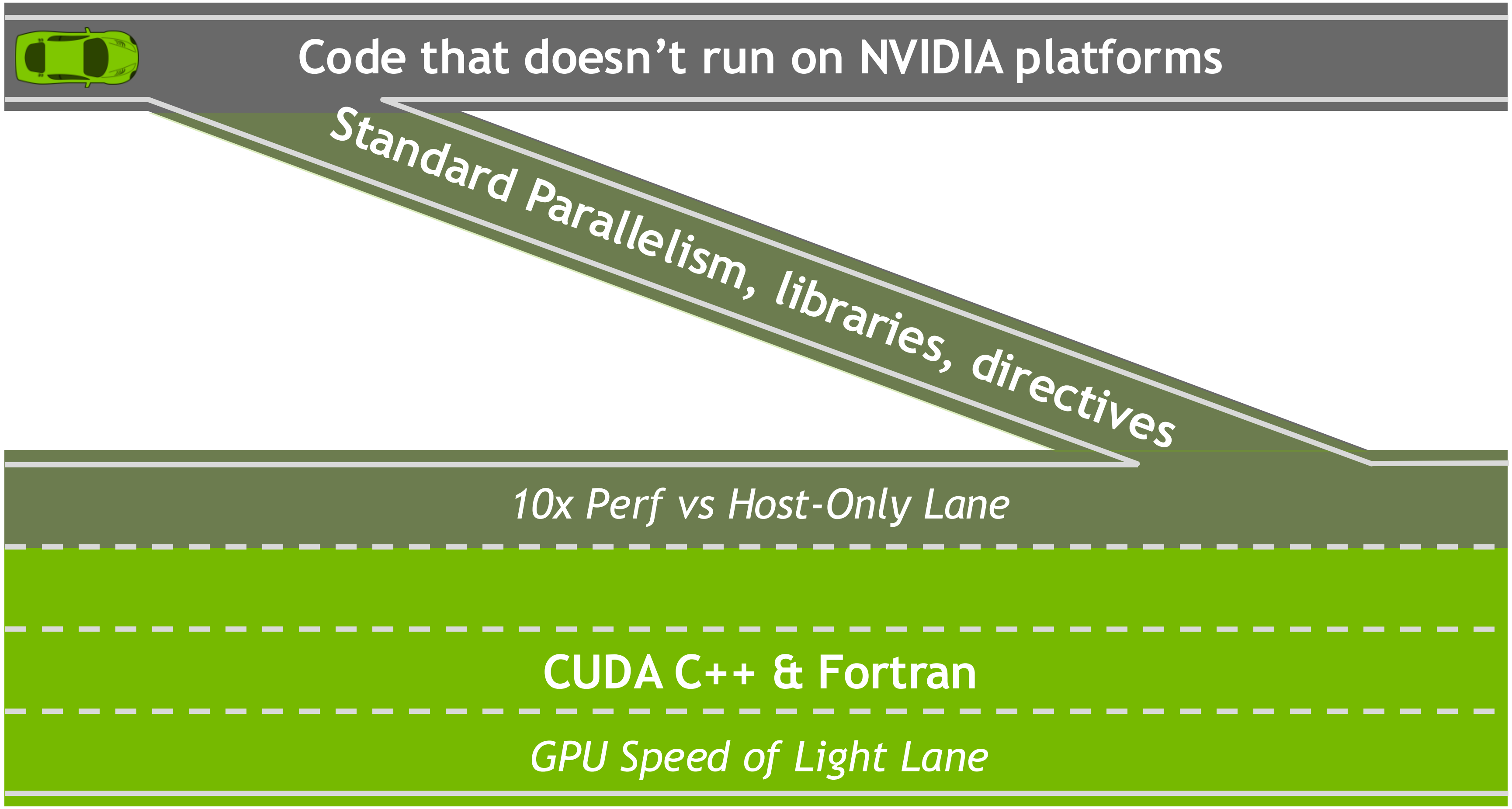
NVIDIA Developer Tools:
Nsight Systems & Nsight Compute Intro

Outline for today

- Quick overview – Developer Tools Place in the Ecosystem
- NSight Systems, Nsight Compute
- Walkthrough of example:
  - Code available at https://github.com/ljdursi/alliance_oct24_nsight
  - Includes Nsight reports in case you're having trouble generating them
- Some advanced topics we didn't see
- Questions and Office Hours
- Let the Accelerator Working group know if there are other topics you'd like covered!

How we think about GPU development

GPU Computing Needs On-Ramps



Programming the NVIDIA Platform

CPU, GPU, and Network

ACCELERATED STANDARD LANGUAGES

ISO C++, ISO Fortran

```
std::transform(par, x, x+n, y, y,
    [=](float x, float y){ return y +
a*x; }
);
```

```
do concurrent (i = 1:n)
   y(i) = y(i) + a*x(i)
enddo
```

```
import cunumeric as np
…
def saxpy(a, x, y):
    y[:] += a*x
```

INCREMENTAL PORTABLE OPTIMIZATION

OpenACC, OpenMP

```
#pragma acc data copy(x,y) {
...
std::transform(par, x, x+n, y, y,
    [=](float x, float y){
        return y + a*x;
});
...
}
```

```
#pragma omp target data map(x,y) {
...
std::transform(par, x, x+n, y, y,
    [=](float x, float y){
        return y + a*x;
});
...
}
```

PLATFORM SPECIALIZATION

CUDA

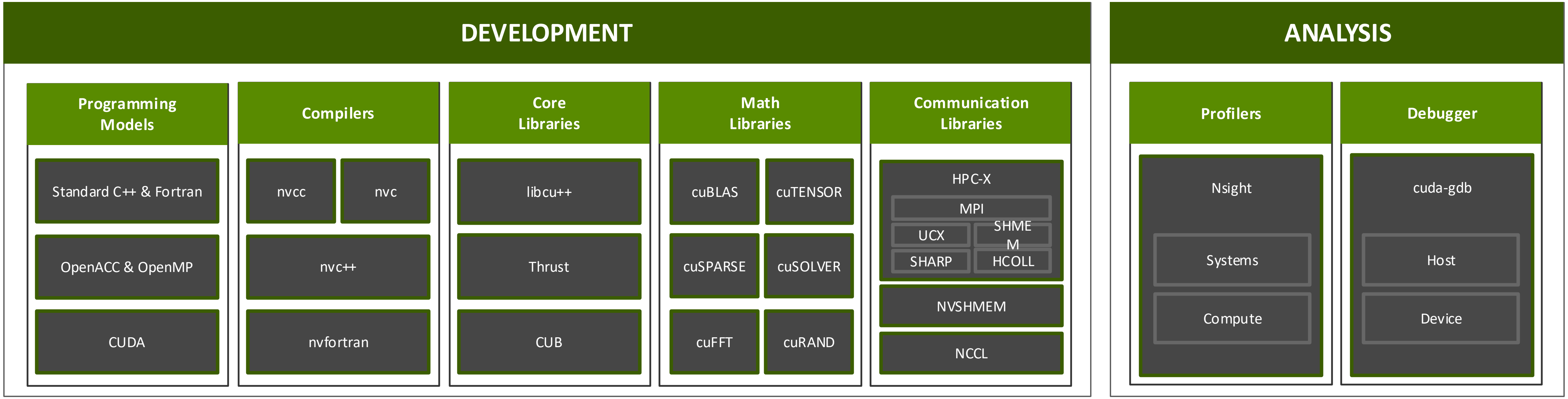
```
__global__
void saxpy(int n, float a,
           float *x, float *y) {
  int i = blockIdx.x*blockDim.x +
          threadIdx.x;
  if (i < n) y[i] += a*x[i];
}

int main(void) {
  ...
  cudaMemcpy(d_x, x, ...);
  cudaMemcpy(d_y, y, ...);

  saxpy<<<(N+255)/256,256>>>(...);

  cudaMemcpy(y, d_y, ...);
```

ACCELERATION LIBRARIES

Core | Math | Communication | Data Analytics | AI | Quantum

NVIDIA HPC SDK

Available at developer.nvidia.com/hpc-sdk, on NGC, via Spack, and in the Cloud

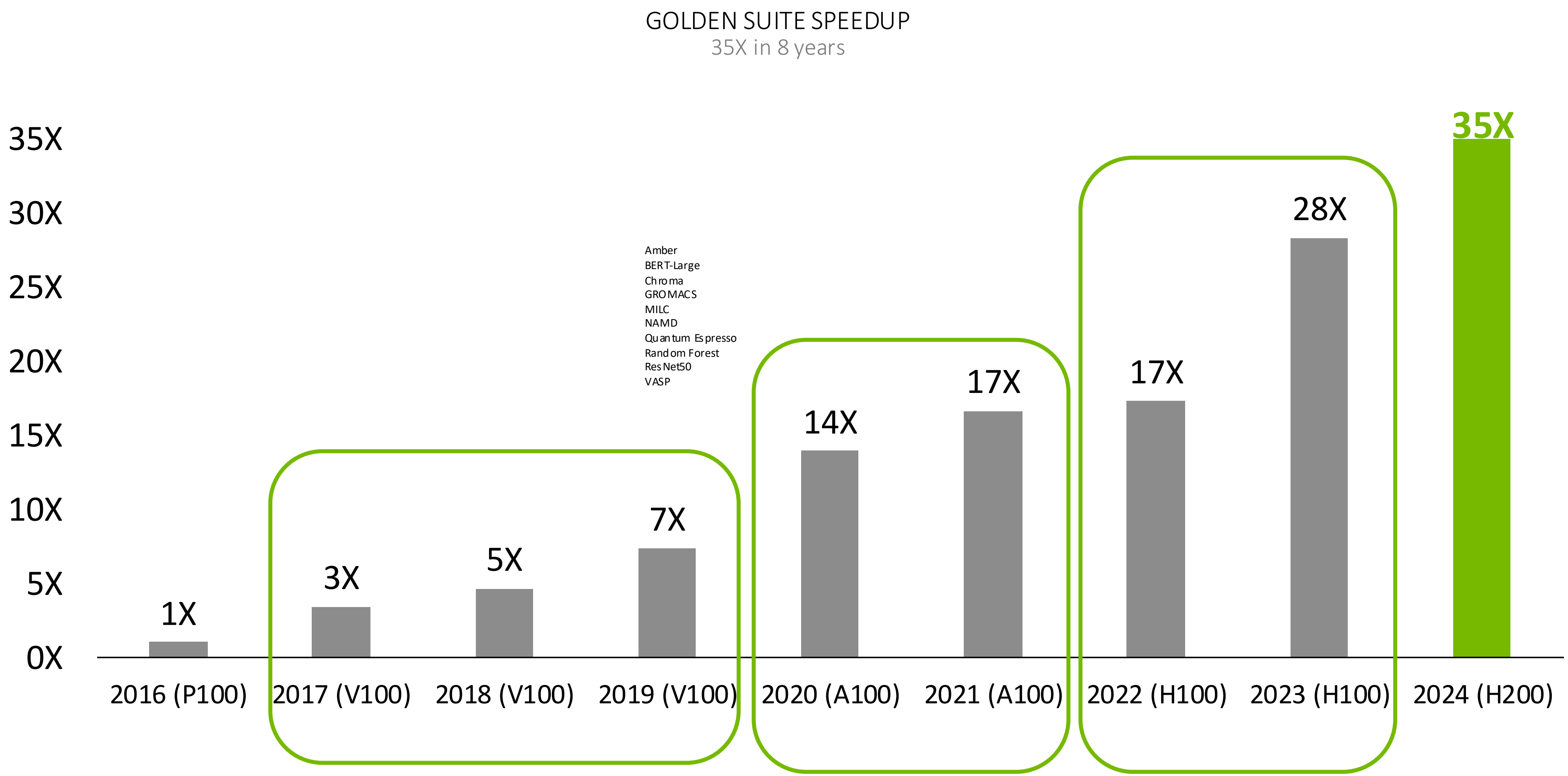


Develop for the NVIDIA Platform: GPU, CPU and Interconnect

Libraries | Accelerated C++ and Fortran | Directives | CUDA

7-8 Releases Per Year | Freely Available

Libraries, Tools Continually Improve Performance

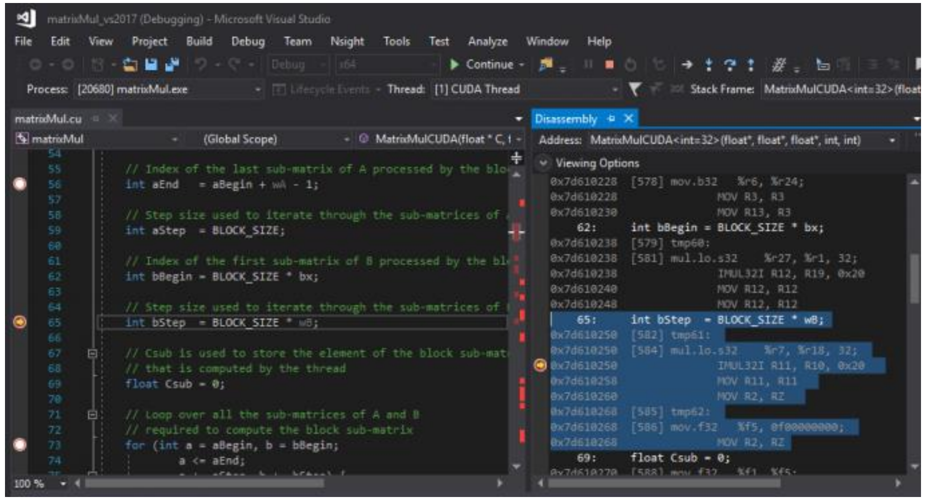


Geometric mean of application speedups vs. P100 in 2016 | benchmark applications | Amber [PME-Cellulose_NVE], Chroma [HMC], GROMACS [ADH Dodec], MILC [Apex Medium], NAMD [stmv_nve_cuda], PyTorch (BERT Large Fine Tuner], Quantum Espresso [AUSURF112-jR]; TensorFlow [ResNet-50], VASP 6 [Si Huge]; Random Forest make_blobs (160000 x 64 : 10).

