NSIGHT PROFILING TOOLS WORKFLOW

Nsight Systems
Comprehensive Workload-Level Performance

Recheck Overall Workload Behavior

Optimize:
Synchronization, Data Movement,
Overlap / Parallelization

Nsight Compute
Detailed CUDA Kernel Performance

Optimize:
GPU Utilization, Kernel Implementation,
Memory Accesses

Nsight Graphics
Detailed Frame/Render Performance

Optimize:
Frame Composition, Rendering, Shaders,
Synchronization

Dive into Top CUDA Kernels
Dive into Graphics Frames

Start Here

Finished, if Performance Satisfactory

Recheck Overall Workload Behavior
NSIGHT SYSTEMS
System Profiler

Key Features:

- System-wide application algorithm tuning
  - Multi-process tree support
- Locate optimization opportunities
  - Visualize millions of events on a very fast GUI timeline
  - Or gaps of unused CPU and GPU time
- Balance your workload across multiple CPUs and GPUs
  - CPU algorithms, utilization and thread state
  - GPU streams, kernels, memory transfers, etc
- Command Line, Standalone, IDE Integration

OS: Linux (x86, Power, Arm SBSA, Tegra), Windows, MacOSX (host)
GPUs: Pascal+

Processes & threads
OpenGL trace
NVTX annotations
CUDA API trace
GPU CUDA Kernel & memory transfer activities
CPU call-stack samples
Thread activity
Thread CPU core
Thread state
ZOOM/FILTER TO EXACT AREAS OF INTEREST
<table>
<thead>
<tr>
<th>Symbol Name</th>
<th>Self, %</th>
<th>Module Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>VolumetricData::compute_volume_gradient()</td>
<td>20.14</td>
<td>/home/johns/vmd/src/gtcbuilds/vmd_LINUXX64.11</td>
</tr>
<tr>
<td>VolumetricData::compute_volume_gradient()</td>
<td>20.14</td>
<td>/home/johns/vmd/src/gtcbuilds/vmd_LINUXX64.11</td>
</tr>
<tr>
<td>BaseMolecule::add_volume_data(char const*, double const*, double const*, double const*, double const*, int, int, int, float*)</td>
<td>18.30</td>
<td>/home/johns/vmd/src/gtcbuilds/vmd_LINUXX64.11</td>
</tr>
<tr>
<td>VMDApp::molecule_add_volumetric(int, char const*, double const*, double const*, double const*, double const*, double const*, int, int, float*)</td>
<td>18.30</td>
<td>/home/johns/vmd/src/gtcbuilds/vmd_LINUXX64.11</td>
</tr>
<tr>
<td>obj_segmentation(void*, Tcl_Interp*, int, Tcl_Obj* const*)</td>
<td>18.30</td>
<td>/home/johns/vmd/src/gtcbuilds/vmd_LINUXX64.11</td>
</tr>
<tr>
<td>[Max depth]</td>
<td>18.30</td>
<td></td>
</tr>
<tr>
<td>BaseMolecule::add_volume_data(char const*, float const*, float const*, float const*, float const*, float const*, float const*, int, int, float*, float*, float*)</td>
<td>1.84</td>
<td>/home/johns/vmd/src/gtcbuilds/vmd_LINUXX64.11</td>
</tr>
<tr>
<td>MolecPlugin::read_volumetric(Molecule**, int, int const*)</td>
<td>1.84</td>
<td>/home/johns/vmd/src/gtcbuilds/vmd_LINUXX64.11</td>
</tr>
<tr>
<td>VMDApp::molecule_load(int, char const*, char const*, FileSpec const*)</td>
<td>1.84</td>
<td>/home/johns/vmd/src/gtcbuilds/vmd_LINUXX64.11</td>
</tr>
<tr>
<td>text_cmd_mol( void*, Tcl_Interp*, int, char const** )</td>
<td>1.84</td>
<td>/home/johns/vmd/src/gtcbuilds/vmd_LINUXX64.11</td>
</tr>
<tr>
<td>TclInvokeStringCommand</td>
<td>1.84</td>
<td>/home/johns/vmd/src/gtcbuilds/vmd_LINUXX64.11</td>
</tr>
<tr>
<td>TclEvalObj::internal</td>
<td>1.84</td>
<td>/home/johns/vmd/src/gtcbuilds/vmd_LINUXX64.11</td>
</tr>
<tr>
<td>TclExecuteByteCode</td>
<td>1.84</td>
<td>/home/johns/vmd/src/gtcbuilds/vmd_LINUXX64.11</td>
</tr>
<tr>
<td>TclCompEvalObj</td>
<td>1.84</td>
<td>/home/johns/vmd/src/gtcbuilds/vmd_LINUXX64.11</td>
</tr>
<tr>
<td>TclEvalObjEx</td>
<td>1.84</td>
<td>/home/johns/vmd/src/gtcbuilds/vmd_LINUXX64.11</td>
</tr>
<tr>
<td>Tcl_Cmd$smarty</td>
<td>1.84</td>
<td>/home/johns/vmd/src/gtcbuilds/vmd_LINUXX64.11</td>
</tr>
<tr>
<td>text_cmd_mol( void*, Tcl_Interp*, int, char const** )</td>
<td>1.84</td>
<td>/home/johns/vmd/src/gtcbuilds/vmd_LINUXX64.11</td>
</tr>
<tr>
<td>TclTextInterp::evalFile(char const*)</td>
<td>1.84</td>
<td>/home/johns/vmd/src/gtcbuilds/vmd_LINUXX64.11</td>
</tr>
<tr>
<td>VMDApp::logfile_read(char const*)</td>
<td>1.84</td>
<td>/home/johns/vmd/src/gtcbuilds/vmd_LINUXX64.11</td>
</tr>
<tr>
<td>VMDReadStartup(VMDApp*)</td>
<td>1.84</td>
<td>/home/johns/vmd/src/gtcbuilds/vmd_LINUXX64.11</td>
</tr>
<tr>
<td>[Max depth]</td>
<td>1.84</td>
<td></td>
</tr>
<tr>
<td>0x7f10ca7022d6</td>
<td>5.13</td>
<td>/usr/lib64/libcuda.so.390.25</td>
</tr>
<tr>
<td>obj_segmentation( void*, Tcl_Interp*, int, Tcl_Obj* const*)</td>
<td>3.44</td>
<td>/home/johns/vmd/src/gtcbuilds/vmd_LINUXX64.11</td>
</tr>
</tbody>
</table>
EXPERT SYSTEM
GPU METRICS SAMPLING
MULTI-REPORT TILING
Visualize more parallel activity

