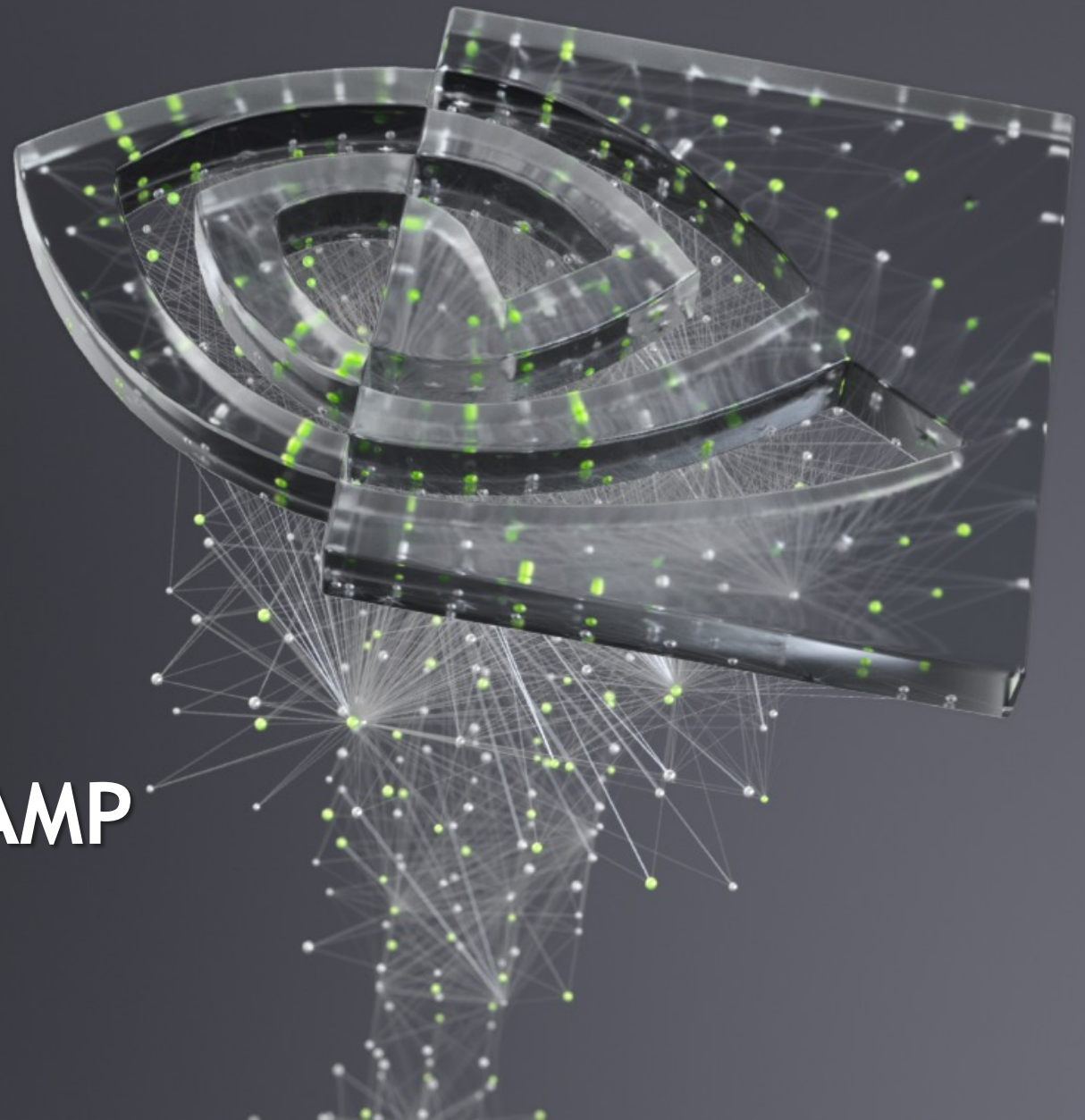


N-WAYS GPU BOOTCAMP

NSIGHT SYSTEM PROFILING

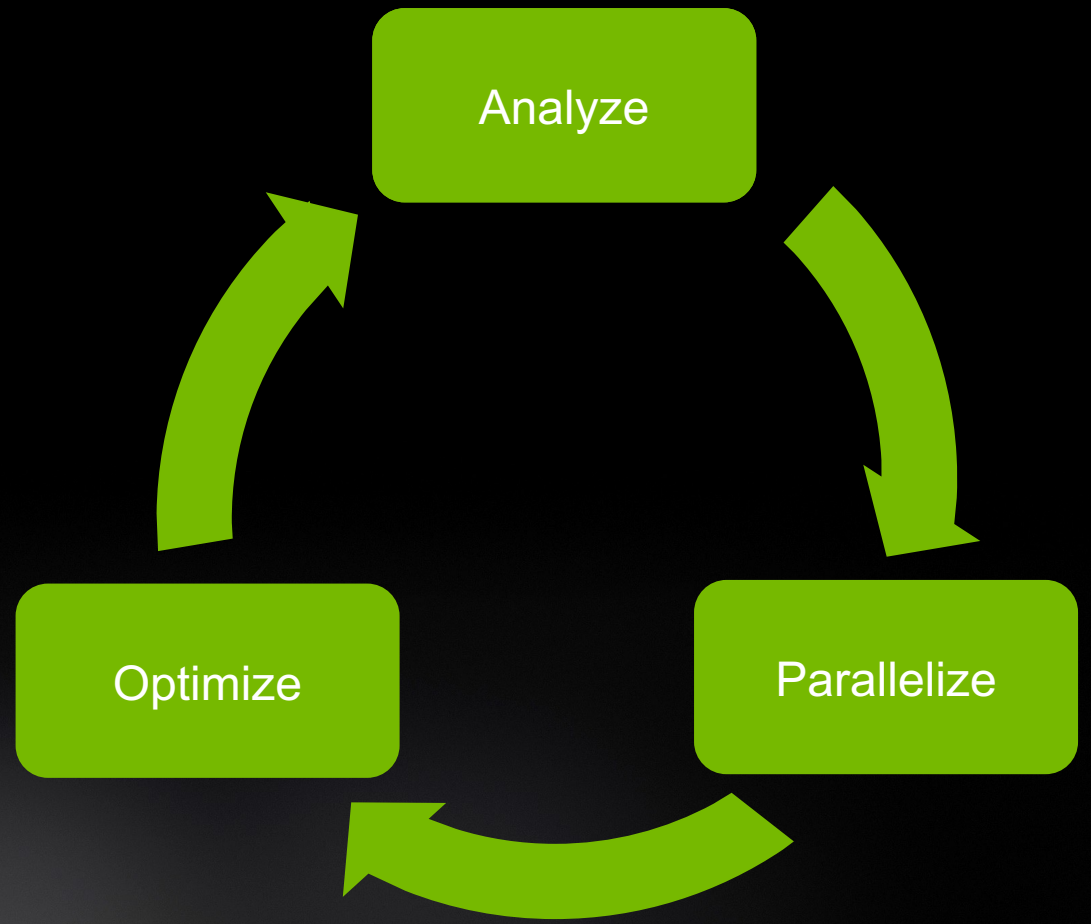
NSIGHT SYSTEM

What to expect?

- Basic introduction to Nsight family of tools
- Using Nsight Systems

DEVELOPMENT CYCLE

- **Analyze** your code to determine most likely places needing parallelization or optimization.
- **Parallelize** your code by starting with the most time consuming parts and check for correctness.
- **Optimize** your code to improve observed speed-up from parallelization.



PROFILING SEQUENTIAL CODE

Profile Your Code

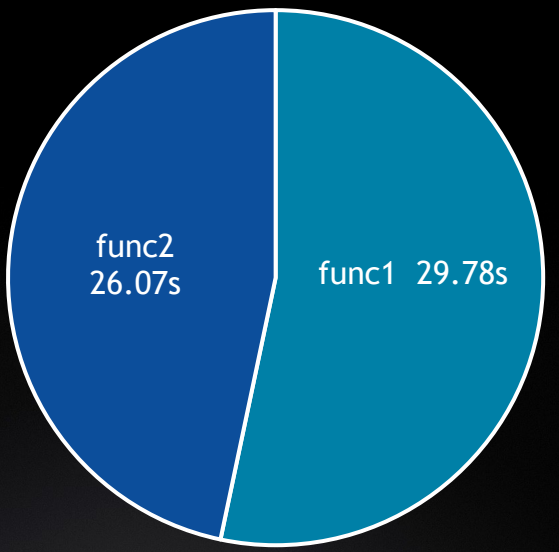
Obtain detailed information about how the code ran.

