Debugging with TotalView on the Blue Gene Q

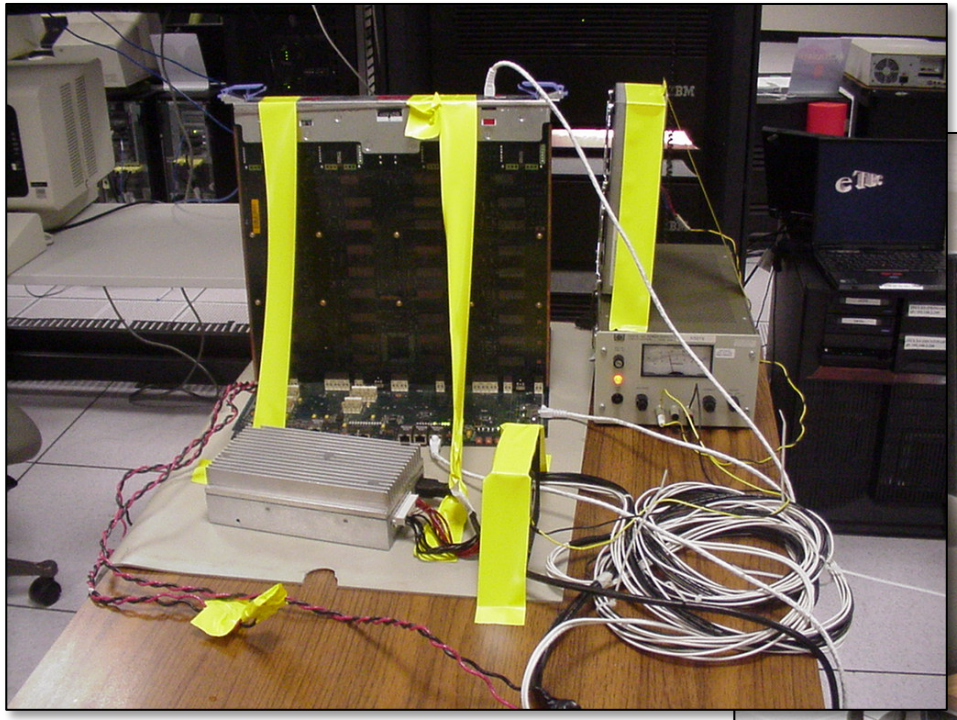
Ed Hinkel,
Sr Sales Engineer
Rogue Wave Software

Early Blue Gene Days

| Copyright © 2013 Rogue Wave Software | All Rights Reserved

TotalView Blue Gene Support

- **TotalView involvement started in 2003 on BG/L**



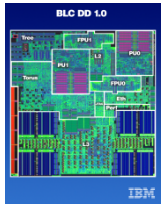
IBM/TV BG/L development system

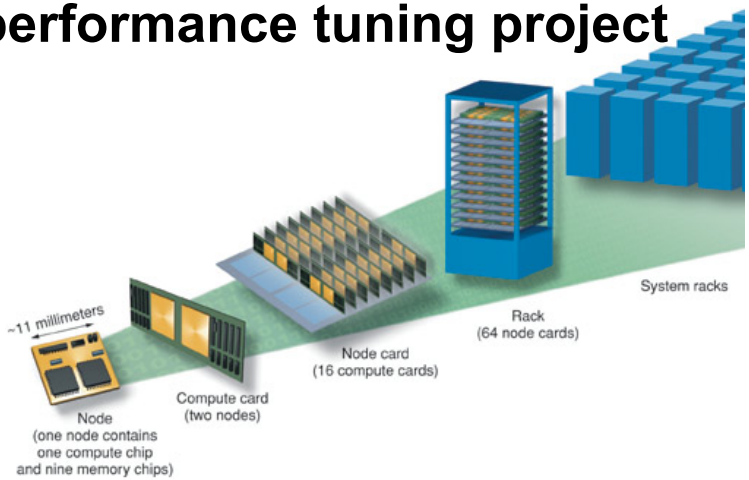
Gotta love that yellow duct tape!

| Copyright © 2013 Rogue Wave Software | All Rights Reserved

TotalView Blue Gene/L Support

- Support for Blue Gene/L since 2005
- Debugging interfaces developed via close collaboration with IBM (CIOD)
- Used on DOE/NNSA/LLNL's Blue Gene/L system containing 212 K cores
  - Heap memory debugging support added
  - Blue Gene/L scaling and performance tuning project



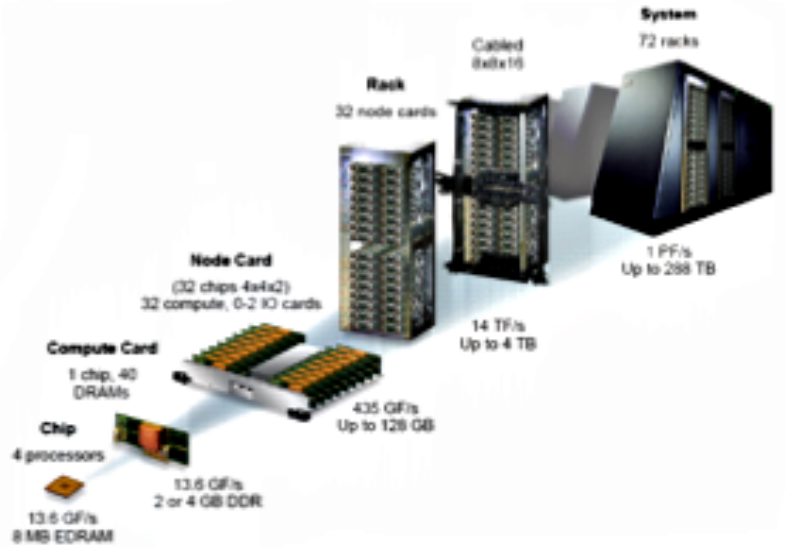


Blue Gene/L work facilitated Blue Gene/P support

| Copyright © 2013 Rogue Wave Software | All Rights Reserved

TotalView Blue Gene/P Support

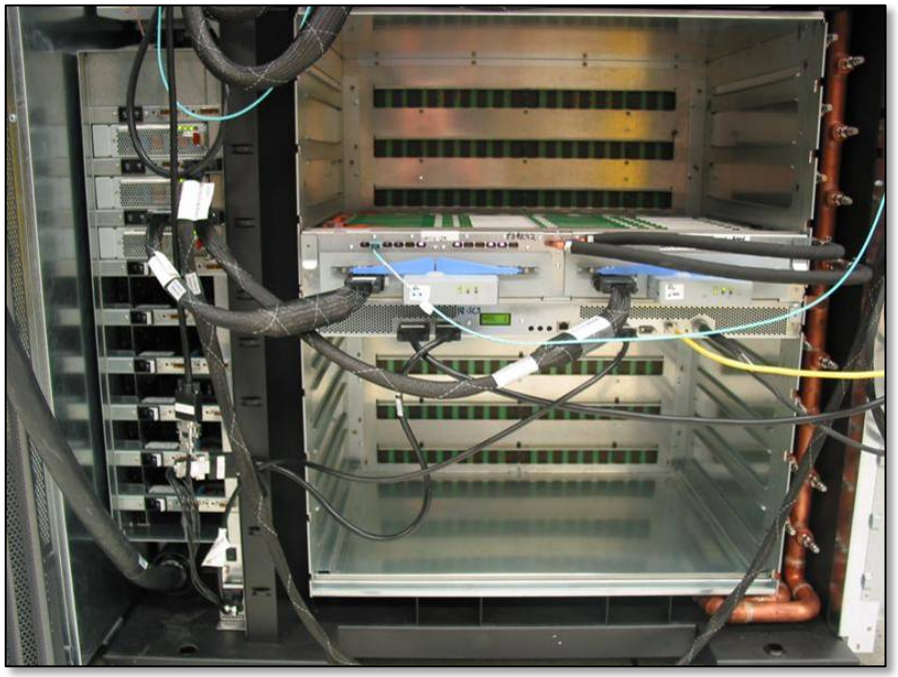
- Continued close collaboration with IBM
- Support for shared libraries, threads, and OpenMP
- TotalView on BG/P has debugged jobs as large as 32,768 cores
- Active workshop participation through the development
  - ANL's ALCF INCITE Performance Workshop
  - Jülich's Blue Gene/P Porting, Tuning, and Scaling Workshops



| Copyright © 2013 Rogue Wave Software | All Rights Reserved

TotalView Blue Gene/Q Support

- Porting TotalView began in June 2011
- Access to Q32 at IBM began in August
- Basic debugging in October 2011
- Used in Synthetic Workload Testing in December (LLNL)
- Fully functional in March 2012

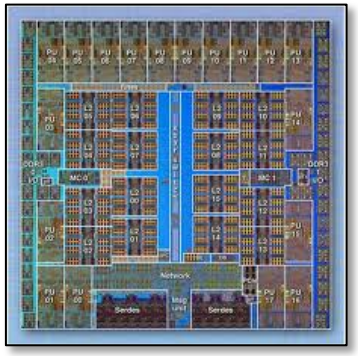


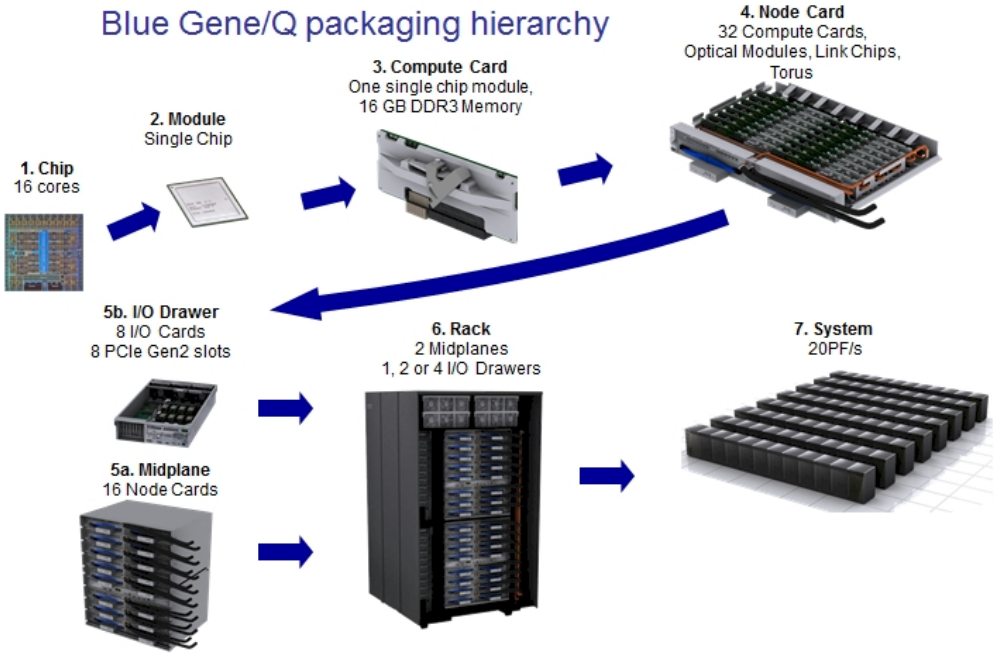
IBM's Q32

| Copyright © 2013 Rogue Wave Software | All Rights Reserved

TotalView Blue Gene/Q Support (cont)

