Debugging Multi-GPU applications on CUDA Accelerated Clusters with TotalView

Chris Gottbrath, Principal Product Manager
May 16\textsuperscript{th}, 2012
Agenda

- Who are Rogue Wave?
- What is TotalView?
- Getting Started With TotalView
- TotalView for CUDA
- Latest Updates and Futures
Rogue Wave Today

- **Customers**
  - 3,000+ in 36 countries
  - Financial services, telecoms, oil and gas, government and aerospace, research and academic

- **History**
  - Founded: 1989
  - Acquired by Battery Ventures: 2007
  - Acquired:
    - Visual Numerics: 2009
    - TotalView Technologies: 2010
    - Acumem: 2010
    - ILOG Visualization Tech 2012

- **Pioneers in C++/object-oriented development**
- **Leading the way in cross-platform, parallel development**
Global Reach

40 years -- 3000+ customers -- 36 countries
Representative Customers

BARCLAYS
CAPITAL

Morgan Stanley

Deutsche Bank

UBS

Bank of America

JPMorganChase

citi

Motorola

Pixar

Disney

at&t

CenturyLink

Nokia

BT

Huawei

RiskMetrics Group

Deutsche Telekom

T

NASA

Raytheon

Lockheed Martin

ExxonMobil

Lawrence Livermore National Laboratory

Siemens

Mizuho

Cisco

Sybase

An SAP Company

Availity

Oracle

IBM

Openwave

vodafone

Los Alamos

Bizerba

ChevronTexaco

SAP
Rogue Wave Solution Portfolio

Enterprise C++ Development

Developing for Parallel Architectures

Parallel, Data-intensive Applications

Leveraging High Performance Data Analytics

Visualization

SourcePro C++

MemoryScape

ThreadSpotter

TotalView

IMSL

PV-WAVE
<table>
<thead>
<tr>
<th>Mathematics</th>
<th>Statistics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Basic Types</td>
<td>Basic Statistics</td>
</tr>
<tr>
<td>Linear Algebra</td>
<td>Time Series &amp; Forecasting</td>
</tr>
<tr>
<td>Eigensystems</td>
<td>Nonparametric Tests</td>
</tr>
<tr>
<td>Interpolation &amp; Approximation</td>
<td>Correlation &amp; Covariance</td>
</tr>
<tr>
<td>Quadrature</td>
<td>Data Mining</td>
</tr>
<tr>
<td>Differential Equations</td>
<td>Regression</td>
</tr>
<tr>
<td>Nonlinear Equations</td>
<td>Analysis of Variance</td>
</tr>
<tr>
<td>Optimization</td>
<td>Transforms</td>
</tr>
<tr>
<td>Special Functions</td>
<td>Goodness of Fit</td>
</tr>
<tr>
<td>Finance &amp; Bond Calculations</td>
<td>Distribution Functions</td>
</tr>
<tr>
<td>Genetic Algorithm</td>
<td>Random Number Generation</td>
</tr>
<tr>
<td></td>
<td>Neural Networks</td>
</tr>
</tbody>
</table>
IMSL C and Fortran libraries both support CUDA

- BLAS Linear Algebra routines
- Uses CUDA where appropriate
  - Just a link time change .. No code changes whatsoever!
- Contact me for more details.
IMSL Fortran GPU Benchmarks

![Chart showing SGEMM Speedup using NVIDIA CUBLAS]

- IMSL BLAS
- High Perf BLAS 1 core
- High Perf BLAS 4 core
Agenda

• Who are Rogue Wave?
• What is TotalView?
• Getting Started With TotalView
• TotalView for CUDA
• Latest Updates and Futures
What is TotalView?

Application Analysis and Debugging Tool: Code Confidently

- Debug and Analyze C/C++ and Fortran on Linux, Unix or Mac OS X
- Debugs host and accelerator together
- Laptops to supercomputers (BG, Cray)
- 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 and OpenACC (early access)
- Includes a Remote Display Client freeing 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
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
  - 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
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
TotalView for CUDA

- Characteristics
  - Full visibility of both Linux threads and GPU device threads
  - Fully represent the hierarchical memory
  - Supports Unified Virtual Addressing and GPUDirect
  - Thread and Block Coordinates
  - Device thread control
  - Handles CUDA function inlining and CUDA stacks
  - Support for C++ and inline PTX
  - Reports memory access errors
  - Handles CUDA exceptions
  - Multi-Device Support
  - Can be used with MPI

- Supports CUDA 3.2, 4.0 and 4.1
What is ThreadSpotter?

