Tools for Managing Clusters of NVIDIA GPUs

San Jose, CA | September 21st
My Background

- At NVIDIA for over 7 years
- Lead architect for an internal, large, from-scratch distributed compute farm
  - Used internally for building, testing, simulation HW/SW
  - We transition from 5 nodes -> 100 nodes -> 1000+ nodes
  - Some HPC problems are universal
Topics

- Historical challenges for GPUs
  - Vs. traditional clusters

- Where we stand today
  - Current tool support

- Future directions
  - Which areas we’re focusing on
Historical challenges for GPUs

A 10,000’ assessment of GPU-based hybrid systems:

- Very appealing due to the huge perf/watt advantage
- But, harder to manage and maintain

We haven’t eliminated this latter deficiency, but we’re making great progress
Historical challenges for GPUs

But first, some context

- What are the key cluster management problems?
- Where do hybrid GPU-based systems lag their CPU-only peers?
- And in those cases, why?
Key Problem Spaces

Visibility into system state
- Utilization rates
- ECC error counts
- Crash dumps
- Job failure cause (e.g. OOM)

Monitoring
- Out-of-band
- Runtime health checks
- Crash dumps retrieval
- Fenced Node
- Diagnostic, RMA
Key Problem Spaces

§ Job isolation
- Clean aborts
- Sandboxing (space and time)
- Security

§ Performance
- Minimal monitoring cost
- Minimal SW stack overhead
- Minimal HW latency
Key Problem Spaces

- Tight CM integration
  - Resource scheduling
  - Health monitoring
  - Maintenance
  - Job cleanup/recovery

- Ecosystem integration
  - Programmatic interfaces
  - API support (WMI, SNMP, etc)
Where We Stand Today

- **Nvidia-smi**
  - Addressing: GPU visibility, monitoring, ecosystem integration

- **Nvidia Healthmon**
  - Addressing: Health monitoring

- **GPU Direct**
  - Addressing: Latency/overhead

- **3rd Party Tools**
  - Addressing: Ecosystem integration
nvidia-smi

- Command-line tool
- Windows Server 2008+ and Linux
- Tesla and Fermi architecture compute parts
- Ships with driver
- Our primary monitoring tool
nvidia-smi Features -- Queries

- Get serial #s
  - Immutable, universally unique
- Get PCI device and location ids
- Get thermals
  - Temps for GPU, memory
  - Fan speeds
- Get ECC counts
  - FB, RF, L1, L2
  - Volatile vs. lifetime
Nvidia-smi Features -- Queries

- Get utilization rates
  - GPU % busy
  - Memory usage
- Get compute mode
- Get driver version
Nvidia-smi Features -- Operations

- Reset ECC error counts
- Set compute mode
- Set driver model
  - TCC vs. WDDM
Nvidia-smi Features - TCC Mode

- For Windows Vista and higher systems
- Treat GPU as generic peripheral, not as graphics device
- Benefits:
  - Execution of compute apps over remote desktop
  - Better performance vs. WDDM
  - Fewer memory restrictions

We recommend running compute work in TCC mode
Nvidia-smi – Example Output

Timestamp : 09/20/2010 11:29:53 PM
Driver Version : 260.68

GPU 0:
Product Name : Tesla C2050
PCI Device/Vendor ID : 6d110de
PCI Location ID : 0:3:0
Display : Connected
Temperature : 68 C
Fan Speed : 30%
Utilization
GPU : 0%
Memory : 3%
ECC errors :
Single bit :
FB : 0
RF : 0
L1 : 0
L2 : 0
Total : 0
Double bit :
FB : 0
RF : 0
L1 : 0
L2 : 0
Total : 0
Nvidia-smi - Example Output

**Timestamp**
Mon Sep 20 02:34:51 2010

**Unit 0:**
- **Product Name:** S2050
- **Product ID:** 01--0001--001
- **Serial Number:** d345b21
- **Firmware Ver:** 6.2
- **Intake Temperature:** 21 C
- **Power Capping:**
  - **Power limit:** 1200 watts
  - **Up event count:** 0
  - **Down event count:** 0
  - **Up latency:** 20000ms
  - **Down latency:** 100ms

**GPU 0:**
- **Product Name:** Tesla S2050
- **Serial:** 0330510041162
- **PCI Device/Vendor ID:** 6de10de
- **PCI Location ID:** 0:7:0
- **Bridge Port:** 0
- **Temperature:** 56 C