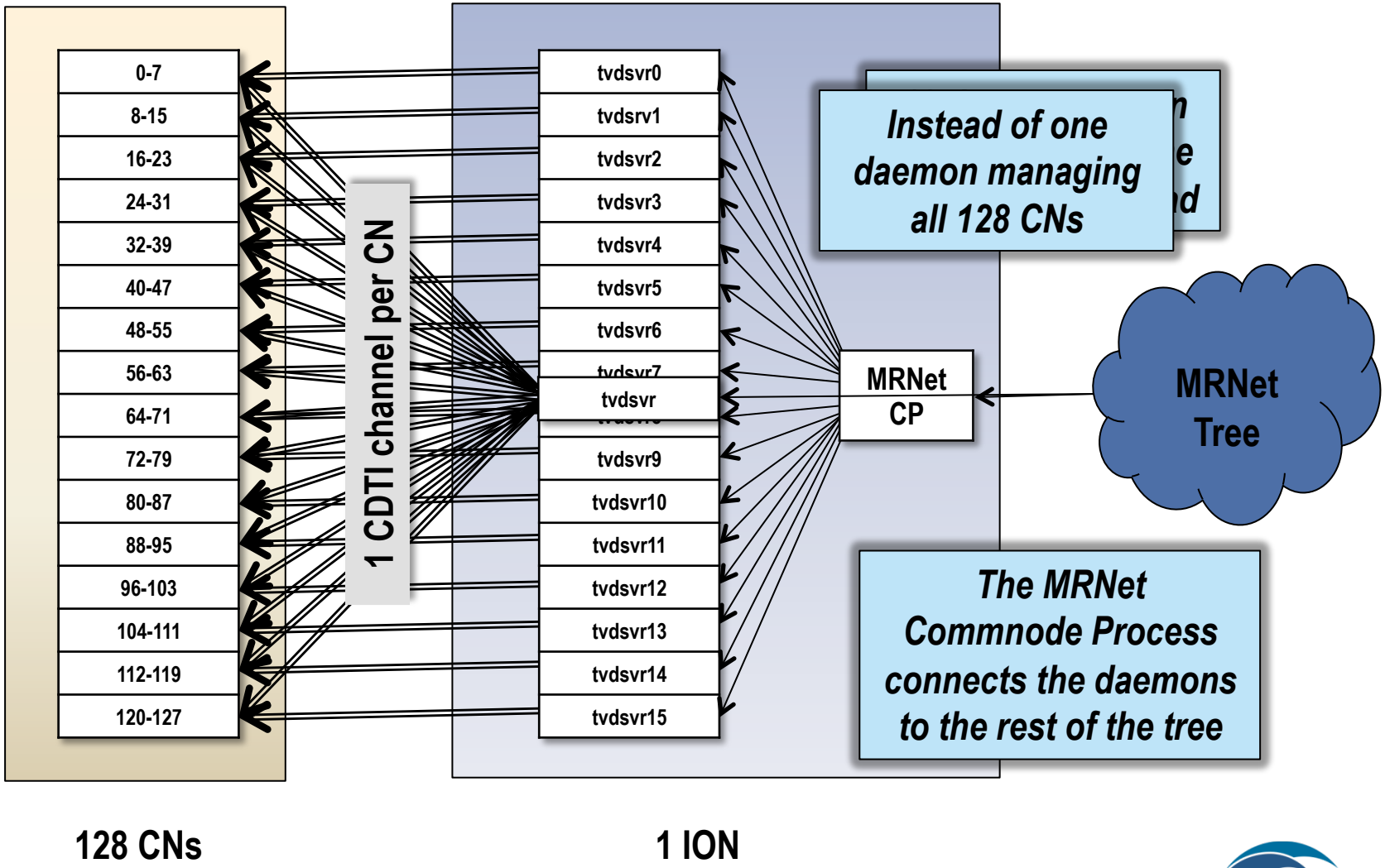
- Thanks to the ongoing collaboration with IBM and the BG Kernel Team, early access versions of TotalView were made available for BG/Q
- At Lawrence Livermore National Laboratory TotalView has now debugged jobs as large as 768,432 cores





| Copyright © 2013 Rogue Wave Software | All Rights Reserved

Solution: TotalView/MRNet Trees on the IO Nodes



128 CNs

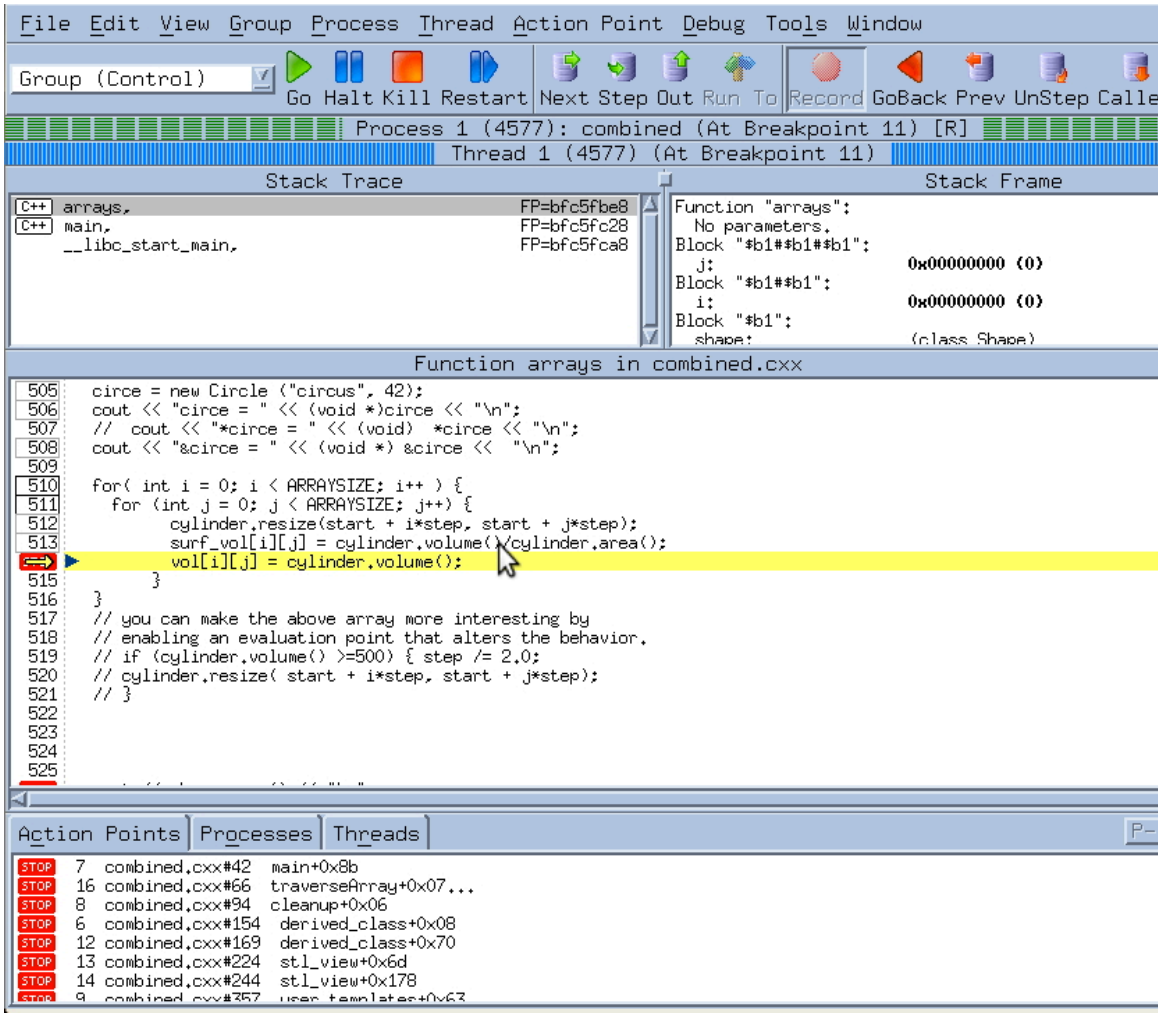
1 ION

| Copyright © 2013 Rogue Wave Software | All Rights Reserved

What is TotalView?