- **Runtime Cache Performance Optimization Tool: Tune into the Multi-Core Era**
  - Realize More of the Performance Offered by Multi/Many-Core Chips
  - Quickly Detects and Prioritizes Issues -- and then Provides Usable Advice!
    - Brings Cache Performance Into Reach for Every Developer
    - Makes Experienced Cache Optimizers Hyper-Efficient

- **Features**
  - Supports Linux x86/x86-64 and Windows
  - Any compiled code
  - Runtime Analysis
    - Low overhead
  - Cache Modeling
    - Prioritizes Issues
    - Identifies Problem Lines of Code
  - Provides Advice
    - Explanations
    - Examples
    - Detailed statistics (if desired)
Agenda

- Who are Rogue Wave?
- What is TotalView?
- Getting Started With TotalView
- TotalView for CUDA
- Latest Updates and Futures
Starting TotalView

Start New Process

Program:
On host: (local)

- Enable ReplayEngine
  Record all program state while running. Roll back your program to any point in the past.

- Enable memory debugging
  Track dynamic memory allocations. Catch common errors, leaks, and show reports.

- Enable CUDA memory checking
  Detect global memory addressing violations and misaligned global memory accesses.

OK Cancel Help
Starting TotalView

Start New Process – Arguments tab
TotalView Startup with MPI TVT Launch

In the Parallel tab, select:

- your MPI preference,
- number of tasks,
- and number of nodes.

... then add any additional starter arguments.
Architecture for Cluster Debugging

- Single Front End (TotalView)
  - GUI
  - debug engine
- Debugger Agents (tvdsvr)
  - Low overhead, 1 per node
  - Traces multiple rank processes
- TotalView communicates directly with tvdsvrs
  - Not using MPI
  - Protocol optimization

Provides Robust, Scalable and efficient operation with Minimal Program Impact
**TotalView Root Window**

- **Hierarchical/Linear Toggle**
- **Host name**
- **Rank # (if MPI program)**
- **TotalView Thread ID #**
- **Expand - Collapse Toggle**
- **Process Status**
- **Action Point ID number**

<table>
<thead>
<tr>
<th>ID</th>
<th>Rank</th>
<th>Host</th>
<th>Process</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>&lt;local&gt;</td>
<td>mismatchLinux0</td>
<td>(1 active threads)</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>&lt;local&gt;</td>
<td>/home/barry/tests/fork_loopLinux</td>
<td>(5 active threads)</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>intrepid этнус.T</td>
<td>/home/barry/tests/fork_loopLinux</td>
<td>(6 active threads)</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>intrepid этнус.T</td>
<td>/home/barry/tests/fork_loopLinux</td>
<td>(1 active thread)</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>intrepid этнус.T</td>
<td>/home/barry/tests/fork_loopLinux</td>
<td>(2 active threads)</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>intrepid этнус.T</td>
<td>/home/barry/tests/fork_loopLinux</td>
<td>(1 active thread)</td>
</tr>
<tr>
<td>9</td>
<td>1</td>
<td>intrepid этнус.T</td>
<td>/home/barry/tests/fork_loopLinux</td>
<td>(1 active thread)</td>
</tr>
<tr>
<td>10</td>
<td>1</td>
<td>intrepid этнус.T</td>
<td>/home/barry/tests/fork_loopLinux</td>
<td>(3 active threads)</td>
</tr>
<tr>
<td>11</td>
<td>1</td>
<td>intrepid этнус.T</td>
<td>/home/barry/tests/fork_loopLinux</td>
<td>(1 active thread)</td>
</tr>
<tr>
<td>12</td>
<td>1</td>
<td>intrepid этнус.T</td>
<td>/home/barry/tests/fork_loopLinux</td>
<td>(1 active thread)</td>
</tr>
<tr>
<td>13</td>
<td>1</td>
<td>&lt;local&gt;</td>
<td>mismatchLinux</td>
<td>(1 active thread)</td>
</tr>
<tr>
<td>14</td>
<td>2</td>
<td>&lt;local&gt;</td>
<td>mismatchLinux</td>
<td>(1 active thread)</td>
</tr>
<tr>
<td>15</td>
<td>3</td>
<td>&lt;local&gt;</td>
<td>mismatchLinux</td>
<td>(1 active thread)</td>
</tr>
</tbody>
</table>

- **Dive to refocus**
- **Dive in new window** to get a second process window

(Rogue Wave Software)
Process Window Overview

Provides detailed state of one process, or a single thread within a process.

A single point of control for the process and other related processes.
### Tabbed Pane

The image shows a tabbed pane with three tabs:

- **Action Points Tab**: all currently defined action points
- **Processes Tab**: all current processes
- **Threads Tab**: all current threads, ID’s, Status

#### Action Points Tab

<table>
<thead>
<tr>
<th>Action Points</th>
<th>Processes</th>
<th>Threads</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>(3084118688)</td>
<td>R</td>
</tr>
<tr>
<td>1.2</td>
<td>(3084114864)</td>
<td>B6</td>
</tr>
<tr>
<td>1.3</td>
<td>(3075722160)</td>
<td>B6</td>
</tr>
<tr>
<td>1.4</td>
<td>(3067329456)</td>
<td>B6</td>
</tr>
<tr>
<td>1.5</td>
<td>(3058936752)</td>
<td>B6</td>
</tr>
</tbody>
</table>
Stepping Commands

Based on PC location
Process Control Concepts

- Each process window is always focused on a specific process.
- Process focus can be easily switched
  - $P^+/P^-$, Dive in Root window and Process tab
- Processes can be ‘held’ - they will not run till unheld.
  - Process > Hold
- Breakpoints can be set to stop the process or the group
- Breakpoint and command scope can be simply controlled
Basic Process Control

Groups

• Control Group
  – All the processes created or attached together

• Share Group
  – All the processes that share the same image

• Workers Group
  – All the threads that are not recognized as manager or service threads

• Lockstep Group
  – All threads at the same PC

• Process, Process (Workers), Process (Lockstep)
  – All process members as above

• User Defined Group
  – Process group defined in Custom Groups dialog
Diving on Variables

Example: Dive on Variable “j” from Stack Frame or SourcePanes

```cpp
for(int i = 0; i < ARRAYSIZE; i++) {
    for(int j = 0; j < ARRAYSIZE; j++) {
        cylinder.resize(start + i*step, start + j*step);
        surf_vol[k][j] = cylinder.volume() / cylinder.area();
        vol[k][j] = cylinder.volume();
    }
}
```

expression: j
address: 0xbf9
value: 0x00000006 (6)
Viewing Arrays

Data Arrays

Structure Arrays
Slicing Arrays

Slice notation is \([\text{start} : \text{end} : \text{stride}]\)
Filtering Arrays
Multi-Dimensional Array Viewer

- See your arrays on a “Grid” display
- 2-D, 3-D... N-D
- Arbitrary slices
- Specify data representation
- Windowed data access

Fast
Visualizing Arrays

- Visualize array data using Tools > Visualize from the Variable Window
- Large arrays can be sliced down to a reasonable size first
- Visualize is a standalone program
- Data can be piped out to other visualization tools
- Visualize allows to spin, zoom, etc.
- Data is not updated with Variable Window; You must revisualize
- `$visualize()` is a directive in the expression system, and can be used in evaluation point expressions
Agenda

• Who are Rogue Wave?
• What is TotalView?
• Getting Started With TotalView
• TotalView for CUDA
• Latest Updates and Futures
GPU architecture

Used in conjunction with conventional CPUs
  Acts as an accelerator to a host process
  Or, perhaps the host processor acts to support the GPU

Distinct architecture
  Distinct processor architecture from the CPU

Many more cores than an SMP
  Multiple streaming multiprocessors
  Potentially 10k+ thread contexts
Programming for the GP-GPU

- **CUDA**
  - Function-like kernels are written for calculations to be performed on the GPU
    - Data parallel style, one kernel per unit of work
  - Presents a hierarchical organization for thread contexts
    - 2D or 3D grid of blocks
    - 3D block of thread
  - Exposes memory hierarchy explicitly to the user
  - Includes routines for managing device memory and data movement to and from device memory using streams
Starting TotalView

You can debug the CUDA host code using the normal TotalView commands and procedures.
TotalView CUDA Debugging Model
Debugging CUDA

- **Block (x,y,z)**: GPU focus thread selector for changing the block (x,y) and thread (x,y,z) indexes of the CUDA thread.
- **Thread (x,y,z)**: Select a line number in a box to plant a breakpoint.
- **CUDA host threads** have a positive TotalView thread ID.
- **CUDA GPU threads** have a negative TotalView thread ID.

CUDA host threads have a positive TotalView thread ID

CUDA GPU threads have a negative TotalView thread ID

Copyright © 2012 Rogue Wave Software | All Rights Reserved
Running to a Breakpoint in the GPU code

CUDA grid and block dimensions, lanes/warp, warps/SM, SMs, etc.

