's codebase. Releases trigger potential "copyleft" unseen vulnerabilities.

Results:

- Delta scans to pinpoint only changed or new OSS files
- One scan for baseline, then scans for subsequent releases
- Decision-memory to limit remediation of scanned code

:: OSS code compliance without a complete re-scan with each release. Vulnerabilities addressed while significantly reducing the time spent

Klocwork finds critical defects

Challenge: Advance the prosthetic arm development, adhere to FDA regulations, and integrate Agile build methodology – all on a tight timeline.

Results:

- Develop to safety-critical and real-time requirements
- Saved 900 person-hours in five months
- Found 225 total defects, 83 critical
- 0.5% false positive rate

“We had a very tight schedule and without Klocwork, we would have had difficulty meeting our objectives.”

Klocwork spots faulty code

Challenge: Mitigate risk of faulty code in embedded development, define consistent practices across all divisions.

Results:

- Standardize on one tool
- Consistent results across five divisions
- Risk reduction for post-release failures and attendant costs

“The ability to find and fix issues as the code is being written, before it leaves developers’ desktops, is very powerful. I’m really pleased with the amount of real defects that are being found, and the ratio of warnings to defects is good.”

IMSL C Numerical Library

- IMSL 8.5 January 2014
- Algorithms
  - Decision Tree Algorithms
  - Apriori Analysis
  - Support Vector Machines: pattern recognition in high dimensional data
  - Kohonen Self Organizing Maps: clustering in high D data
  - Vector Self Regression
  - Vector Error Correction Model
- Features
  - Work with data > physical memory
  - Works with streaming data
  - CUDA 5.5

IMSL embedded analytics

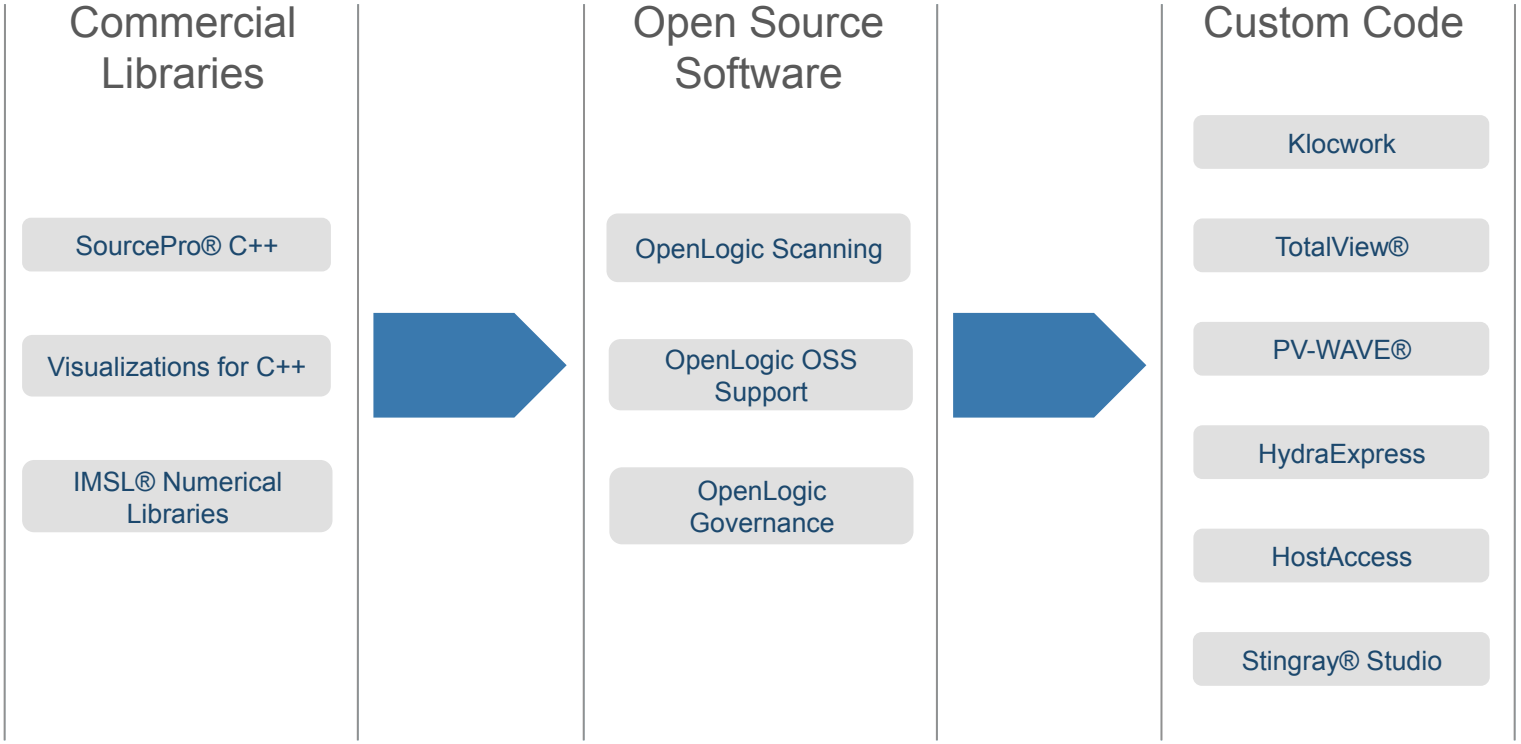
Challenge: Find a standardized analytic to embed, that scales, and covers a number of use cases for flagship in-memory database.