Open multiple reports
Correlated on same timeline
NSIGHT COMPUTE
Kernel Profiling Tool

Key Features:
- Interactive CUDA API debugging and kernel profiling
- Built-in rules expertise
- Fully customizable data collection and display
- Command Line, Standalone, IDE Integration, Remote Targets

OS: Linux (x86, Power, Tegra, Arm SBSA), Windows, MacOSX (host only)

GPUs: Volta+

Targeted metric sections

Built-in expertise for Guided Analysis and optimization

Customizable data collection and presentation
Metrics for peak performance ratios

Visual memory analysis chart
BASELINES

- Comparison of results directly within the tool with "Baselines"

- Supported across kernels, reports, and GPU architectures
- Source/PTX/SASS analysis and correlation
- Source metrics per instruction and aggregated (e.g. PC sampling data)
- Metric heatmap

SOURCE VIEW
STANDALONE SOURCE VIEWER

- View of side-by-side assembly and correlated source code for CUDA kernels
- No profile required
- Open .cubin files directly
- Helps identify compiler optimizations and inefficiencies
OCCUPANCY CALCULATOR
Model Hardware Usage and Identify Limiters

- Model theoretical hardware usage
- Understand limitations from hardware vs. kernel parameters
- Configure model to vary HW and kernel parameters
- Opened from an existing report or as a new activity
▪ Visualize multiple levels of the memory hierarchy
▪ Identify bottlenecks caused by memory limitations
▪ Determine how modifying algorithms may (or may not) impact performance
REPLAY MODES

https://docs.nvidia.com/nsight-compute/ProfilingGuide/index.html#replay

Kernel Replay
(interactive and non-interactive)

Range Replay
(non-interactive)

Application Replay
(non-interactive)
FURTHER INFORMATION

Download


https://developer.nvidia.com/nsight-systems

https://developer.nvidia.com/nsight-compute

Documentation

https://docs.nvidia.com/nsight-systems/

https://docs.nvidia.com/nsight-compute

Support is available via:


More information at:
