
NVIDIA
GPU PROFILING

AGENDA

The need for GPU Profiling

- Why Profile Your Code
- When to Profile Your Code
- Are there opportunities to improve performance?

GPU Profiling Tools

Nsight Suite of Profilers

- Nsight Systems
  - NVTX: NVIDIA Tools Extension API
- Nsight Compute

Profiling Best Practices: Tips and Techniques for Efficient Workloads

---

Resources

WHY PROFILE YOUR CODE?

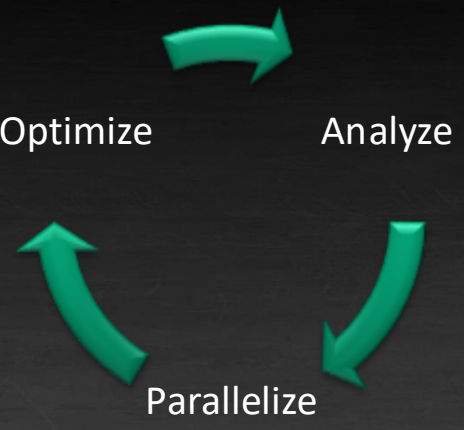
When is this useful?

- Code profiling gives you deeper insights into how your code performs at runtime.
- Allows you to focus your efforts on the problem areas of your application if optimizations are needed.
- Optimization does not always refer to execution time or latency, you can also optimize memory usage or throughput.
- Profiling can help you find errors or easy optimization opportunities.
- Often the way code executes on a device doesn't match what we would expect based on reading the code – profiling can help you figure out why.

WHEN TO PROFILE YOUR CODE

Is it time to profile yet?

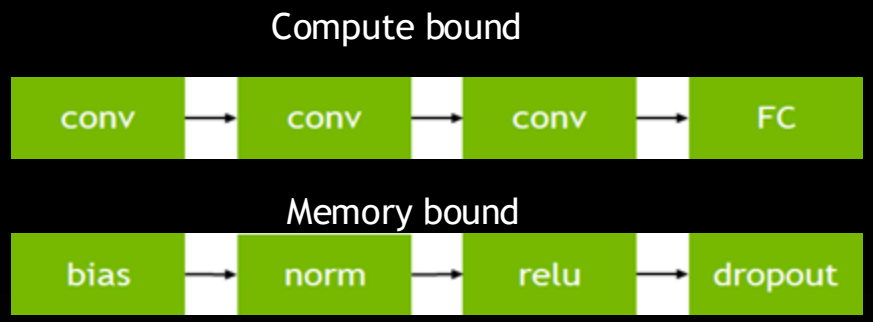
- Focus your efforts initially on getting a readable, maintainable, and correct version of your program finished.
    - Correctness beats everything else, nobody cares how fast or memory efficient your program is if it is unstable or gives the wrong result.
- Avoid premature optimization, bottlenecks can typically be attributed to a small fraction (<10%) of the overall code. Humans often aren't great at predicting which parts of the code that will be.
- Make sure your code is modular, break tasks into small easily readable functions with descriptive names.
- Decide what to optimize, it may not be latency/execution time. It might be more important to optimize for memory usage or cost/throughput.
- Follow the Development Cycle:
    - Analyze
    - Parallelize
    - Optimize



FINDING PERFORMANCE OPPORTUNITIES

Models can be data bound by the data pipeline, compute or memory

- GPU utilization as it relates to model code
  - Time being spent on ops in every iteration
  - Time spent on GPU/CPU
  - Data types used for operations
- Bottlenecks could be attributed to
  - Input data pipeline: data loading, preprocessing etc
  - Compute (math) limited operations
  - Memory limited operations
  - Other aspects such as overall system tuning
- Categories of operations in DNNs based on bottleneck
  - Element wise: ReLU, memory bound
  - Reduction: Batch norm, memory bound
  - Dot product: Convolution, math bound



Compute heavy ops see speed-ups from GPUs

DEEP LEARNING OPTIMIZATION

Performance Analysis at System and DNN Level & Visualization

System Level Tuning

- System Tuning
  - Thread Synchronization, Multi GPU and node communication
  - Memory management & Kernel profiling
- Leveraging/Optimizing Hardware
- Input Pipeline Optimization
- Many others….

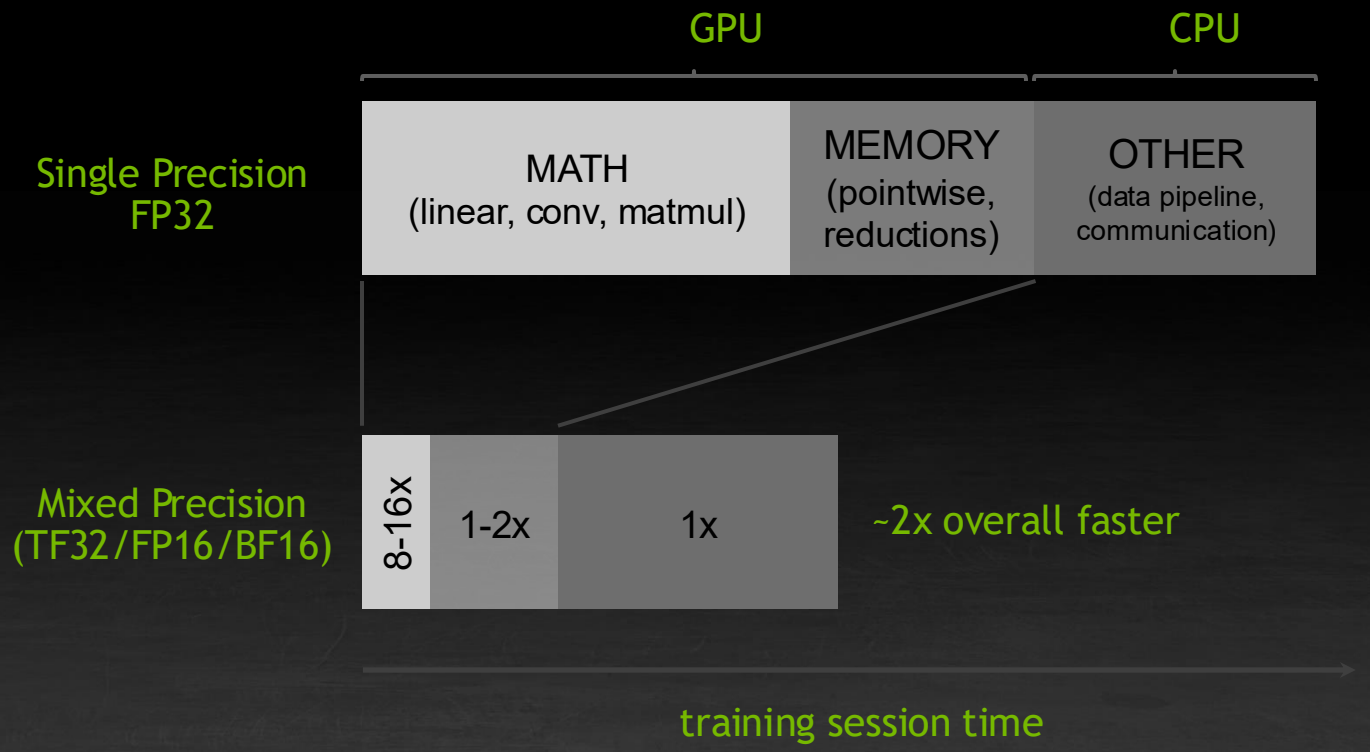
DNN Level Tuning

- Algorithm Techniques & Data Representations
- Pruning
- Calibration
- Quantization
- Many others….

LIMITS OF PERFORMANCE OPTIMIZATION

End-to-end perf depends on training composition

- **Amdahl's Law:**
If you speed up part of your training session (GPU work), then the remaining parts (CPU work) limit your overall performance



DL PROFILING NEEDS OF DIFFERENT PERSONAS

Researchers

Fast development of best performant models for research, challenge and domains

Data Scientists & Applied Researchers

Reduce Training time, focus on data, develop and apply the best models for the applications

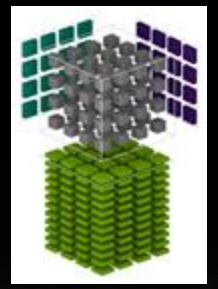
Sysadmins & DevOps

Optimized utilization and uptime, monitor GPU workloads, leverage hardware

TOOLS & TECHNOLOGIES

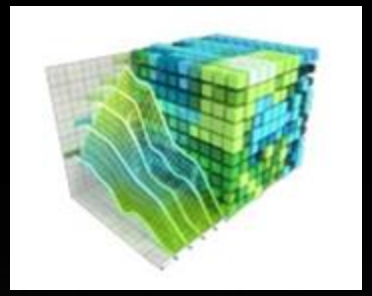
Profiling and monitoring tools for all users