Results:

- Optimized and consistent results across multiple platforms
- Embed to perform analytics where the data lives
- Reliable within the database context – no additional overhead or risk
- Breadth of analytics to cover uses within all industry verticals
- Standardized for seamless integration

:: Product differentiation in highly competitive BI and database market

Solutions for all code bases



Global, diversified customer base

Used by 3,000 customers in over 50 countries across diverse industries to develop mission-critical applications and software

TotalView Overview

Hybrid and Accelerated Applications

- What do we see
  - NVIDIA Tesla GP-GPU computational accelerators
  - Intel Xeon Phi Coprocessors
  - Complex memory hierarchies (numa, device vs host, etc)
  - Custom languages such as CUDA and OpenCL
  - Directive based programming such as OpenACC and OpenMP
  - Core and thread counts going up
- A lot of complexity to deal with if you want performance
  - C or Fortran with MPI starts to look “simple”
  - Everything is Multiple Languages / Parallel Paradigms
  - Up to 4 “kinds” of parallelism (cluster, thread, heterogeneous, vector)
  - Data movement and load balancing

How does Rogue Wave help?

TotalView debugger

- Troubleshooting and analysis tool
  - Visibility Into
  - Control Over
- Scalability
- Usability
- Support for HPC platforms and languages

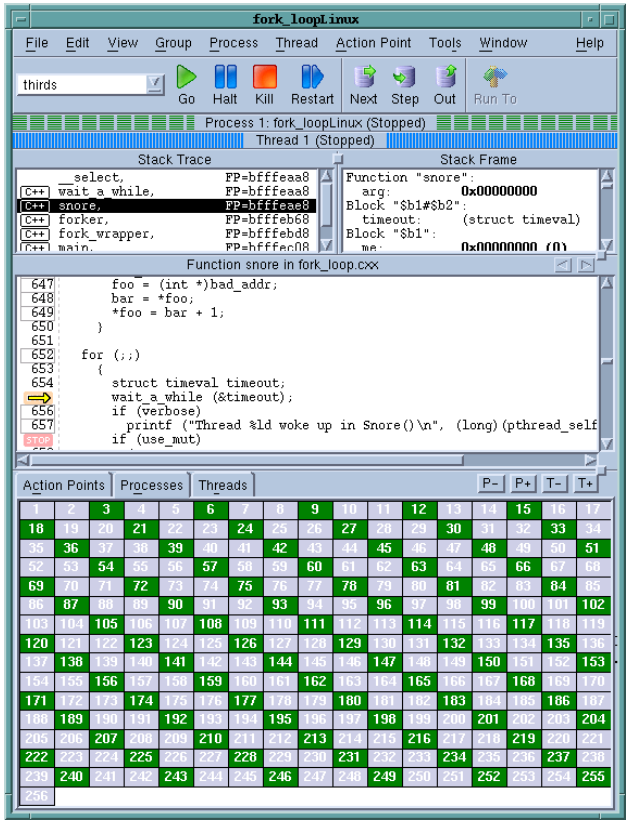
What is TotalView®?