A comprehensive debugging solution for demanding parallel and multi-core applications

- **Wide compiler & platform support**
  - C, C++, Fortran 77 & 90, UPC
  - Unix, Linux, OS X
- **Handles Concurrency**
  - Multi-threaded Debugging
  - Multi-process Debugging
- **Integrated Memory Debugging**
- **Supports Multiple Usage Models**
  - Powerful and Easy GUI – Highly Graphical
  - CLI for Scripting
  - Remote Display Debugging
  - Unattended Batch Debugging



TotalView on BG/Q

- **BG/Q TotalView is as functional as BG/P TotalView**
  - MPI, OpenMP, pthreads, hybrid MPI+threads
  - C, C++, Fortran, assembler; IBM and GNU compilers
  - Basics: source code, variables, breakpoints, watchpoints, stacks, single stepping, read/write memory/registers, conditional breakpoints, etc.
  - Memory debugging, message queues, binary core files, etc.
- **PLUS, features unique to BG/Q TotalView**
  - QPX (floating point) instruction set and register model
  - Fast compiled conditional breakpoints and watchpoints
  - Asynchronous thread control

| Copyright © 2013 Rogue Wave Software | All Rights Reserved

ROGUE WAVE SOFTWARE

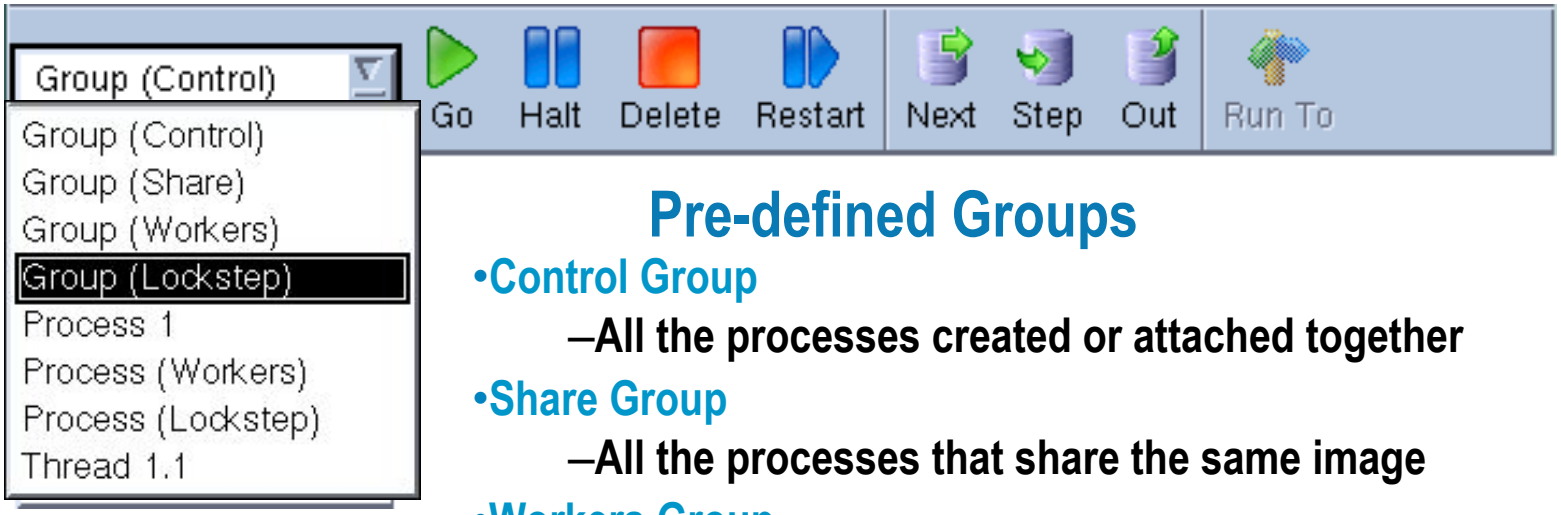
Key BG/Q TotalView Features

- **Asynchronous thread control**
  - Allows you to individually control the execution of threads
  - Run and halt individual threads
  - Single-step a group of threads in lockstep
  - Hold and release the execution of individual threads
  - Create stop-thread and thread barrier breakpoints

| Copyright © 2013 Rogue Wave Software | All Rights Reserved

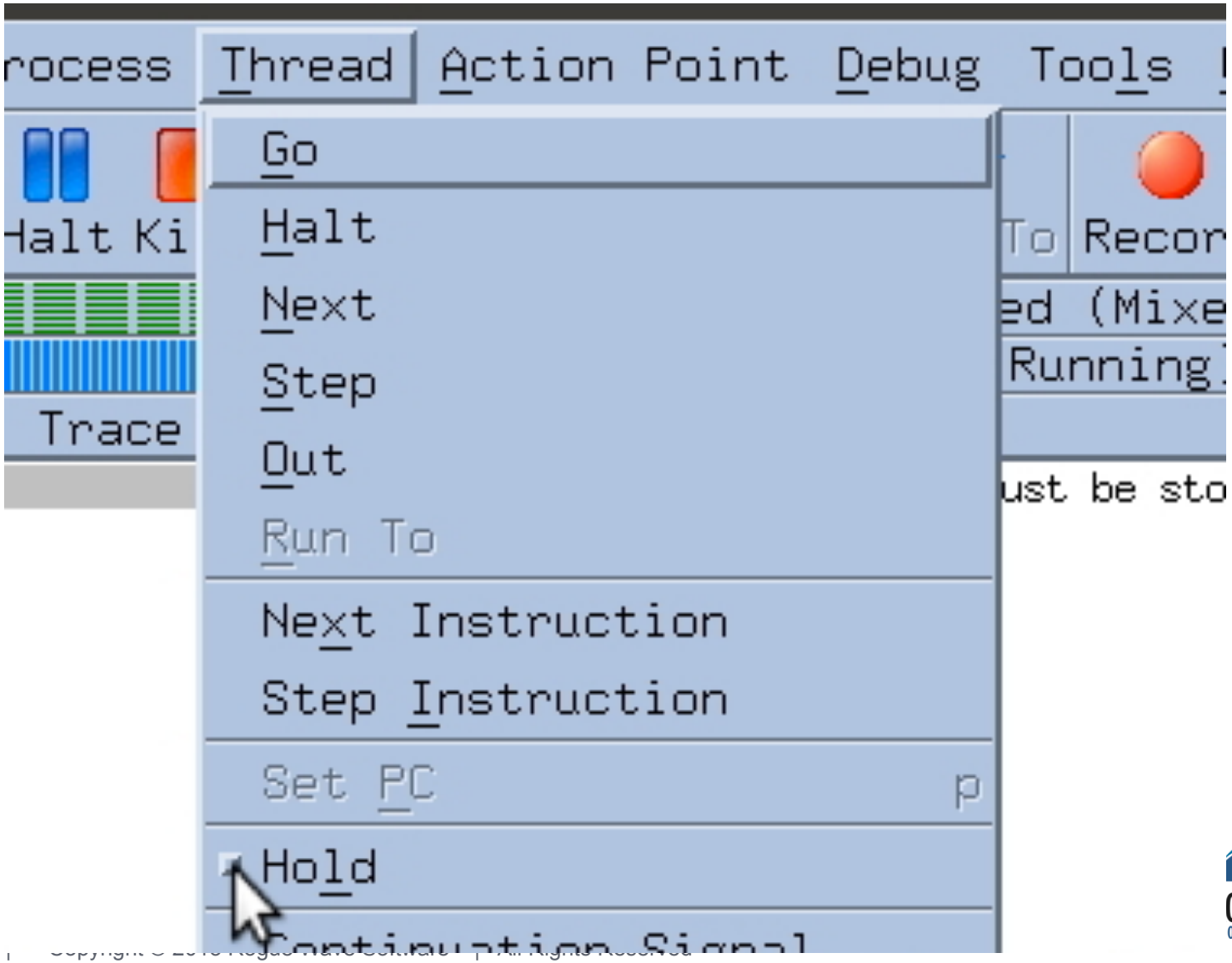
ROGUE WAVE SOFTWARE

Scope: Basic Thread/Process Control



Pre-defined Groups

- Control Group
  - All the processes created or attached together
- Share Group
  - All the processes that share the same image
- Workers Group
  - All the processes or threads that are not recognized as manager or service processes or threads
- Lockstep Group
  - All threads at the same PC
- Call Graph Group
  - All processes going through the same node in the call graph
- User Defined Group
  - Process group defined in Custom Groups dialog



Copyright © 2010 Rogue Wave Software, Inc. All rights reserved.

Setting Breakpoints

Advanced BG/Q TotalView Features

- **Fast compiled conditional breakpoints and watchpoints**
  - Conditional breakpoints and watchpoints execute in as little as 7 μsecs
  - Conditional expressions are compiled and dynamically patched into the process
  - Evaluation is performed in parallel by the triggering thread