This can include information such as:

- Total runtime
- Runtime of individual routines
- Hardware counters

Identify the portions of code that took the longest to run. We want to focus on these “hotspots” when parallelizing.

Hotspot Analysis




NVIDIA NSIGHT FAMILY

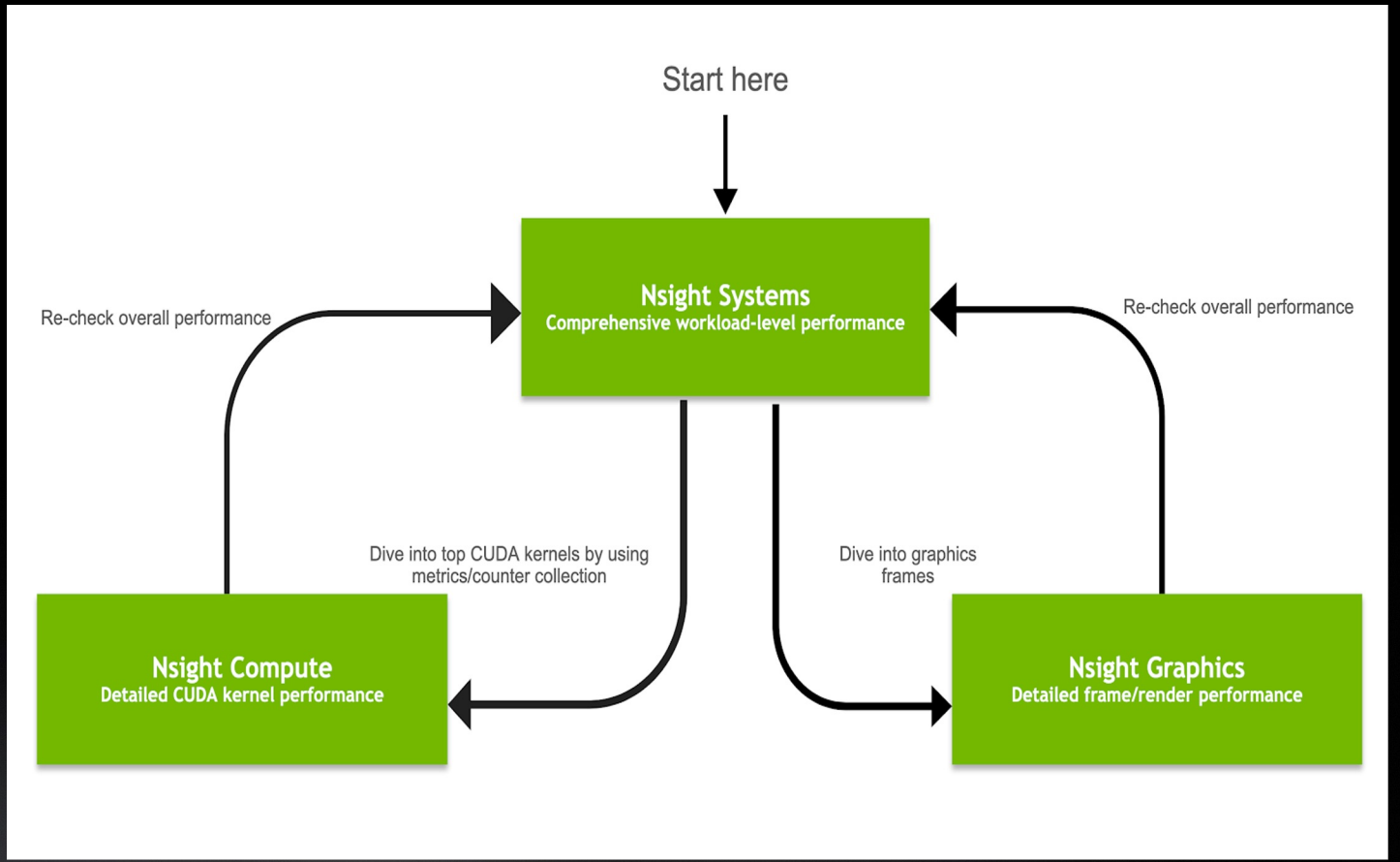
Nsight Product Family