**GPU 1:**
- **Product Name:** Tesla S2050
- **Serial:** 0330510041163
- **PCI Device/Vendor ID:** 6de10de
- **PCI Location ID:** 0:8:0
- **Bridge Port:** 2
- **Temperature:** 49 C

**ECC errors:**
- **Single Bit:**
  - FB: 0
  - RF: 0
  - L1: 0
  - L2: 0
  - **Total:** 0
- **Double Bit:**
  - FB: 0
  - RF: 0
  - L1: 0
  - L2: 0
  - **Total:** 0

**Fan Tachs:**
- #00: 3570 Status: NORMAL
- #01: 3432 Status: NORMAL
- #02: 3628 Status: NORMAL
- #03: 3474 Status: NORMAL
- #04: 3556 Status: NORMAL
- #05: 3492 Status: NORMAL
- #06: 3580 Status: NORMAL
- #07: 3410 Status: NORMAL
- #08: 3464 Status: NORMAL
- #09: 3508 Status: NORMAL
- #10: 3646 Status: NORMAL
- #11: 3450 Status: NORMAL
- #12: 3634 Status: NORMAL
- #13: 3388 Status: NORMAL

**PSU:**
- **Voltage:** 11.98 V
- **Current:** 31.66 A
- **State:** Normal

**LED:**
- **State:** GREEN
Nvidia Healthmon

- Simple tool for assessing health of GPU node.

- Performs:
  - Basic device query
  - Basic memory test (on ECC products)
  - PCIe Bandwidth test (host->device and device->host)

- 15 - 20 seconds
Nvidia Healthmon - Example Output

NVIDIA Tesla Health Monitor v0.1
Device Enumeration:
1 devices detected

Device 0: Tesla C2050
Compute capability: 2.0
Amount of memory: 2817720320 bytes
ECC: enabled
Number of SMs: 14
Core clock: 1147 MHz
Watchdog timeout: disabled
Compute mode: default (supports multiple simultaneous contexts)

GPU Functional Validation

Device 0: Tesla C2050
Allocated 2682684702 bytes
Test PASSED

PCIe Bandwidth

Device 0: Tesla C2050
Host-to-device bandwidth: 3142.090088 MB/s
Device-to-host bandwidth: 2973.980469 MB/s
Bidirectional device bandwidth: 4322.295898 MB/s
Test PASSED
3rd Party Tools - Bright Computing
3rd Party Tools - Bright Computing

<table>
<thead>
<tr>
<th>3rd Party Tools - Bright Computing</th>
<th>Bright Cluster Manager</th>
<th>Click to edit Master text styles</th>
</tr>
</thead>
<tbody>
<tr>
<td>RESOURCES</td>
<td></td>
<td>Second level</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

![Bright Cluster Manager screenshot](image-url)
Future Directions

Based on feedback from current & future customers

Main areas of focus:
- Programmatic interfaces
- Monitoring/visibility
- Out-of-band
- SW ecosystem
Programmatic Interfaces

- Ship a new library with the driver, NVML
- Provide API that encapsulates nvidia-smi functionality, plus additional goodness
- Future nvidia-smi built on top of NVML
- Building block for 3rd party tools
Out-of-band Monitoring

- Move GPU monitoring off CPU
- Thermals, utilization, ECC errors, etc

- Crash dumps
- Get some useful data even if GPU/driver is hung

- OEM engagement to build this in to real products
SW Ecosystem

- Lots of tools that we can enable or create ourselves
  - Not always clear which route is better
  - General philosophy is to enable others to create tools, e.g. with NVML

- Homegrown?
  - Windows Perf Counters
  - Drop-in WMI, SNMP, etc clients
GPUDirect

San Jose, CA | September 21st
GPUDirect v1 Definition

- Allows 3rd Party to share pinned system memory on Linux
- Requires a Linux kernel patch to support `get_driver_pages()`
- Requires 3rd Party drivers to add support for `get_driver_pages()`
- Initial support for Infiniband and Mellanox
GPUDirect v1 Motivation

- Remove a copy in system memory
- Handle the failure/teardown safely through a callback mechanism
GPUDirect v1 Improvements

- Remove a copy
- Allows MPI to use RDMA with CUDA pinned system memory
- Improved scalability in clusters since the CPU will do fewer memory copies
- AMBER sees ~ 6% improvement
We’d Love to Hear From You

Feedback from our customers/collaborators helps us refine our efforts and focus.

- Chat with us after this session
- Cluster Management Pod @ table 22 (Back wall, Exhibit Hall)
- Today 6 - 8pm
- Wednesday 11am - 2pm & 6 - 8pm
- Thursday 11am - 2pm