| Copyright © 2013 Rogue Wave Software | All Rights Reserved

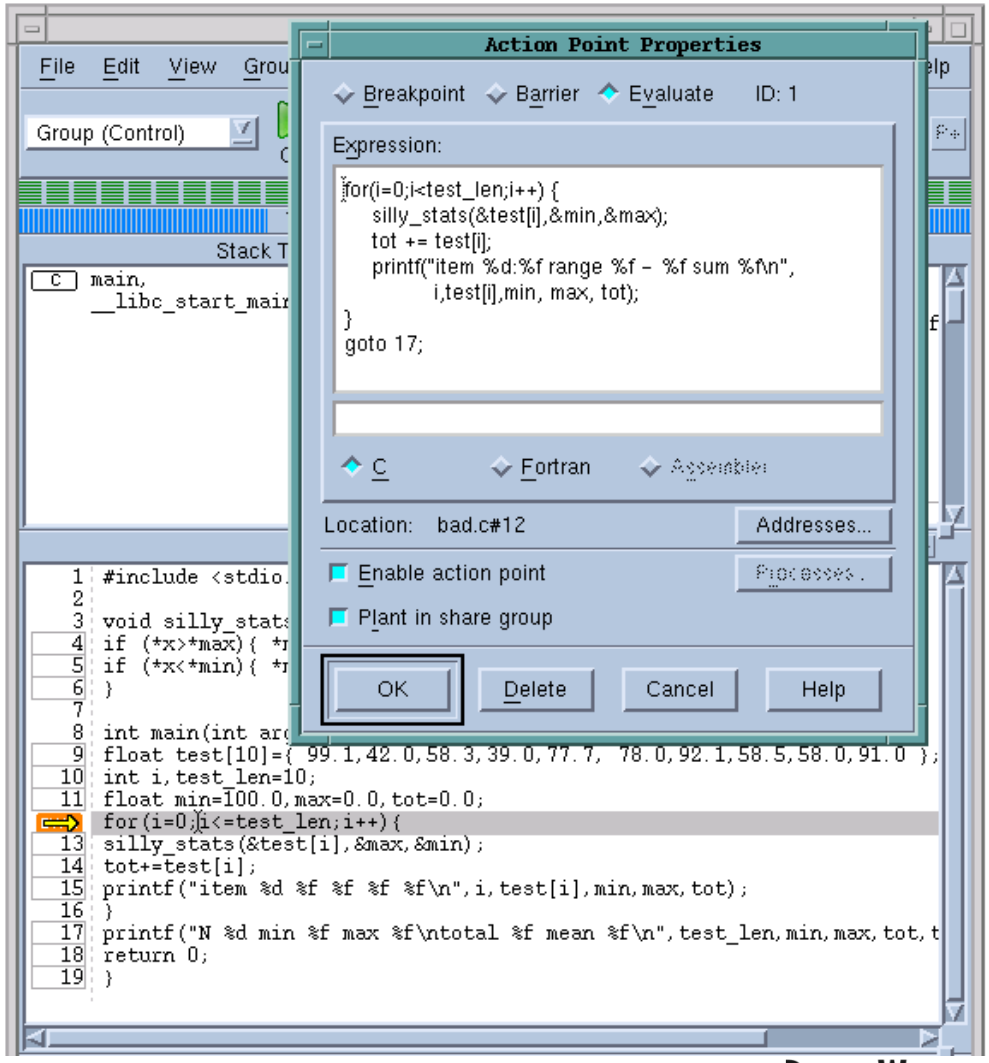
ROGUE WAVE SOFTWARE

Evaluation Breakpoints

Test Code Changes on the Fly!

- Test small source code patches
- Call functions
- Set variables
- Test conditions
- C/C++ or Fortran
- Can't use C++ constructors
- Use program variables
- Can't modify variables or call functions with replay engine

```
item 0:99.099998  range 99.099998 - 99.099998  sum 99.099998
item 1:42.000000  range 42.000000 - 99.099998  sum 141.100006
item 2:58.299999  range 42.000000 - 99.099998  sum 199.400009
item 3:39.000000  range 39.000000 - 99.099998  sum 238.400009
item 4:77.699997  range 39.000000 - 99.099998  sum 316.100006
item 5:78.000000  range 39.000000 - 99.099998  sum 394.100006
item 6:92.099998  range 39.000000 - 99.099998  sum 486.200012
item 7:58.500000  range 39.000000 - 99.099998  sum 544.700012
item 8:58.000000  range 39.000000 - 99.099998  sum 602.700012
item 9:91.000000  range 39.000000 - 99.099998  sum 693.700012
N 10 min 39.000000 max 99.099998
total 693.700012 mean 69.370001
```



ROGUE WAVE SOFTWARE

TotalView

- **TotalView on Blue Gene/Q Today**
  - Lawrence Livermore Labs (LLNL) - USA
  - IDRIS - France
  - CINECA - Italy
  - JULICH - Germany
  - IBM uses TotalView internally for debugging and testing.
  - TotalView is installed on IBM's Blue Gene On Demand Center Q32 (if anyone has access to that system).

| Copyright © 2013 Rogue Wave Software | All Rights Reserved

ROGUE WAVE SOFTWARE

TotalView at Argonne

- Licensing
  - BG/P: 2048 processes (Latest version available 8.9.0.0)
  - BG/Q: 8192 processes (Research license)
- Startup overview
  - Compile–g–O0
    - OMP code compile -qsmp=omp:noauto:noopt
  - BG/P: softenv key “+totalview” or BG/Q: /soft/debuggers/totalview
  - Need X11 server and ssh –X forwarding
  - [BG/P] Start interacive job with *isub*
  - [BG/Q] Copy job scripts from /soft/debuggers/scripts/totalview
- More details:
- – [BG/P] http://www.alcf.anl.gov/resource-guides/totalview

| Copyright © 2013 Rogue Wave Software | All Rights Reserved

TotalView Scripts

/soft/debuggers/scripts/totalview-examples/

- **To submit:**

```
#!/bin/bash
qsub -t 60 -n 128 --mode script -O LOG --env DISPLAY=$DISPLAY ./runtv.sh
echo “After your job starts, do a 'tail -f LOG' to see output”
```

- **The job script runtv.sh :**

```
#!/bin/sh
# Modify the totalview arguments for your situation
echo "Starting Cobalt job script"
echo "DISPLAY is $DISPLAY"
/soft/debuggers/totalview -args runjob -p 1 -n 128 --block
$COBALT_PARTNAME --verbose 2 --envs
PAMID_VERBOSE=1 :yourprogram.exe
```

| Copyright © 2013 Rogue Wave Software | All Rights Reserved

Techniques for Debugging Complex Codes

- **Mechanize**
- **Minimize**
- **Visualize**
- **… and Don’t Forget the Memory**

| Copyright © 2013 Rogue Wave Software | All Rights Reserved

ROGUE WAVE SOFTWARE

Mechanize

Extended Automation Capabilities

 Copyright © 2010 Rogue Wave Software | All Rights Reserved

Tvscript

- **Non-Interactive Batch Debugging –**
  - Work in the "main" batch queue
  - Don't have to baby-sit job waiting on it to run
  - Use scripting to perform checks that would be tedious to do by hand
  - Verification through automated processes (nightly build and test)

| Copyright © 2010 Rogue Wave Software | All Rights Reserved

ROGUE WAVE SOFTWARE

Non-Interactive Batch Debugging with TVscript

- Run multiple debugging sessions without the need for recompiling, unlike with printf
- TVscript syntax:

  tvscript [options] [filename] [-a program_args]
- More complex actions-to-events are possible, utilizing TCL within a CLI file
- TVscript lets you define what events to act on, and what actions to take

TVscript uses a simple, Event/Action interface	
Typical Events • Action_point • Any_memory_event • Guard_corruption error	**Typical Actions** • Display_backtrace [-level *level-num*] • List_leaks • Save_memory • Print [-slice {*slice_exp*] {*variable* | *exp*}

| Copyright © 2010 Rogue Wave Software | All Rights Reserved

ROGUE WAVE SOFTWARE

Unattended Debugging with Tvscript

Example

The following tells tvscript to report the contents of the *foreign_addr* structure each time the program gets to line 85

-create_actionpoint "#85=>print foreign_addr”

Typical output sample with tvscript:

```
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
! Print
!
! Process:
!   ./server (Debugger Process ID:  1, System ID:  12110)
! Thread:
!   Debugger ID:  1.1, System ID:  3083946656
! Time Stamp:
!   06-26-2008 14:04:09
! Triggered from event:
!   actionpoint
! Results:
!    foreign_addr = {
!      sin_family = 0x0002 (2)
!      sin_port = 0x1fb6 (8118)
!      sin_addr = {
!        s_addr = 0x6658a8c0 (1717086400)
!      }
!      sin_zero = ""
!    }
!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
```

Minimize

Reduce the Scope of Effort

| Copyright © 2013 Rogue Wave Software | All Rights Reserved

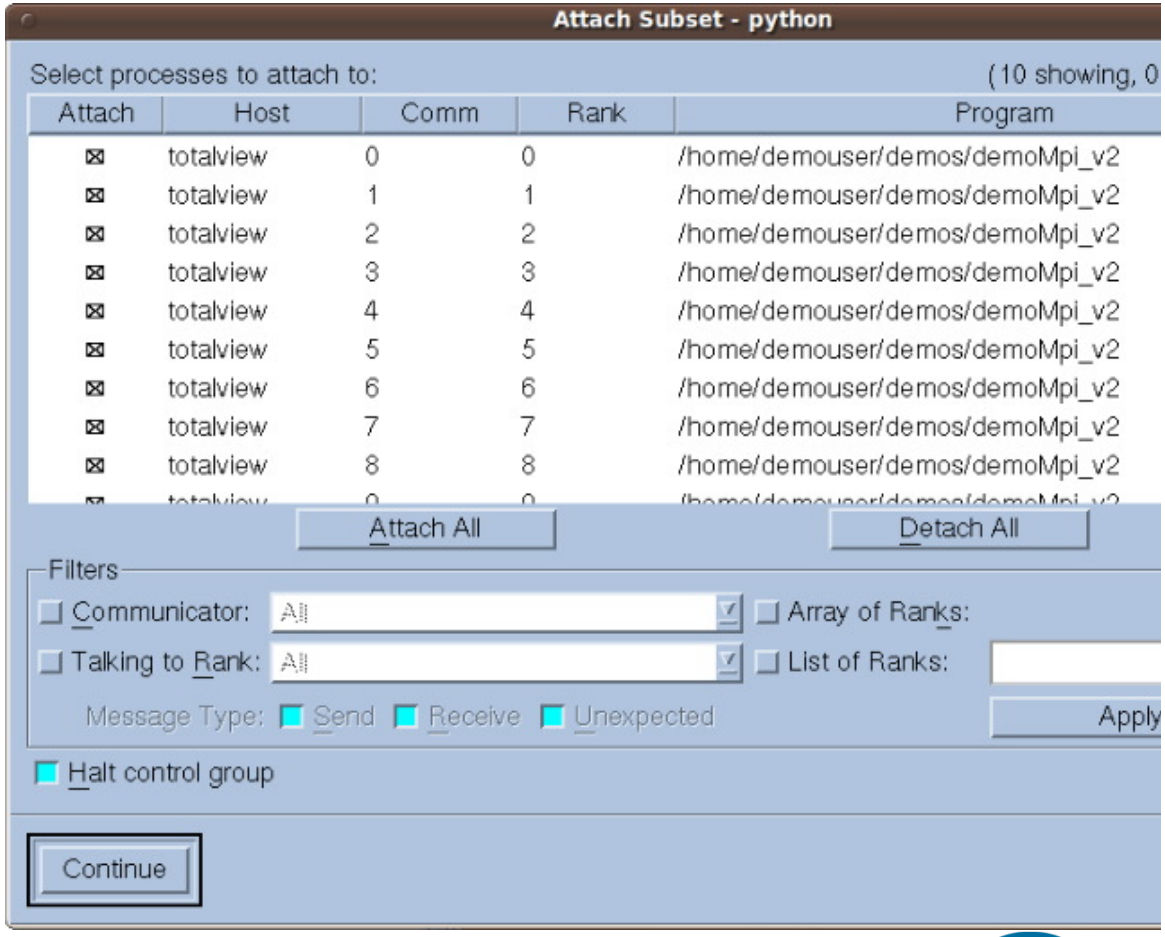
Subset Debugging
With TotalView

 Copyright © 2010 Rogue Wave Software | All Rights Reserved

Subset Attach

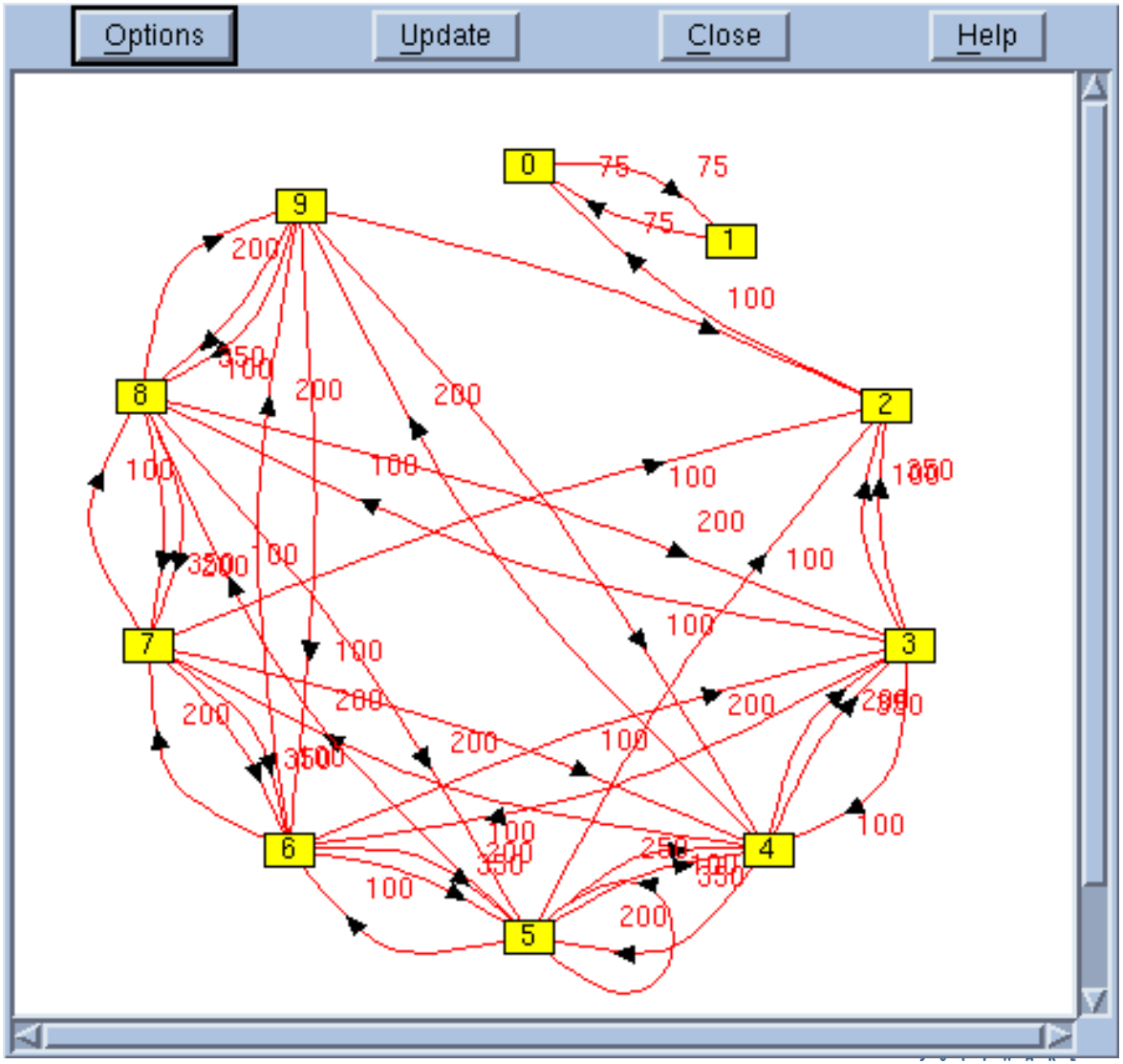
You need not be attached to the entire job

- You can be attached to different subsets at different times through the run
- You can attach to a subset, run till you see trouble and then 'fan out' to look at more processes if necessary.
- This greatly reduces overhead
- It also reduces license size requirements



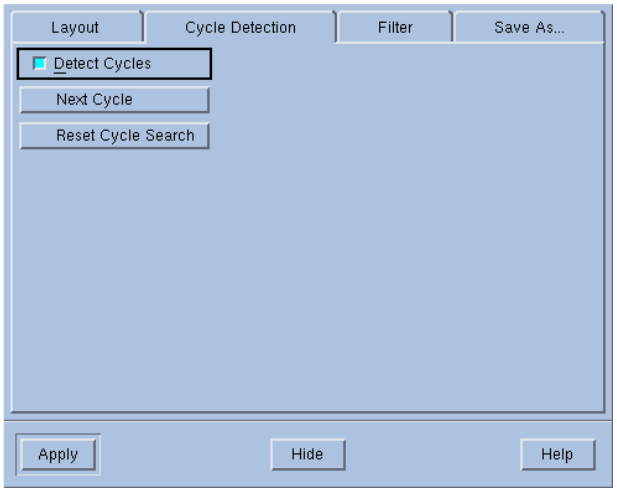
Message Queue Graph

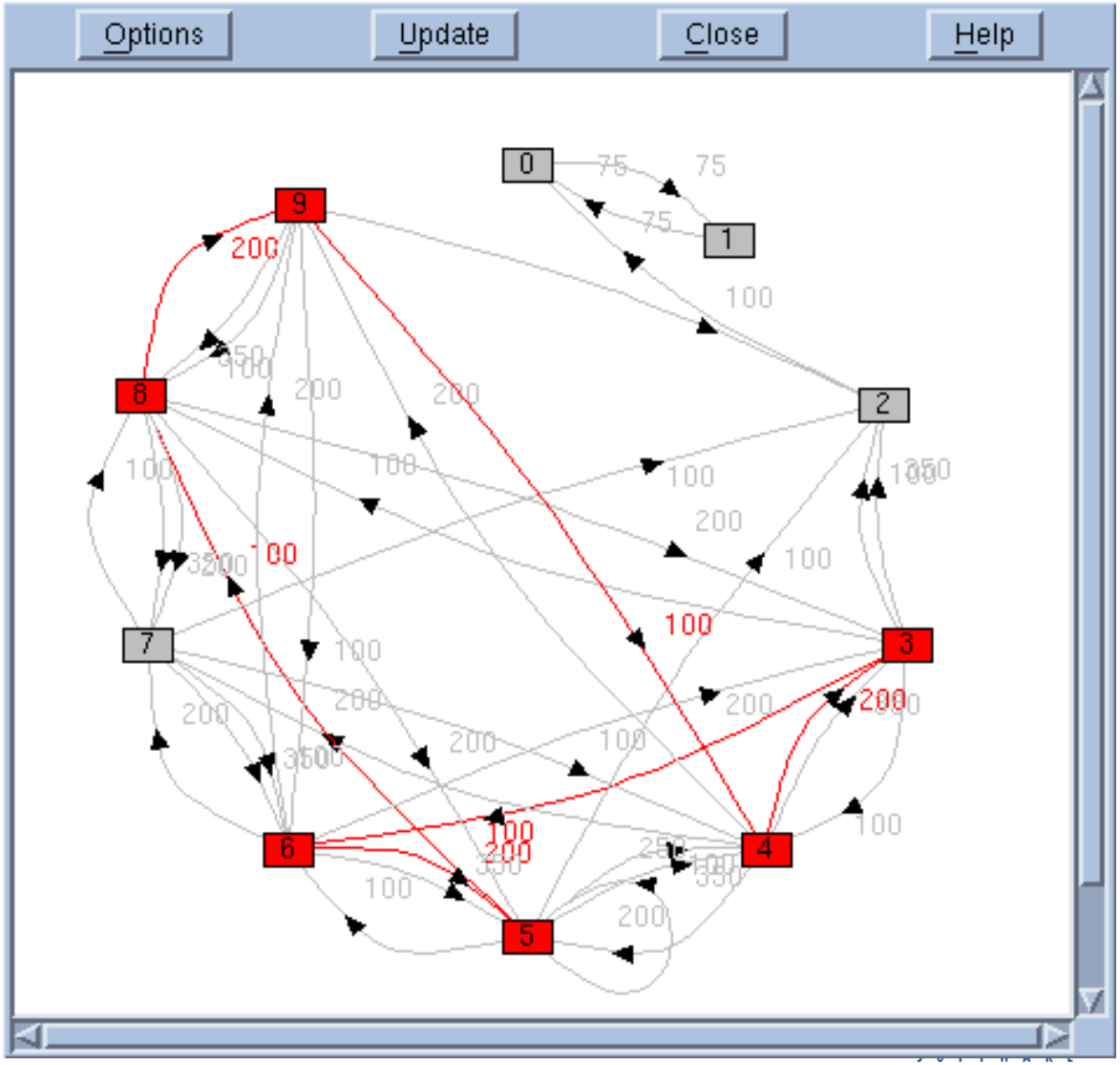
- Hangs & Deadlocks
- Pending Messages
  - Receives
  - Sends
  - Unexpected
- Inspect
  - Individual entries
- Patterns