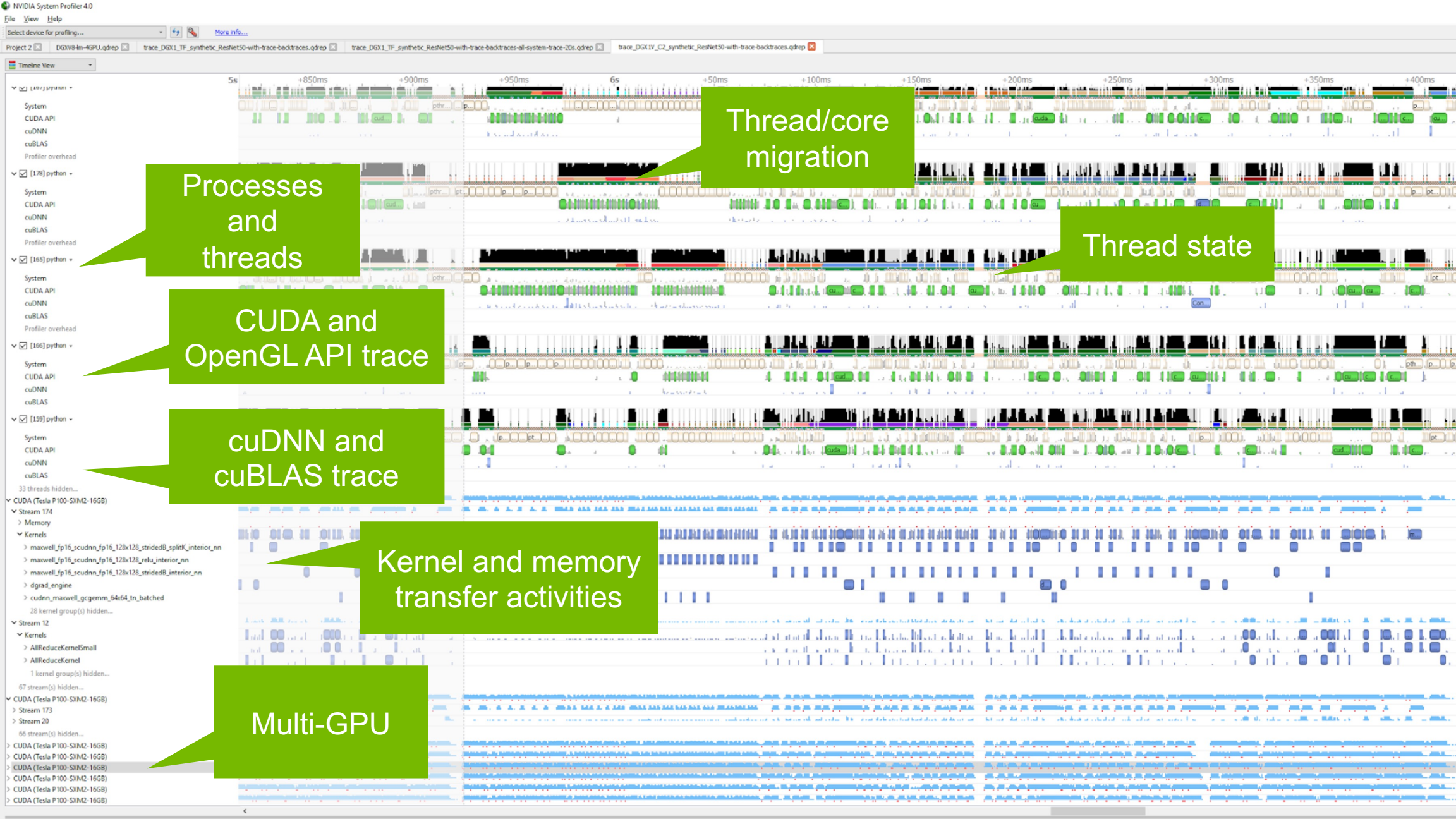
Workflow

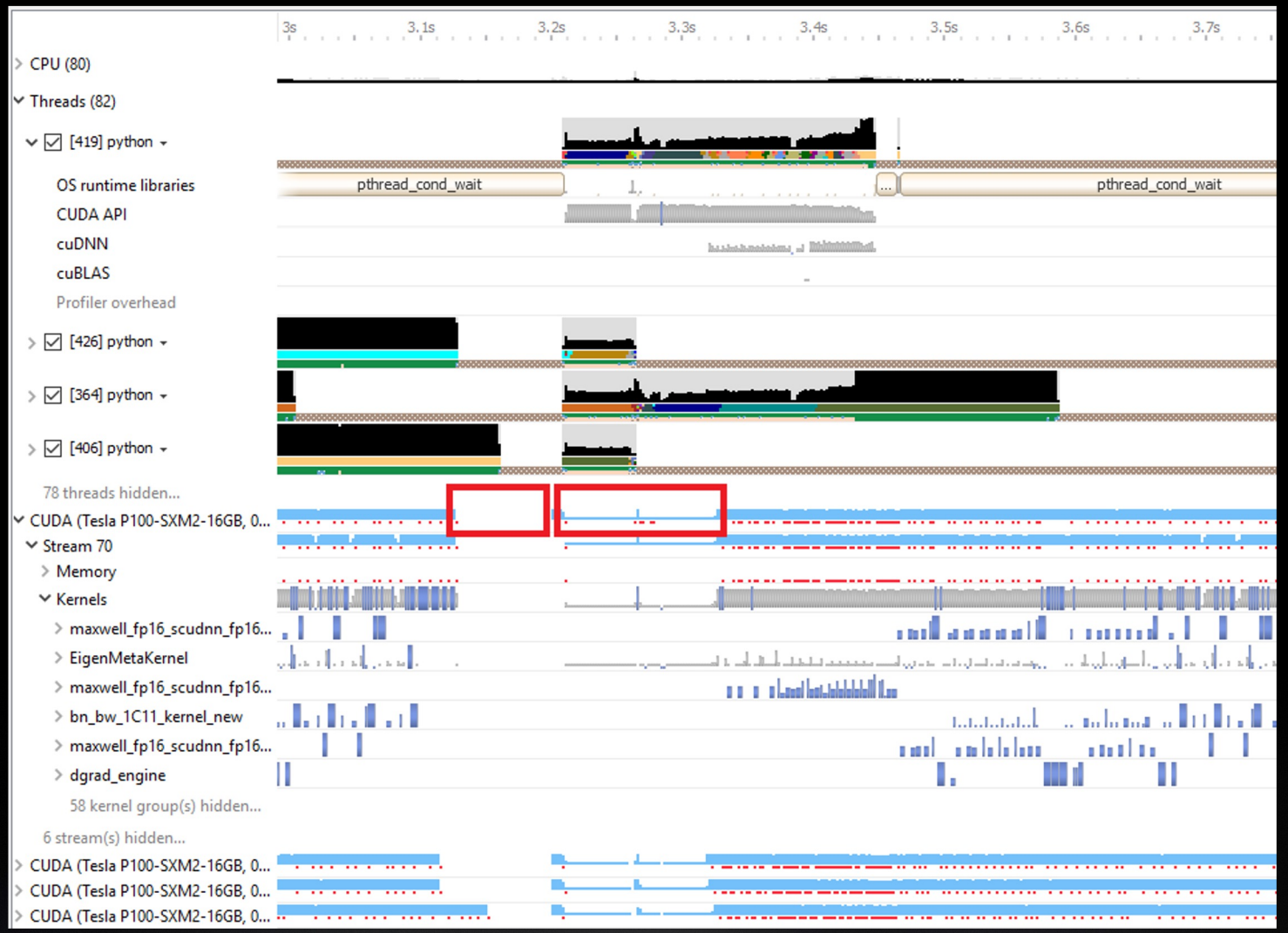
Nsight Systems - **Analyze application algorithm system-wide**