Developer Tools Ecosystem

Developer tools

Debuggers: cuda-gdb, Nsight Visual Studio Edition

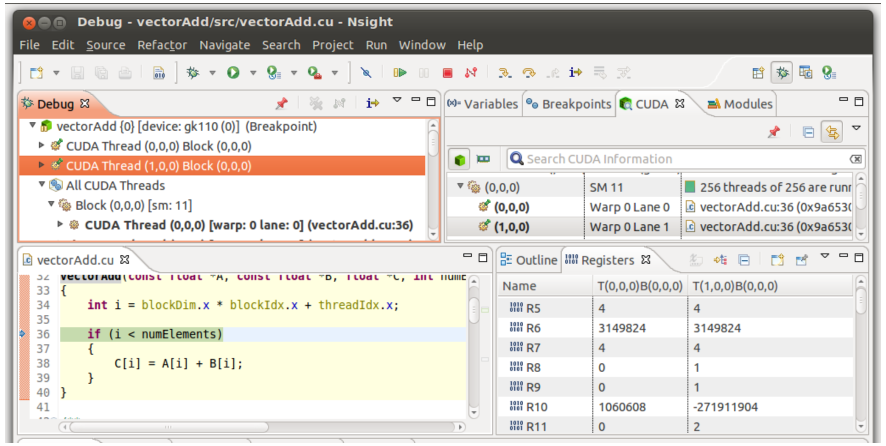


Profilers: Nsight Systems, Nsight Compute, NVIDIA Tools eXtension (NVTX)

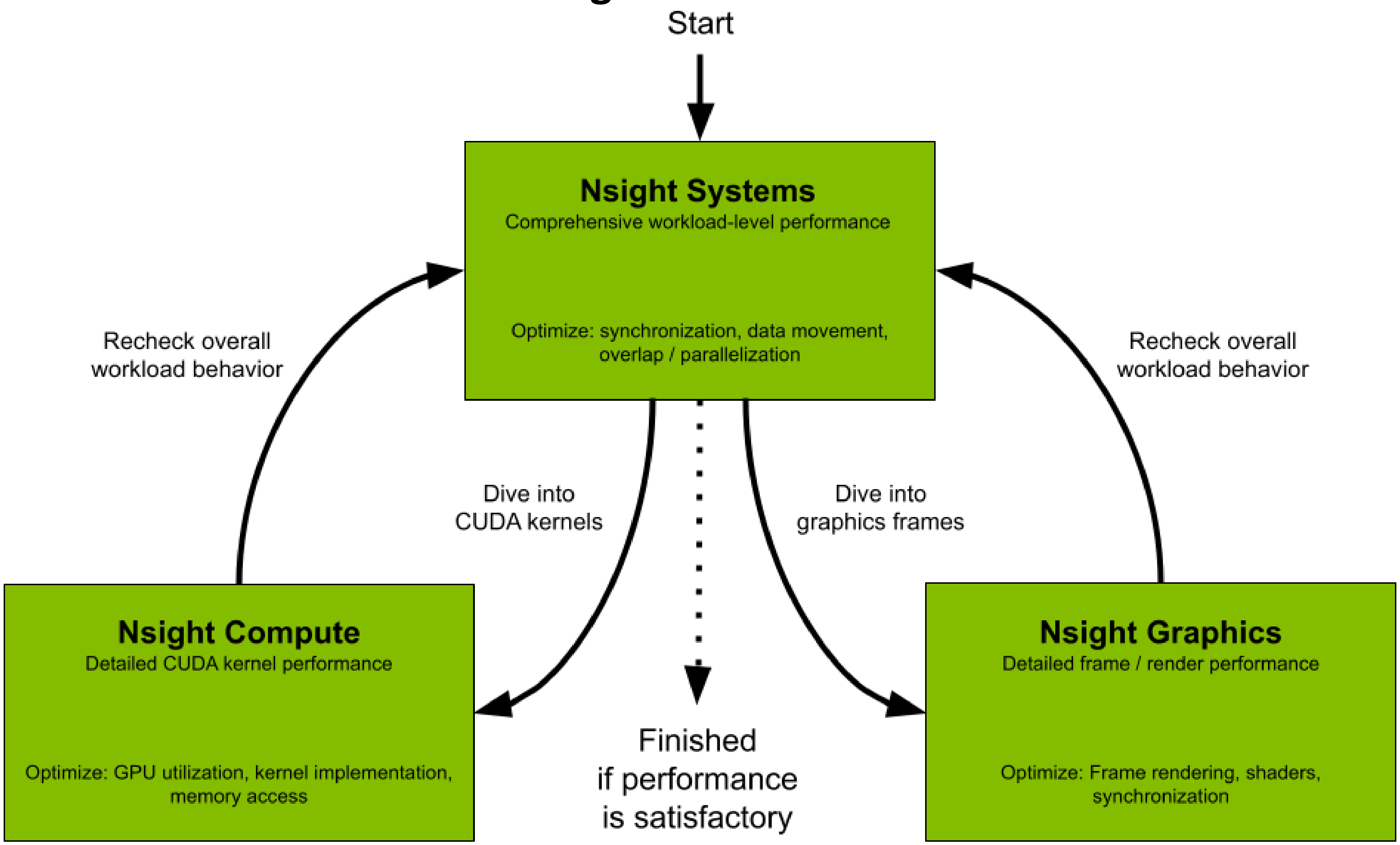
Correctness Checker: Compute Sanitizer

```
$ compute-sanitizer --leak-check full memcheck_demo
========= COMPUTE-SANITIZER
Mallocing memory
Running unaligned_kernel
Ran unaligned_kernel: no error
Sync: no error
Running out_of_bounds_kernel
Ran out_of_bounds_kernel: no error
Sync: no error
========= Invalid __global__ write of size 4 bytes
=========     at 0x60 in memcheck_demo.cu:6:unaligned_kernel(void)
=========     by thread (0,0,0) in block (0,0,0)
=========     Address 0x400100001 is misaligned
```

IDE integrations: Nsight Eclipse Edition
Nsight Visual Studio Edition
Nsight Visual Studio **Code** Edition



Nsight Overview

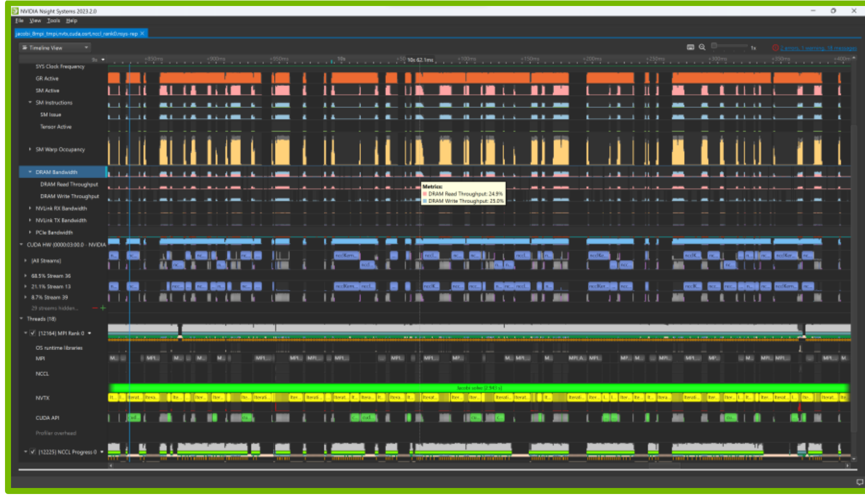


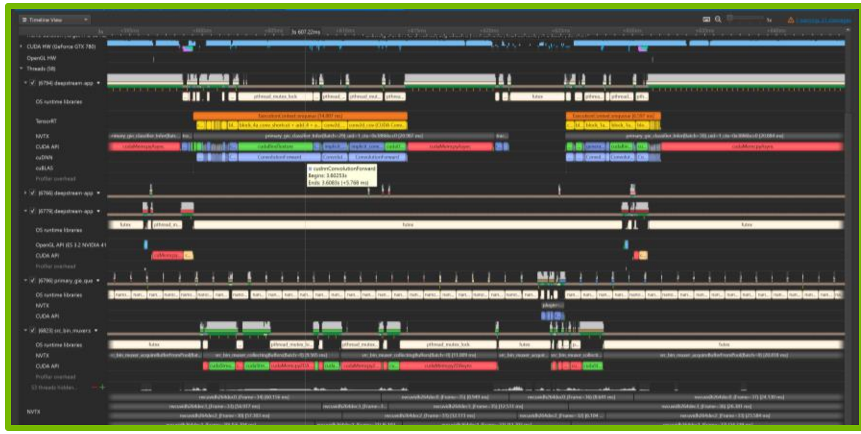
Nsight Systems

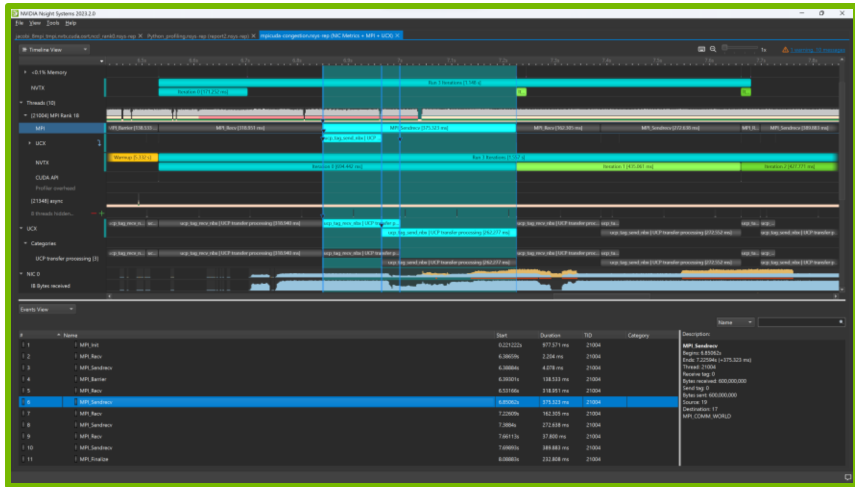
System Profiler

Key Features:

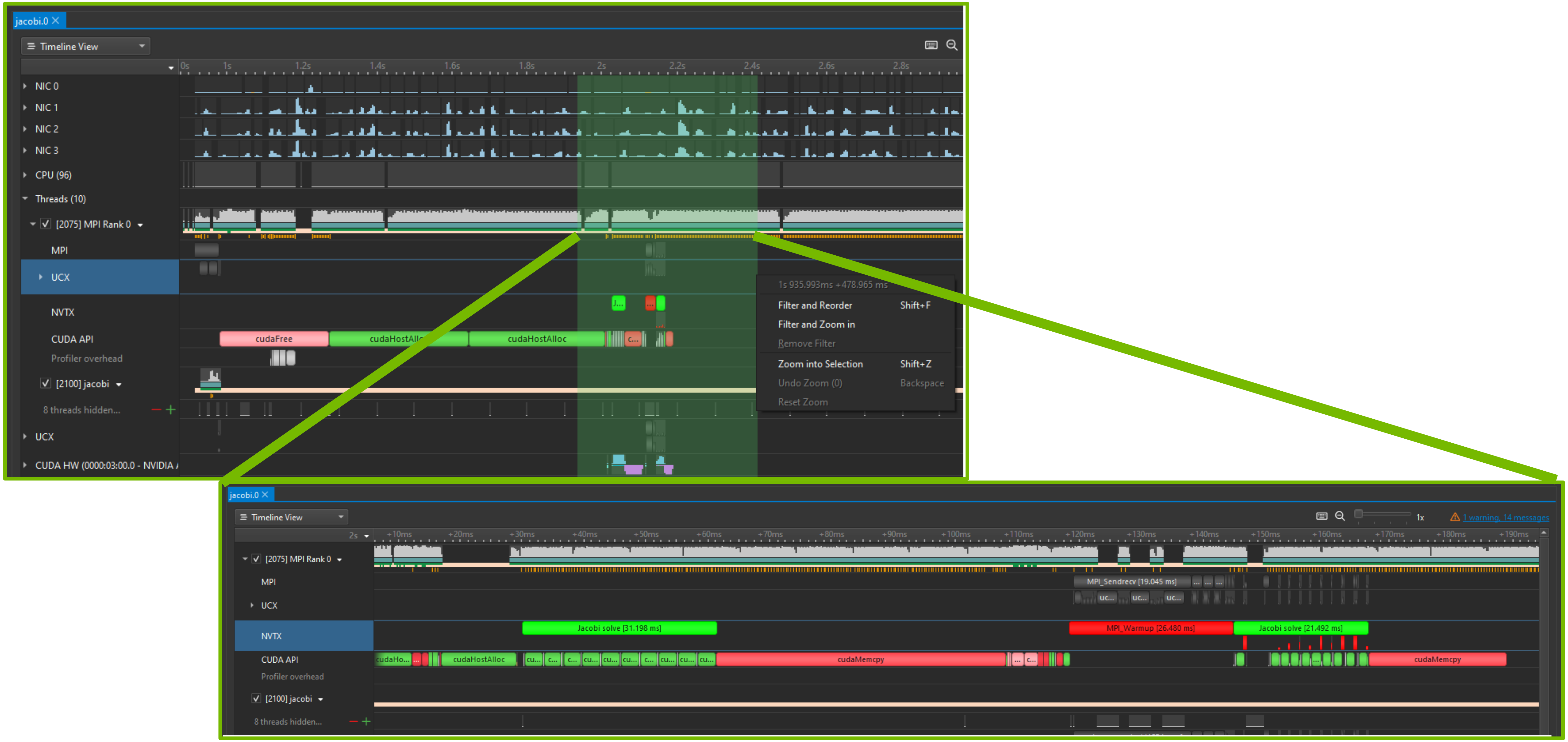
- System-wide application algorithm tuning
  - Multi-process tree support
- Locate optimization opportunities
  - Visualize millions of events on a very fast GUI timeline
  - Identify gaps of unused CPU and GPU time
- Balance your workload across multiple CPUs and GPUs
  - CPU algorithms, utilization and thread state
    GPU streams, kernels, memory transfers, etc
- Command Line, Standalone, IDE Integration
- OS: Linux (x86, ARM, Tegra), Windows, macOS X (host)
- GPUs: Pascal+
- Docs/product: https://developer.nvidia.com/nsight-systems







Zoom/Filter to Exact Areas of Interest



NVIDIA

NVIDIA Tools eXtension (NVTX)

- Decorate application source code with annotations (markers, ranges, nested ranges, …) to help visualize execution with debugging, tracing and profiling tools
- Header-only library https://github.com/NVIDIA/NVTX/tree/release-v3/c.

```
#include <nvtx3/nvToolsExt.h>
```

- Marker:

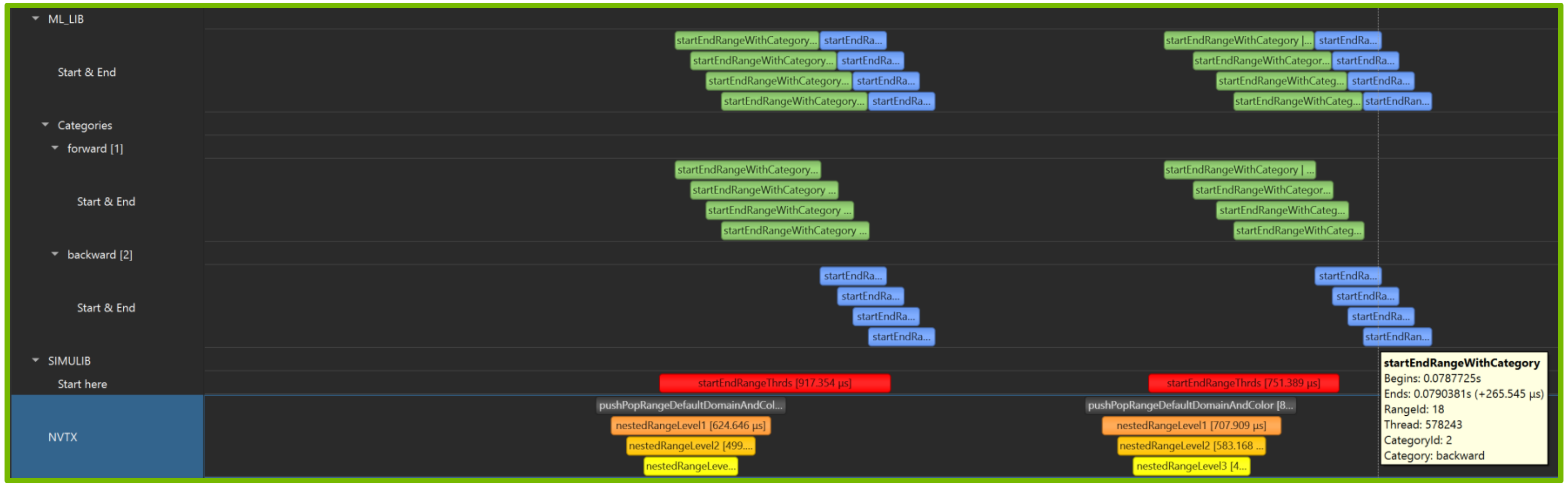
```
nvtxMark("This is a marker");
```

- Push-Pop range

```
nvtxRangePush("This is a push/pop range");
// Do something interesting in the range
nvtxRangePop();  // Pop must be on same thread as corresponding Push
```

- Start-End range

```
nvtxRangeHandle_t handle = nvtxRangeStart("This is a start/end range");
// Somewhere else in the code, not necessarily same thread as Start call:
nvtxRangeEnd(handle);
```



API references https://nvidia.github.io/NVTX/doxygen/index.html and https://nvidia.github.io/NVTX/doxygen-cpp/index.html

NVIDIA

Python and NVTX

- Annotate Python code with NVTX

```
# demo.py

import time
import nvtx

@nvtx.annotate(color="blue")
def my_function():
    for i in range(5):
        with nvtx.annotate("my_loop", color="red"):
            time.sleep(i)

my_function()
```

- pip install nvtx - https://pypi.org/project/nvtx/

Application Profiles with Nsight Systems

$ nsys profile -o report -stats=true ./myapp.exe

- Generated file: report.qdrep (or report.nsys-rep)
  Open for viewing in the Nsight Systems UI

- When using MPI, recommended to use *nsys* after mpirun/srun:
  $ mpirun -n 4 nsys profile ./myapp.exe

- Kernel's `perf_event_paranoid` value on the node has to be set to be 2 or less (docs link) or other processes (even those of the same user) can't access CPU performance information.

Nsight Compute

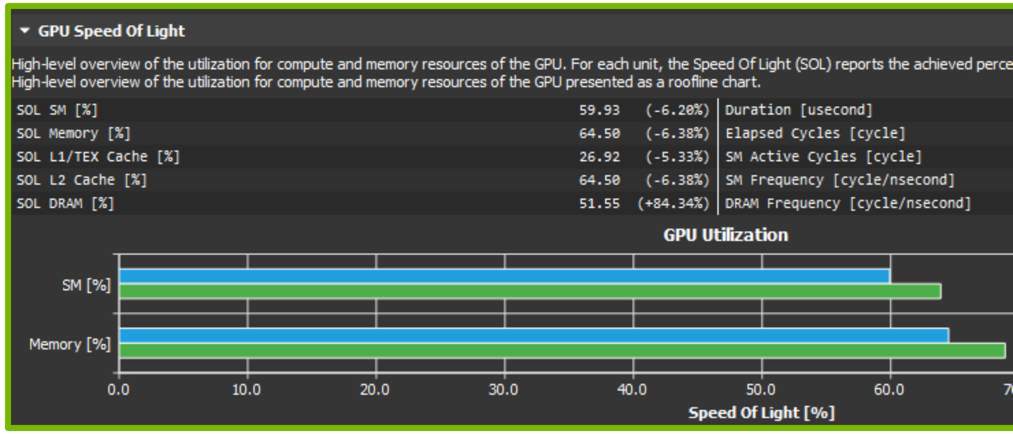
Kernel Profiler

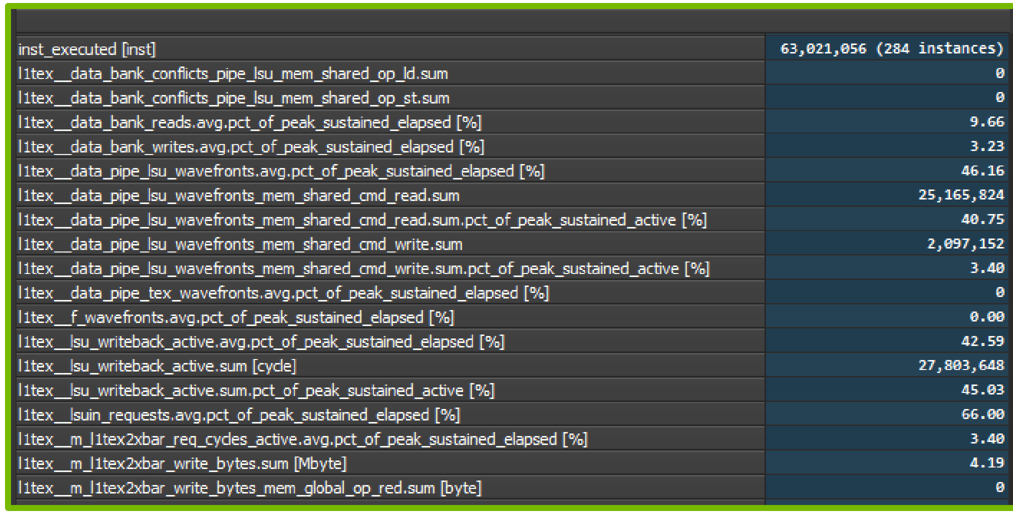
Key Features:

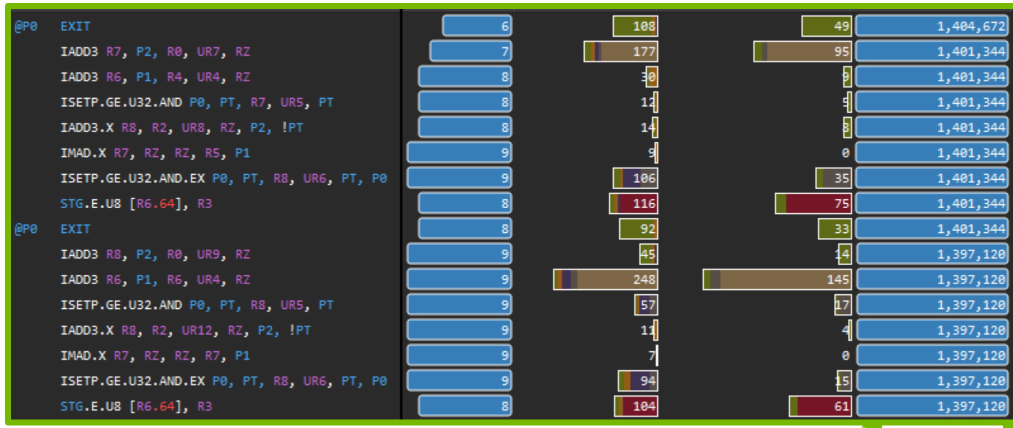
- Interactive CUDA API debugging and kernel profiling
- Built-in rules expertise
- Fully customizable data collection and display
- Command Line, Standalone, IDE Integration, Remote Targets

- OS: Linux (x86, Power, Tegra, Arm SBSA), Windows, macOS X (host only)
- GPUs: Volta+