Message Queue Debugging

- Filtering
  - Tags
  - MPI Communicators
- Cycle detection
  - Find deadlocks



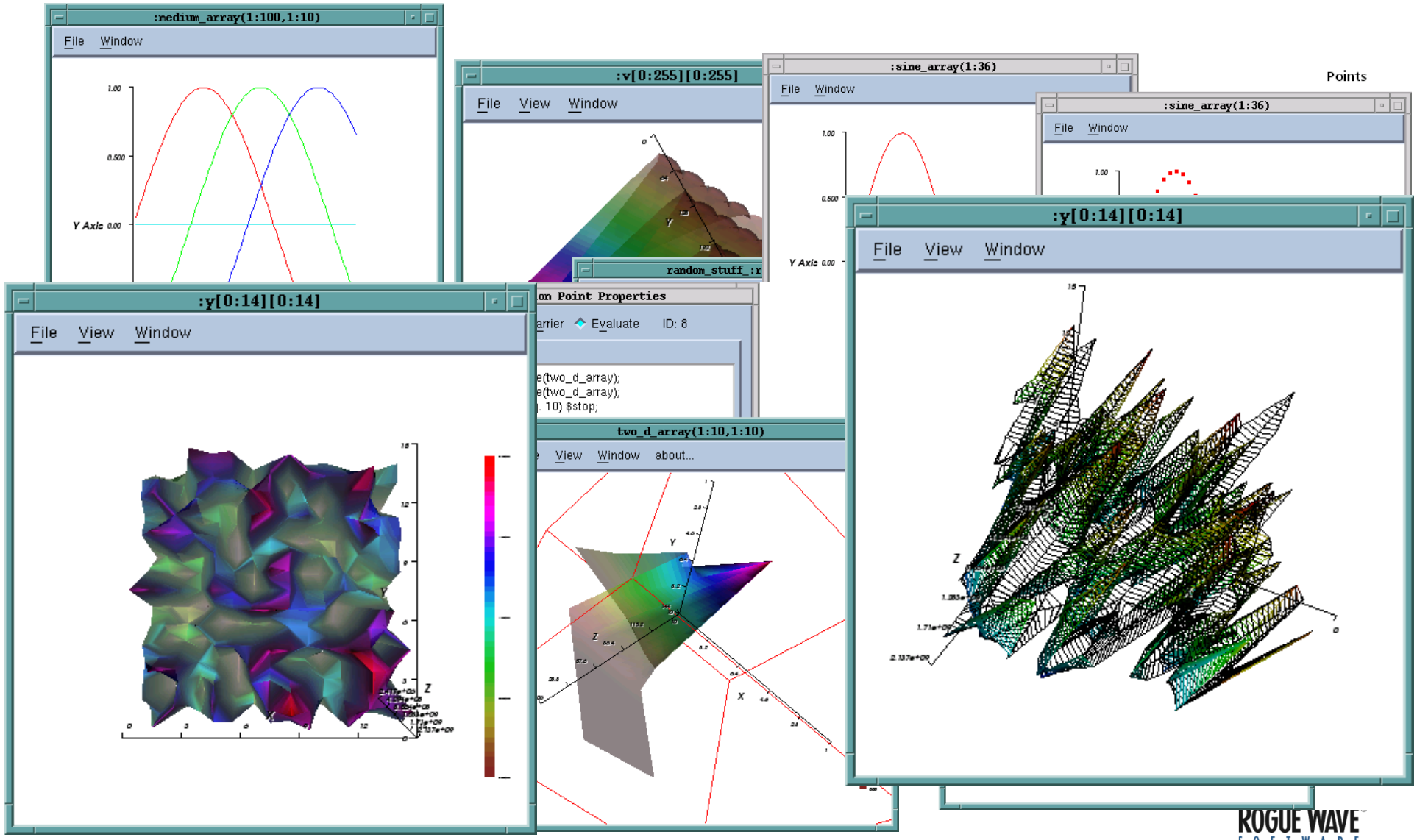


28

Visualize

| Copyright © 2013 Rogue Wave Software | All Rights Reserved

Visualization



| Copyright © 2013 Rogue Wave Software | All Rights Reserved

Visualization

Get the big picture – Observe anomalies – Utilize Pattern recognition – Save time!

/home/demouser/demos/combined

File Edit View Group Process Thread Action Point Debug Tools Window Help

Group (Control) Go Halt Kill Restart Next Step Out Run To GoBack Prev UnStep Caller BackTo Live

Process 1 (3660): combined (At Breakpoint 10)

Thread 1 (3075738400) (At Breakpoint 10)

Stack Trace

```
C++  arrays,              FP=bf9dd618
C++  main,                FP=bf9dd658
     __libc_start_main,   FP=bf9dd6e8
```

Stack Frame

```
Function "arrays":
  No parameters.
Local variables:
  shape:          (class Shape)
  circle:         (class Circle)
  cylinder:       (class Cylinder)
  cylly:          (class Cylinder)
  circe:          0x09c67070 -> (class Circle)
  surf_vol:       (double[*][*])
  vol:            (double[*][*])
```

Function arrays in combined.cxx

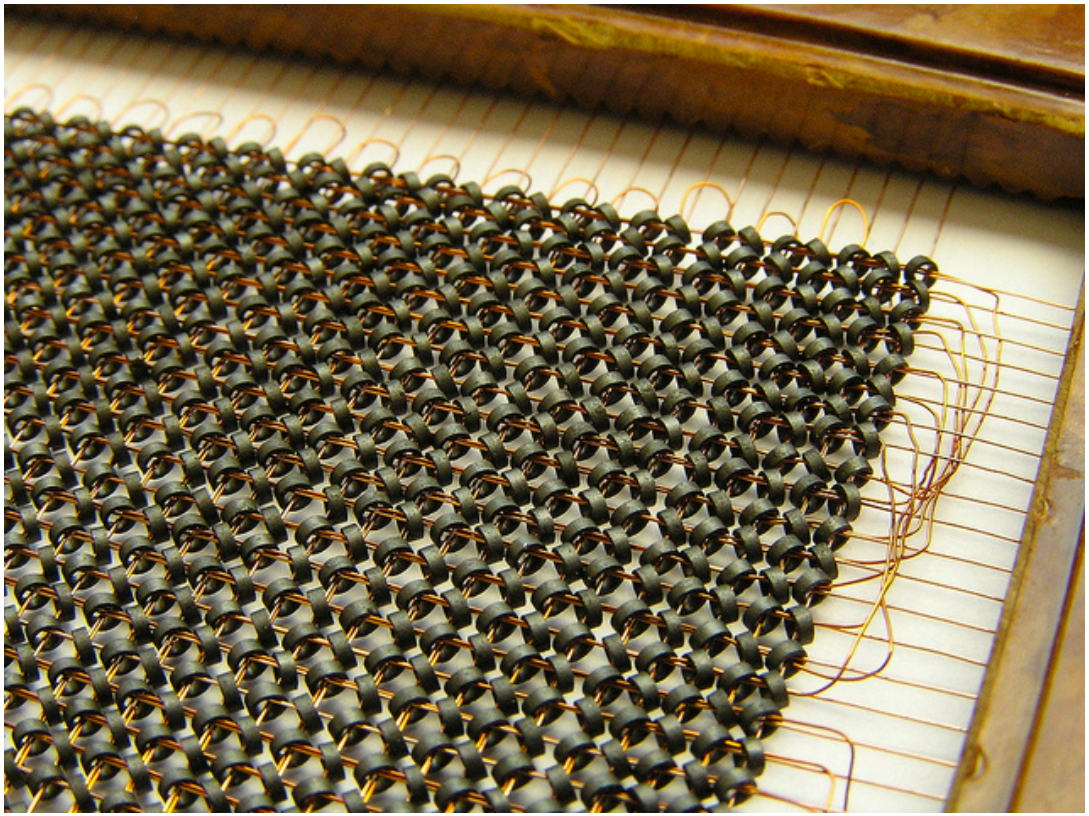
```
505    circe = new Circle ("circus", 42);
506    cout << "circe = " << (void *)circe << "\n";
507    //  cout << "*circe = " << (void)  *circe << "\n";
508    cout << "&circe = " << (void *) &circe <<  "\n";
509
510    for( int i = 0; i < ARRAYSIZE; i++ ) {
511      for (int j = 0; j < ARRAYSIZE; j++) {
512            cylinder.resize(start + i*step, start + j*step);
513            surf_vol[i][j] = cylinder.volume()/cylinder.area();
STOP           vol[i][j] = cylinder.volume();
515          }
516    }
517
518    // you can make the above array more interesting by
```

Action Points | Processes | Threads P- P+ T- T+

```
1.1        (3075738400) B10     in arrays
```

ROGUE WAVE SOFTWARE

… And Don’t Forget the Memory!



| Copyright © 2013 Rogue Wave Software | All Rights Reserved

MemoryScape

Memory bugs often go undetected until the worst possible time

- Symptoms often surface long after the actual damage is done
- Some only surface after hours or even days of operation
- In many cases, the programs affected are "innocent bystanders"