Application Analysis and Debugging Tool: Code Confidently

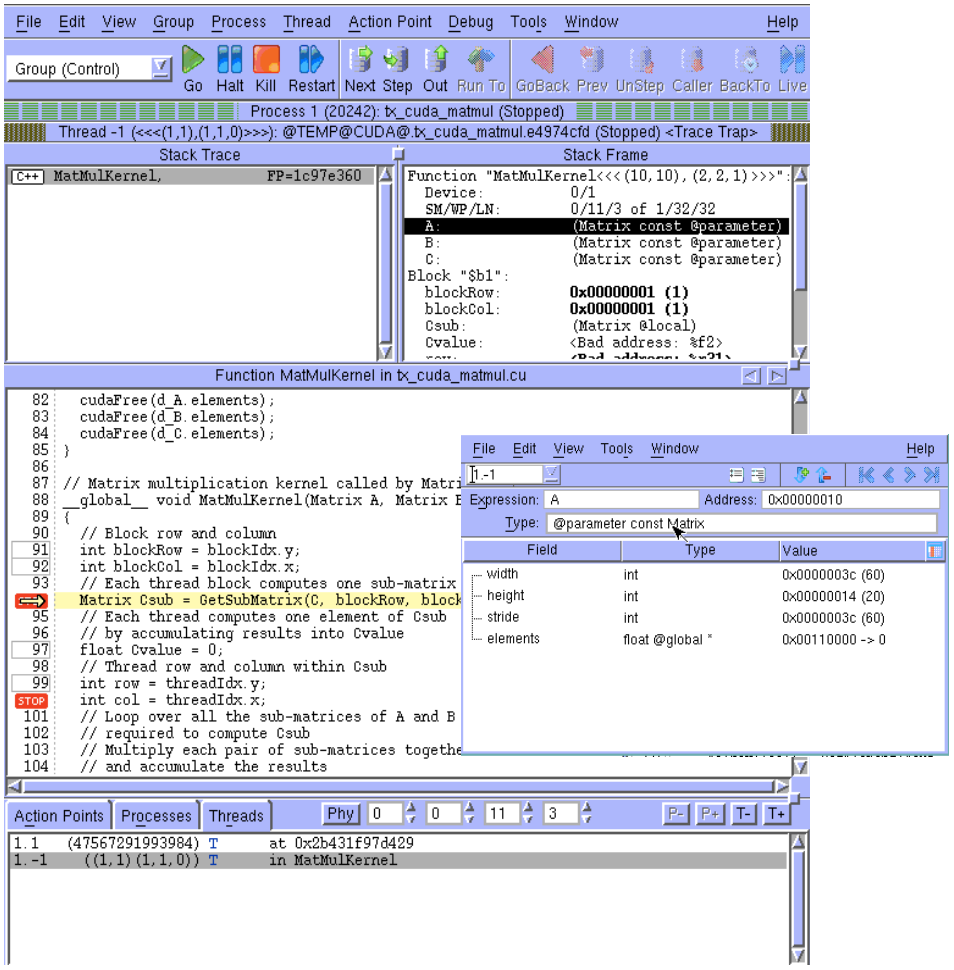
- Debug and Analyse C/C++ and Fortran on Linux™, Unix or Mac OS X
- Laptops to supercomputers
- Makes developing, maintaining, and supporting critical apps easier and less risky

Major Features

- Easy to learn graphical user interface with data visualization
- Parallel Debugging
  - MPI, Pthreads, OpenMP™, GA, UPC
  - CUDA™, OpenACC®, and Intel® Xeon Phi™ coprocessor
- Low tool overhead resource usage
- Includes a Remote Display Client which frees you to work from anywhere
- Memory Debugging with MemoryScape™
- Deterministic Replay Capability Included on Linux/x86-64
- Non-interactive Batch Debugging with TVScript and the CLI
- TTF & C++View to transform user defined objects



TotalView for the NVIDIA ® GPU Accelerator



- NVIDIA Tesla K40
- NVIDIA CUDA 5.0 and 5.5 new in 8.13
  - With limited support for dynamic parallelism
- Cray CCE OpenACC
- Features and capabilities include
  - Support for MPI based clusters and multi-card configurations
  - Flexible Display and Navigation on the CUDA device
    - Physical (device, SM, Warp, Lane)
    - Logical (Grid, Block) tuples
  - CUDA device window reveals what is running where
  - Support for types and separate memory address spaces
  - Leverages CUDA memcheck

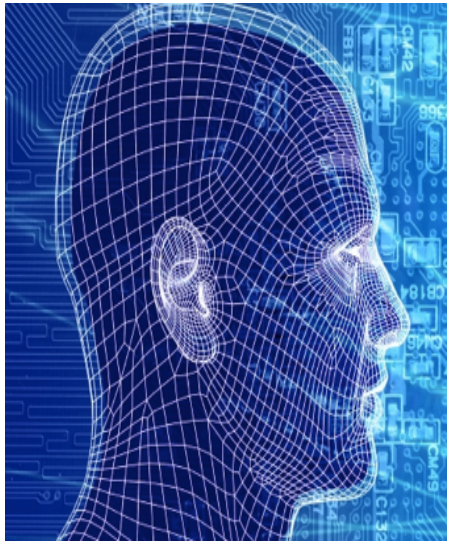
RogueWave SOFTWARE

HRL Laboratories

- “In the first full day of using TotalView, we were quickly able to solve the bug that had us stumped for weeks. “

  Kirill Minkovich, HRL Laboratories

- Neural Network Simulation in the Defense Industry
- Need to adopt new technology *and* achieve massive scaling on a deadline
- Project went from failing to over-achieving



TotalView for the Intel® Xeon Phi™ coprocessor

Supports All Major Intel Xeon Phi Coprocessor Configurations

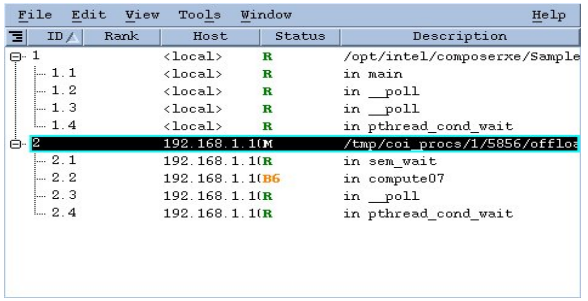
- Native Mode
  - With or without MPI
- Offload Directives
  - Incremental adoption, similar to GPU
- Symmetric Mode - New in 8.13
  - Host and Coprocessor
- Multi-device, Multi-node
- Clusters