Parameter, register, local and shared variables

Dive on a variable name to open a variable window

PC arrow for the warp

Stack backtrace (3.2) and inlined functions (3.1)

GPU focus thread logical coordinates
Stepping GPU Code

- single-step operation advances all of the GPU hardware threads in the *same* warp
- To advance the execution of more than one warp, you may either:
  - set a breakpoint and continue the process, or
  - select a line number in the source pane and select “Run To”. 
GPU Device Status Display

- Display of PCs across SMs, Warps and Lanes
- Updates as you step
- Shows what hardware is in use
- Helps you map between logical and hardware coordinates

Example of Divergent GPU threads
Different PC for two groups of Lanes
State of Lanes inside the warp
GPU Memory Hierarchy

- Hierarchical memory
  - Local (thread)
    - Local
    - Register
  - Shared (block)
  - Global (GPU)
    - Global
    - Constant
    - Texture
  - System (host)
TotalView Type Storage Qualifiers

@parameter Address is an offset within parameter storage.

@local Address is an offset within local storage.

@shared Address is an offset within shared storage.

@constant Address is an offset within constant storage.

@global Address is an offset within global storage.

@register Address is a PTX register name.
CUDA Variables

- Storage qualifiers appear in the data type
Control of Threads and Warps

- Warps advance synchronously
  - They share a PC
- Single step operation advances all GPU threads in the same warp
- Stepping over a __syncthreads() call will advance all relevant threads
- To advance more than one warp
  - Continue, possibly after setting a new breakpoint
  - Select a line and “Run To”
CUDA Segmentation Faults

- TotalView displays segmentation faults as expected
- Must enable CUDA memory checking
CUDA Built-in Runtime Variables

- Supported built-in runtime variables are:
  - `struct dim3_16 threadIdx;`
  - `struct dim2_16 blockIdx;`
  - `struct dim3_16 blockDim;`
  - `struct dim2_16 gridSize;`
  - `int warpSize;`
HRL Case Study

- Center for Neural and Emergent Systems at HRL
- Using a CUDA accelerated cluster to model the brain

- “In the first full day of using TotalView, we were quickly able to solve the bug that had us stumped for weeks. With TotalView we were able to step into a specific thread, and then into specific CUDA kernels to identify what went wrong. We could resolve the bugs quickly, and focus our development effort on adding features.”

- “We noticed a dramatic drop in our development cycle – what used to take us more than two weeks to develop and fully test now takes less than one week. By scaling down the development cycle we were able to add more features, even going beyond the requirements of our release cycle. Most important, we were able to focus on the performance of our code, resulting in much better utilization of our existing hardware and allowing us to scale past 100 GPUs.”

- For more information look at the HRL case study on the following page

  http://www.roguewave.com/resources/case-studies.aspx
Agenda

• Who are Rogue Wave?
• What is TotalView?
• Getting Started With TotalView
• TotalView for CUDA
• Latest Updates and Futures
New Capabilities in TotalView 8.10

- **CUDA 4.1**
- **Visual Dive Indicator**
- **Cray-specific enhancements**
  - Improved Cray Compiler Edition Support
  - Replay on Cray XE
  - CUDA support on Cray XK
  - Early Access Preview for OpenACC on the Cray with CCE 8
- **Reverse Debugging**
  - Replay on Demand
  - C++View and ReplayEngine interoperability
- **TVScript Scalability Improvements**
TV 8.10 support for CUDA 4.1 specific features

- Works with the CUDA 4.1 SDK and Runtime
  - New Compiler Front End
  - Slight differences in the debug API
- Support for no copy pinned memory
  - This was broken at the driver level in 4.0
- New support for CUDA device assertions
- New support for multiple CUDA contexts from the same process on the same device
- Support for CUDA on the Cray XK environment
OpenACC

- Pragma Based
  - Similar to OpenMP
  - Supports C and Fortran
- OpenACC Vendors
  - Cray
  - PGI
  - CAPS
- TotalView early access with Cray CCE 8
- Working with PGI and CAPS towards support
Coming Soon

- **Massive Scalability**
  - Collaboration with LLNL and Tri-lab partners
  - Targeting the Cielo, Sequioa 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

- **Intel MIC**

- **Talk to us about participation in early access programs**
Thank you!

Rogue Wave Software

Developing parallel, data-intensive applications is hard. We make it easier.

sales@roguewave.com

Note: TotalView is available free for Students!