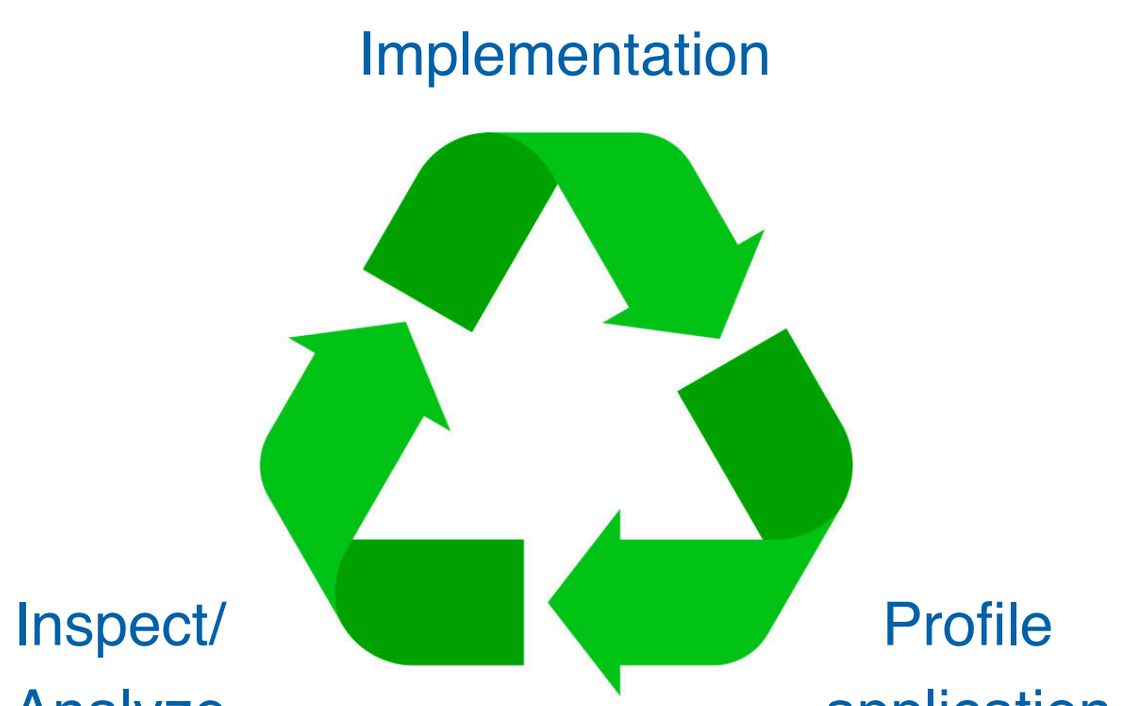
Introduction to NVIDIA profiling tools

Martin Kwok(Fermilab)

Traineeships in Advanced Computing for High Energy Physics (TAC-HEP)
31 Oct, 2024

Overview of profilers

- GPU programing is often an iterative process
  - Profiling often provides important insight to achieving better performance
- Nvidia has two main profiling tools* for different level of analysis
- Nsight System
  - A ***system-wide*** performance analysis tool
    - Overview of program timeline (Host/GPU trace)
    - Data movement/ synchronization
- Nsight Compute
  - An interactive ***kernel*** profiler for CUDA applications
    - GPU utilization
    - Memory access
    - Source-code reference



*older tools called `nvprof/NVVP` are being deprecated

Nsight system

- System-wide application algorithm tuning
- Visualization
  - Locate optimization opportunities
  - See gaps of unused CPU and GPU times

Command line interface:

`nsys`

Generate report

`.nsys-rep`

Graphical interface:

`nsight-sys`

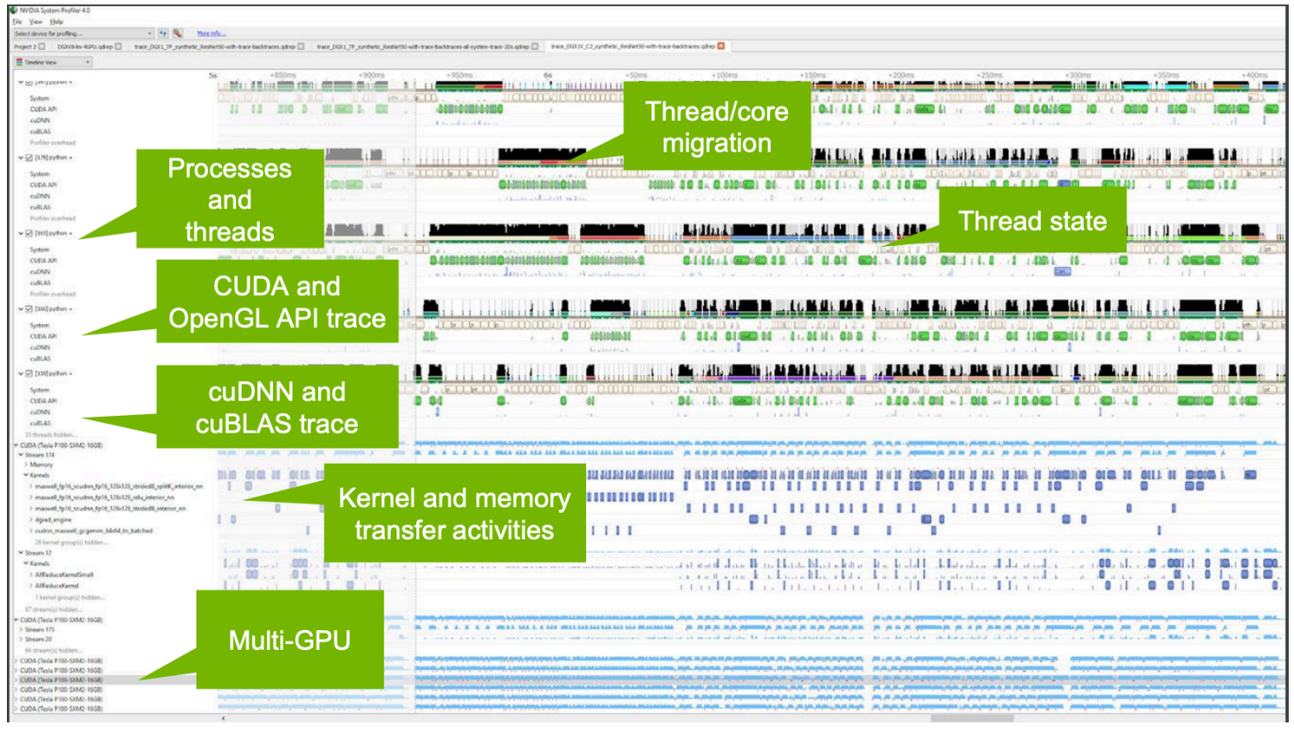
```
Operating System Runtime API Statistics:

 Time (%)  Total Time (ns)  Num Calls    Avg (ns)      Med (ns)     Min (ns)   Max (ns)    StdDev (ns)       Name
 --------  ---------------  ---------  ------------  ------------  --------  -----------  ------------  ----------
     96.9   14,319,334,568        154  92,982,692.0  100,139,902.0     2,560  100,143,937  24,886,371.8  poll
      2.3      342,595,408        508     674,400.4       11,695.0     1,010  165,912,878   7,603,508.4  ioctl
      0.6       88,713,038         27   3,285,668.1        7,170.0     1,360   71,884,588  13,887,463.8  mmap
      0.1       21,921,139         28     782,897.8        2,435.0     1,150   20,423,878   3,854,048.7  fopen
      0.0          867,626          9      96,402.9       49,310.0    16,340      502,894     153,391.8  sem_timedwait
      0.0          803,508         27      29,759.6        3,540.0     2,890      494,224      93,578.8  mmap64
      0.0          582,824          5     116,564.8        1,720.0     1,220      313,762     158,570.3  fcntl
      0.0          469,212          4     117,303.0      116,650.5    44,550      191,361      75,627.4  pthread_create
      0.0          357,502          5      71,500.4       15,090.0     4,110      316,282     136,960.1  fread
      0.0          239,796         44       5,449.9        4,335.0     1,350       26,160       4,006.3  open64
      0.0           95,481         17       5,616.5        4,050.0     1,290       23,360       5,337.4  munmap
      0.0           34,021          1      34,021.0       34,021.0    34,021       34,021           0.0  fgets
      0.0           29,700          6       4,950.0        3,755.0     1,960        9,430       3,350.0  open
      0.0           23,702         10       2,370.2        1,870.5     1,160        4,731       1,382.2  write
      0.0           14,880          9       1,653.3        1,670.0     1,030        2,700         523.7  read
      0.0           12,250          2       6,125.0        6,125.0     2,060       10,190       5,748.8  socket
      0.0           10,701          5       2,140.2        1,510.0     1,160        5,260       1,750.8  fclose
      0.0            8,961          1       8,961.0        8,961.0     8,961        8,961           0.0  connect
      0.0            6,260          1       6,260.0        6,260.0     6,260        6,260           0.0  pipe2
      0.0            1,160          1       1,160.0        1,160.0     1,160        1,160           0.0  bind

[5/8] Executing 'cudaapisum' stats report

CUDA API Statistics:

 Time (%)  Total Time (ns)  Num Calls    Avg (ns)      Med (ns)     Min (ns)   Max (ns)    StdDev (ns)          Name
 --------  ---------------  ---------  ------------  ------------  --------  -----------  ------------  --------------------
     46.1      354,695,212          3  118,231,737.3  155,726,568.0  27,430,083  171,538,561  79,032,968.5  cudaMallocHost
     38.8      298,764,028          7   42,680,575.4   42,571,950.0  42,566,451   43,103,845     195,558.2  cudaDeviceSynchronize
     14.4      110,635,279          3   36,878,426.3   11,177,207.0  11,058,656   88,399,416  44,618,525.3  cudaFreeHost
      0.7        5,006,049          3    1,668,683.0    2,249,408.0     148,381    2,608,260   1,328,789.8  cudaFree
      0.1          797,166          3      265,722.0      190,171.0     130,801      476,194     184,675.5  cudaMalloc
      0.0          108,870         21        5,184.3        3,480.0       3,000       26,820       5,167.7  cudaMemcpyAsync
      0.0           68,871          7        9,838.7        5,860.0       5,430       32,430       9,986.5  cudaLaunchKernel
      0.0           25,740          1       25,740.0       25,740.0      25,740       25,740           0.0  cudaStreamCreate
```



Fermilab

Nsight system - Command Line Interface(CLI)

- Basic usage:
  `nsys profile [options for profile][application][app. options]`
- Example:
  `nsys profile -o fileName --stats=true ./cuda`
- Result:
  - stats=true gives report printouts
- Reports written in **filename.nsys-rep & filename.sqlite**

osrtsum

```
Operating System Runtime API Statistics:

 Time (%)  Total Time (ns)  Num Calls    Avg (ns)      Med (ns)     Min (ns)   Max (ns)    StdDev (ns)       Name
 --------  ---------------  ---------  ------------  ------------  --------  -----------  ------------  --------------
     96.9   14,319,334,568        154  92,982,692.0  100,139,902.0     2,560  100,143,937  24,886,371.8  poll
      2.3      342,595,408        508     674,400.4       11,695.0     1,010  165,912,878   7,603,508.4  ioctl
      0.6       88,713,038         27   3,285,668.1        7,170.0     1,360   71,884,588  13,887,463.8  mmap
      0.1       21,921,139         28     782,897.8        2,435.0     1,150   20,423,878   3,854,048.7  fopen
      0.0          867,626          9      96,402.9       49,310.0    16,340      502,894     153,391.8  sem_timedwait
      0.0          803,508         27      29,759.6        3,540.0     2,890      494,224      93,578.8  mmap64
      0.0          582,824          5     116,564.8        1,720.0     1,220      313,762     158,570.3  fcntl
      0.0          469,212          4     117,303.0      116,650.5    44,550      191,361      75,627.4  pthread_create
      0.0          357,502          5      71,500.4       15,090.0     4,110      316,282     136,960.1  fread
      0.0          239,796         44       5,449.9        4,335.0     1,350       26,160       4,006.3  open64
      0.0           95,481         17       5,616.5        4,050.0     1,290       23,360       5,337.4  munmap
      0.0           34,021          1      34,021.0       34,021.0    34,021       34,021           0.0  fgets
      0.0           29,700          6       4,950.0        3,755.0     1,960        9,430       3,350.0  open
      0.0           23,702         10       2,370.2        1,870.5     1,160        4,731       1,382.2  write
      0.0           14,880          9       1,653.3        1,670.0     1,030        2,700         523.7  read
      0.0           12,250          2       6,125.0        6,125.0     2,060       10,190       5,748.8  socket
      0.0           10,701          5       2,140.2        1,510.0     1,160        5,260       1,750.8  fclose
      0.0            8,961          1       8,961.0        8,961.0     8,961        8,961           0.0  connect
      0.0            6,260          1       6,260.0        6,260.0     6,260        6,260           0.0  pipe2
      0.0            1,160          1       1,160.0        1,160.0     1,160        1,160           0.0  bind
```

cudaapisum

```
[5/8] Executing 'cudaapisum' stats report

CUDA API Statistics:

 Time (%)  Total Time (ns)  Num Calls    Avg (ns)      Med (ns)     Min (ns)    Max (ns)    StdDev (ns)          Name
 --------  ---------------  ---------  ------------  ------------  ----------  -----------  ------------  --------------------
     46.1      354,695,212          3  118,231,737.3  155,726,568.0  27,430,083  171,538,561  79,032,968.5  cudaMallocHost
     38.8      298,764,028          7   42,680,575.4   42,571,950.0  42,566,451   43,103,845     195,558.2  cudaDeviceSynchronize
     14.4      110,635,279          3   36,878,426.3   11,177,207.0  11,058,656   88,399,416  44,618,525.3  cudaFreeHost
      0.7        5,006,049          3    1,668,683.0    2,249,408.0     148,381    2,608,260   1,328,789.8  cudaFree
      0.1          797,166          3      265,722.0      190,171.0     130,801      476,194     184,675.5  cudaMalloc
      0.0          108,870         21        5,184.3        3,480.0       3,000       26,820       5,167.7  cudaMemcpyAsync
      0.0           68,871          7        9,838.7        5,860.0       5,430       32,430       9,986.5  cudaLaunchKernel
      0.0           25,740          1       25,740.0       25,740.0      25,740       25,740           0.0  cudaStreamCreate
```


Fermilab

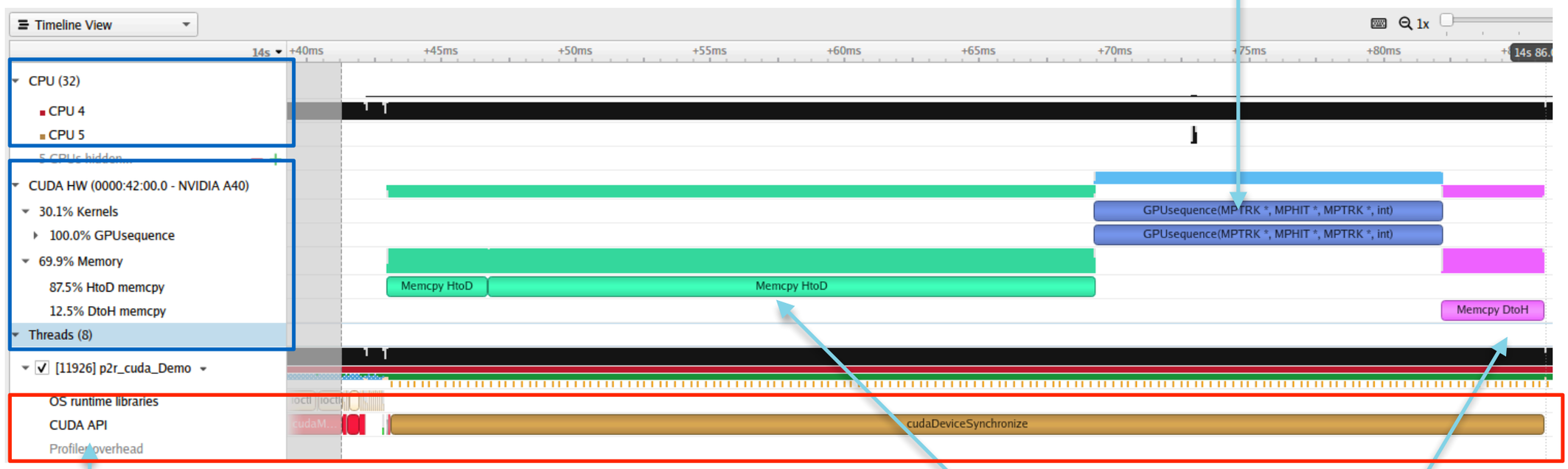
Nsight system - Command Line Interface(CLI)

- Reading back the reports:
  - Print out all reports:
    `nsys stats fileName.nsys-rep`
  - Reading one particular report
    `nsys stats --report [option] fileName.nsys-rep --timeunit us`
  - To see what reports are available:
    `nsys stats --help-reports`

```
apigpusum[:base|mangled] -- API & GPU Summary (CUDA API + kernels + mem ops)
cudaapisum -- CUDA API Summary
cudaapitrace -- CUDA API Trace
dx12gpumarkersum -- DX12 GPU Command List PIX Ranges Summary
gpukernsum[:base|mangled] -- CUDA GPU Kernel Summary
gpumemsizesum -- GPU Memory Operations Summary (by Size)
gpumemtimesum -- GPU Memory Operations Summary (by Time)
gpusum[:base|mangled] -- GPU Summary (kernels + memory operations)
gputrace -- CUDA GPU Trace
kernexecsum[:base|mangled] -- Summary of kernel launch and exec times
kernexectrace[:base|mangled] -- Kernel launch and exec time trace
khrdebuggpusum -- OpenGL KHR_debug GPU Range Summary
khrdebugsum -- OpenGL KHR_debug Range Summary
nvtxgpuproj -- NVTX range projection
nvtxkernsum[:base|mangled] -- NVTX Range Kernel Summary
nvtxppsum -- NVTX Push/Pop Range Summary
nvtxpptrace -- NVTX Push/Pop Range Trace
nvtxsesum -- NVTX Start/End Range Summary
nvtxsssum -- DEPRECATED - Use nvtxsesum instead
nvtxsum -- NVTX Range Summary
openaccsum -- OpenACC Summary
openmpevtsum -- OpenMP Event Summary
osrtsum -- OS Runtime Summary
pixsum -- PIX Range Summary
umcpupagefaults -- Unified Memory CPU Page Faults Summary
unifiedmemory -- Unified Memory Analysis Summary
unifiedmemorytotals -- Unified Memory Totals Summary
vulkangpumarkersum -- Vulkan GPU Range Summary
vulkanmarkerssum -- Vulkan Range Summary
wddmqueuesdetails -- WDDM Queues Utilization Summary
```

Nsight system - GUI

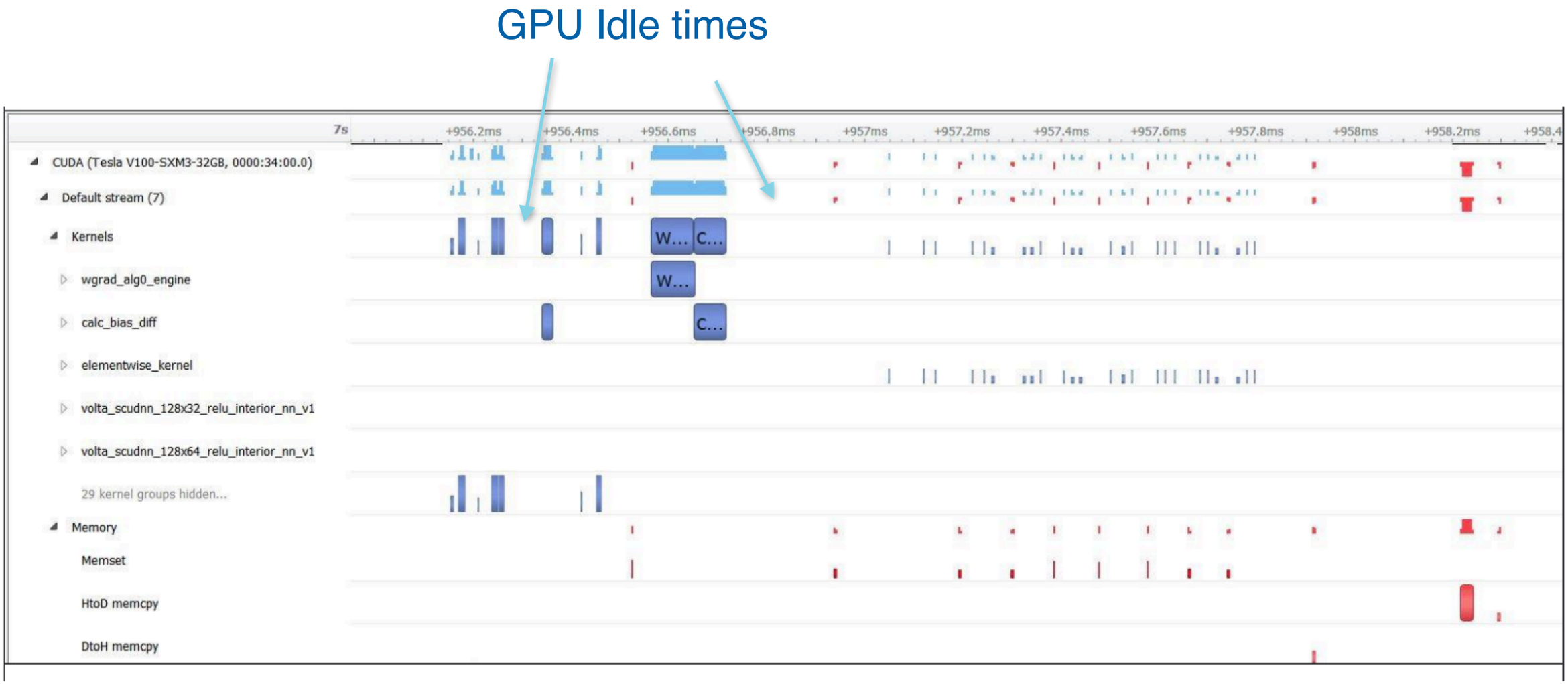
- After generating the reports, start the GUI with `nsight-sys`
- Visualize the program trace



Fermilab

Nsight system - GUI

- A more realistic example
  - Multiple kernels and/or streams

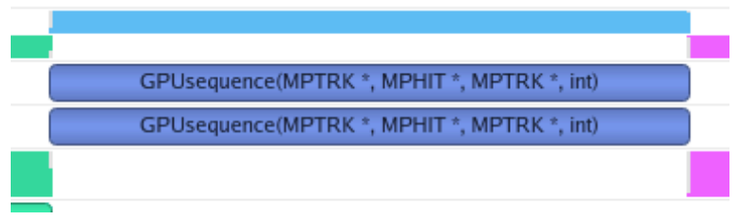


DEMO

Nsight compute - CLI

- `Nsight-system` provides the overview
  - GPU Kernel itself is a “blue box”
- `Nsight-compute` looks inside the blue box via hardware counters and software instrumentations
  - GPU utilization
  - Memory access
- In general, more “intrusive” to original program
  - Gather much more details
  - Could have non-negligible overheads depending on what data are being collected

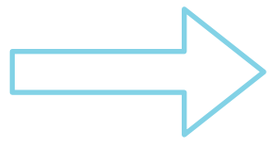
CUDA Kernel



Command line interface:

`ncu`

Generate report



`.ncu-rep`

Graphical interface:

`ncu-ui`

Fermilab

Nsight compute - CLI

- Basic Usage:
  `ncu [ncu options] [program] [program-arguments]`
- Examples:
- `ncu --set=[full] -o p2r_cuda_ncu_demo ./p2r_cuda_Demo`
  - Profile a specific kernel
  - `ncu --kernel-name GPUsequence ./p2r_cuda_Demo`
- What can ncu collect? [`ncu --list-sets`]
  - A metric is a characteristic of an application that is calculated from one or more event values
  - A section is a group of metrics

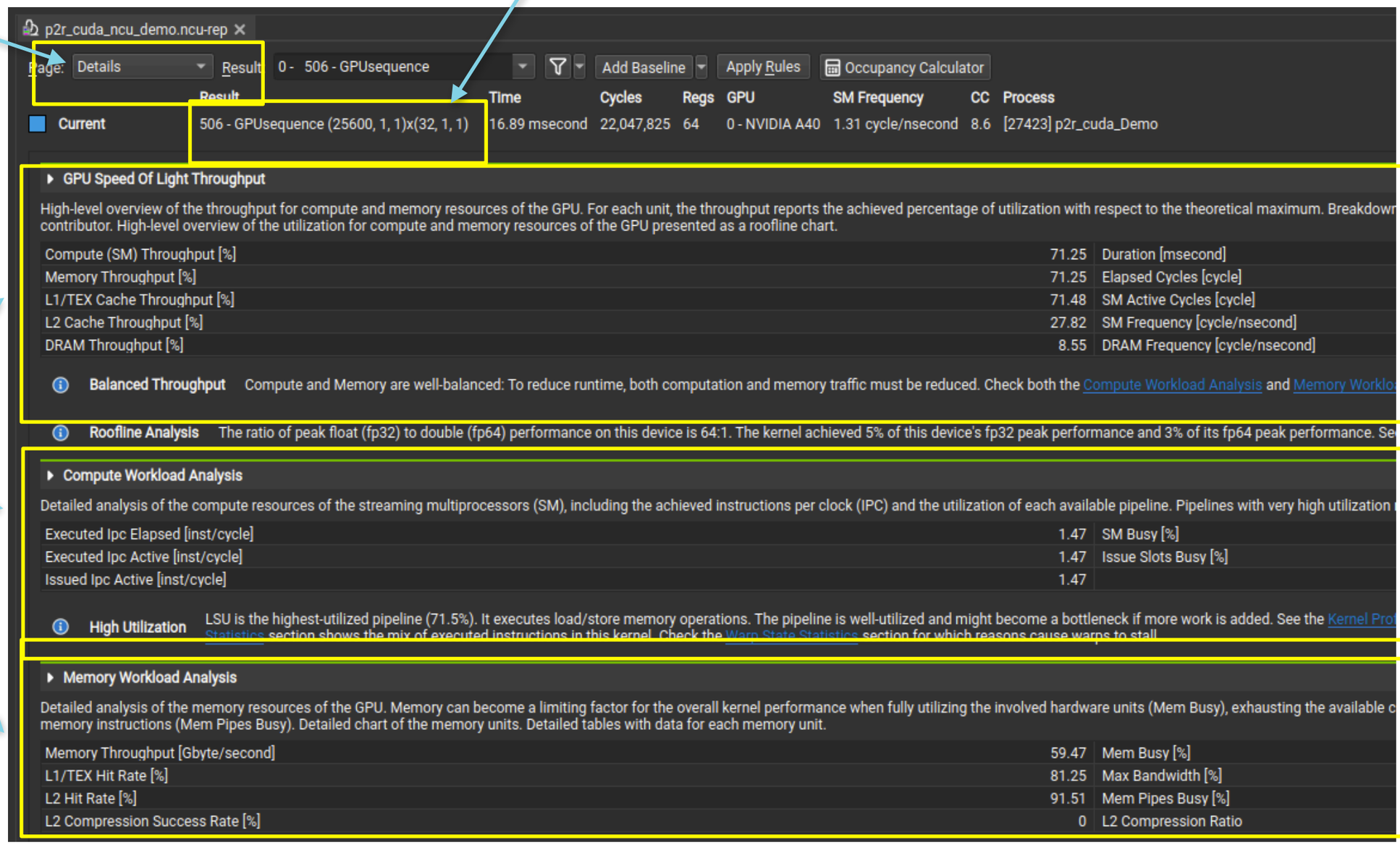
```
---------- ------------------------------------------------------------------------------- ------- ----------------
Identifier Sections                                                                        Enabled Estimated Metrics
---------- ------------------------------------------------------------------------------- ------- ----------------
default    LaunchStats, Occupancy, SpeedOfLight                                            yes     36
detailed   ComputeWorkloadAnalysis, InstructionStats, LaunchStats, MemoryWorkloadAnaly     no      181
           sis, Occupancy, SchedulerStats, SourceCounters, SpeedOfLight, SpeedOfLight_
           RooflineChart, WarpStateStats
full       ComputeWorkloadAnalysis, InstructionStats, LaunchStats, MemoryWorkloadAnaly     no      198
           sis, MemoryWorkloadAnalysis_Chart, MemoryWorkloadAnalysis_Tables, Nvlink_Ta
           bles, Nvlink_Topology, Occupancy, SchedulerStats, SourceCounters, SpeedOfLi
           ght, SpeedOfLight_RooflineChart, WarpStateStats
roofline   SpeedOfLight, SpeedOfLight_HierarchicalDoubleRooflineChart, SpeedOfLight_Hi     no      48
           erarchicalHalfRooflineChart, SpeedOfLight_HierarchicalSingleRooflineChart,
           SpeedOfLight_HierarchicalTensorRooflineChart, SpeedOfLight_RooflineChart
source     SourceCounters                                                                  no      67
```

Nsight compute - GUI

Page:
[Summary I Details I Source]

Kernel name

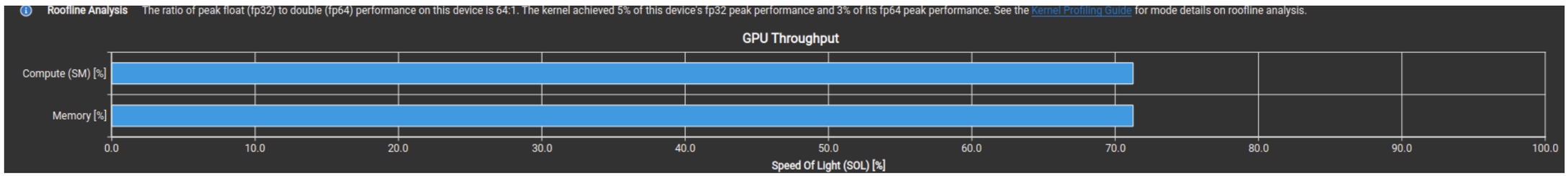
Sections



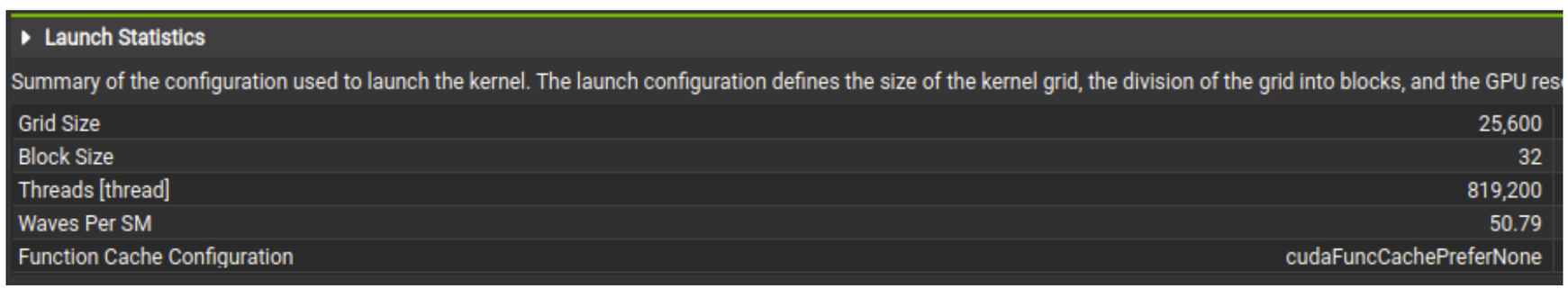
Fermilab

Nsight compute - GUI

- Speed of Light (SOL) -
  Compare the fraction of achieved compute/memory throughput w.r.t. theoretical maximum
  - Higher is better
  - If the an application is slow, typically it will show up as low utilization of SM/Memory



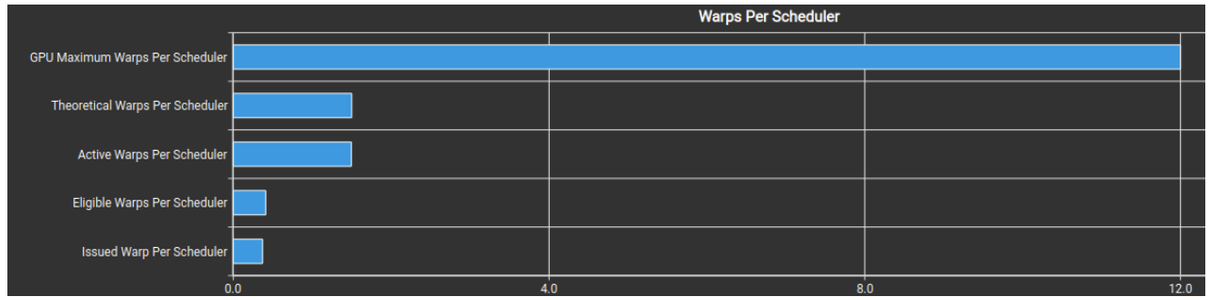
- Launch Statistics
  - Grid, blocks, Number of threads
    - Have you launched enough threads?
  - Waves Per SM
    - How many blocks each SM has to compute to finish the grid of work



Nsight compute - GUI

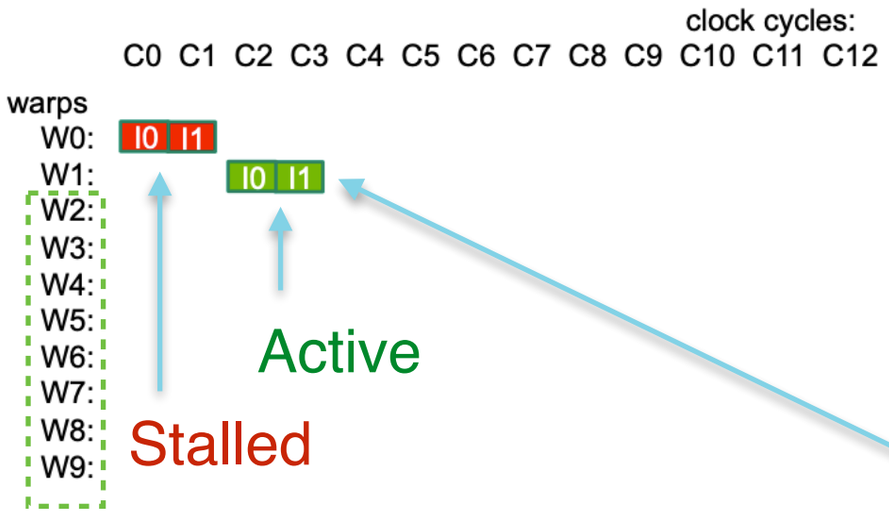
- Deeper dive into SM utilization:
  - How the work is distributed
- Scheduler Statistics
  Warp State Statistics
  Compute Workload

Scheduler statistics

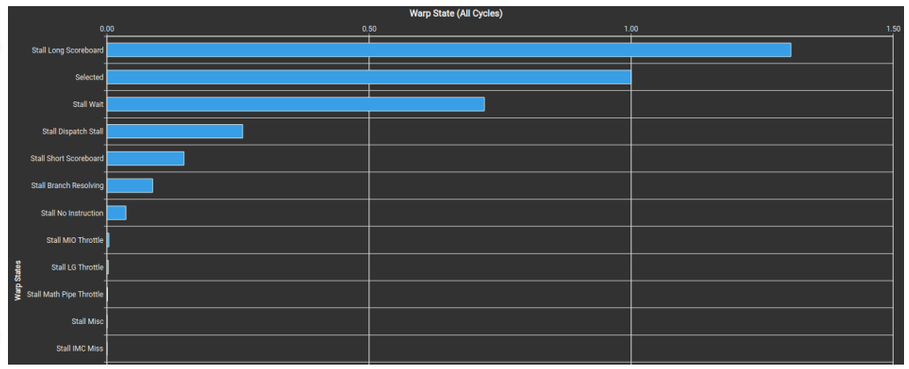


Warp Activities

Scheduler 1:

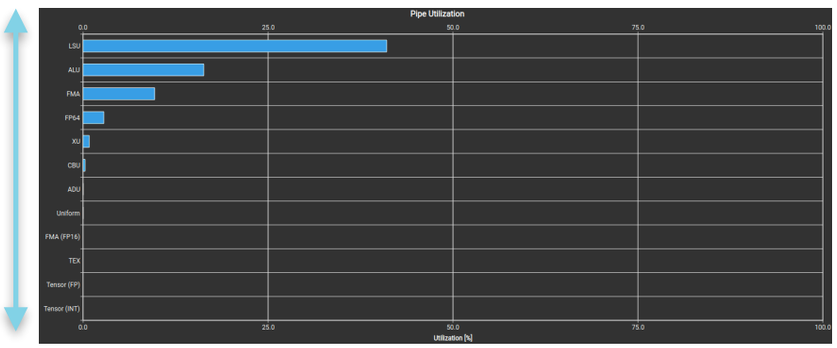


Warp state statistics



Warp states
(# cycles spent in each state for each instructions)

Compute Workload



Operations
(LSU, ALU, FMA etc)

Fermilab

Nsight compute - GUI

- Deeper dive into *memory* utilization
- Memory Chart [reference]