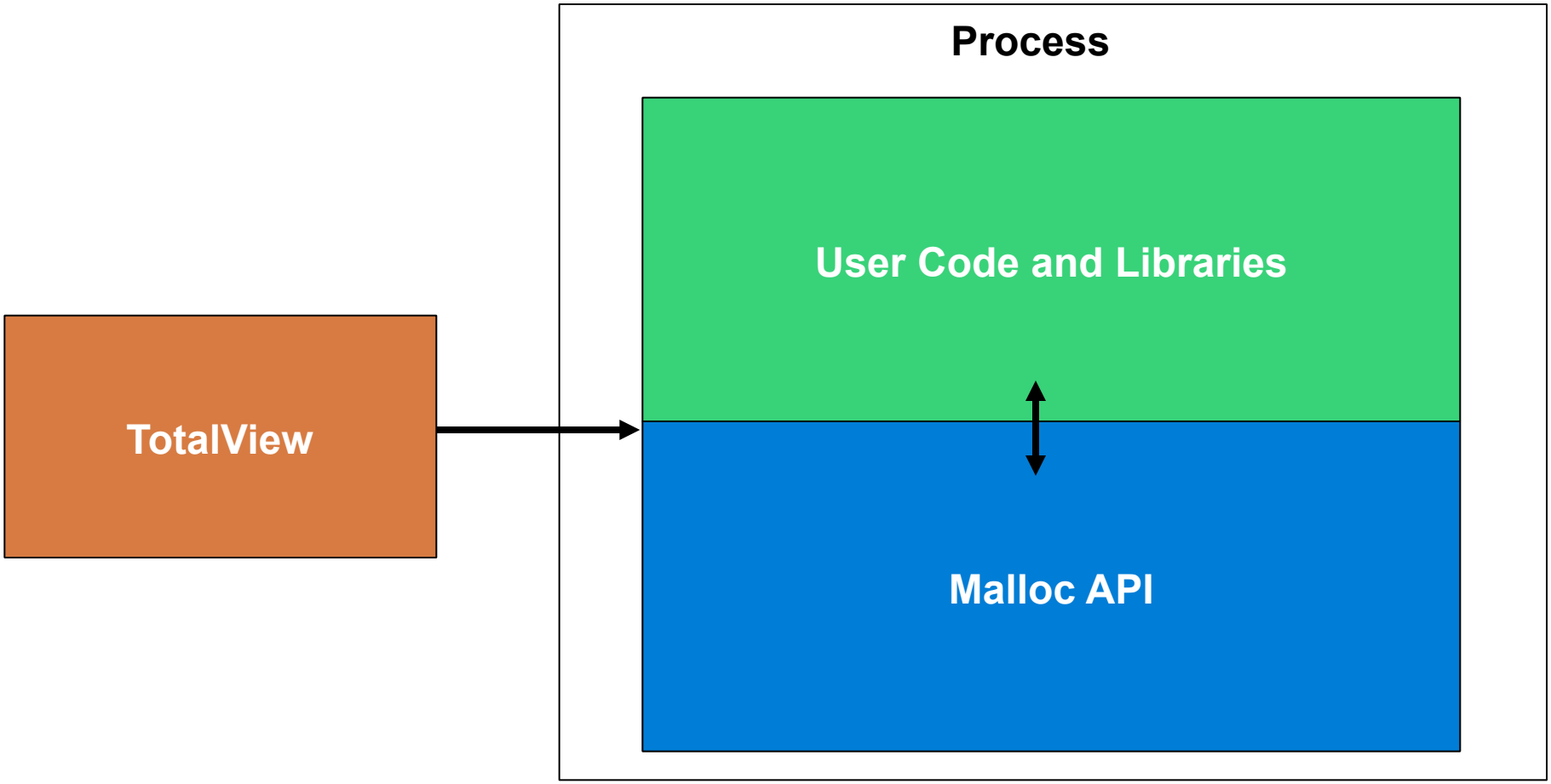
MemoryScape: Fully Integrated in TotalView

- Programs run nearly full speed
- Low performance overhead
- Low memory overhead • Efficient memory usage

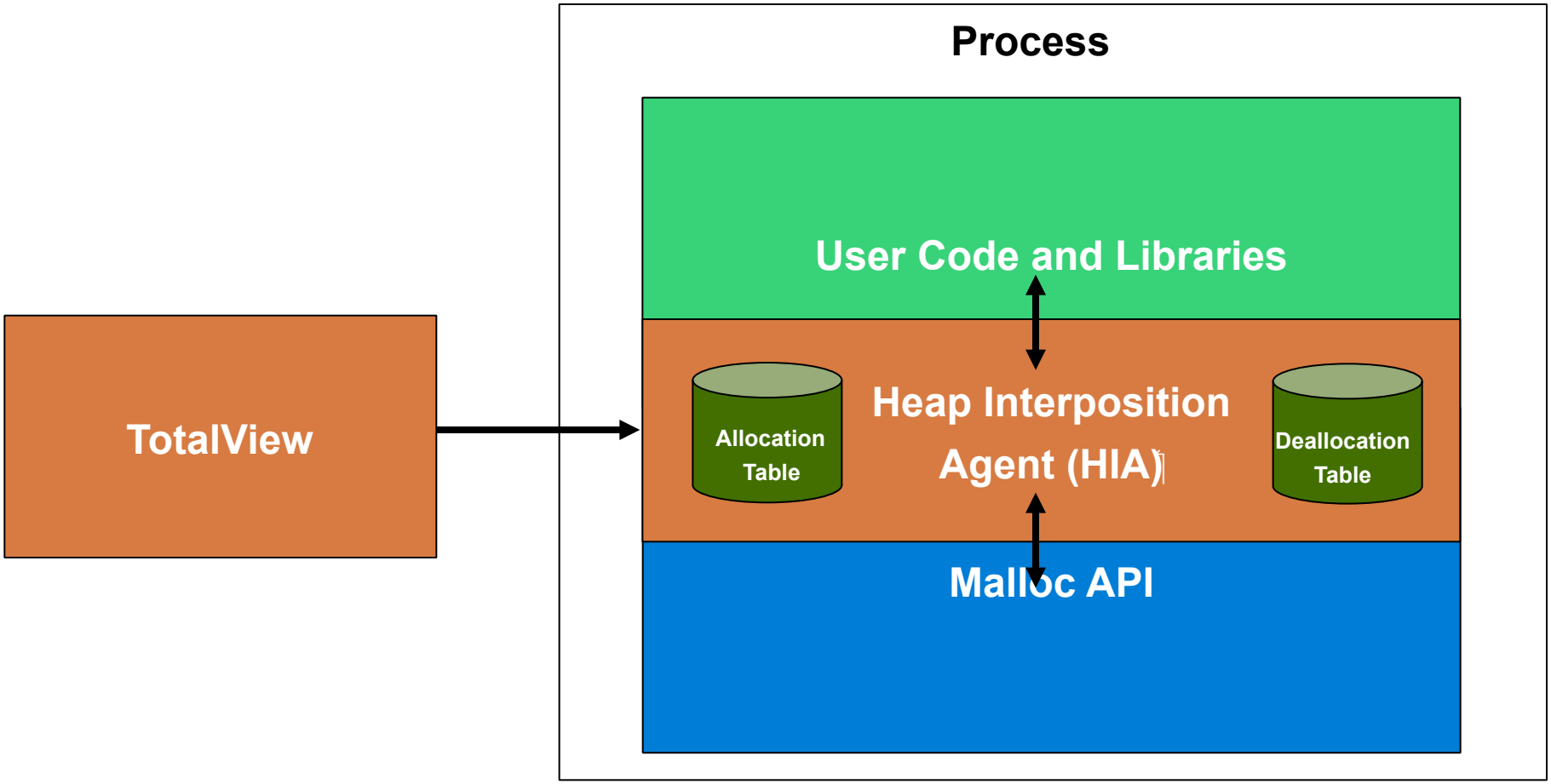
| Copyright © 2013 Rogue Wave Software | All Rights Reserved

ROGUE WAVE SOFTWARE

The Agent and Interposition



The Agent and Interposition



Linking Your Application with the HIA Agent

Platform	Compiler	Binary Interface	Additional linker options
Cray XT, XE, XK CLE	-		`-L<path> -ltvheap_cnl_static`
Cray XT3 Catamount	-	64	`-L<path> -lgmalloc -ltvheap_xt3`
IBM Blue Gene/L (static)	-	32	`-L<path> -ltvheap_bluegene`
IBM Blue Gene/P (dynamic)	-	32	`-L<path> -ltvheap -Wl,-rpath,<path>`
IBM Blue Gene/P (static)	-	32	`-L<path> -ltvheap_bluegene_p`
IBM Blue Gene/Q (dynamic)	IBM/GCC	64	`-L<path> -ltvheap_64 -Wl,-rpath,<path>`
IBM Blue Gene/Q (static)	IBM/GCC	64	`-L<path> -Wl,@<path>/tvheap_bgqs.ld`

As with Blue Gene/P, if your program is dynamically linked you can avoid having to link the heap agent with your program if you set theLD_PRELOAD variable in your program's environment. For full details, see related documentation at:

http://www.roguewave.com/support/product-documentation/totalview.aspx

| Copyright © 2013 Rogue Wave Software | All Rights Reserved

MemoryScape

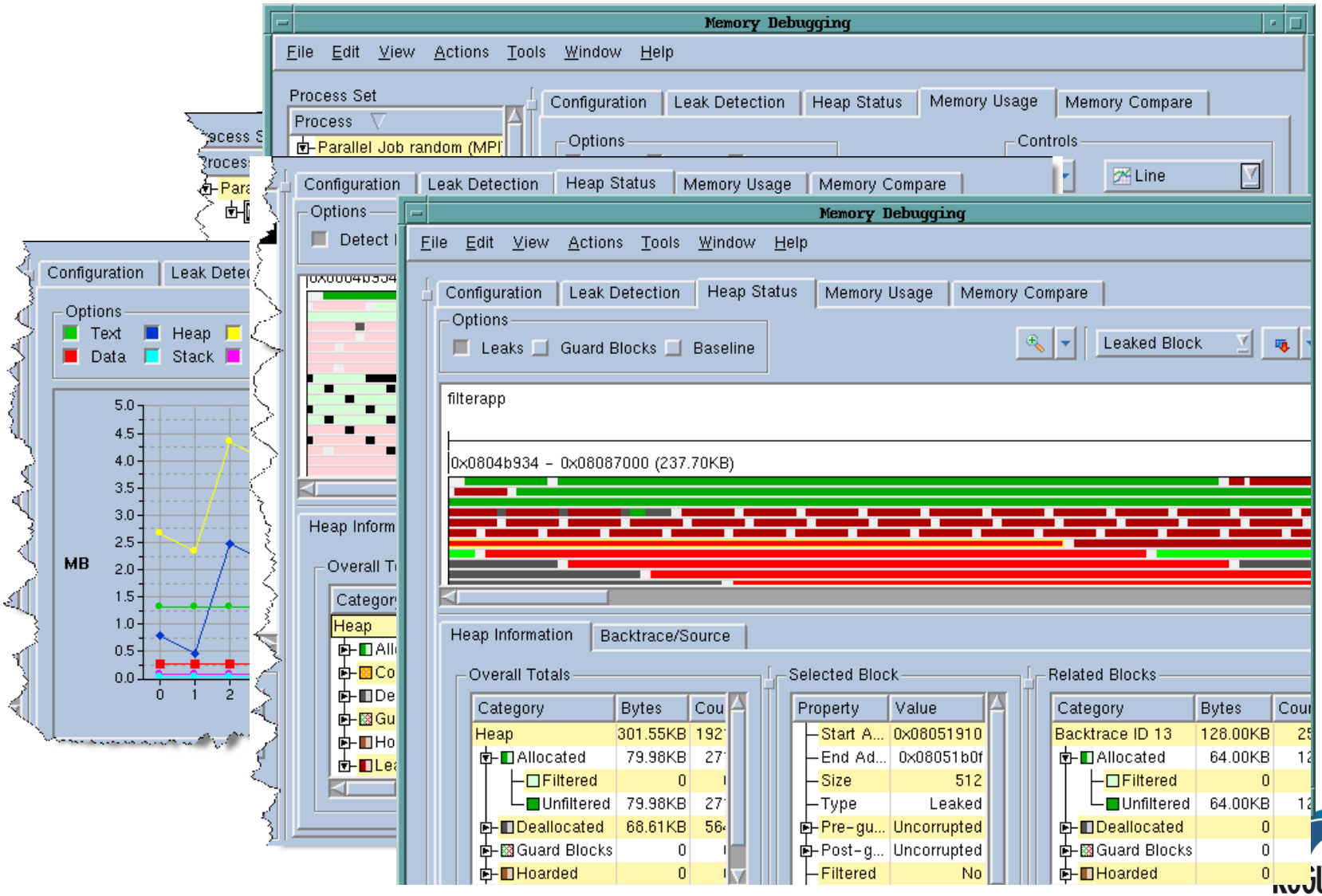
MemoryScape Feature Highlights

- Automatic allocation problem detection
- Heap Graphical View
- Leak detection
- Block painting
- Dangling pointer detection
- Deallocation/reallocation notification
- Memory Corruption Detection - Guard Blocks
- Memory Hoarding
- Memory Comparisons between processes
- Collaboration features

| Copyright © 2013 Rogue Wave Software | All Rights Reserved

ROGUE WAVE SOFTWARE

MemoryScape



| Copyright © 2013 Rogue Wave Software | All Rights Reserved

Memory Event Details Window

Memory Event Details - Process 1 (5406): filterapp-mpi - 1

Process 1 (5406): filterapp-mpi - 1 Time: 06:48:41

Event: Guard corruption error - Bounds error: The guard area around a block has been overwritten

Event Location | Allocation Location | Deallocation Location | Block Details

Block Information

0x0959ef20 — 64 bytes — 0x0959ef5f

Status: Allocated

Flags: Operation in Progress

Hexadecimal Count: 80

Bytes 1 2 4 8

0x0959ef18	0x77	0x77	0x77	0x77	0x77	0x77	0x77	0x77	0x10	0x00
0x0959ef22	0x00	0x00	0x0f	0x00	0x00	0x00	0x0e	0x00	0x00	0x00
0x0959ef2c	0x0d	0x00	0x00	0x00	0x0c	0x00	0x00	0x00	0x0b	0x00
0x0959ef36	0x00	0x00	0x0a	0x00	0x00	0x00	0x09	0x00	0x00	0x00
0x0959ef40	0x08	0x00	0x00	0x00	0x07	0x00	0x00	0x00	0x06	0x00
0x0959ef4a	0x00	0x00	0x05	0x00	0x00	0x00	0x04	0x00	0x00	0x00
0x0959ef54	0x03	0x00	0x00	0x00	0x02	0x00	0x00	0x00	0x01	0x00
0x0959ef5e	0x00	0x00	0x00	0x00	0x00	0x00	0x99	0x99	0x99	0x99

Generate

Memory File

Close | View in Block Properties window | Help

Memory Corruption Report

Corrupted Memory Report

Options

Enable Filtering

	Preceding Block	Corrupted Block	Following Block
1	0x0959c248 64 bytes 0x0959c287	0x0959c2a0 64 bytes 0x0959c2df	0x0959c338 37 bytes 0x0959c35c
2	0x0959eaa8 64 bytes 0x0959eae7	0x0959eb00 64 bytes 0x0959eb3f	0x0959eb58 64 bytes 0x0959eb97
3	0x0959ebb0 64 bytes 0x0959ebef	0x0959ec08 64 bytes 0x0959ec47	0x0959ec60 64 bytes 0x0959ec9f
4	0x0959ecb8 64 bytes 0x0959ecf7	0x0959ed10 64 bytes 0x0959ed4f	0x0959ed68 64 bytes 0x0959eda7
5	0x0959edc0 64 bytes 0x0959edff	0x0959ee18 64 bytes 0x0959ee57	0x0959ee70 64 bytes 0x0959eeaf

Backtrace/Source | Memory Content

Backtrace

Process	Function	Line #	Source Information
59			
	malloc	166	malloc_wrappers_dlope
	corrupt_data	**108**	**main.cxx**
	main	295	main.cxx
	__libc_start_main		libc.so.6
	start		filterapp-mpi

Source

/home/demouser/src/main.cxx

```
108   p1 = (int *) malloc( size * sizeof(
109   p2 = (int *) malloc( size * sizeof(

111 // Common corruption cases.  Oop
112   for( i=0; i<=size; i++ )
```


ROGUE WAVE SOFTWARE

Block Summary Data

MemoryScape 2T.0.0-1

File Tools Window Help

Home | Memory Reports | Manage Processes | Memory Debugging Options | Tips

Summary | Leak Detection | Heap Status | Memory Usage | Corrupted Memory | Memory Comparisons

April 12, 2007

Save Data
Export Memory Data...

Heap Status Reports
Source Report
Backtrace Report

Other Reports Categories
Leak Detection Reports
Memory Usage Reports
Corrupted Memory Report
Compare Memory Usage

Other Tasks
Manage Filters

Process Selection

Process
filterapp (3496)
hia_events (3452)
random (0)
random.0 (File: random
random.10 (File: rando

Heap Status Graphical Report

Options
Detect Leaks | Enable Filtering

Leaked Block

Process 37 (3496): filterapp

0x0a040010 - 0x0a0c4f78 (531.85KB)

Memory block:

Type	Leaked
Filtered	No
Size	41
Start Address	0x0a0410b8
End Address	0x0a0410e0
Backtrace ID	8

Point of allocation:

File	myClassB.cxx
Method	myClassB::myClassB
Line	8

Guard Blocks:

Pre-guard

size	8 bytes
pattern	0x77777777

Post-guard

size	8 bytes
pattern	0x99999999

Heap Information | Bac

Overall Totals

Category
Heap
Allocated
Corrupted Guard
Deallocated
Guard Blocks
Hoarded

Related Blocks

Category
Backtrace ID 8
Allocated
Corrupted Guard E
Deallocated
Guard Blocks
Hoarded

Cursor rollover provides rapid access to block summary data

What's New

- <u>Increased Scalability</u>
  - Leveraging TotalView's Architecture
  - Efficient Use of Cluster Resources
    - Extremely light weight debug agents; Minimal memory footprint
    - More space on the compute nodes for user application code
  - Tree-Based Overlay Network - MRNET
    - Broadcast of Operations; Aggregation of Events and Data
- Replay Enhancements
  - Record on Demand
- OpenACC Support
- Intel PHI (MIC) Support

| Copyright © 2013 Rogue Wave Software | All Rights Reserved

Developing for Parallel Architectures

TotalView®

- Code debugging
  - Highly scalable interactive GUI debugger
    - Easy to use -- without sacrificing detail that users need to debug
    - Used from workstations to the largest supercomputers
  - Powerful features for debugging multi-threaded, multi-process, and MPI parallel programs
  - Compatible with wide variety of compilers across several platforms and operating systems
- Memory Debugging
  - Parallel memory analysis and error detection
  - Easily integrated into the validation process
- Reverse Debugging
  - Parallel record and deterministic replay within TotalView
  - Run programs “backwards” to find bugs
  - Now with Record On-Demand
- GPU CUDA Debugging
  - Full Hybrid Architecture Support
  - Asynchronous Warp Control
  - Multi-Device and MPI Support
- Intel PHI (MIC) Support

ROGUE WAVE® SOFTWARE

 Copyright © 2013 Rogue Wave Software | All Rights Reserved

http://www.roguewave.com/support/product-documentation/totalview.aspx

Logout | ehinkel | Download Evaluation | Contact Us

Search

HOME | PRODUCTS | SERVICES | RESOURCES | SUPPORT | COMPANY

Home > Support > Product Documentation > TotalView

TotalView Documentation

TotalView 8.11	
Online Help and Documentation 8.11	HTML
TotalView New Features and Change Log	PDF | 226.1 KB
TotalView Installation Guide	PDF | 226.3 KB
TotalView Release Notes	PDF | 359.3 KB
TotalView Getting Started Guide	PDF | 764.1 KB
TotalView Quick View Guide	PDF | 818.6 KB
TotalView Evaluation Guide	PDF | 82.2 KB
TotalView CLI Command Summary Guide	PDF | 73.5 KB
TotalView User Guide	PDF | 6.4 MB
TotalView Reference Guide	PDF | 2.1 MB
TotalView Platforms and System Requirements	PDF | 72.2 KB

MemoryScape 3.3	
MemoryScape New Features and Change Log	PDF | 226.1 KB
MemoryScape Installation Guide	PDF | 223.5 KB
MemoryScape User Guide	PDF | 2.2 MB
MemoryScape Evaluation Guide	PDF | 52.6 KB

SUPPORT

- › Programs & Policies
- › Contact Support
- › File a Support Request
- › Latest Versions List
- › Request a Download/Upgrade
- › User Forums
- › Knowledge Base
- › Product Documentation

Thanks!

Developing parallel, data-intensive applications is hard.

We make it easier.

www.roguewave.com