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





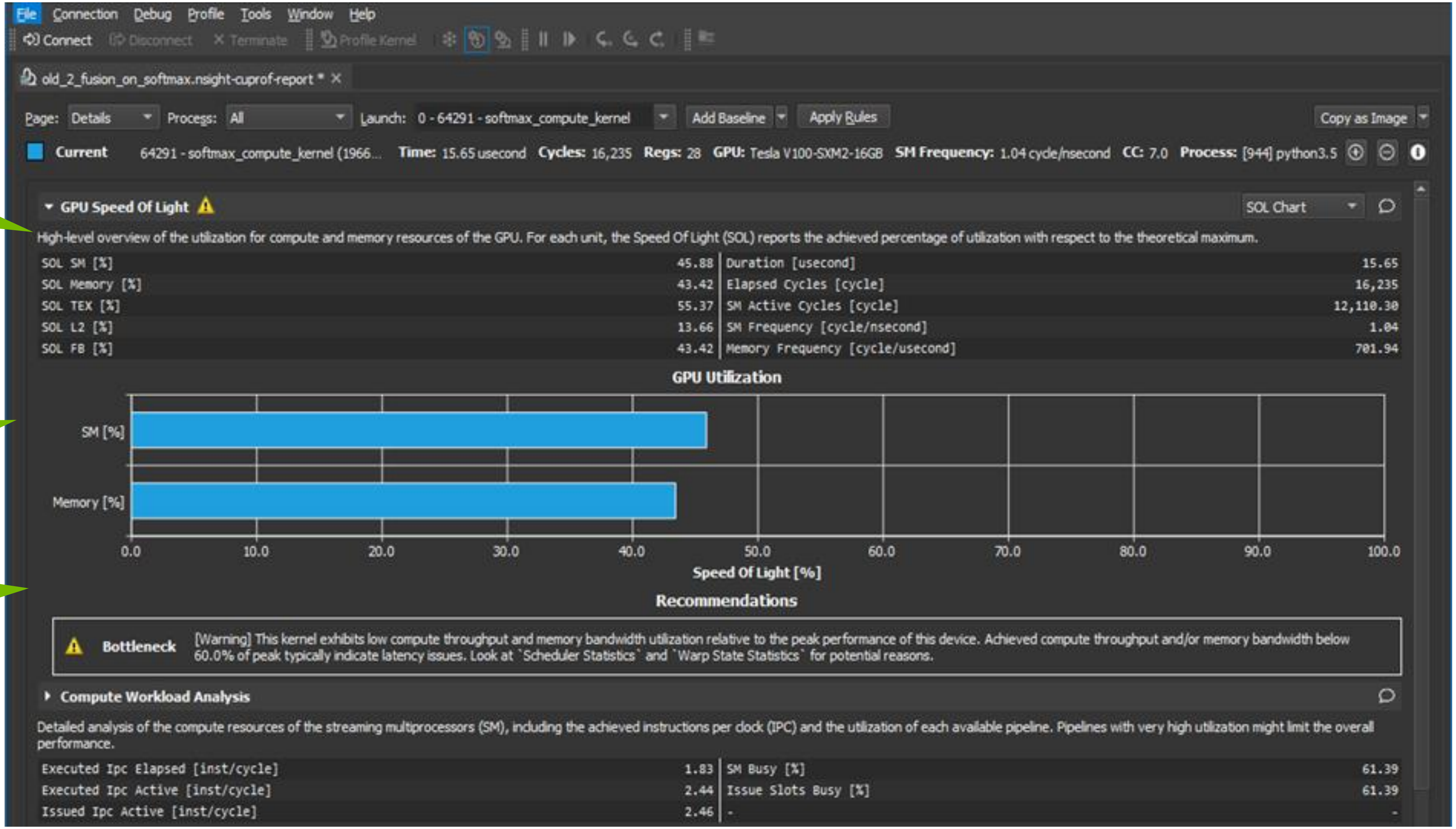


Nsight Compute GUI Interface

Targeted metric sections

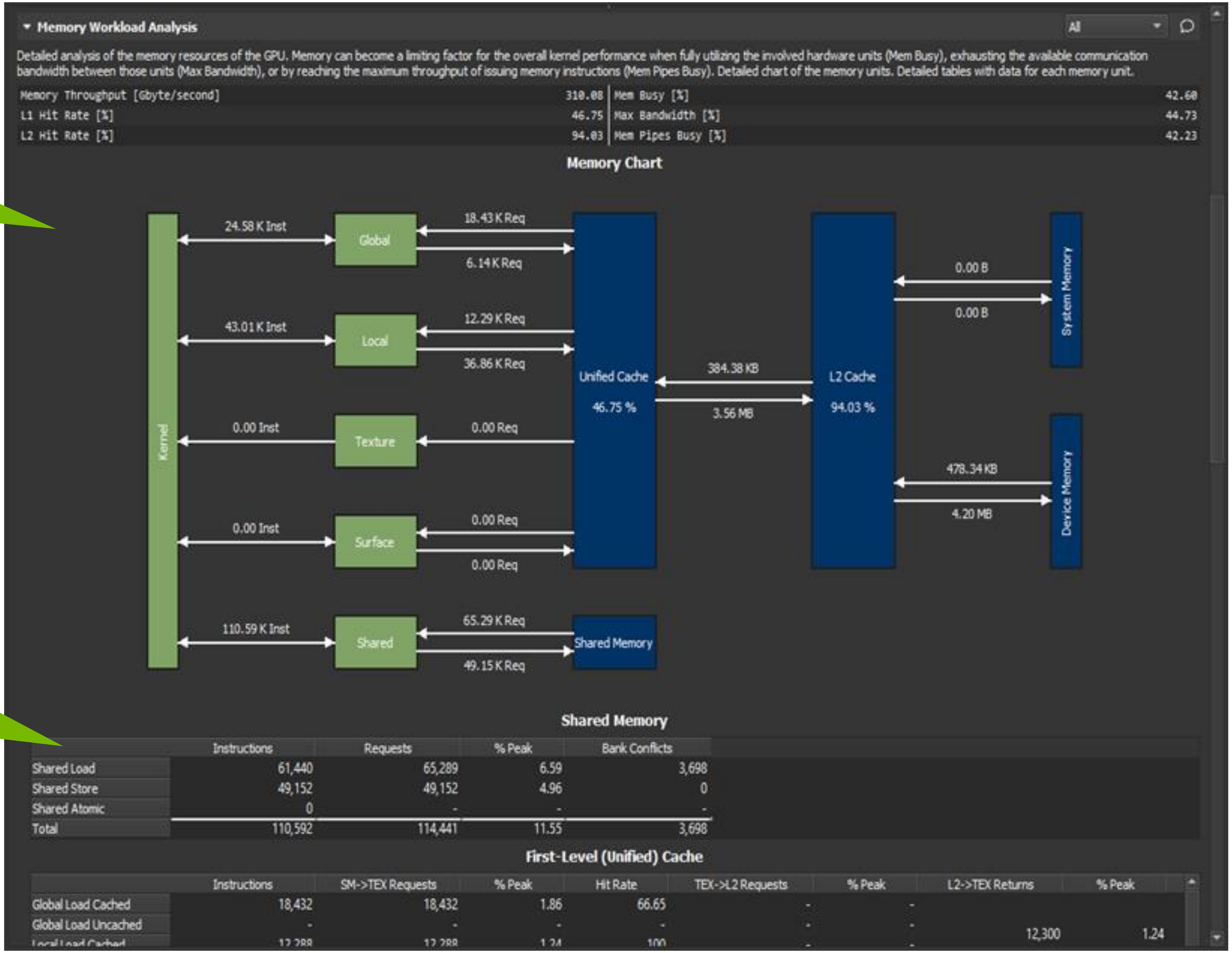
Customizable data collection and presentation

Built-in expertise for Guided Analysis and optimization

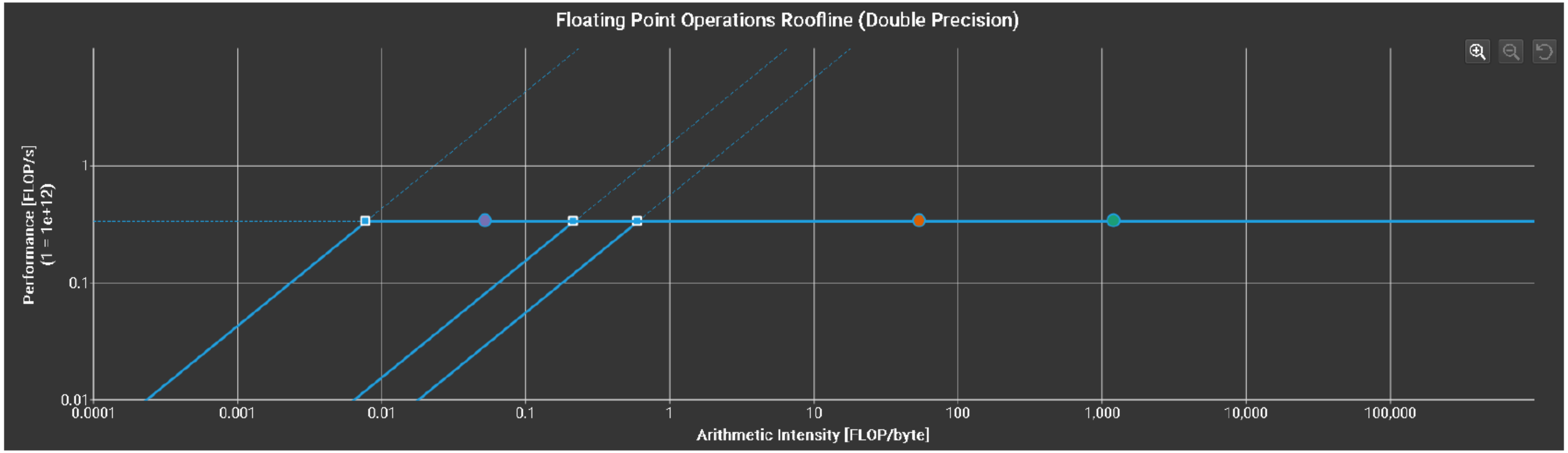


Visual memory analysis chart

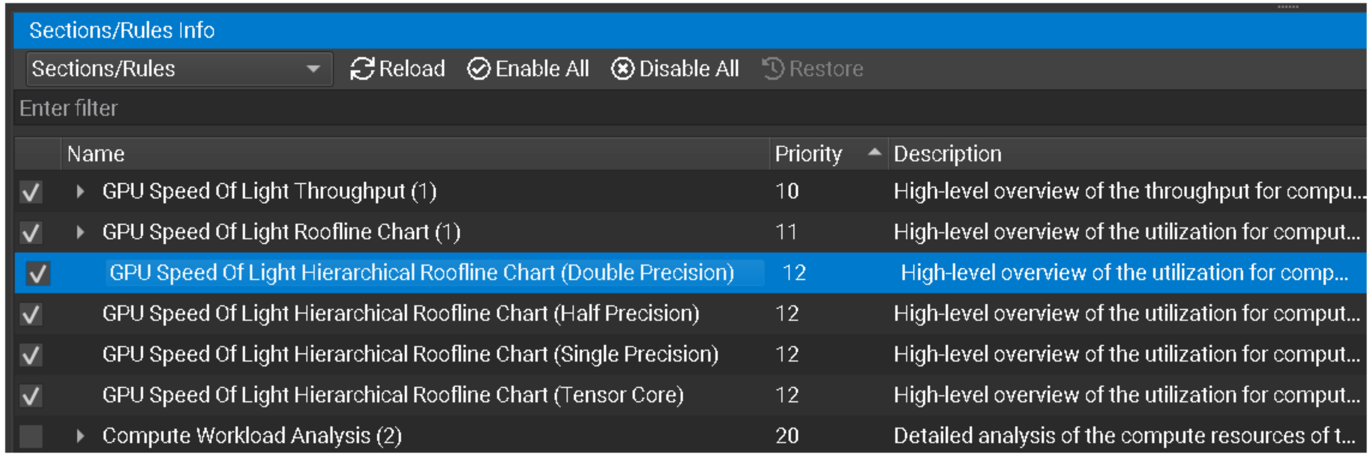
Metrics for peak performance ratios



Hierarchical Roofline



- Visualize multiple levels of the memory hierarchy
- Identify bottlenecks caused by memory limitations
- Determine how modifying algorithms may (or may not) impact performance