Nsight Compute - **Debug/optimize CUDA kernel**

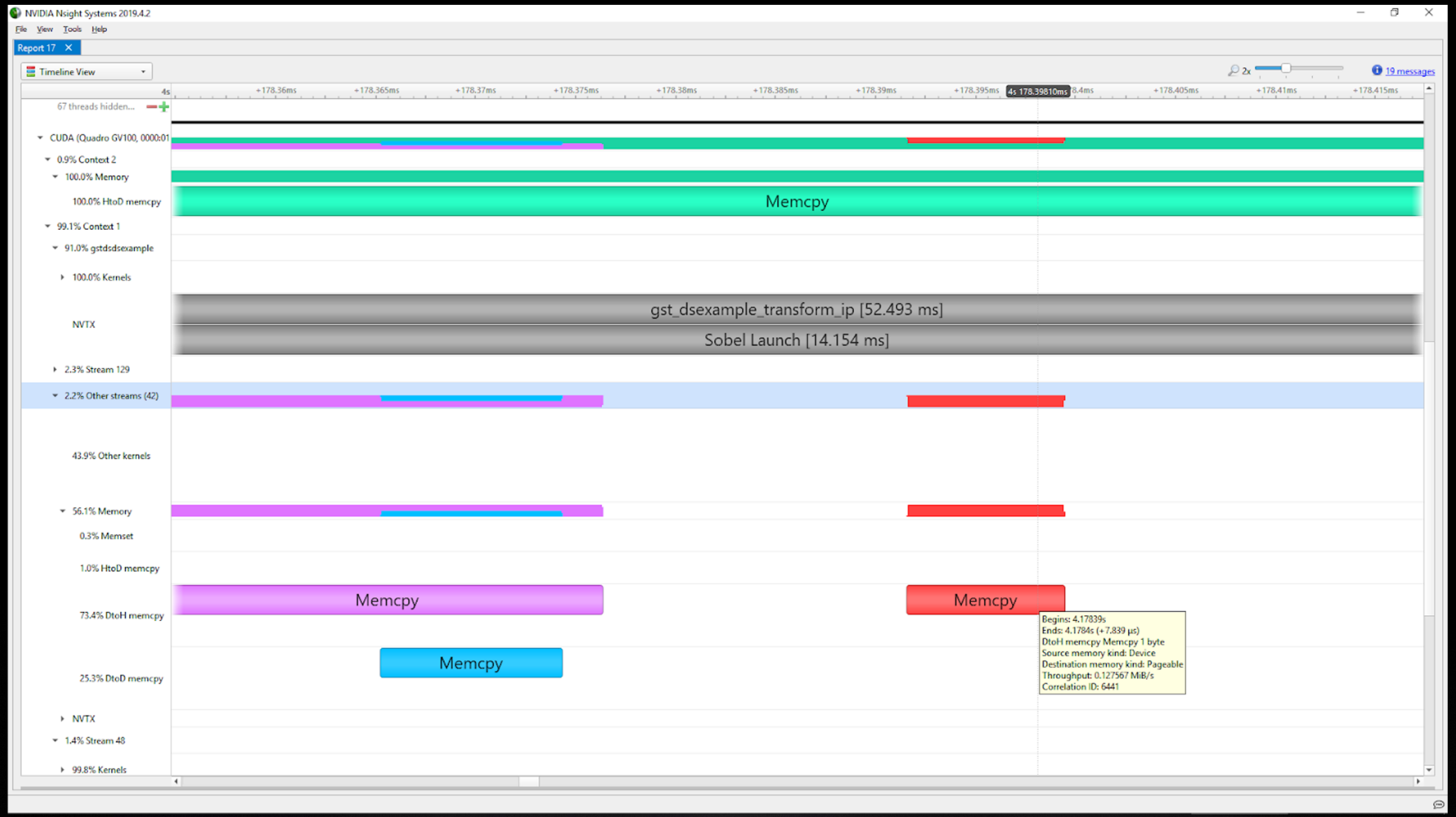
Nsight Graphics - **Debug/optimize graphics workloads**




Thread/core migration
Processes and threads
Thread state
CUDA and OpenGL API trace
cuDNN and cuBLAS trace
Kernel and memory transfer activities
Multi-GPU



GPU IDLE AND LOW UTILIZATION LEVEL OF DETAIL



CUDA MEMORY TRANSFER COLOR PALLETTE SHOW DIRECTION AND PAGEABLE MEMORY HAZARDS

PROFILING WITH NSIGHT SYSTEM AND NVTX

PROFILING SEQUENTIAL CODE

Using Command Line Interface (CLI)

NVIDIA Nsight Systems CLI provides

- Simple interface to collect data
- Can be copied to any system and analysed later
- Profiles both serial and parallel code
- For more info enter `nsys --help` on the terminal

To profile a serial application with NVIDIA Nsight Systems, we use NVIDIA Tools Extension (NVTX) API functions in addition to collecting backtraces while sampling.