Researchers & Developers



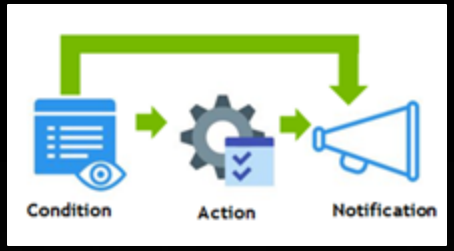
NVTX
Nsight Systems
Nsight Compute

Data Scientists & Applied Researchers



Nsight Systems with NVTX

Sysadmins & DevOps



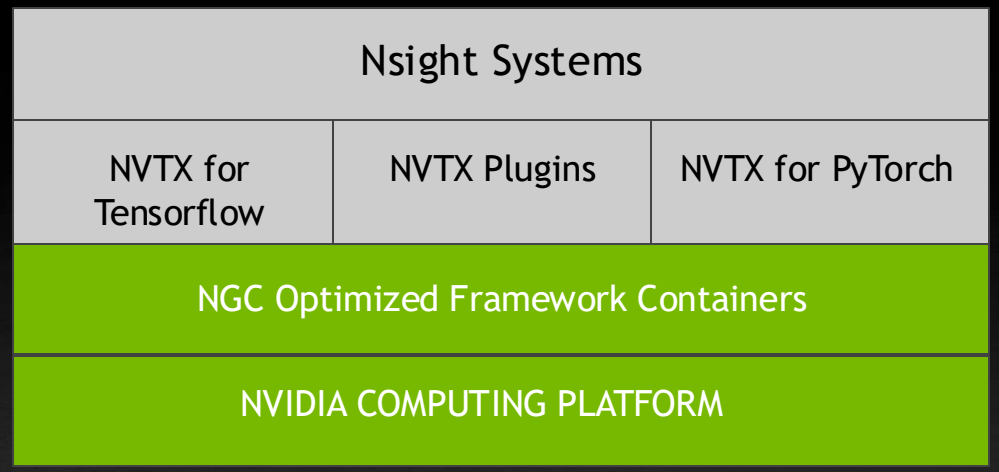
Data Center Monitoring Tools
DCGM, NVML

Skills in Algorithms | **Skills in Domains & Applications** | **Skills in Systems**

NVIDIA PROFILING STACK

The layers that make the cake

- **Nsight Systems** and **Nsight Compute** have been built using CUDA Profiling Tools Interface(CUPTI)
- **NVTX** NVIDIA Tools Extension Library is a way to annotate source code with markers
  - NVTX markers are used to annotate and focus on sections of code important to the user
  - TensorFlow optimized by NVIDIA (aka nvidia-tensorflow) contain support for NVTX markers.
- NVTX plugins are python bindings for users to add markers easily




NSIGHT SUITE OF PROFILERS

NSIGHT PRODUCT FAMILY

Standalone Performance Tools, IDE Plugins

Standalone Performance Tools

Nsight Systems System wide tracing, application algorithm tuning

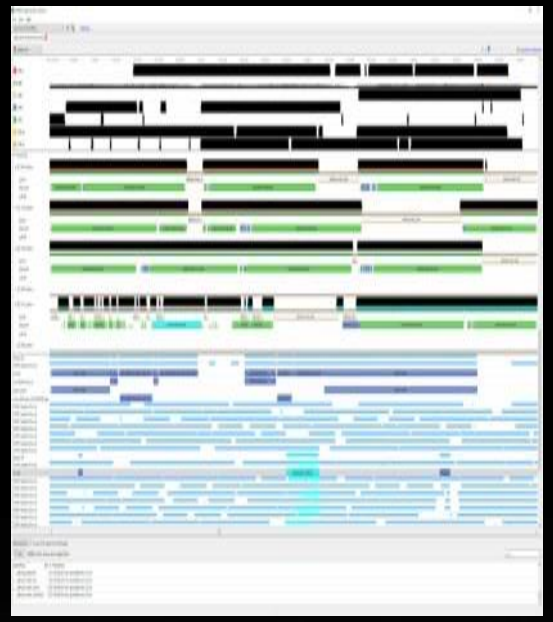
Nsight Compute Debug/Optimize specific CUDA kernels

Nsight Graphics Debug/Optimize specific graphics API and Shaders

IDE Plugins

Nsight Visual Studio/Eclipse Edition editor, debugger, performance analysis

NSIGHT PRODUCT FAMILY



Nsight Systems

System-wide application algorithm tuning

Nsight Compute

CUDA API Debugging & Kernel Profiling

Nsight Graphics

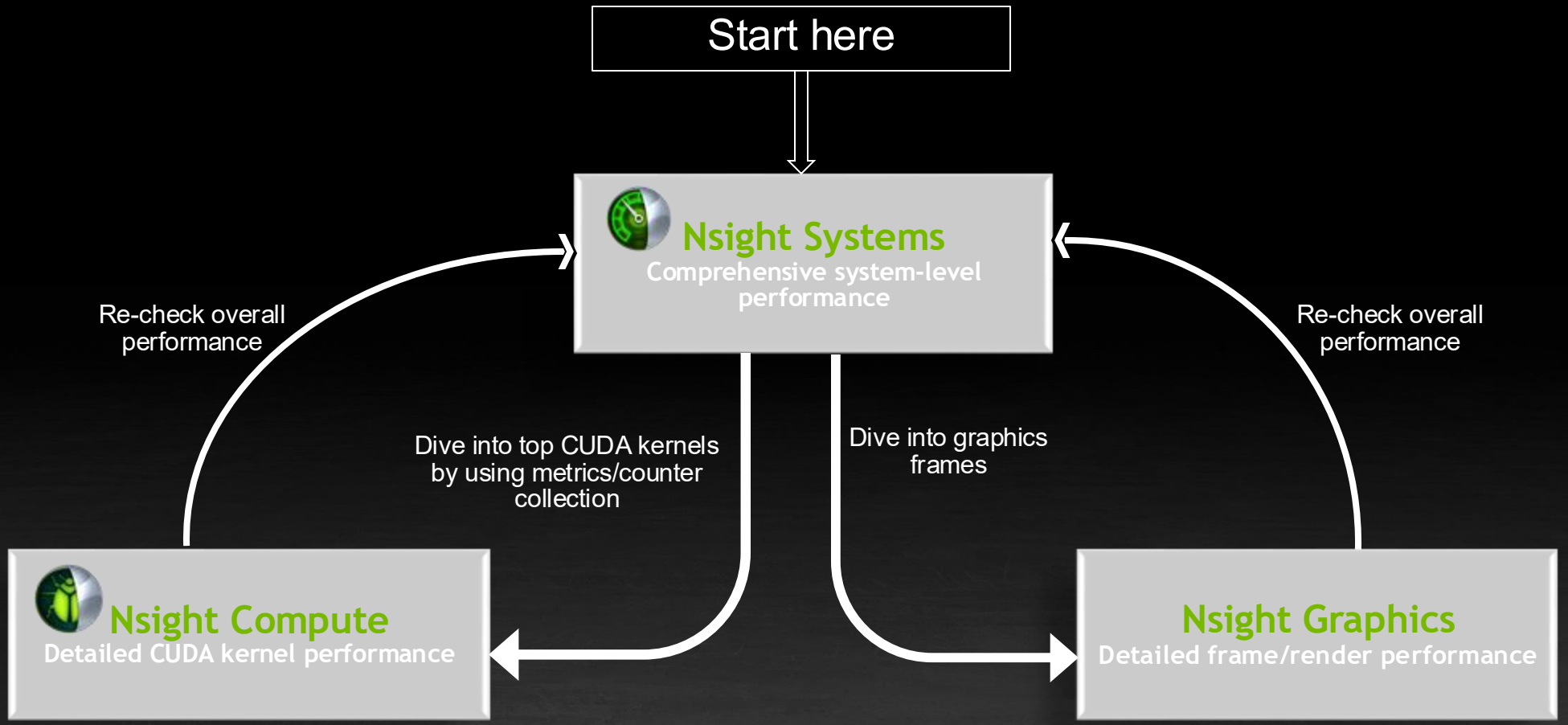
Graphics Debugging & Profiling

IDE Plugins

Nsight Eclipse Edition/Visual Studio (Editor, Debugger)

NSIGHT (STANDALONE)TOOLS WORKFLOW

Nsight Systems, Nsight Compute and Nsight Graphics



NSIGHT SYSTEMS

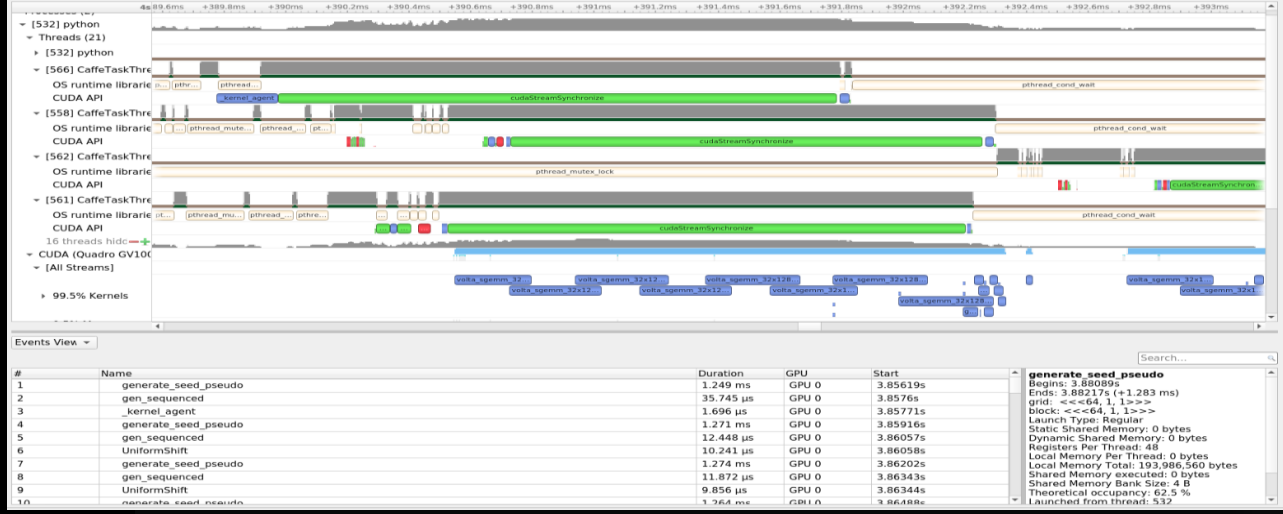
System Profiler

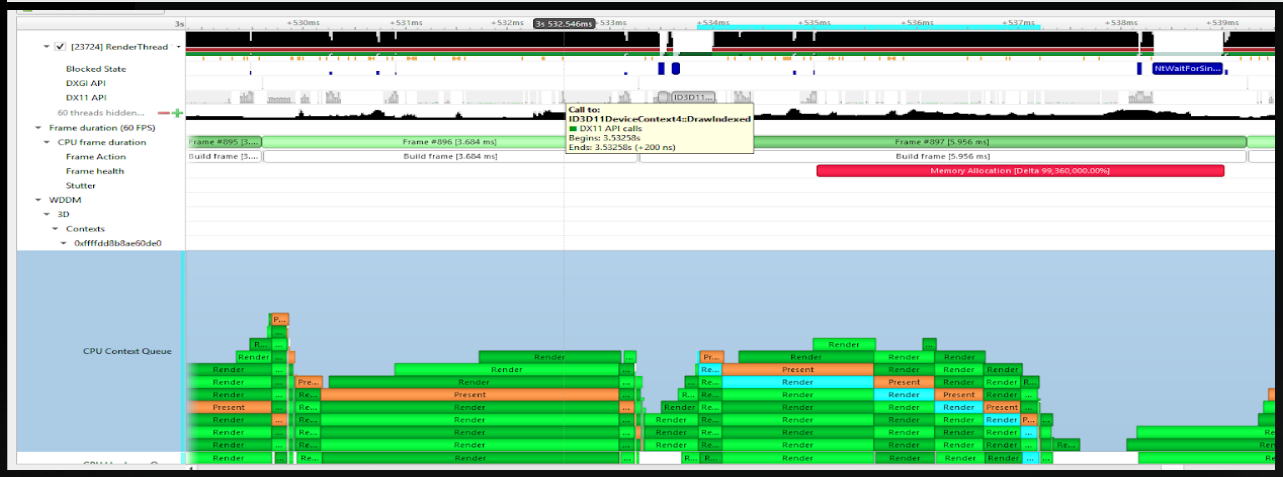
Key Features:

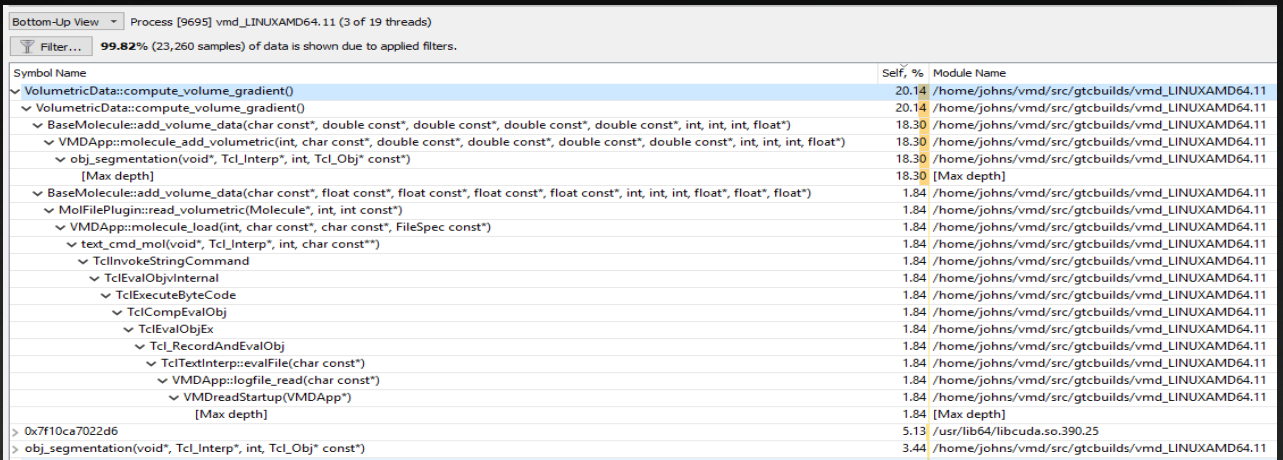
- System-wide application algorithm tuning
  - Multi-process tree support
- Locate optimization opportunities
  - Visualize millions of events on a very fast GUI timeline
  - Or gaps of unused CPU and GPU time
- Balance your workload across multiple CPUs and GPUs
  - CPU algorithms, utilization and thread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

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

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







Additionally:

Trace:
- TensorRT
- Direct3D11,12,DXR
- Vulkan
- OpenGL
- OpenACC
- MPI
- OpenMP
- Ftrace
- ETW
- WDDM
- GPU Context Switch

Export:
- SQLite
- HDF5
- JSON

Architectures:
- X86_64
- Power
- Arm SBSA
- Tegra

NVIDIA

NVTX: NVIDIA TOOLS EXTENSION API

An Annotation Tool for Profiling Code in Python and C/C++

- Code Annotation tool to mark functions or chunks of code.
- NVTX and Nsight Systems together are powerful tools for visualizing CPU and GPU performance
- Workflow:
  - Use decorators *@nvtx.annotate()* or context manager *with nvtx.annotate(..):* to mark code to be measured.
  - Run it with Nsight Systems.
    - *$ nsys profile -t nvtx,osrt --force-overwrite=true --stats=true --output=quickstart python nvtx-quickstart.py*
    - *$ nsys –help or nsys [specific command] --help*
  - Qdrep file and sqlite database generated to be viewed on Nsight Systems UI.

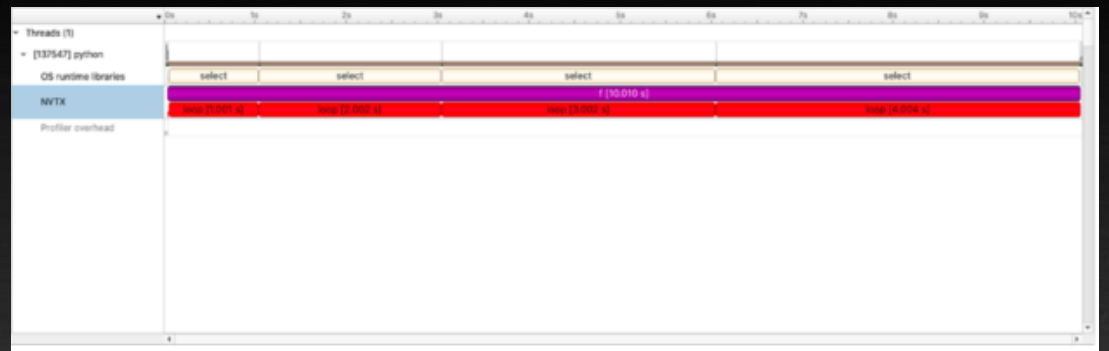


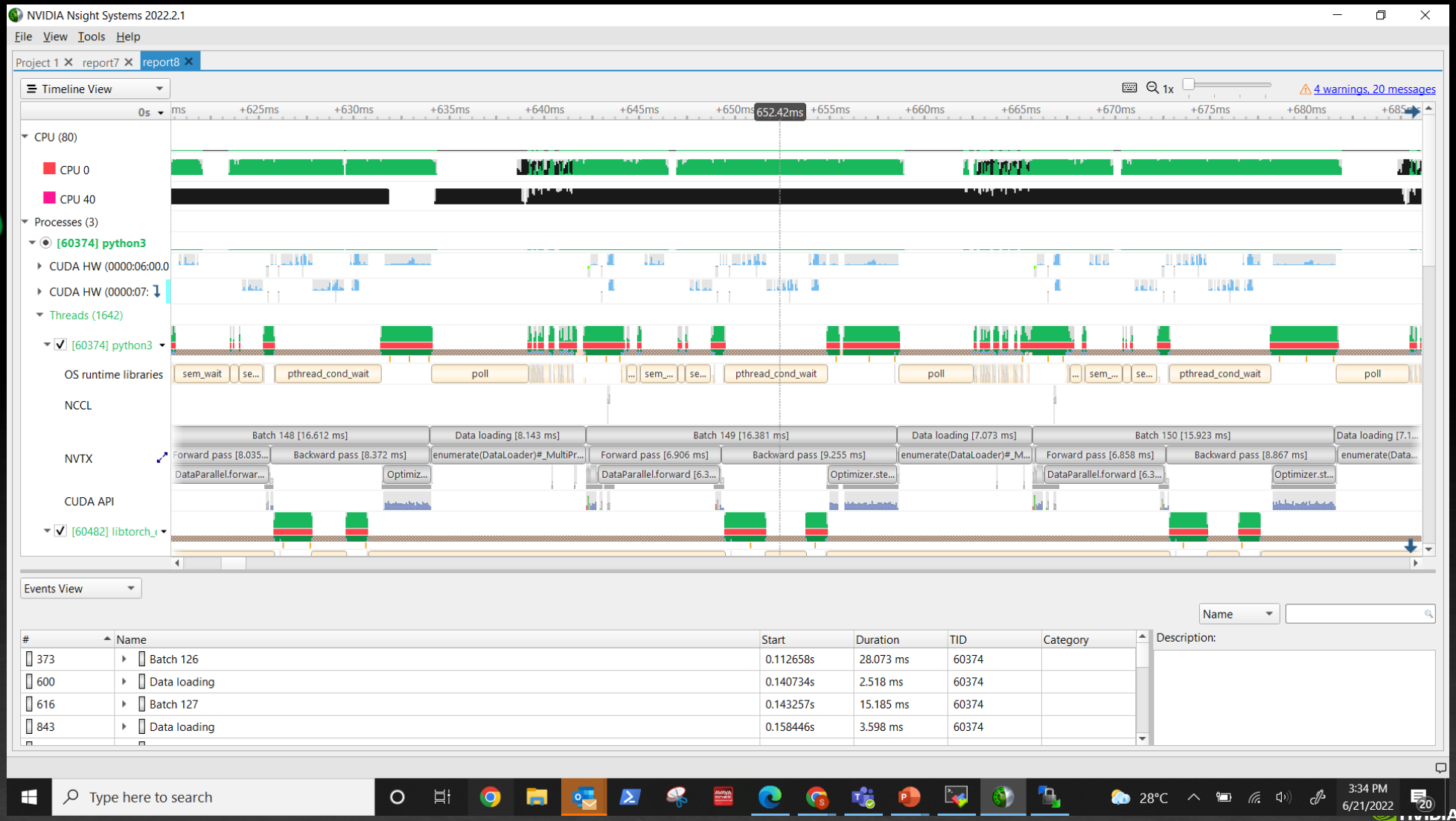
Figure 1: NVIDIA Nsight Systems timeline visualization of the workflow

```
import time
import nvtx


@nvtx.annotate("f()", color="purple")
def f():
    for i in range(5):
        with nvtx.annotate("loop", color="red"):
            time.sleep(i)

f()
```

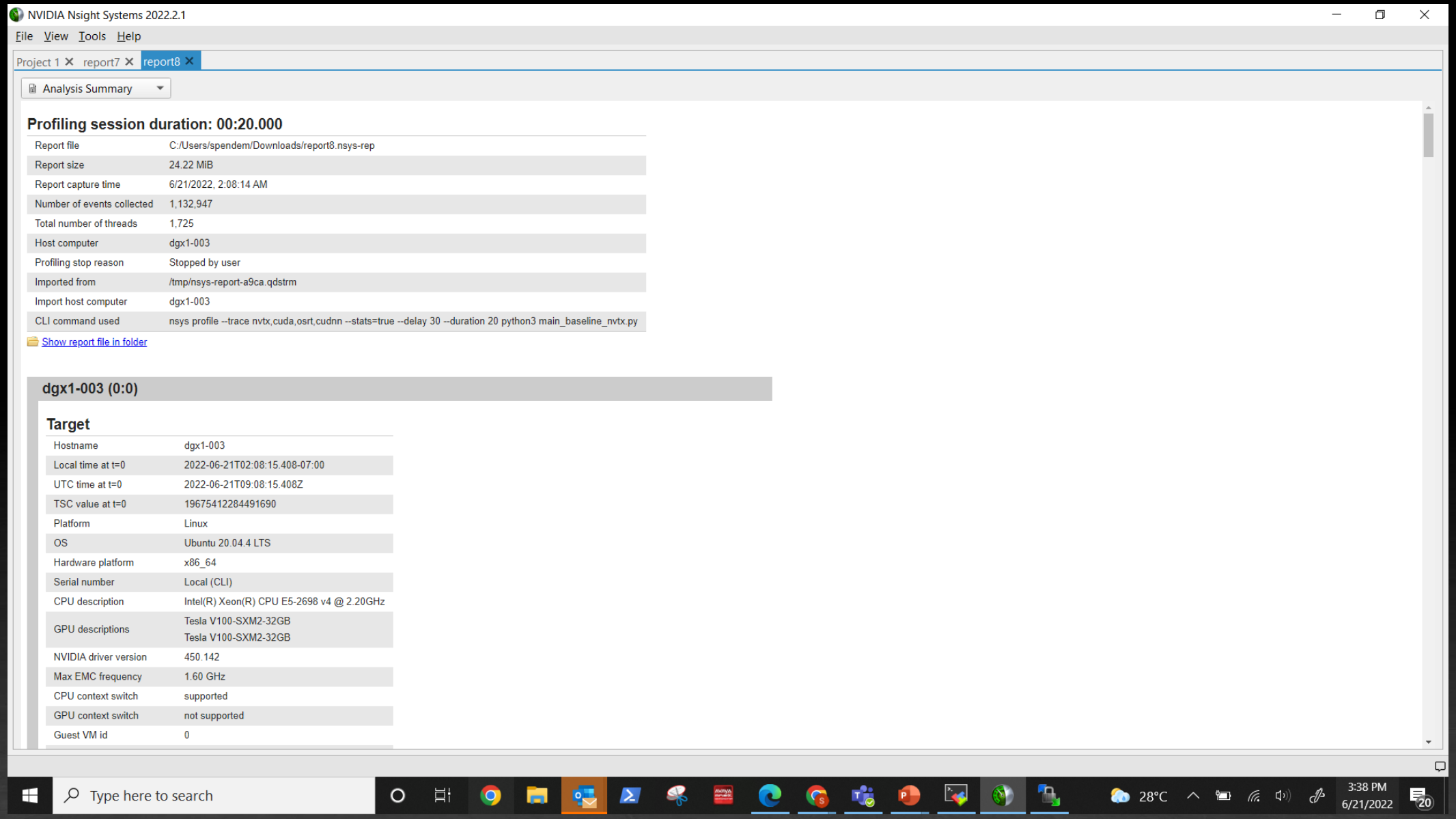
NSIGHT SYSTEMS

Key Features

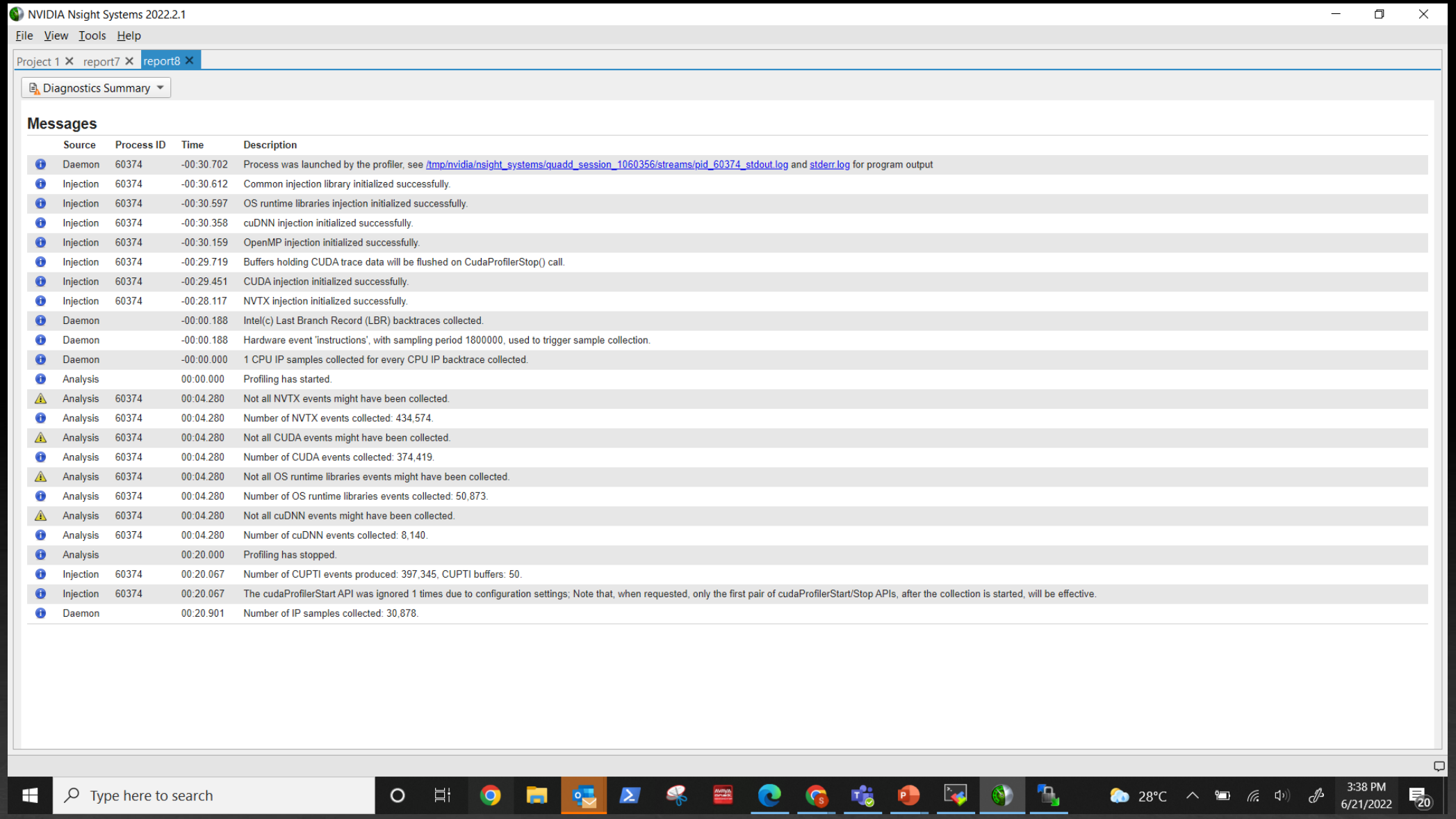
- Timeline View
- Event table



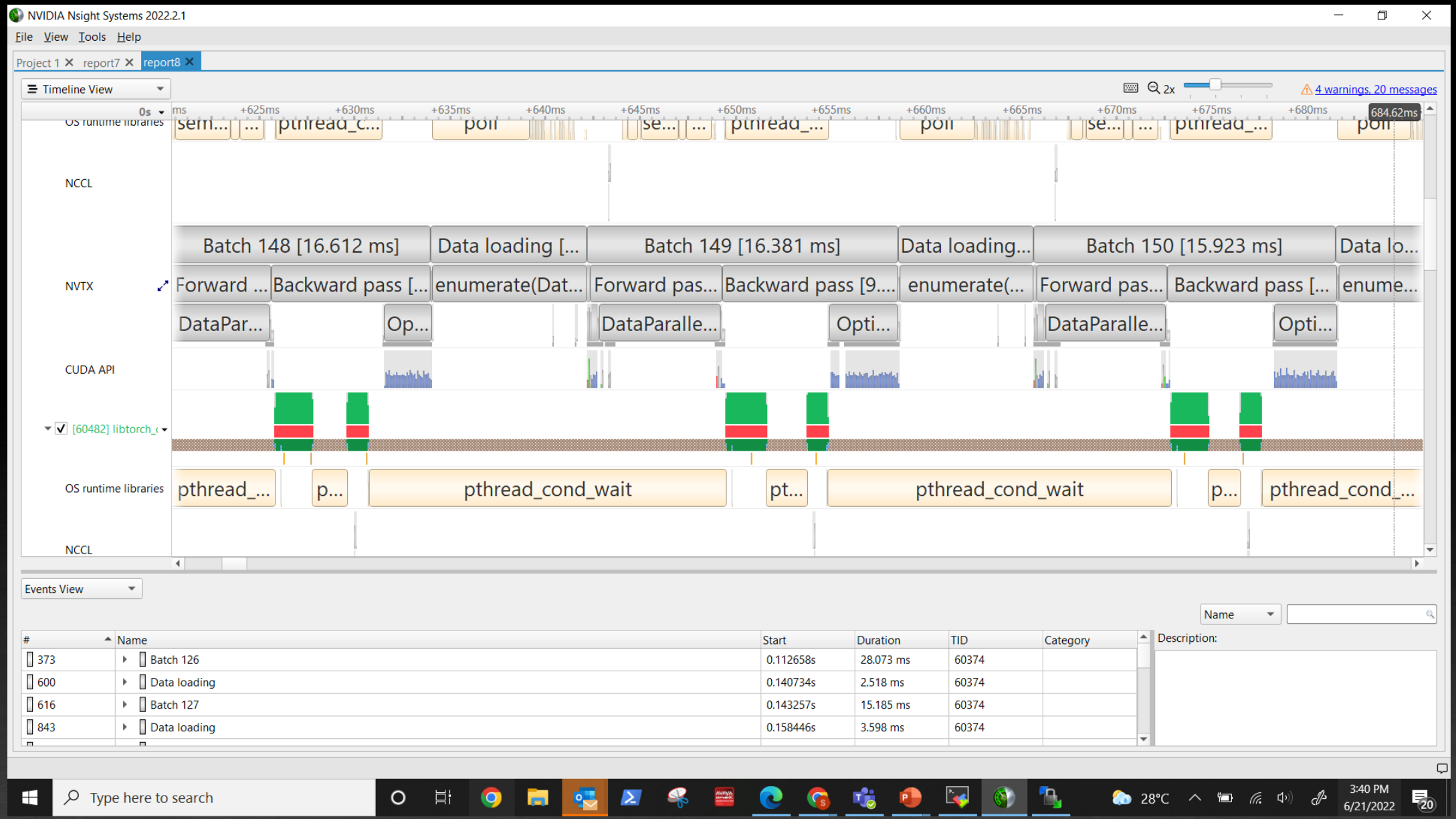
Analysis Summary



Diagnotics Summary



NVTX Trace


NVIDIA Nsight Systems 2022.2.1
File View Tools Help
Project 1 report7 report8
Timeline View
2x
4 warnings, 20 messages
0s
+625ms +630ms +635ms +640ms +645ms +650ms +655ms +660ms +665ms +670ms +675ms +680ms
684.62ms
OS runtime libraries
NCCL
Batch 148 [16.612 ms]
Data loading [...
Batch 149 [16.381 ms]
Data loading...
Batch 150 [15.923 ms]
NVTX
Forward ...
Backward pass [...
enumerate(Dat...
Forward pas...
Backward pass [9....
enumerate(...
Forward pas...
Backward pass [...
DataPar...
Op...
DataParalle...
Opti...
DataParalle...
Opti...
CUDA API
[60482] libtorch_
OS runtime libraries
pthread_...
p...
pthread_cond_wait
pt...
pthread_cond_wait
p...
pthread_cond_...
NCCL
Events View
Name
Name Start Duration TID Category
373 Batch 126 0.112658s 28.073 ms 60374
600 Data loading 0.140734s 2.518 ms 60374
616 Batch 127 0.143257s 15.185 ms 60374
843 Data loading 0.158446s 3.598 ms 60374
Description:
Type here to search
28°C
3:40 PM
6/21/2022

Other Features

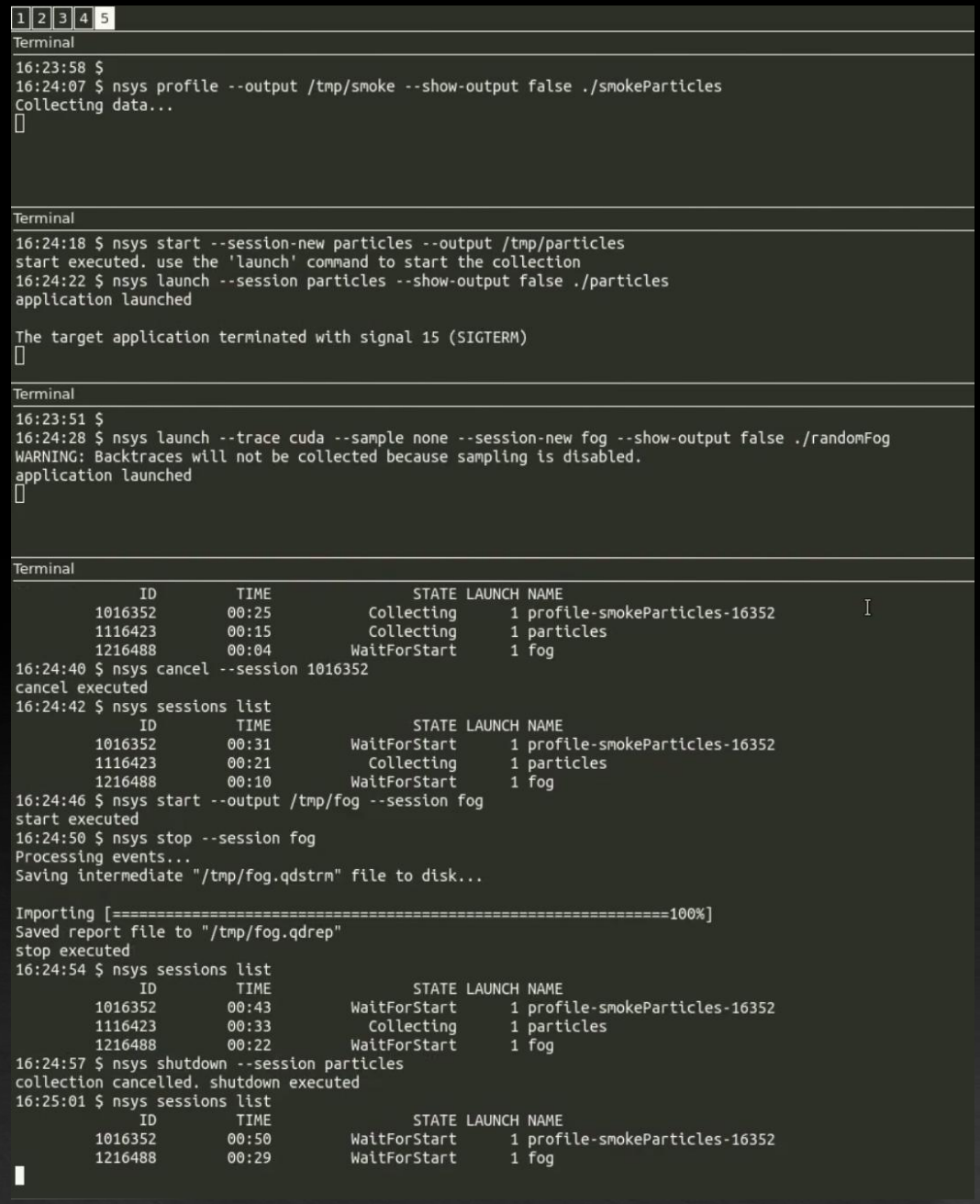
CLI improvement

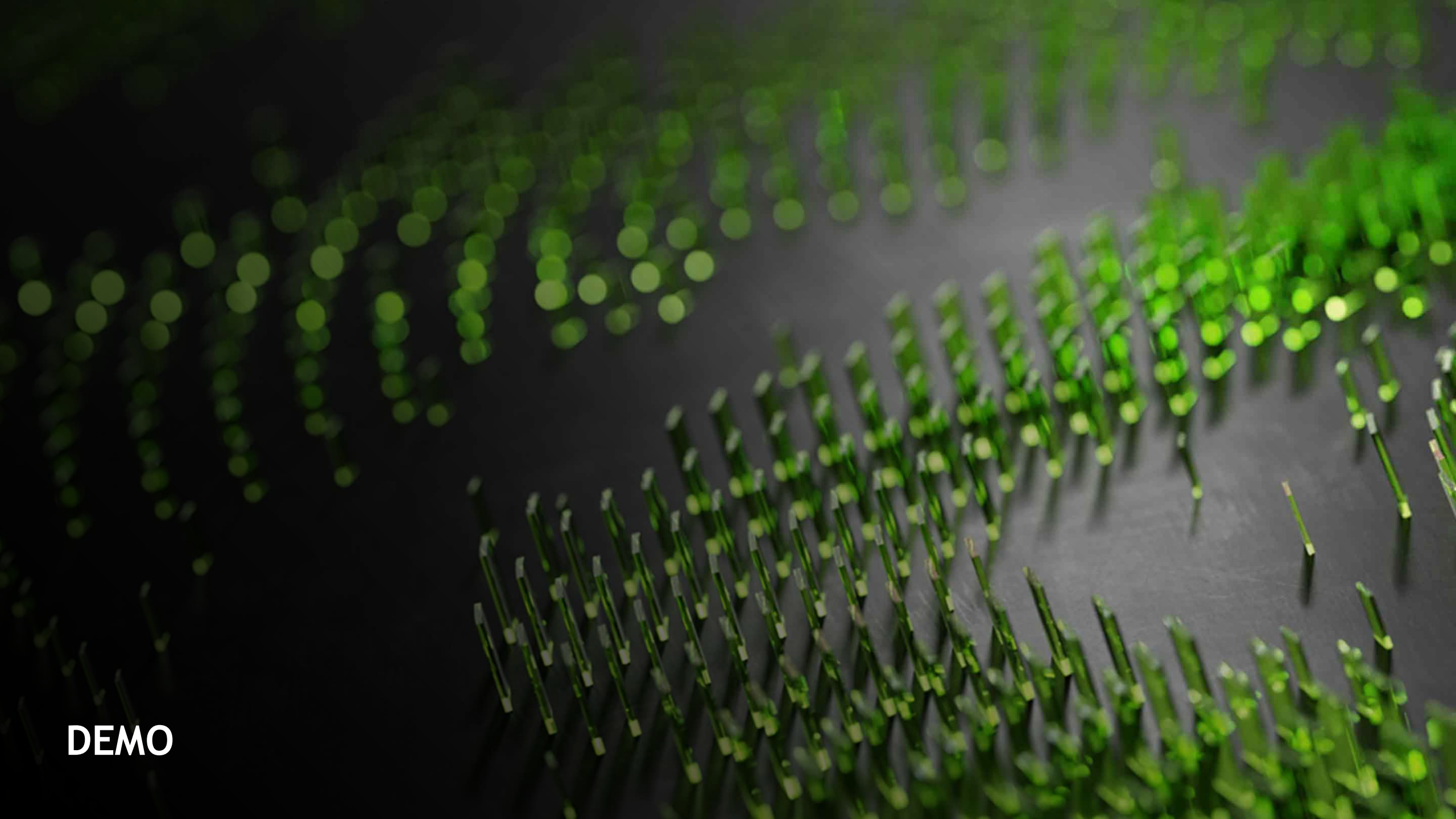
- Simultaneous sessions & management
- `nsys launch` forwards terminal and signals
- `nsys stats` command
- `nsys export` command

CUDA context “All Streams” aggregation tree

Windows WDDM GPU memory usage, paging, evictions

Graphics trace enhancements




DEMO

NSIGHT COMPUTE

Kernel Profiling Tool

Key Features:

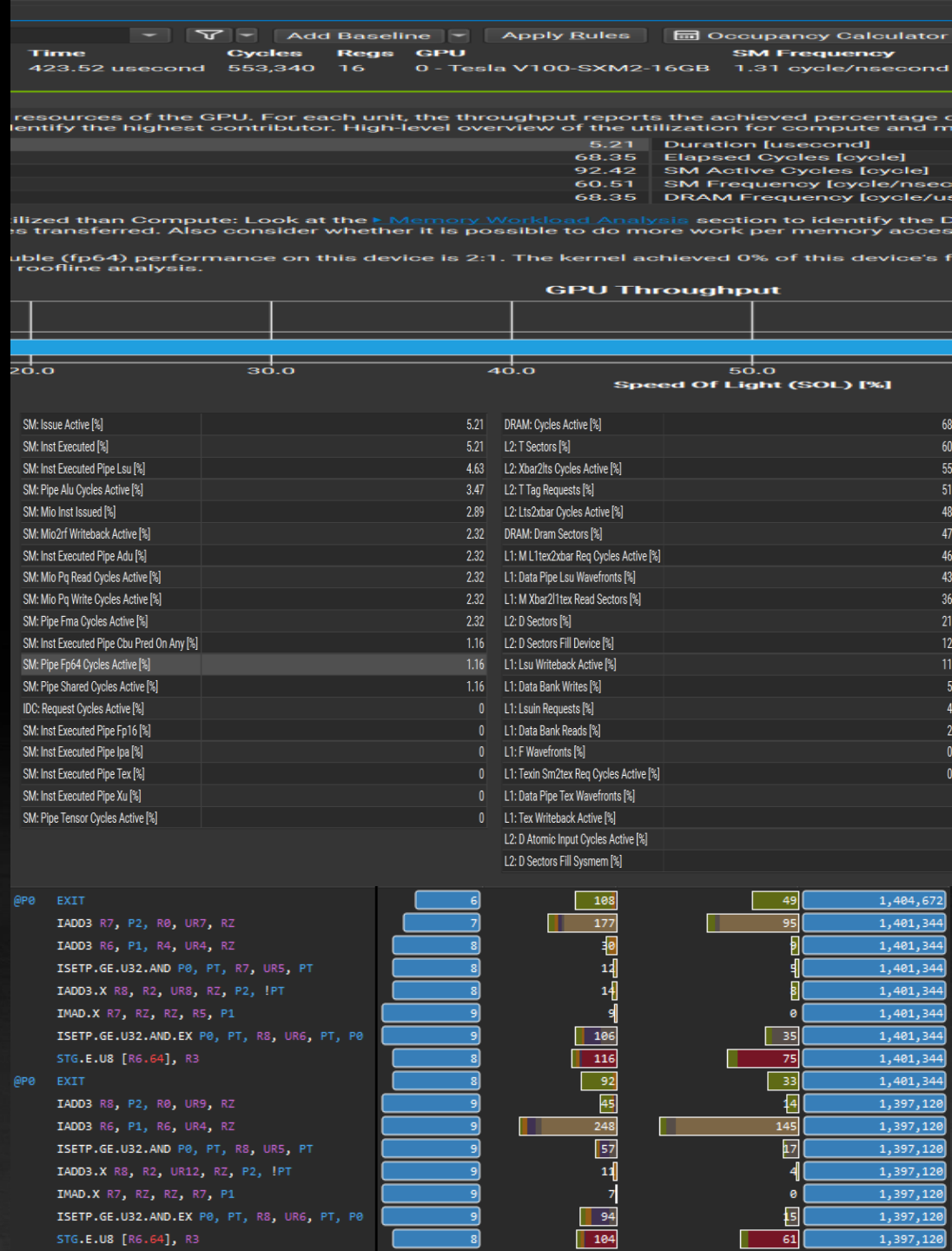
- Interactive CUDA API debugging and kernel profiling
- Fast Data Collection
- Compare performance metrics across different runs

$ ncu --set full -k <kernel_name> -s 4 -c 1 -o myreport ./myapp

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

GPUs: Volta, Turing, A100 GPUs

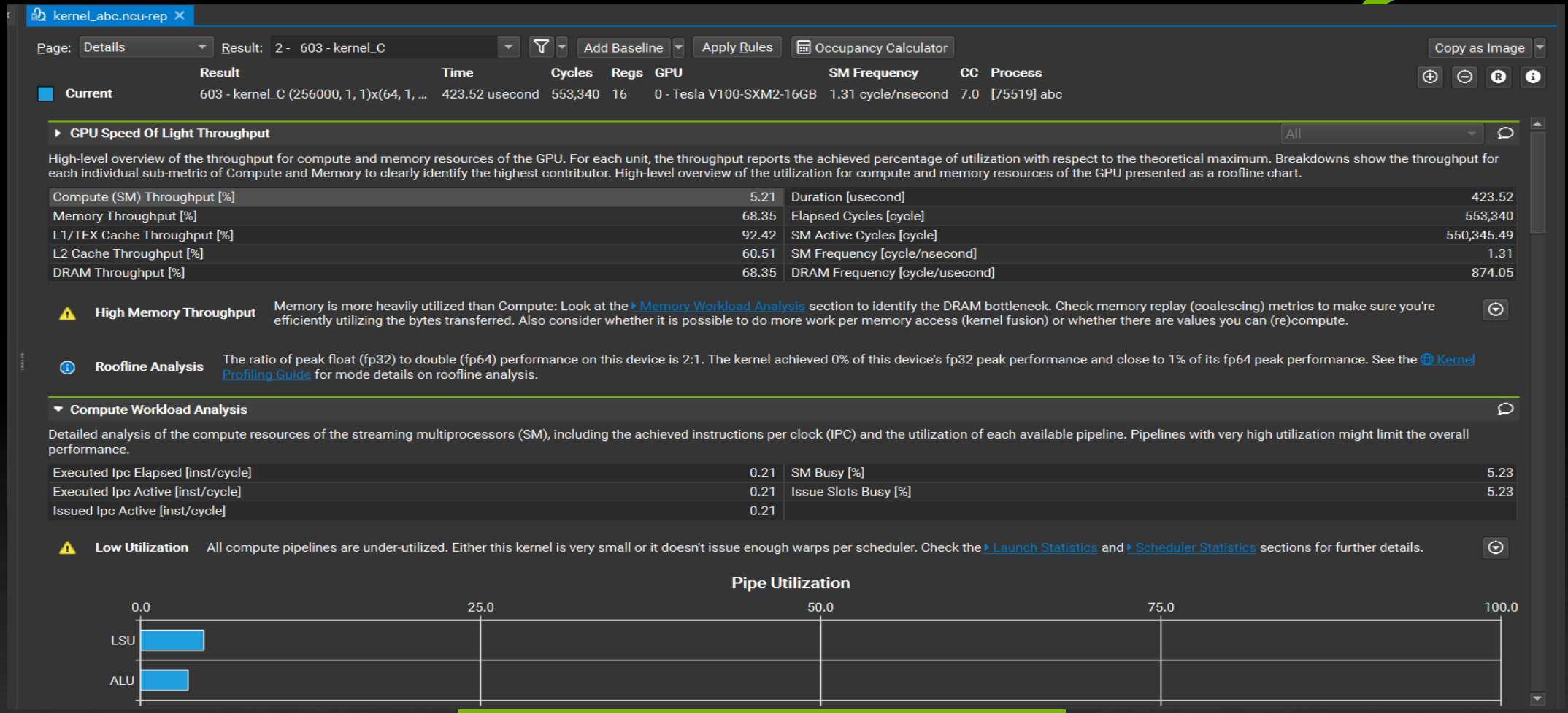
Documentation: https://developer.nvidia.com/nsight-compute



NSIGHT COMPUTE

Profile Report - Details Page

All Data on Single Page

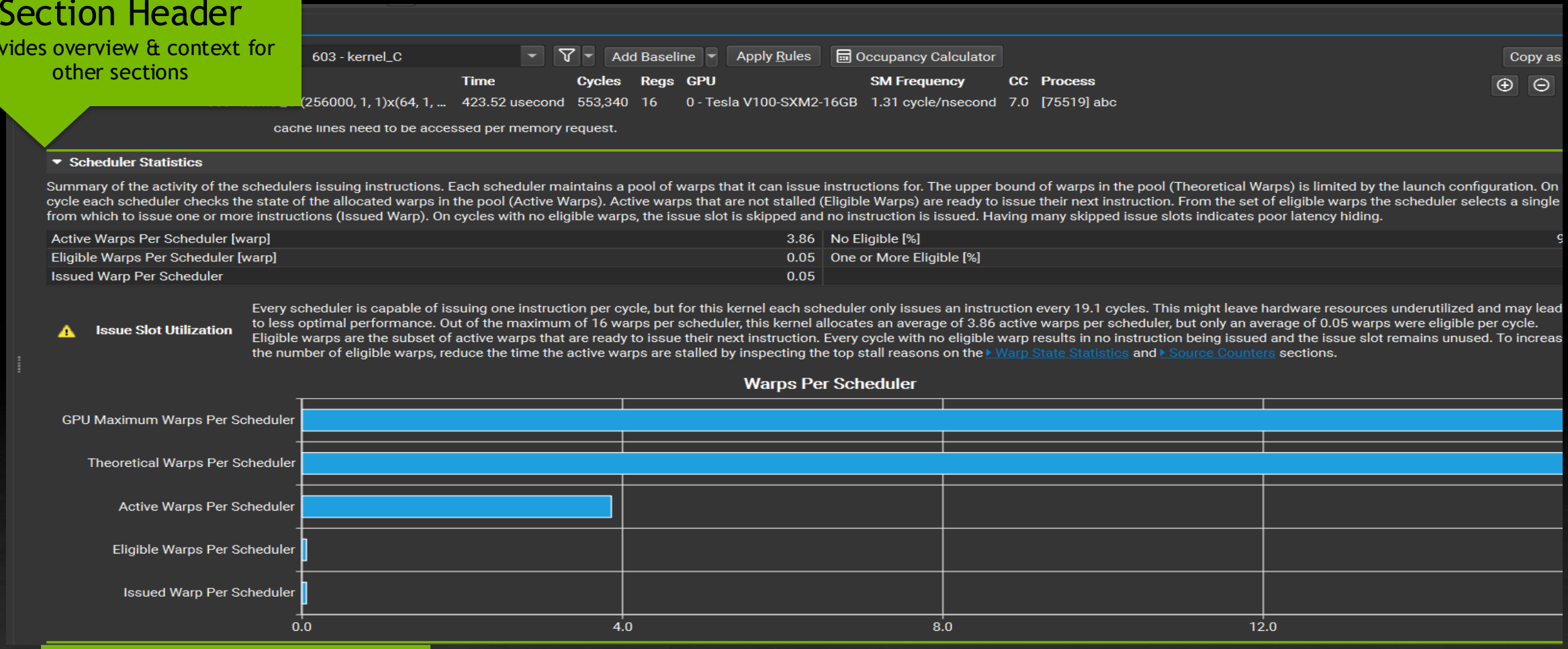


Ordered from Top-Level to Low-Level

NSIGHT COMPUTE

Section Example

Section Header
provides overview & context for other sections

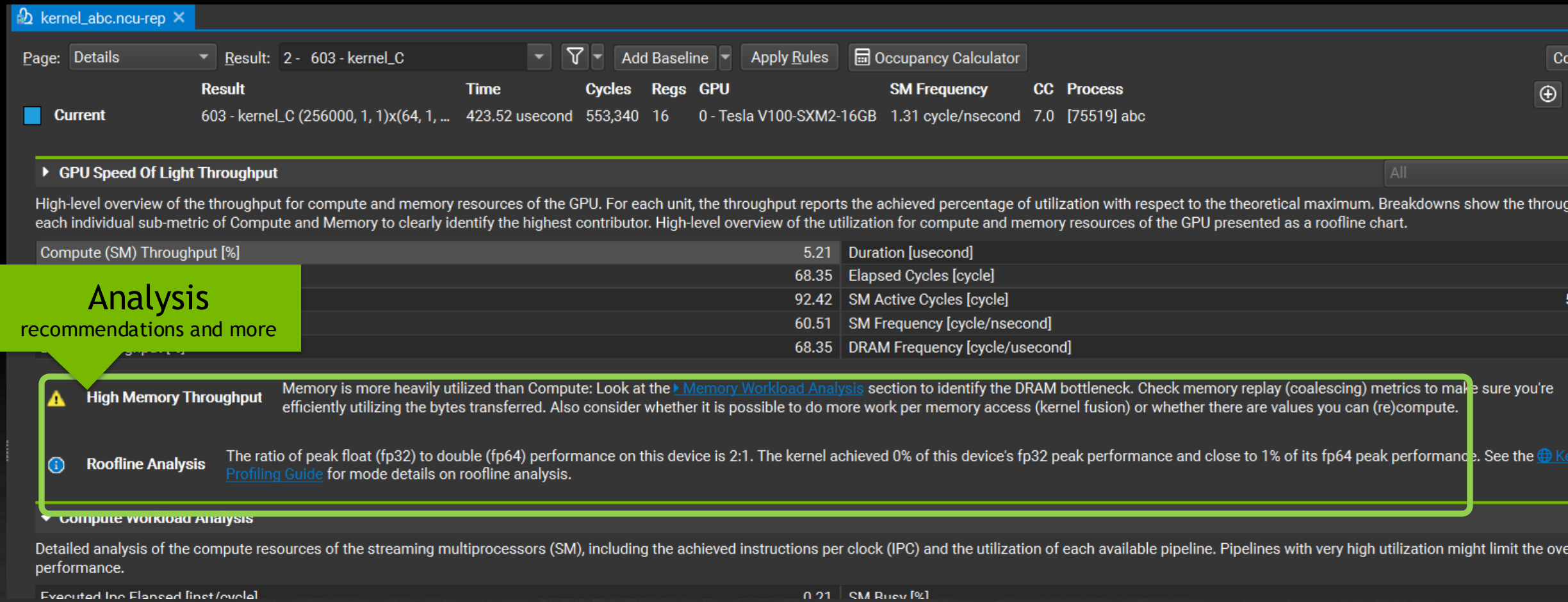


Section Config
completely data driven
add/modify/change sections

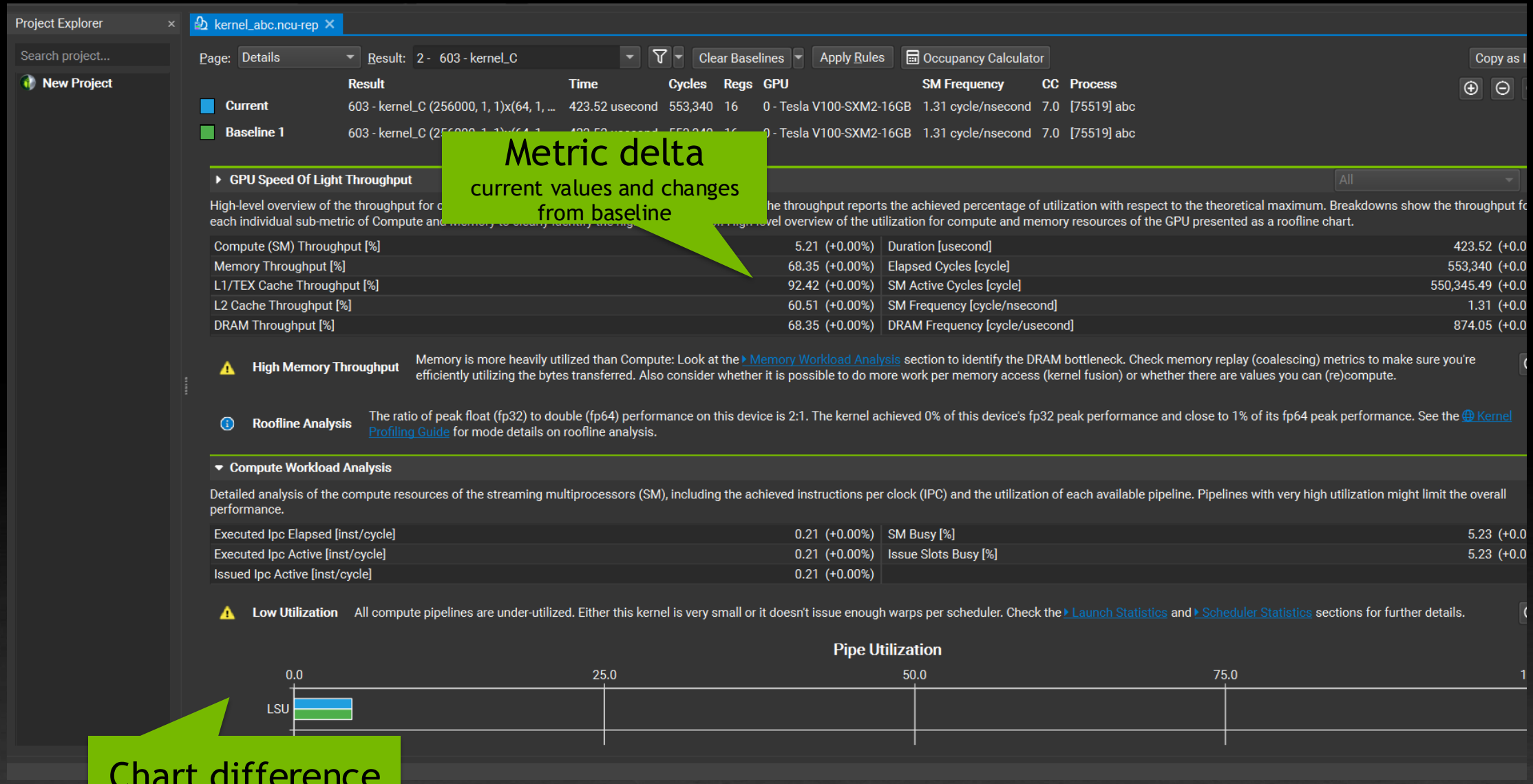
NVIDIA

NSIGHT COMPUTE