Sections/Rules Info

Sections/Rules | Reload | Enable All | Disable All | Restore

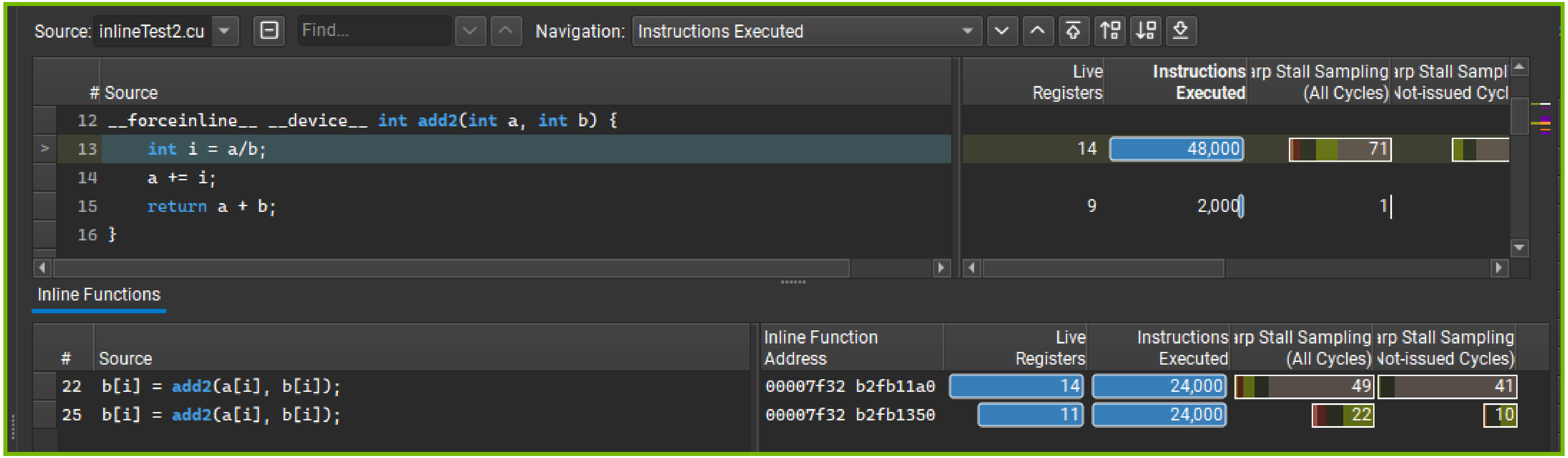
Enter filter

	Name	Priority	Description
✓	GPU Speed Of Light Throughput (1)	10	High-level overview of the throughput for compu...
✓	GPU Speed Of Light Roofline Chart (1)	11	High-level overview of the utilization for comput...
✓	GPU Speed Of Light Hierarchical Roofline Chart (Double Precision)	12	High-level overview of the utilization for comp...
✓	GPU Speed Of Light Hierarchical Roofline Chart (Half Precision)	12	High-level overview of the utilization for comput...
✓	GPU Speed Of Light Hierarchical Roofline Chart (Single Precision)	12	High-level overview of the utilization for comput...
✓	GPU Speed Of Light Hierarchical Roofline Chart (Tensor Core)	12	High-level overview of the utilization for comput...
	Compute Workload Analysis (2)	20	Detailed analysis of the compute resources of t...

Inline Function Table

Shipped with Nsight Compute 2023.1

- Metrics can be analyzed per inline site or aggregated for the entire function
- Use compiler `--lineinfo` flag to generate symbols
- Identify specific underperforming calls and outliers



Kernel Profiles with Nsight Compute

```
$ ncu –k mykernel –o report ./myapp.exe
```

- Generated file: report.ncu-rep
  - Open for viewing in the Nsight Compute UI

- (Without the –k option, Nsight Compute with profile everything and take a long time)

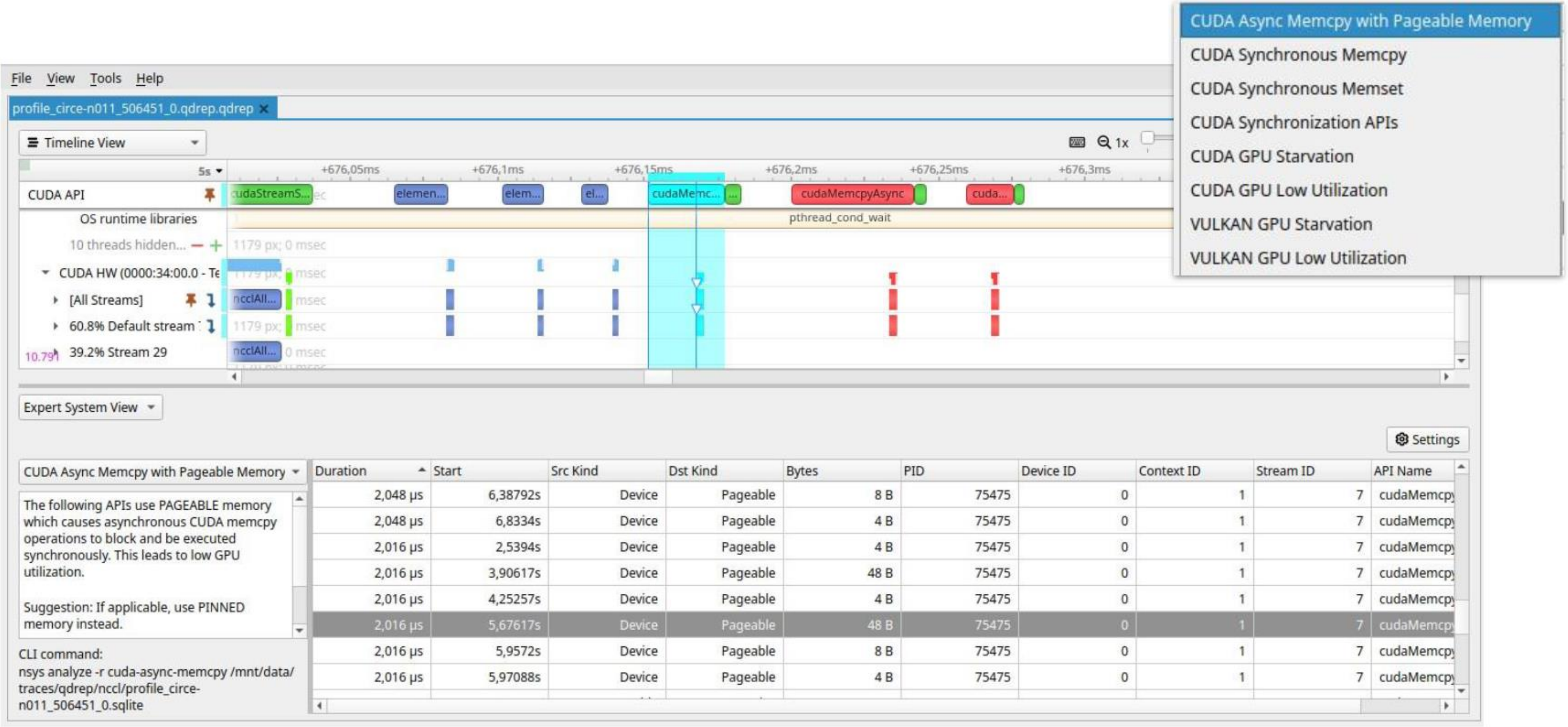
- ncu needs access to the GPU performance counters, which [may require a parameter change to the nvidia kernel module] (`options nvidia NVreg_RestrictProfilingToAdminUsers=0`)

Let’s Get Profiling!

Some Advanced Features We Didn’t See Today

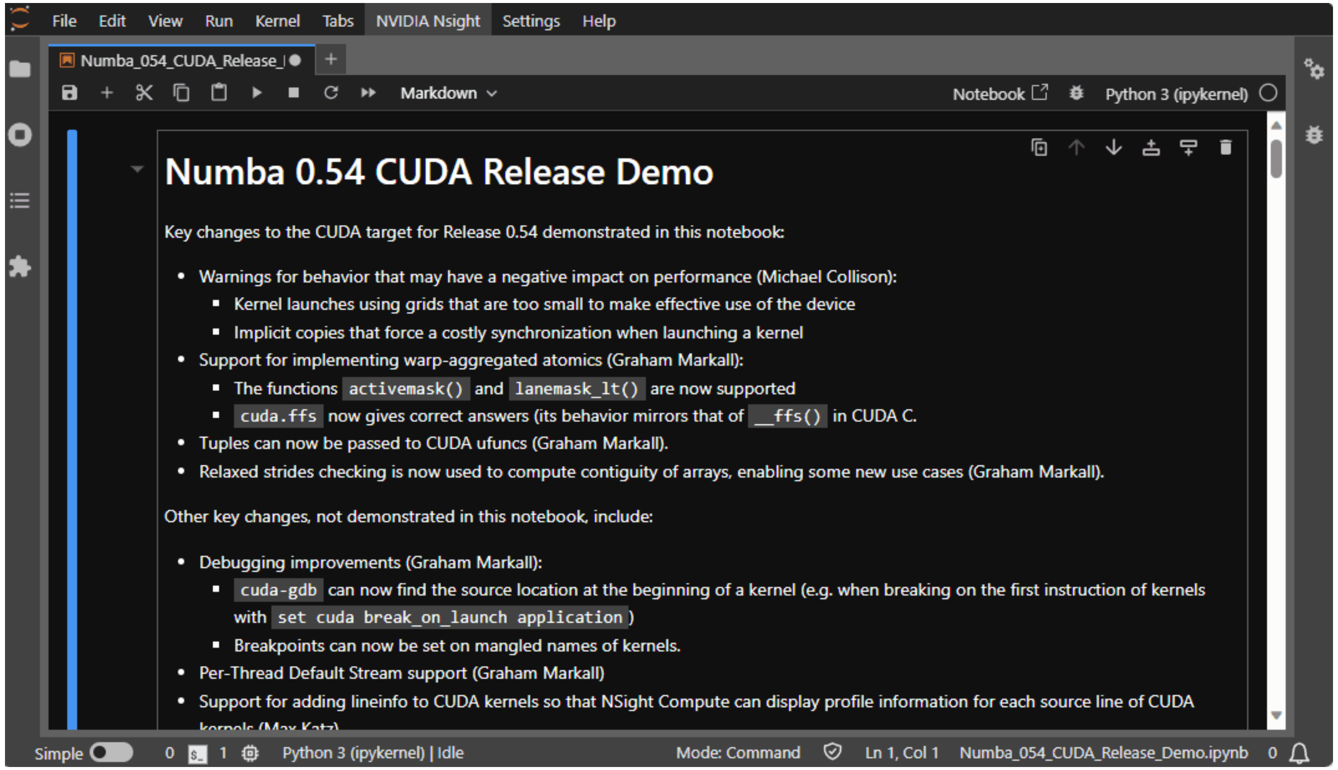
Expert Systems & Statistics

Built-in data analytics with advice



JupyterLab Integration Updates

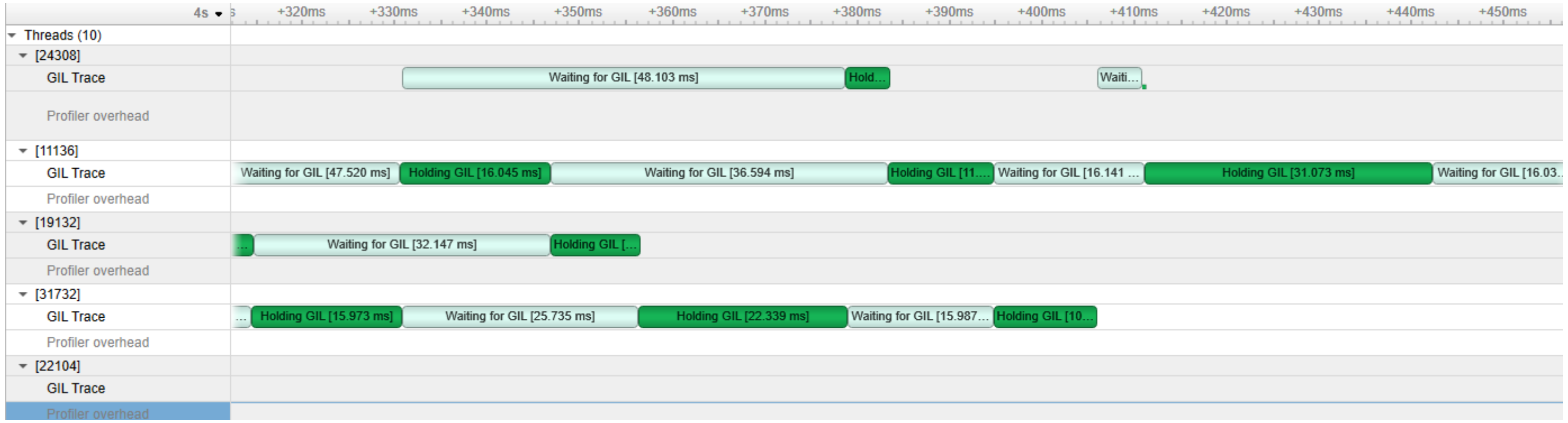
- Extension to JupyterLab
- Profile individual Jupyter cells
- Text-based results can be viewed directly in Jupyter
- Launch **new** remote GUI streaming container directly in JupyterLab
  - Servers without X, Windowing Manager, …
  - Container with X, WM, & WebRTC server
  - Dockerfile inside Nsight Systems Installer
- See it in action:
  - DLIT61667: Profilers, Python, and Performance: Nsight Tools for Optimizing Modern CUDA Workloads

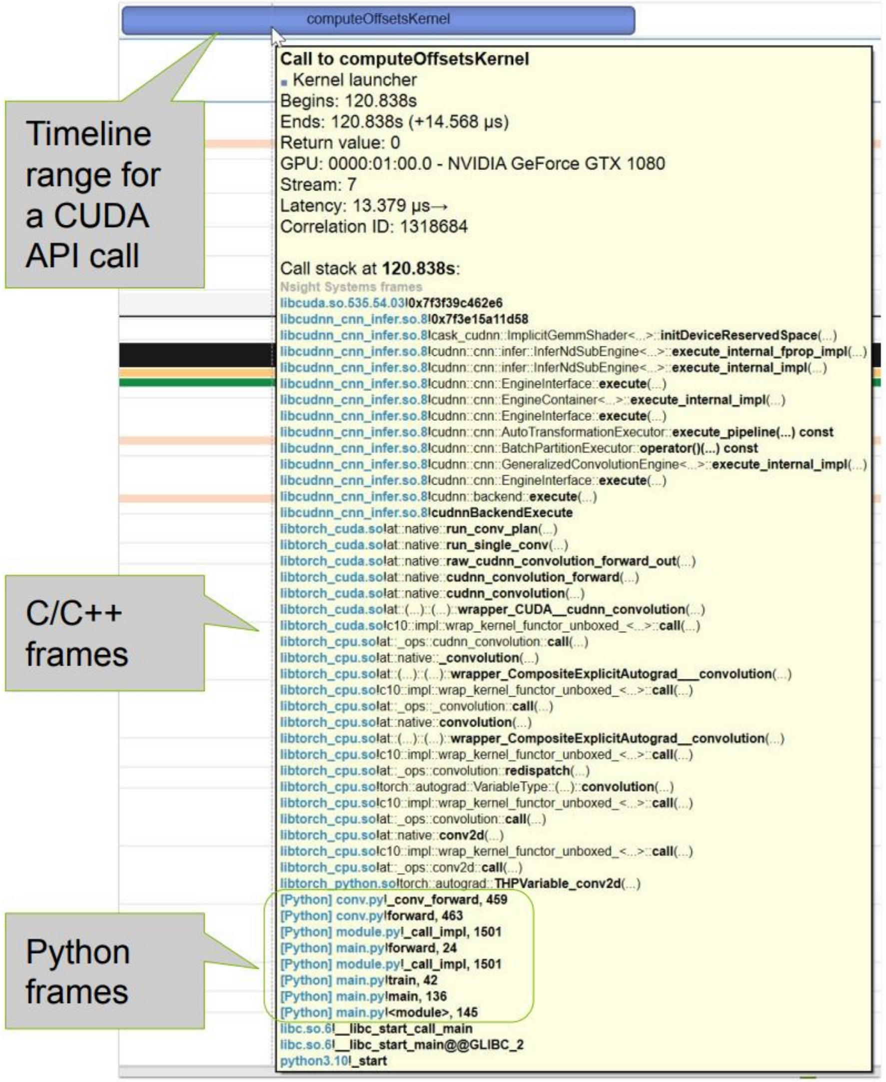


NVIDIA

Python Profiling Updates

- Python Call Stacks Samples and CUDA API Backtrace
  - Identify where you are and how you got there
- Global Interpreter Lock (GIL) trace
  - Common performance limiter in Python
- See annotated code ranges built into in popular frameworks and libraries such as:
  - RAPIDS, Spark, CV-CUDA, and more…

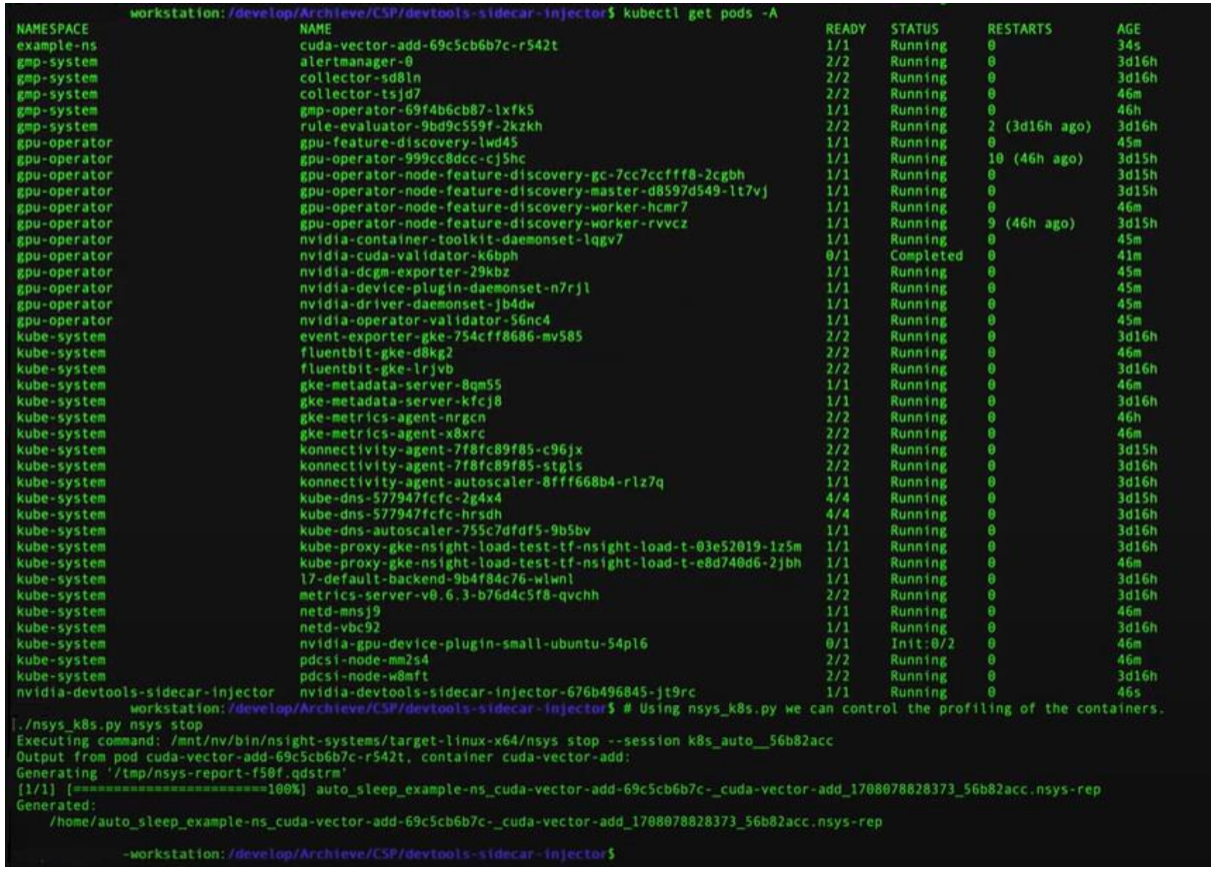


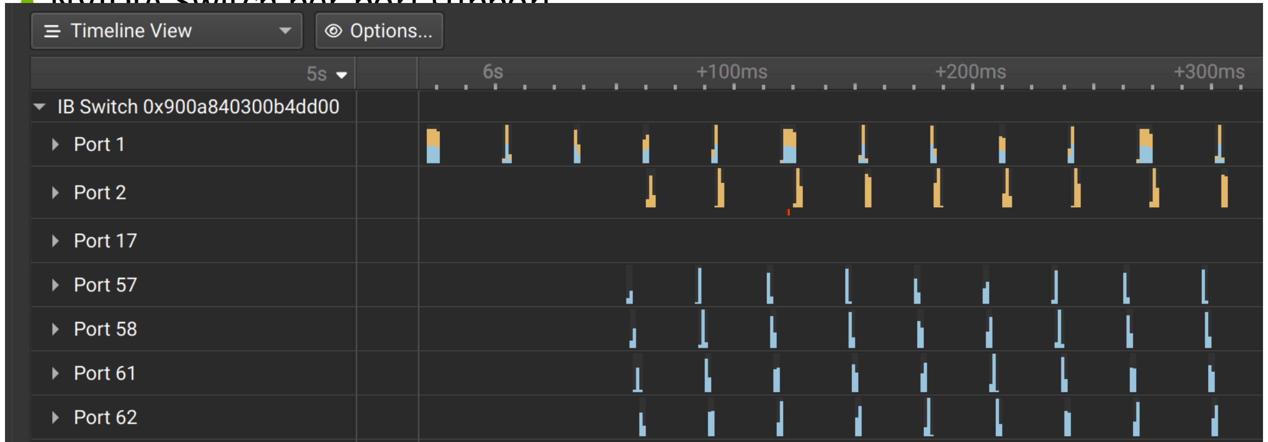


NVIDIA

Cluster and Recipe Framework Improvements

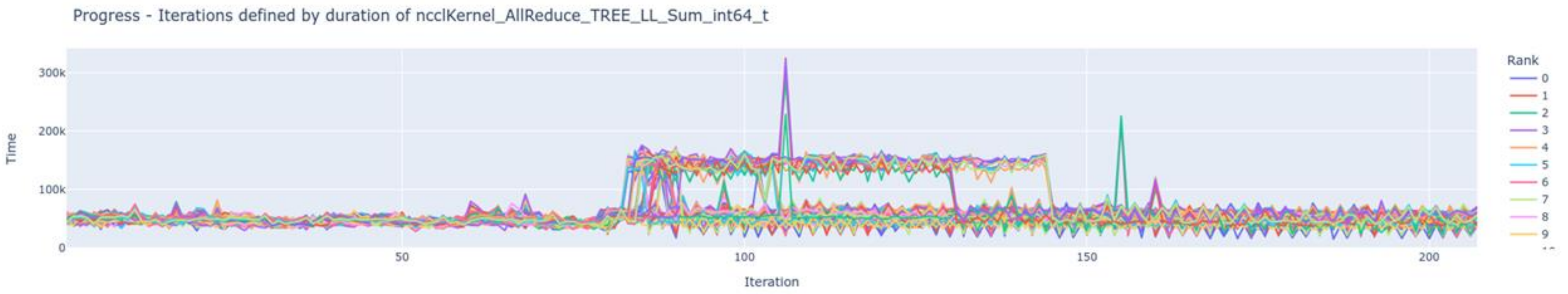
- Nsight Systems enhanced support for Kubernetes
- Nsight Systems analysis framework:
  - User programmable and predefined recipes to:
    - Process and analyze complex and large reports or collection of reports
    - Understand how compute cold-spots relate to communications
    - Generate multi-node heatmaps to show :
      - InfiniBand congestion
      - InfiniBand, Ethernet, and NVLink throughputs
      - Overlapped compute and networking
- NVIDIA Switch per port support