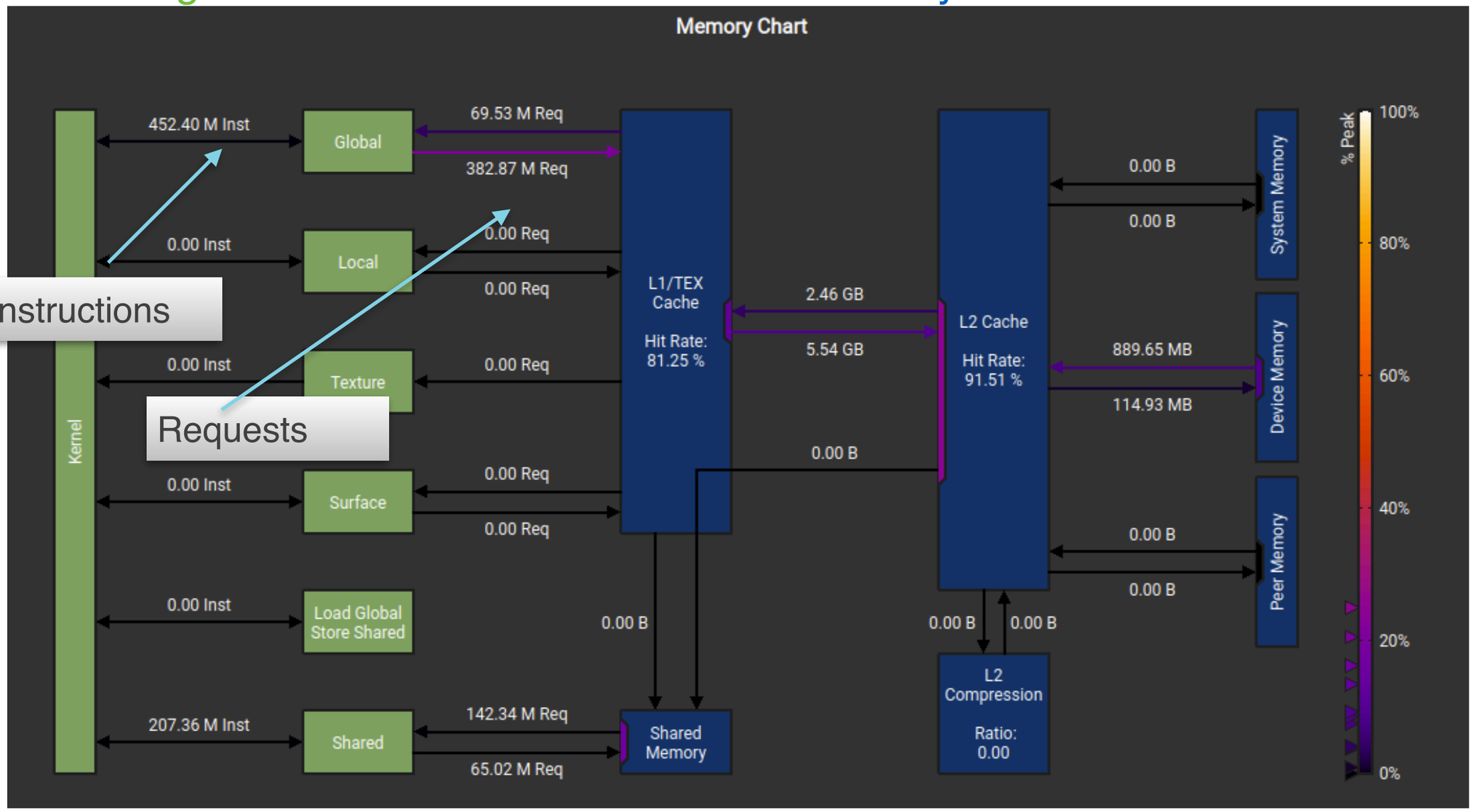
Logical Units

Physical Units

Kernel

instructions

Requests

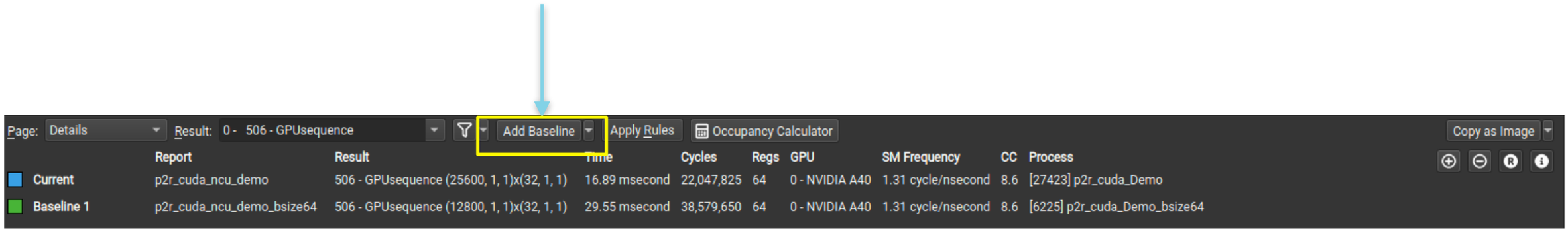


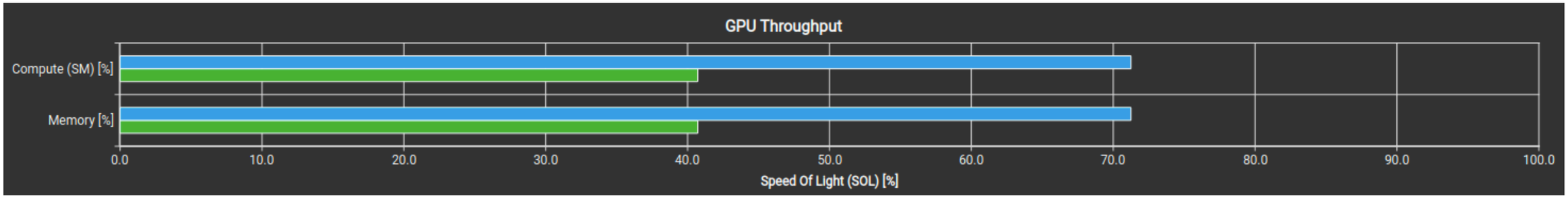
Fermilab

Nsight compute - baseline

- Often difficult to understand the all the metrics in an *absolute* sense
- But the ***relative*** difference is easier to interpret

- Can use any report as the “Baseline”
  - Open another report to compare all the metrics side-by-side



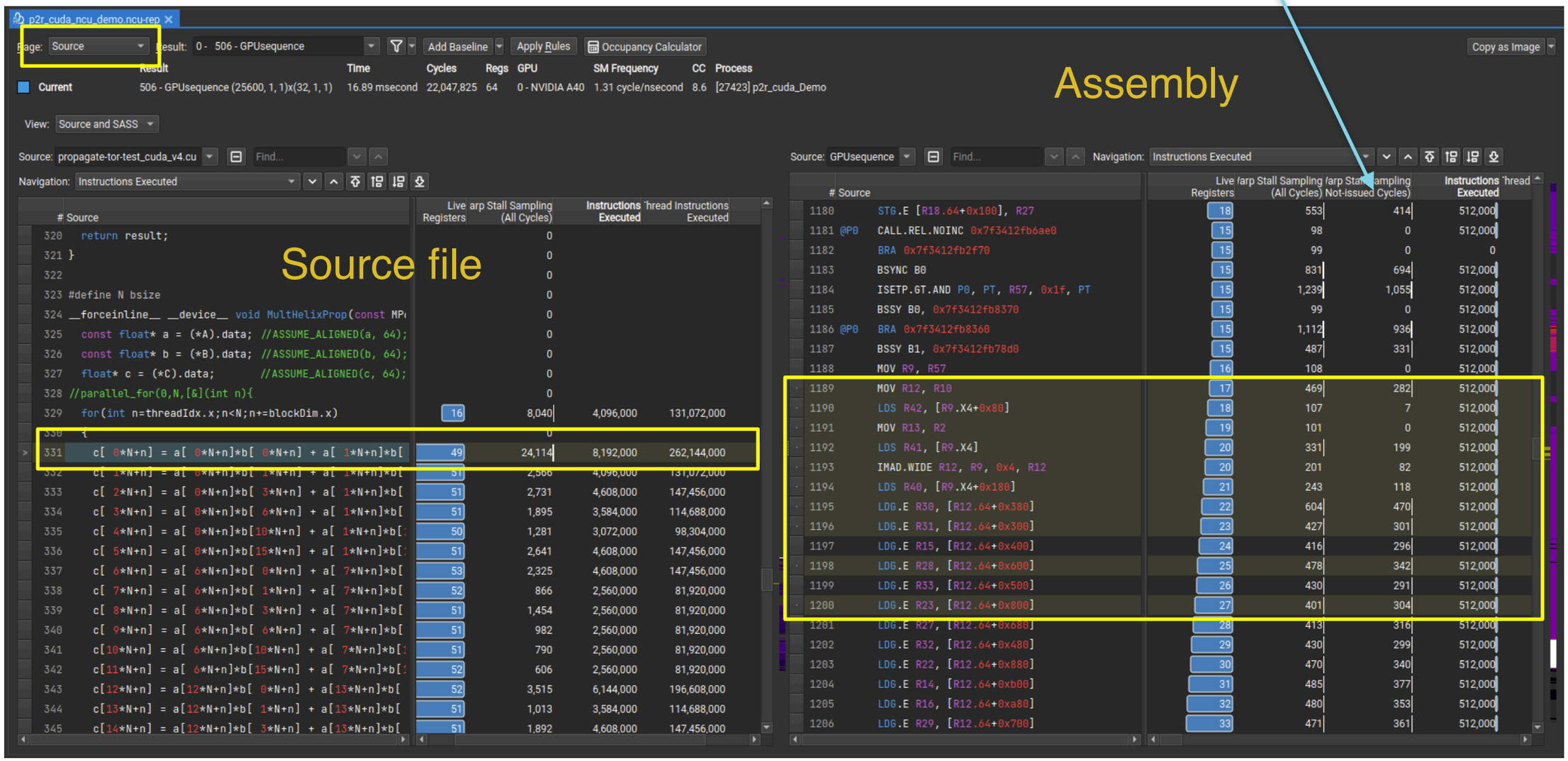


Nsight compute - Source correlation

- Another powerful functionality: source correlation
  - [Compile applications with `-lineinfo` in nvcc]

Sort by metric values for insight
e.g. hot-spot/long stall

Source page



Fermilab

DEMO

More advanced topics

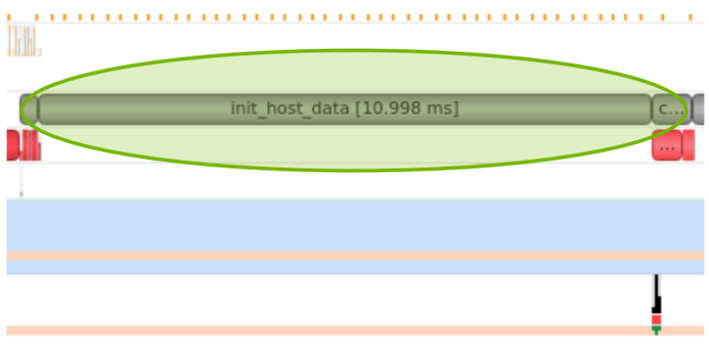
- NSight system
  - NVTX: User defined code regions for more accurate
- NSight-compute
  - User-defined metrics
  - Multi-node profiling

NVTX regions

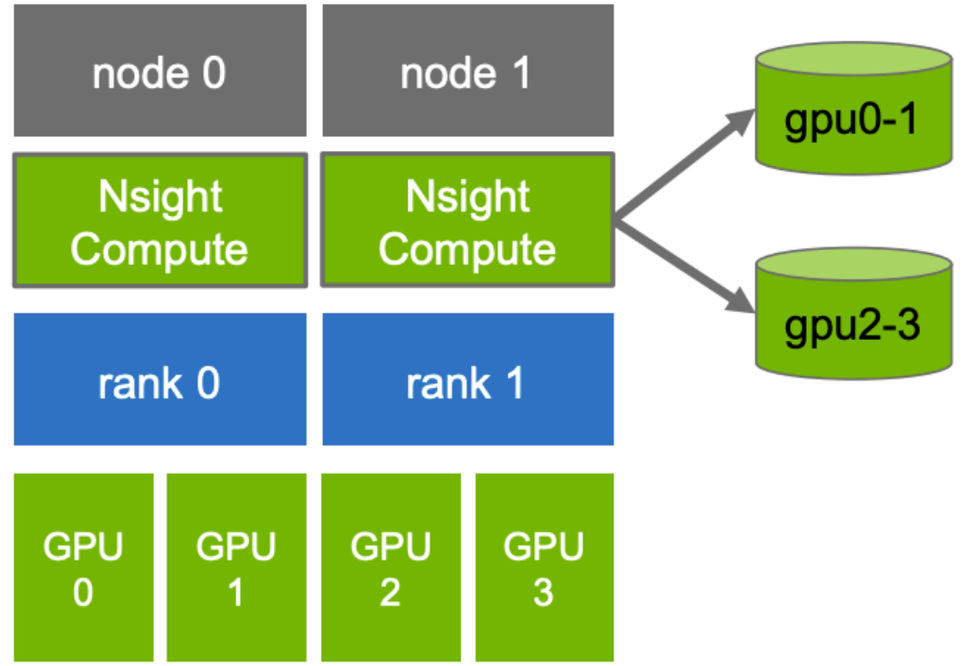
Code

```
#include "nvToolsExt.h"
...
void myfunction( int n, double * x )
{
    nvtxRangePushA("init_host_data");
    //initialize x on host
    init_host_data(n,x,x_d,y_d);
    nvtxRangePop();
}
...
```

Timeline



Muplti-node processing



Fermilab

Summary

- Introduced basic functions of NVIDIA's profiling tools
- NSight-system
  - CPU+GPU program trace for overview
- NSight-compute
  - Deep-dive into individual kernel
  - Baseline analysis
  - Source correlation

- You should be able to collect the profiling results with both tools!
  - Interpretation of results / Identifying problems requires more expertise

Resources

- Nsight system user guide
  - https://docs.nvidia.com/nsight-systems/UserGuide/index.html
- Nsight compute user guide
  - Command line interface
    https://docs.nvidia.com/nsight-compute/NsightComputeCli/index.html
  - GUI
    https://docs.nvidia.com/nsight-compute/NsightCompute/index.html

- Free tutorials for nsight-system/compute
  - Compute Accelerator Forum
    - https://indico.cern.ch/event/962112/
  - https://www.olcf.ornl.gov/calendar/nvidia-profiling-tools-nsight-systems/
  - https://www.olcf.ornl.gov/calendar/nvidia-profiling-tools-nsight-compute/
  - General CUDA training:
    - https://www.olcf.ornl.gov/calendar/fundamental-cuda-optimization-part1/