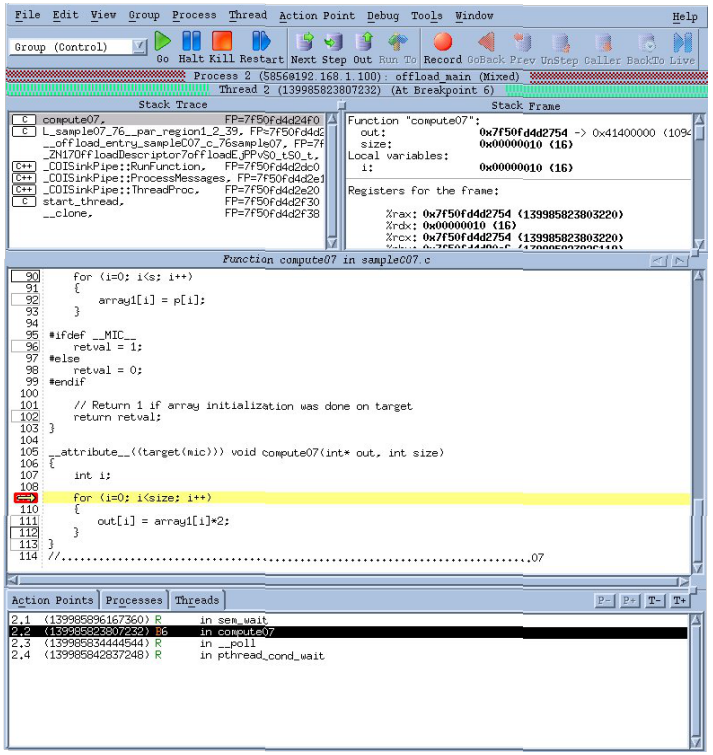
User Interface

- MPI Debugging Features
  - Process Control, View Across, Shared Breakpoints
- Heterogeneous Debugging
  - Debug Both Xeon and Intel Xeon Phi Processes

Memory Debugging - New in 8.13

- Both native and symmetric mode





Beacon Project

- 210 Tflops, 11,520 Xeon Phi Coprocessor Cores
- #1 on the Dec 2012 Green 500

- 9 scientific teams optimizing apps including: MHD, Plasma, Cosmology, Chemistry, QCD, Bio-informatics…

- Porting & optimization
  - Hybrid MPI + OpenMP
  - Many more threads than previous paradigms

- Subtle issues might present themselves
  - And did

Debugging on Beacon with TotalView

- OpenMP Hybridization of Boltzman BGK
  - Correctness issues came up with the OpenMP code
  - Troubleshooting with TotalView
    - Native mode debugging on the Xeon Phi
    - Thread level examination of the OpenMP region
    - Comparison of data between threads
    - … Clarified otherwise puzzling results
  - Developers were able to resolve the correctness issue
  - Ultimately obtained performance gains on the Xeon Phi
    - And correct results

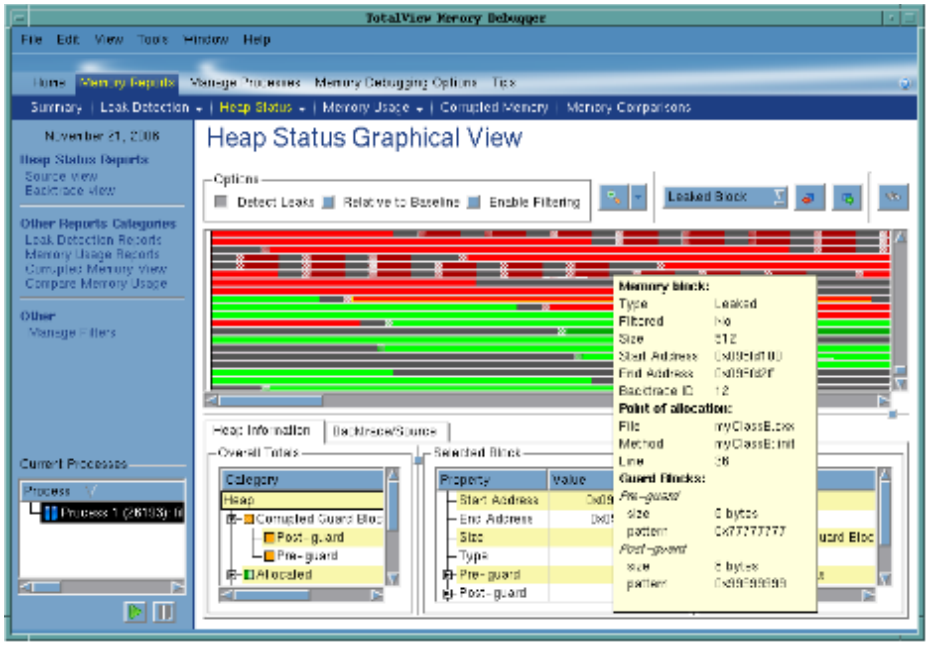
Debugging on Beacon with TotalView

- Porting Gyro Tokamak Plasma Simulation to the Xeon Phi
  - Intermittent crash due to Out Of Memory (OOM) Condition
  - Troubleshooting with TotalView
    - TotalView was used to diagnose the issue across MPI processes
    - Work was not being distributed evenly
    - … the routine had an invalid assumption
  - Developers were able to resolve the OOM error
  - Better load balancing also improved the performance of the code

What Is MemoryScape®?