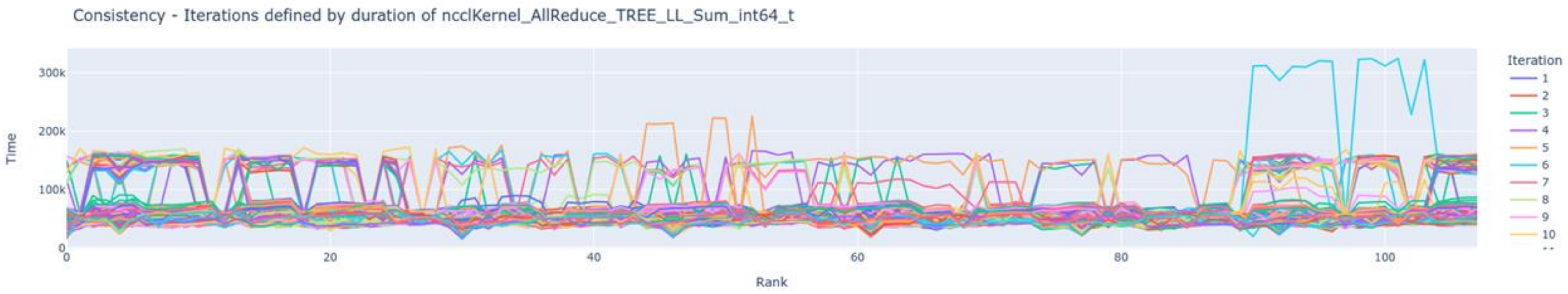




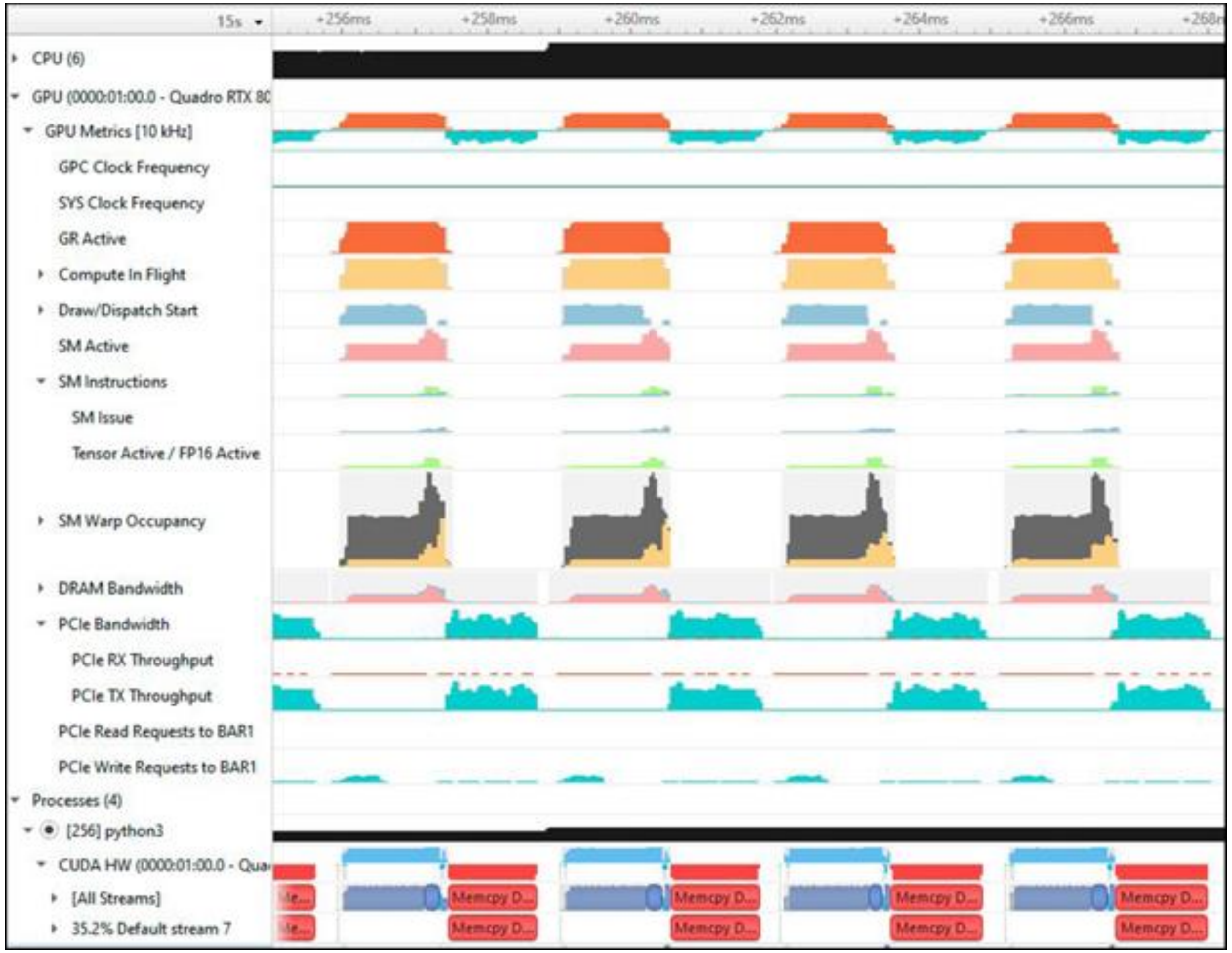
MULTI-REPORT ANALYSIS

Ex: Jupyter notebook output of NCCL barrier time across 128 GPUs
https://www.nvidia.com/en-us/on-demand/session/gtcspring23-s51421/





GPU Metrics Sampling



- Useful GPU utilization metrics, but no kernel names / correlation

Interpreting GPU Sampling Metrics

- GR Activity -> GPU is doing work
  - SM, NVENC, NVDEC, Graphics
- SM Activity -> Utilizing width of GPU
  - If low, modify kernel grid dimension or increase batch size
- SM Instruction Issued -> GPU is performing lots of instructions
  - Stalled waiting on memory?
  - Not enough warps to cover memory latency? Issue larger kernel block dimensions.
- SM Instructions tensor activity -> Tensor core utilization
  - Performance up, SM instructions can drop (depending on arch)
  - Can be limited by shared memory, waiting for loads
- Note: Requires disabling ***DCGM*** and DL built-in profilers

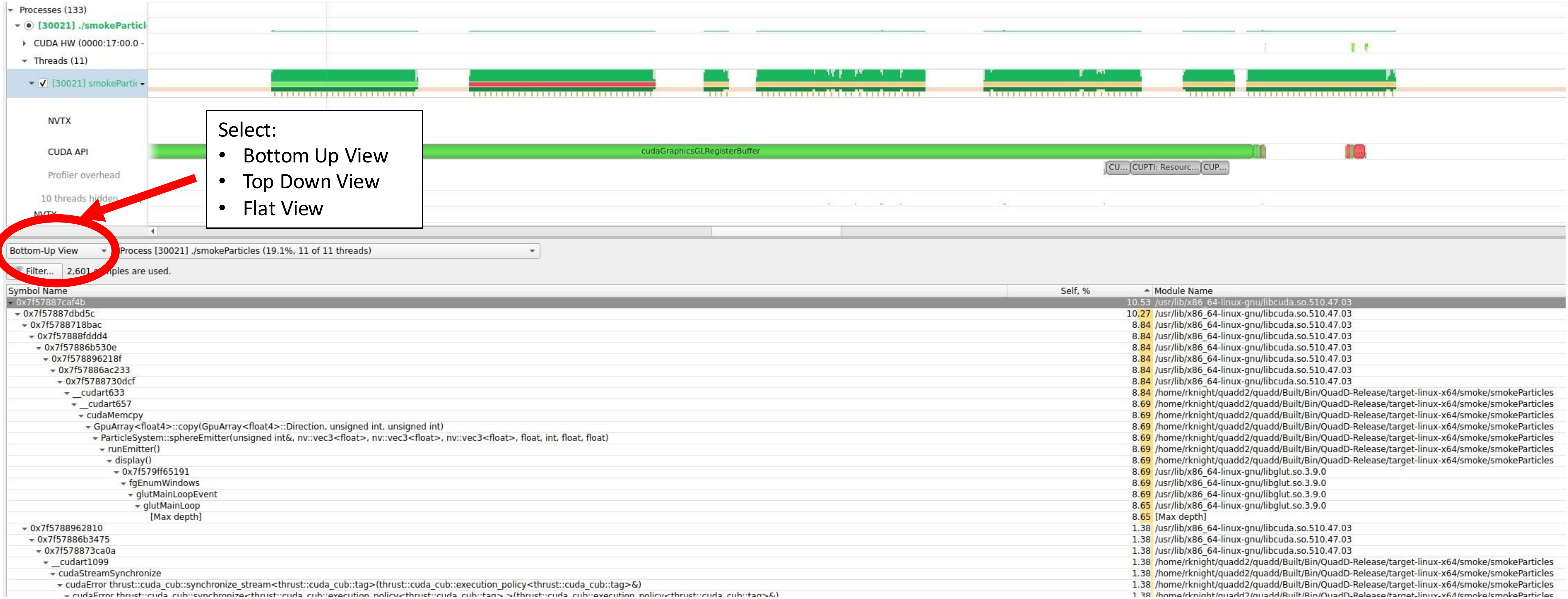
CPU IP/Backtrace Sampling

Placing a mouse cursor on a specific sample (orange-yellow tick mark) causes backtrace tooltip to be displayed

Stack depth is defined by callstack mechanism selected
See nsys CLI --`backtrace` switch

- Intel Last Branch Register (fast, limited depth)
- Frame Pointers (fastest, probably requires recompile)
- DWARF (best depth, most overhead – default on ARM systems)
  - Also, see nsys CLI --samples-per-backtrace switch

CPU IP/Backtrace Sampling Summary / Histogram



The summary enables CPU hotspot identification by identifying hot paths.

Top-Down view:

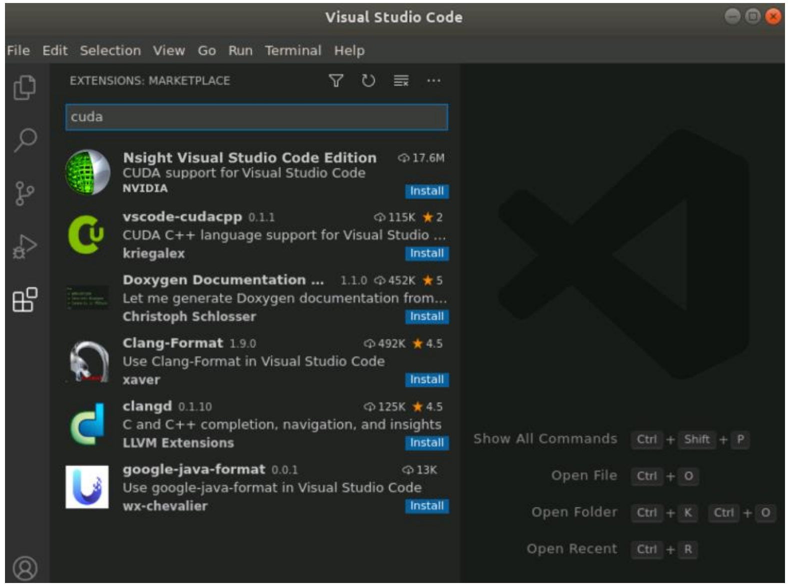
The **Self** column shows the percentage of IP/backtrace samples that were collected while a specific function was executing.

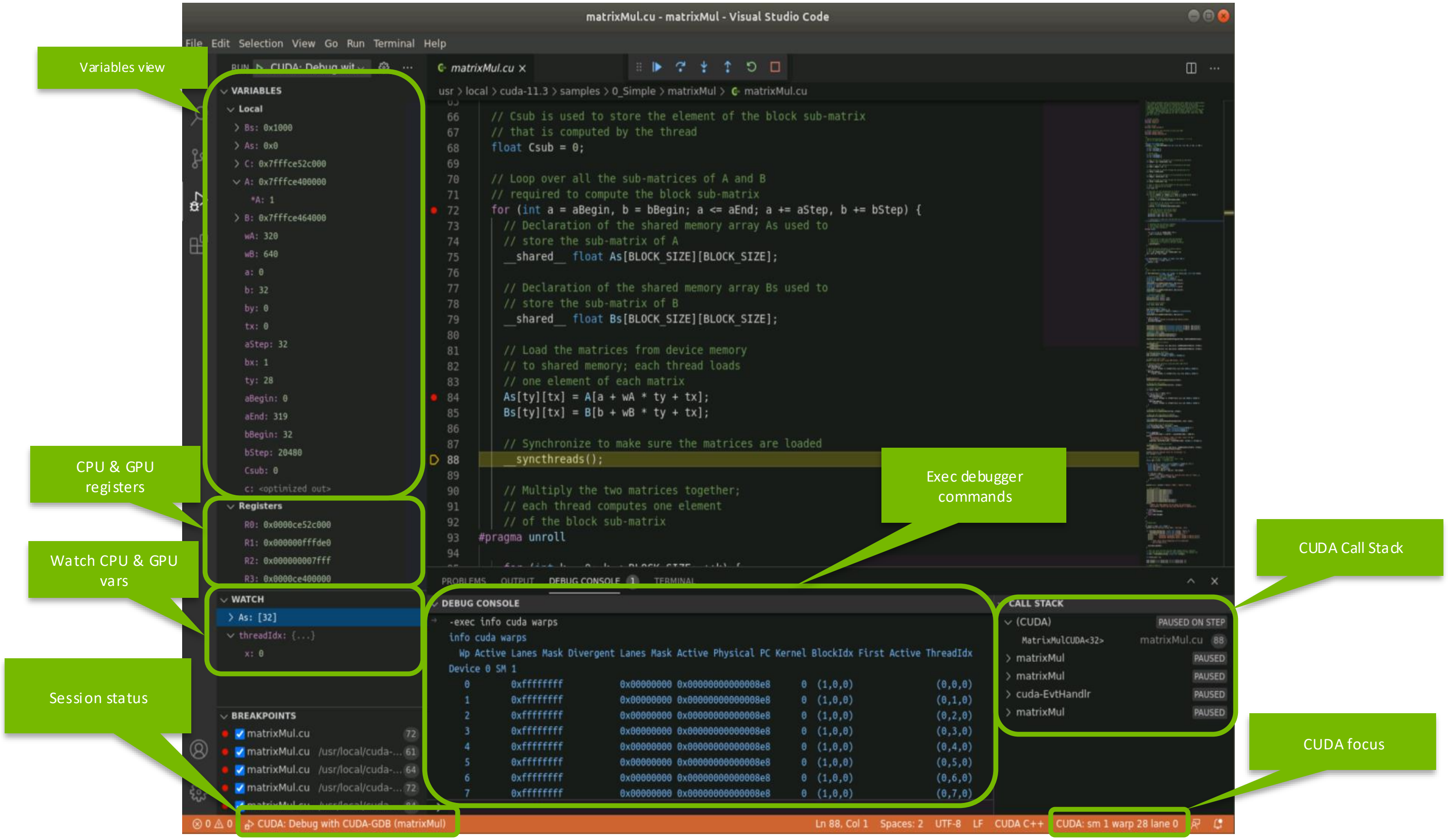
The Total column shows the percentage of IP/backtrace samples that were collected while that function and all of its children functions were executing.