Unguided Analysis / Rules System



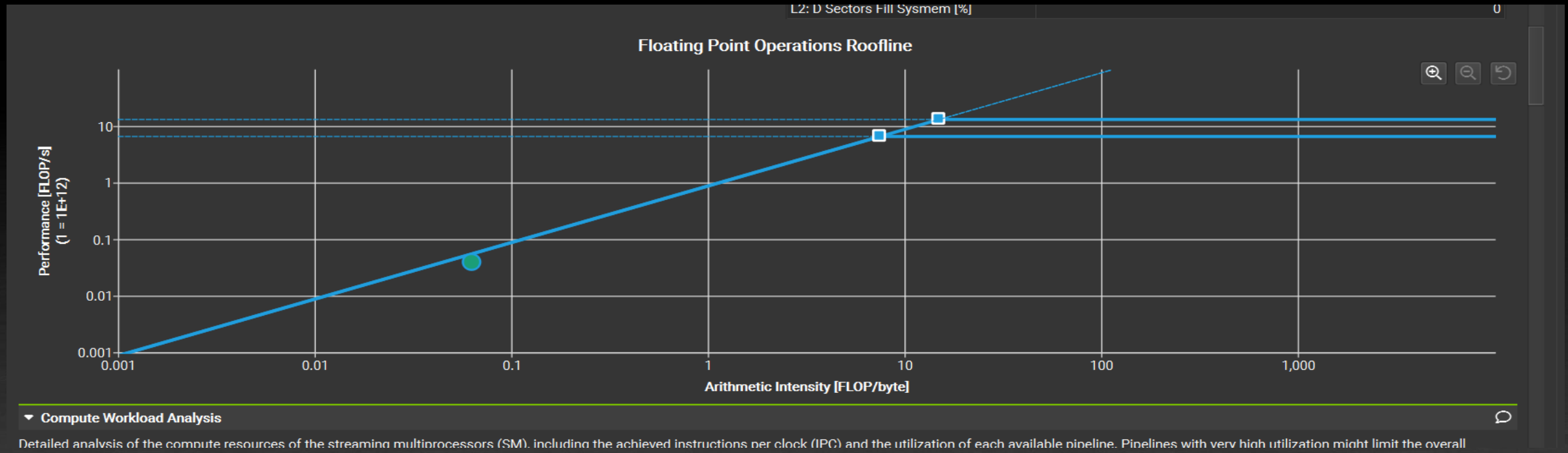
NSIGHT COMPUTE

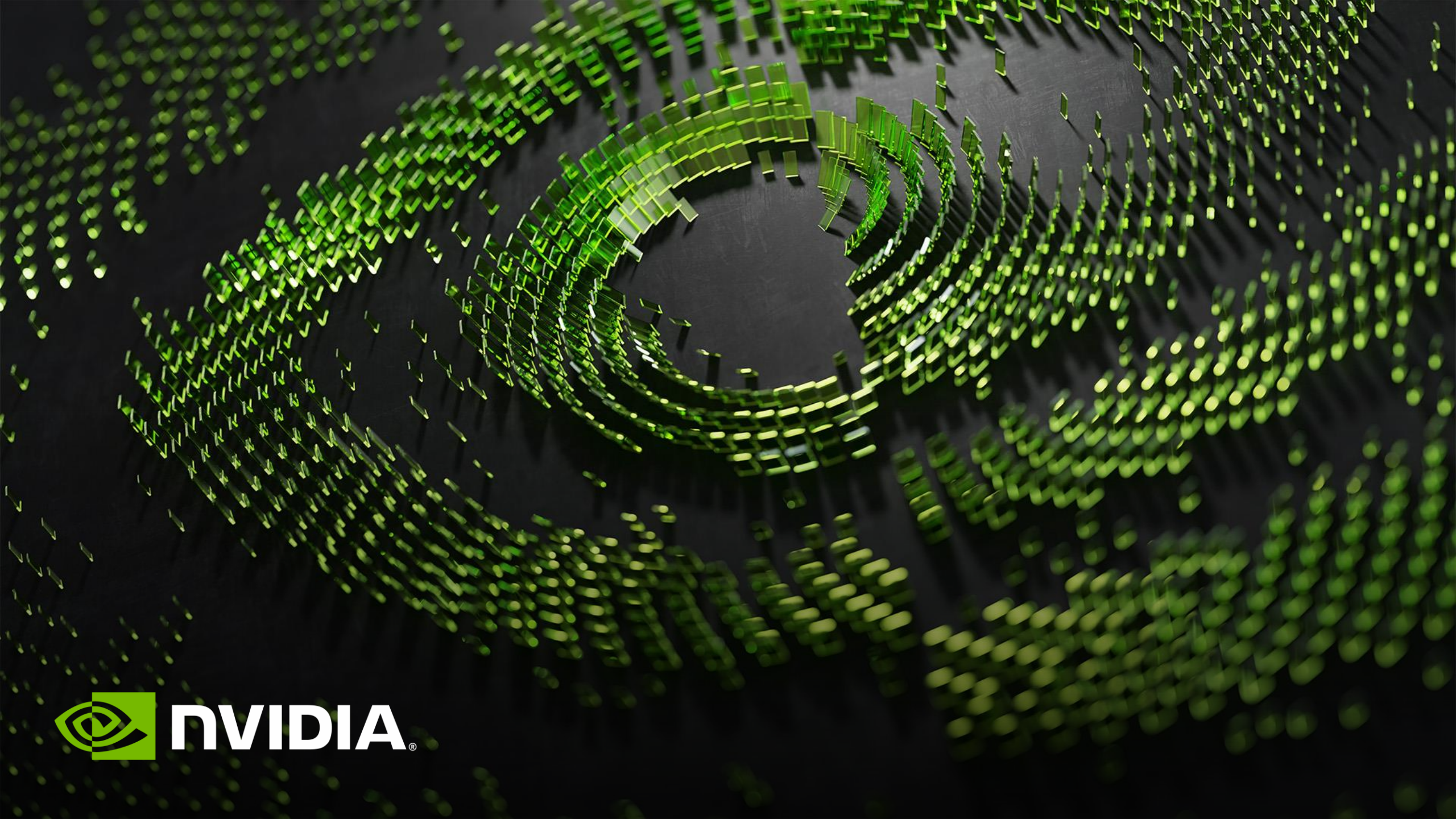


NSIGHT COMPUTE

New Roofline analysis

- Efficient way to evaluate kernel characteristics, quickly understand potential directions for further improvements or existing limiters
- Inputs: Arithmetic Intensity (FLOPS/bytes)
  Performance (FLOPS/s)
- Ceilings: Peak Memory Bandwidth
  Peak FP32/FP64 Performance




NVIDIA