- Runtime Memory Analysis : Eliminate Memory Errors
  - Detects memory leaks *before* they are a problem
  - Explore heap memory usage with powerful analytical tools
  - Use for validation as part of a quality software development process

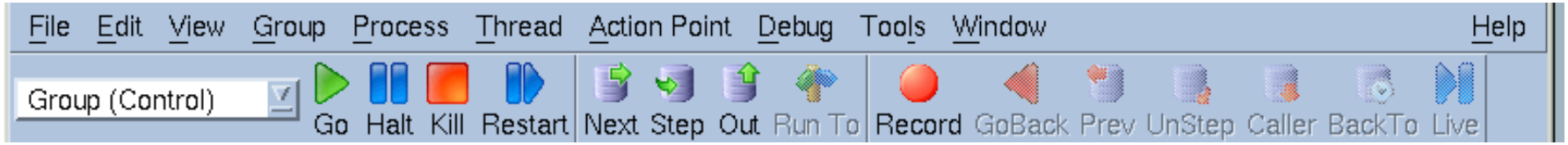
- Major Features
  - Included in TotalView, or Standalone
  - Detects
    - Malloc API misuse
    - Memory leaks
    - Buffer overflows
  - Supports
    - Now includes Intel® Xeon Phi™
    - C, C++, Fortran
    - Linux, Unix, and Mac OS X
    - MPI, pthreads, OMP, and remote apps
  - Low runtime overhead
  - Easy to use
    - Works with vendor libraries
    - No recompilation or instrumentation



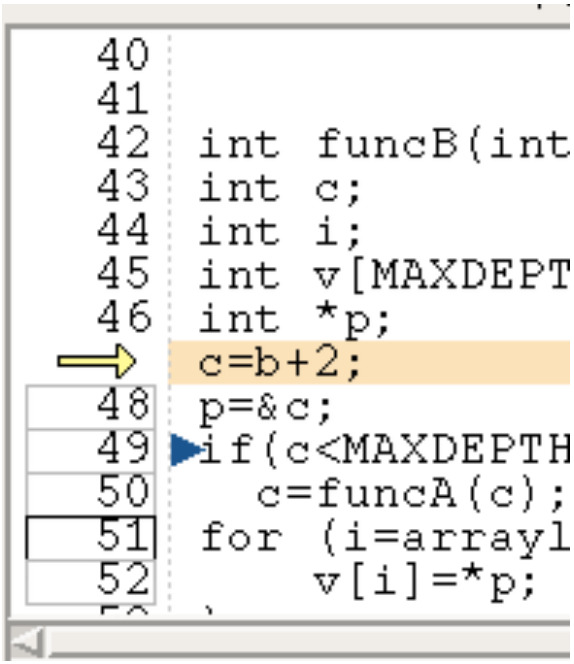
Dassault Systems

- ***"MemoryScape enabled us to identify memory issues, and by using its scripting interface, we were able to automate the evaluation process. Now, the system automatically uncovers any hidden latent errors in our code with every build, allowing our developers to proactively fix potential errors prior to release."***
- *"MemoryScape enabled us to … proactively fix potential errors prior to release."*
  - Nick Monyatovsky, Software Engineer
- Computer Aided Engineering ISV for Aero/Auto/Industry
  - **Nick Monyatovsky, Software Engineer at SIMULIA**
- Struggling with intermittent errors
- Continuous Integration – Better Product Quality

Deterministic Replay Debugging



- Reverse Debugging: Radically simplify your debugging
  - Captures and Deterministically Replays Execution
    - Not just "checkpoint and restart"
  - Eliminate the Restart Cycle and Hard-to-Reproduce Bugs
  - Step Back and Forward by Function, Line, or Instruction



- Specifications
  - A feature included in TotalView on Linux x86 and x86-64
    - No recompilation or instrumentation
    - Explore data and state in the past just like in a live process, including C++View transformations
  - Replay on Demand: enable it when you want it
  - Supports MPI on Ethernet, Infiniband, Cray XE Gemini
  - Supports Pthreads, and OpenMP

Cambridge Study

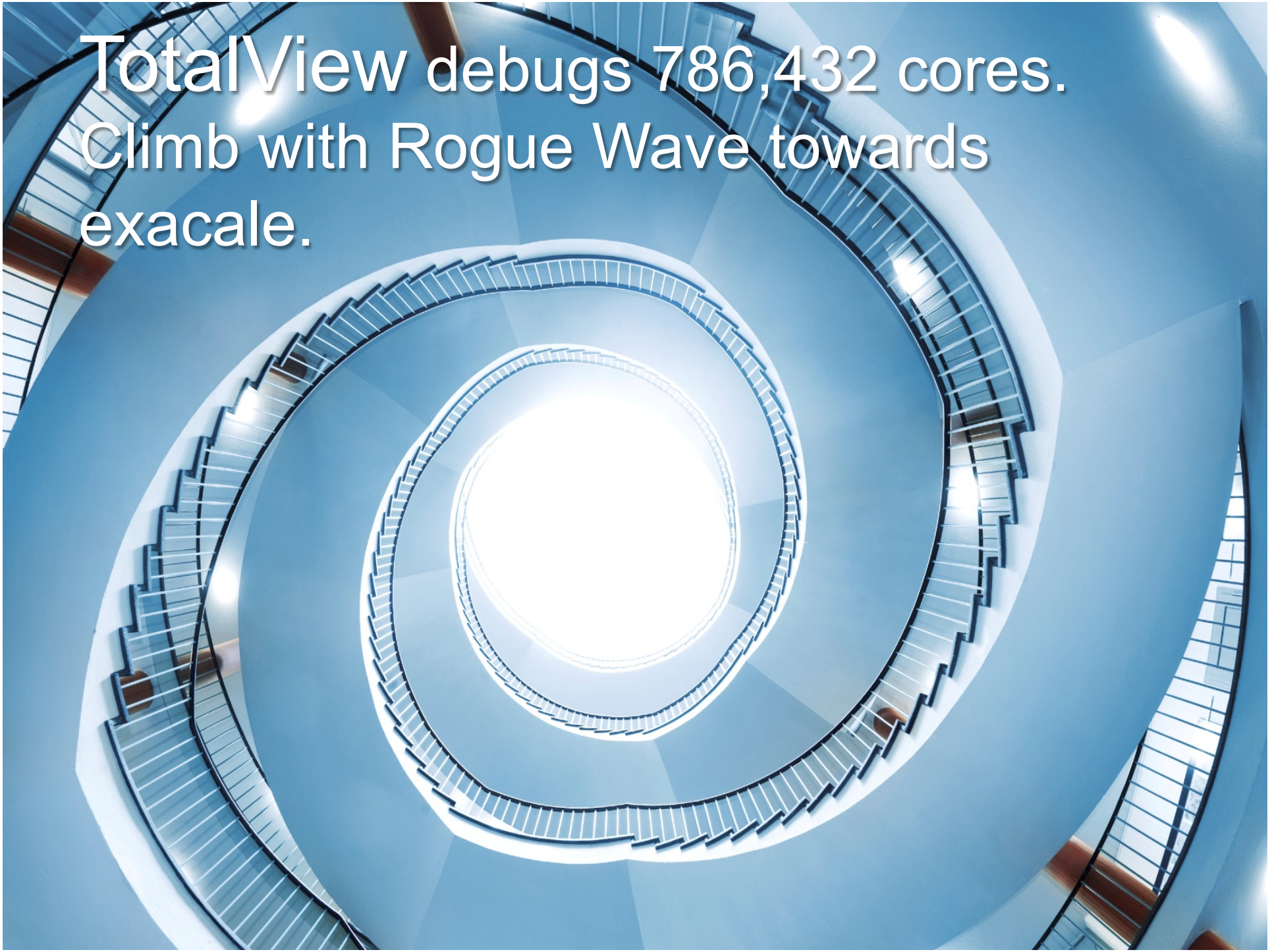
UNIVERSITY OF CAMBRIDGE
Judge Business School

- Survey conducted by the Judge Business School at Cambridge University concluded that Reverse Debuggers allow users, on average, to spend 13% less of their programming time debugging.
  - Programming was 50% of total work week on average
  - Debugging was 50% of programming time without reverse debugging
  - Debugging was 37% of programming time with reverse debugging
  - That frees up 130 hours (>3 work weeks, 6.5% total time) per developer per year for design and new feature development
- The survey looked at total value (salaries & overhead) of debugging as a task and they determined that this savings could, across the whole world economy, be work $41 billion in increased productivity.
  - The productivity improvement should be worth $2,500 per developer per year (salary only) or $5,000 per year with overhead.
- http://www.roguewave.com/company/news-events/press-releases/2013/university-of-cambridge-reverse-debugging-study.aspx

Current Work
and
Future Plans

Multi-phase R&D Projects Underway

- Massive Scalability
  - Collaboration with LLNL and Tri-lab partners
  - Targeting Cray, Blue Gene and Linux Clusters
- Shiny new GUI
  - Sleek, Modern and Fast
  - Configurable
  - Improved Usability
  - Provides aggregation capabilities for big data and scale
  - Leveraging math and stat expertise from IMSL
- Working with customers through early access programs
  - Customer input is key to the success of both programs


TotalView debugs 786,432 cores.
Climb with Rogue Wave towards exacale.

Some more details on the 786,432 core test

- The test was performed on 48 racks of Sequoia
- The test code
  - Implements a Jacobi Linear Equation Solver
  - The test code is a hybrid MPI + OpenMP code
  - 16 threads per process, one process per node
- The test operations
  - Start up
  - Setting breakpoints / removing breakpoints
  - Single stepping all threads
- Tests performed at a variety of scales to understand scalability

Second test - Oversubscription

- Same framework
  - same code
  - same machine
- Oversubscription
  - Scheduled more than one thread per physical core
  - This is a reasonable use case since the BG/Q supports 4 logical threads per core
- TotalView Debugged 1,048,576 threads

Currently…

Tuning the debugger to run faster on your codes

- Collaborative technique called: Application Driven Tuning
- Using applications provided by our tri-lab collaborators
  - Such as ALE3d and IRS
- Using three target architectures
  - Linux/x86
  - Blue Gene/Q
  - Cray XE/XK/XC

Start Up

Focusing on shortening the time to start up

- When you start up an application under TotalView
  - Launch the debugger
  - Launch the application
  - Attach to the application
  - Prepare for debugging (partially digest debug information)
- The time to launch is sensitive to
  - Time to start the application without the debugger
  - Time to run to MPI_INIT
  - Number of shared libraries loaded
  - Size and complexity of application

Caching object data from the cluster

TotalView was doing unneeded work

- TotalView needs to know about subtle differences in objects
  - Libraries aren't always 100% the same across the cluster
  - Different versions of libraries will have functions and lines at different places
  - Objects get loaded at different addresses
- Cray runtime copies object files across into the node during parallel launch
  - at a unique address per rank (based on the apid)
- Cray DVS makes a shared file system available to the compute nodes
  - The application program runs inside a chroot
  - The debugger agents (tvdsvr) not inside the chroot
- These two effects meant that TV was seeing phantom differences in the cluster
  - Teaching TV to work around these effects greatly reduced start up times

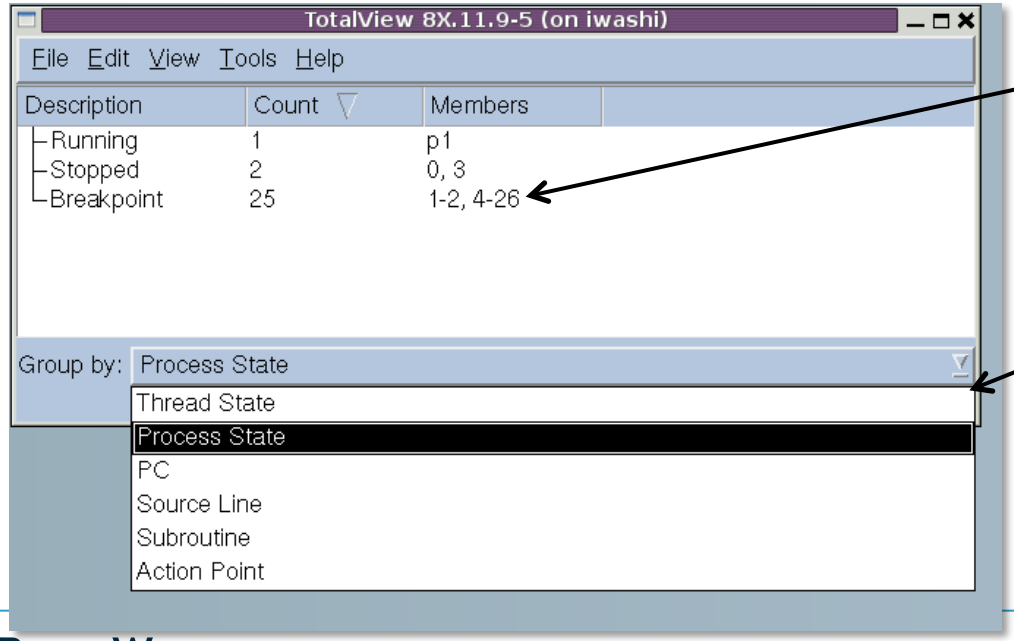
TotalView Scalable Early Access Version

Making these improvements available to early users

- SEA Platforms
  - Linux-x86_64, Blue Gene/Q, Cray X-series
- All versions represents a development “snapshot”
- New advancements are released as they become available
  - The “good”: access to the latest features/performance
  - The “bad”: features partially implemented, changed w/o notice
  - The “ugly”: not production quality, not fully tested
- MRNet must be explicitly enabled
  - Lots of details are in *TotalView Scalability Using MRNet*
- The prototype UI features must be explicitly enabled

New-Style Root Window (SEA2+)

- A prototype new-style root window w/ “-demo_ui”
- Displays aggregated program information
- Intended to eventually replace the old-style root window
- Menu items that are not yet implemented are disabled



- Diving selects a representative of the group and refocuses the process window
- Current aggregations
- Hierarchical groupings planned

Compressed *ptlist* Syntax

- Aggregation requires a compact process/thread set representation (for both CLI *and* GUI output)
- General syntax of a *ptlist*

  ptlist : *pcount* ':' *tcount* '[' *ptrange* [',' *ptrange*] … ']'

  ptrange : *prange* '.' *trange*

  prange : *rank* ['-' *rank*]

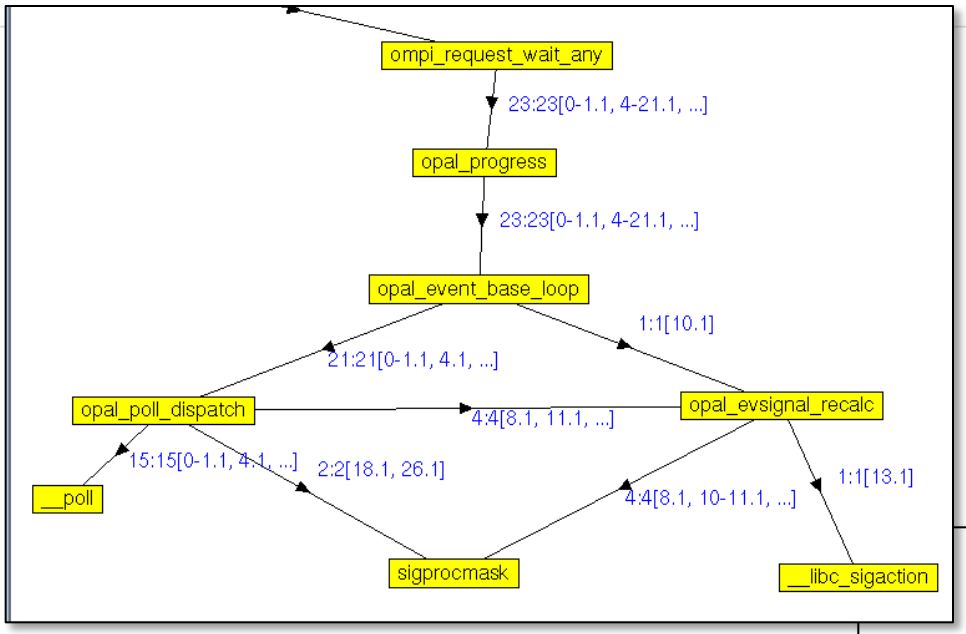
  | 'p' *dpid* ['-' *dpid*]

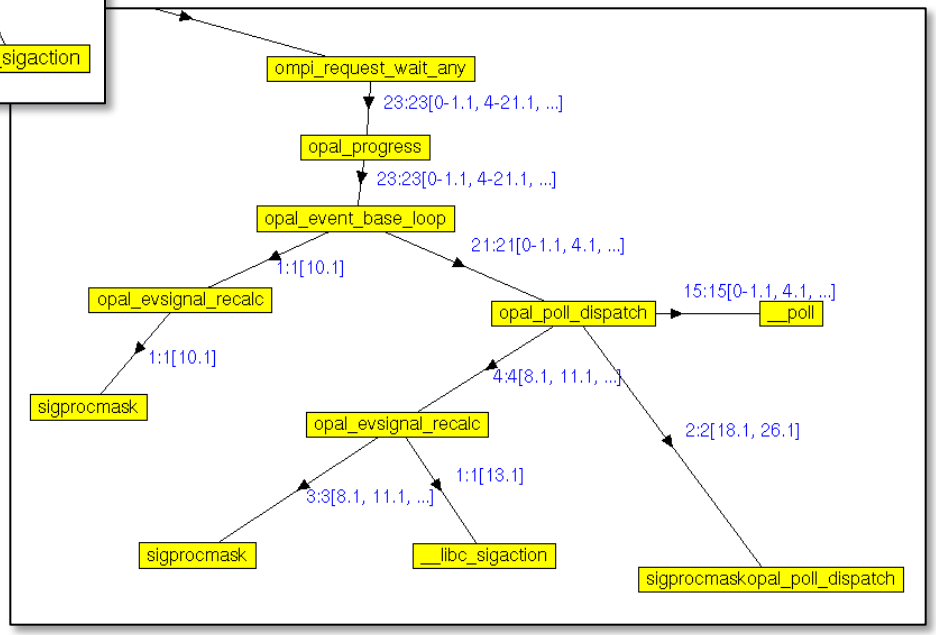
  trange : *dtid* ['-' *dtid*]
- Inspired by STAT and previous TotalView implementations

- Example

```
28:28[0-26.1, p1.1]
```

Call Graph vs. Call Tree (SEA3+)





TotalView Scalable Early Access Summary

Please give it try!

- We value your feedback
- Enable MRNet and the demo UI
  - totalview –mrnet –demo_ui …
- Many infrastructure changes are in place already
  - Though not all operations parallelized yet
- User interface changes in prototype phase
  - More improvements coming in existing UI
  - Remaining improvements coming in new UI

- Questions?

Next Release

Plans subject to change without notice

- TotalView 8.14
  - CUDA 6.0 Support
    - Including Unified Memory
  - Co-array Fortran support
  - C++ Type Transform Support for Unordered Collections
    - Unordered Map
    - Unordered Set
    - Unordered Multimap
    - Unordered Multiset
  - Platform updates

What we do

Rogue Wave helps organizations simplify complex software development, improve code quality, and shorten cycle times

Thanks!

- To learn more / sign up for the Scalability Early Experience Program please contact me: chris.gottbrath@roguewave.com
- Let us know if you want to have a training workshop session: sales@roguewave.com
- Visit the website
  - http://www.roguewave.com/products/totalview.aspx
  - Videos (3 new videos on Xeon Phi)
  - Documentation
  - Sign up for an evaluation
  - Contact customer support & post on the user forum
- Visit us at ISC & SC