Nsight Visual Studio Code Edition

Visual Studio Code extensions that provides:

- CUDA code syntax highlighting
- CUDA code completion
- Build warning/errors
- Debug CPU & GPU code
- Remote connection support via SSH
- Available on the VS Code Marketplace now!





https://developer.nvidia.com/nsight-visual-studio-code-edition

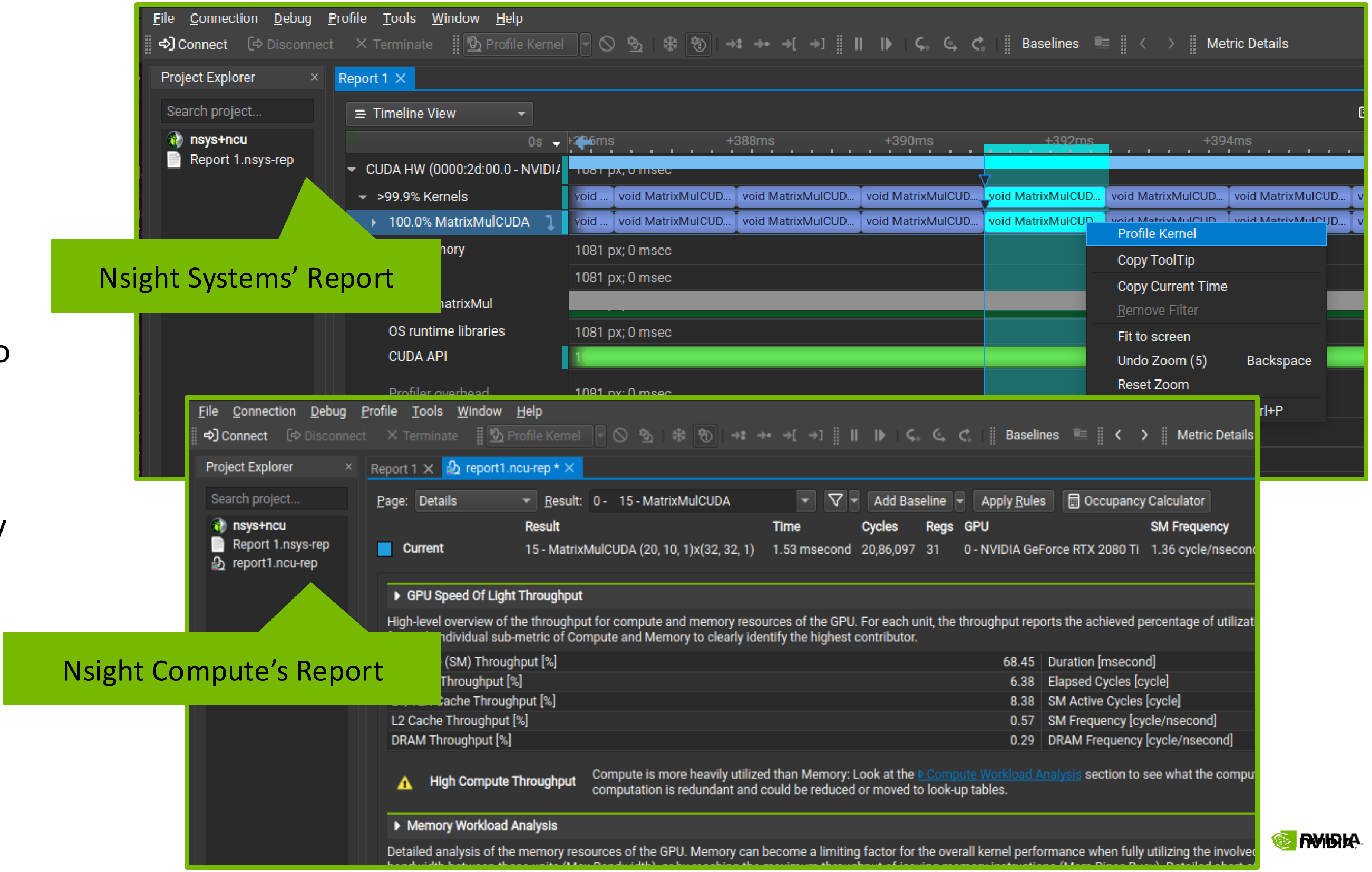
Integrating NSIGHT SYSTEMS basic trace INTO nsight COMPUTE

- Use “System Trace” activity in the connection dialog

- Identify long kernels or compute-bound bottlenecks

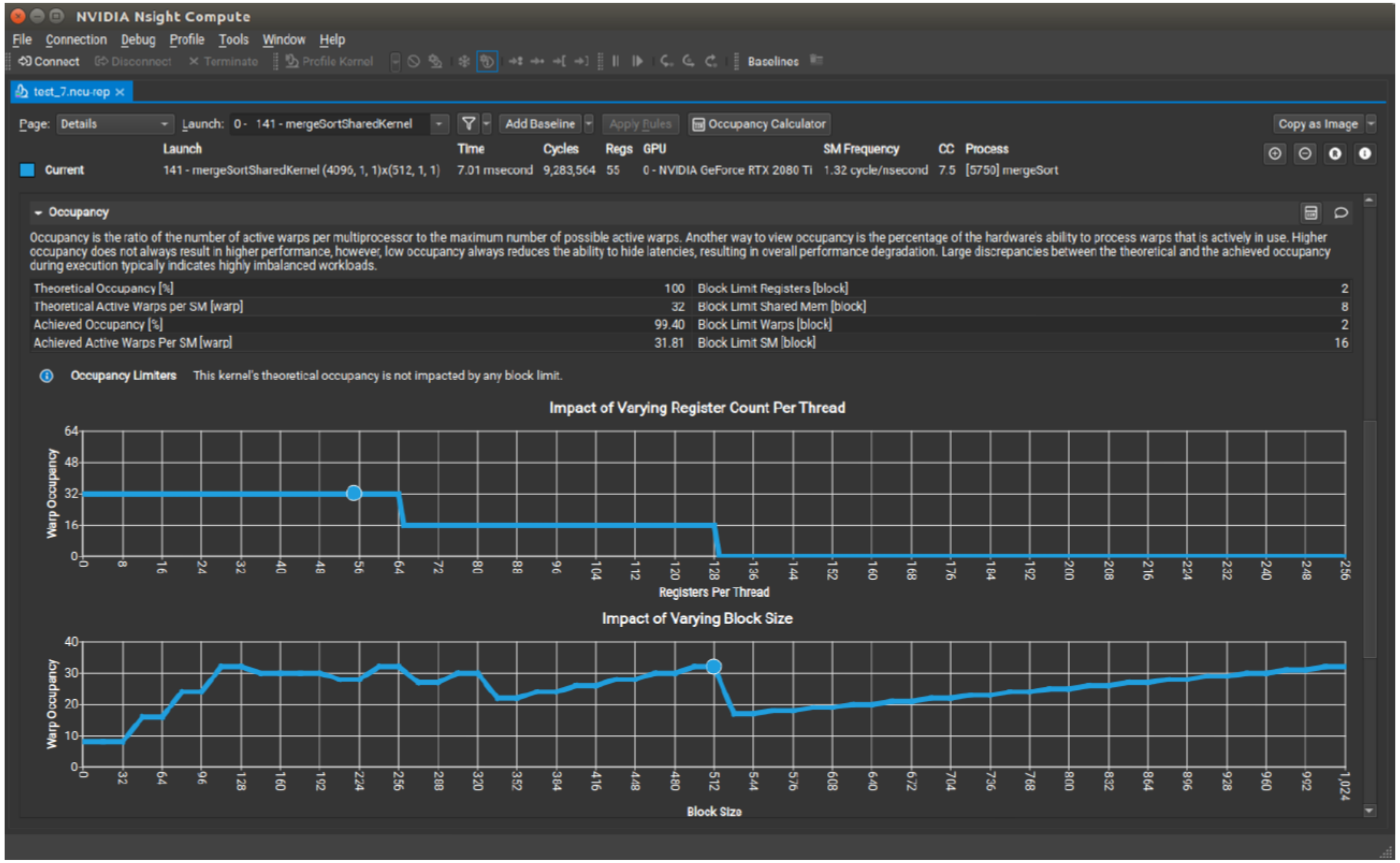
- Right-click kernel in timeline to quickly launch profile

- Nsight Compute automatically filters to selected kernel



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

Additional Links

- Nsight Systems User Guide
    - https://docs.nvidia.com/nsight-systems/UserGuide/index.html
- Nsight Compute
    - User Guide: https://docs.nvidia.com/nsight-compute/NsightCompute/index.html
    - Profiling Guide: https://docs.nvidia.com/nsight-compute/ProfilingGuide/index.html
- Nsight Tutorials
    - https://developer.nvidia.com/tools-tutorials
    - NVIDIA/nsight-training: Training material for Nsight developer tools (github.com)
- CUDA GDB User Guide
    - https://docs.nvidia.com/cuda/cuda-gdb/index.html
- Compute Sanitizer Documentation
    - https://docs.nvidia.com/compute-sanitizer/ComputeSanitizer/index.html
- Example we walked through
    - openhackathons-org/HPC_Profiler: Profiling with NVIDIA Nsight Tools Bootcamp (github.com)

Questions - Contact Me!

- **Email**
  - jdursi@nvidia.com

- **Alliance Federation Staff Office Hours**
  - Will try this for a few months
  - First Tuesday of the month, 3:30 ET (4:30 AT, 2:30 CT, 1:30 MT, 12:30 PT)
  - https://meet.google.com/fcd-yofg-bbk
  - 5 Nov, 3 Dec, 7 Jan
  - Bring any questions you have, I’ll answer what I can and get back to you on what I can’t.

What should we cover next?

- **Advanced Nsight**
  - Multinode, Reports analysis
  - Jupyter-Notebook, VS Code…
- **Profiling tools for Pytorch**
  - Torch profiler, using nsys with torch, common optimizations
- **Other Dev Tools**
  - Compute-sanitizer, cuda-gdb
- **MIG & MPS**
- **Scientific python options**
  - CuPy, Numba, RAPIDS, cuNumeric, JAX
- **AI for PDEs – Modulus**
- **CUDA-Q and cuQuantum**
- **Federated Learning…**

Free Training, Leading to Certification as DLI Instructors

- **First Two Weeks of November**
- **Trainings Listed here:**
  - https://events.nvidia.com/faculty-development-virtual-workshops-higher-ed
- **Aimed at faculty, but email me if interested**

NVIDIA

Questions to Guide Profile Analysis

- What is hot?
  - Can I make it faster, shrink the problem, parallelize it? (Not always…)
  - Reduce precision?

- What is cold?
  - Fill the gaps in the timeline
  - Can I take advantage of unused hardware?
  - Unnecessary dependencies or syncs?

General Optimization Tips

- Using tensor cores?
  - Minimize conversions/transposes
- Increase grid and batch size to utilize GPUs width
- Conventional parallelism – more worker threads!
- Parallel pipelining
  - No data dependency? Parallelize!
  - Prefetch next batch/iteration during computation
- Can I reorder sooner?

General Optimization Tips

- Fuse tiny kernels, copies, memsets.
  - Check out CUDA Graphs
- Overlap/oversubscribe with MPS
- Multi-buffering
  - Don’t make everyone wait on the same piece of memory
  - Double, triple buffer
- Avoid moving data back to the CPU
  - Pre-allocate and recycle!
- Minimize managed memory page faults
  - Prefetch!

- Pytorch
  - DNN Layer annotations are disabled by default
  - ++ ”*with torch.autograd.profiler.emit_nvtx():*”
  - Manually with *torch.cuda.nvtx.range_(push/pop)*
  - TensorRT backend is already annotated

- Tensorflow
  - Annotated by default with NVTX in NVIDIA TF containers
  - TF_DISABLE_NVTX_RANGES=1 to disable for production

NVIDIA