PROFILING SEQUENTIAL CODE

NVIDIA Tools Extension API (NVTX) library

What is it?

- A C-based Application Programming Interface (API) for annotating events
- Can be easily integrated to the application
- Can be used with NVIDIA Nsight Systems

Why?

- Allows manual instrumentation of the application
- Allows additional information for profiling (e.g: tracing of CPU events and time ranges)

How?

- Import the header only C library `nvToolsExt.h`
- Wrap the code region or a specific function with `nvtxRangePush()` and `nvtxRangPop()`

```c
#include <string.h>
#include <stdio.h>
#include <stdlib.h>
#include <omp.h>
#include "laplace2d.h"
#include <nvtx3/nvToolsExt.h>

int main(int argc, char** argv)
{
    const int n = 4096;
    const int m = 4096;
    const int iter_max = 1000;

    const double tol = 1.0e-6;
    double error = 1.0;

    double *restrict A    = (double*)malloc(sizeof(double)*n*m);
    double *restrict Anew = (double*)malloc(sizeof(double)*n*m);

    nvtxRangePushA("init");
    initialize(A, Anew, m, n);
    nvtxRangePop();

    printf("Jacobi relaxation Calculation: %d x %d mesh\n", n, m);

    double st = omp_get_wtime();
    int iter = 0;

    nvtxRangePushA("while");
    while ( error > tol && iter < iter_max )
    {
        nvtxRangePushA("calc");
        error = calcNext(A, Anew, m, n);
        nvtxRangePop();

        nvtxRangePushA("swap");
        swap(A, Anew, m, n);
        nvtxRangePop();

        if(iter % 100 == 0) printf("%5d, %0.6f\n", iter, error);

        iter++;
    }
    nvtxRangePop();

    double runtime = omp_get_wtime() - st;

    printf(" total: %f s\n", runtime);

    deallocate(A, Anew);

    return 0;
}
```

jacobi.c
(starting and ending of ranges are highlighted with the **same** color)

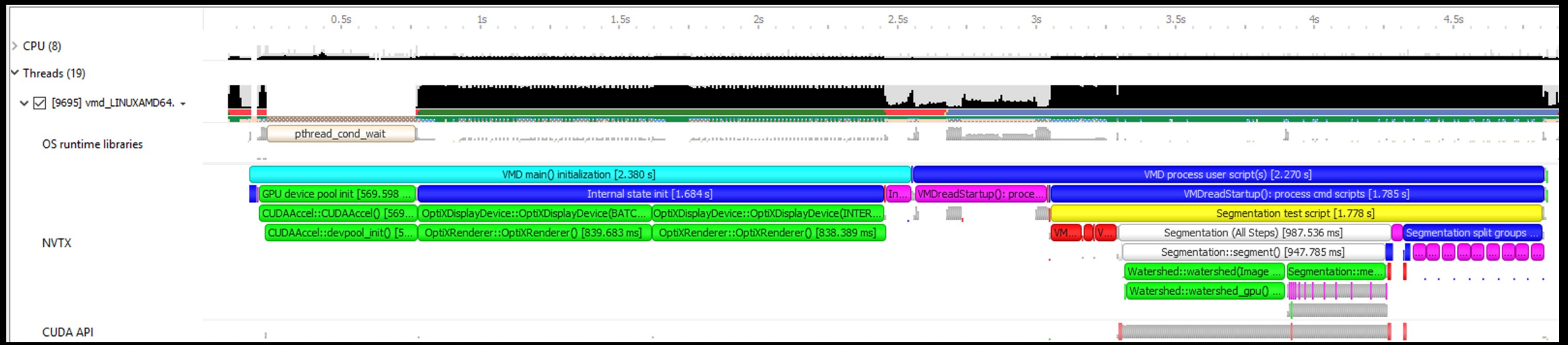
-t	Selects the APIs to be traced (nvtx in this example)
--status	if true, generates summary of statistics after the collection
-b	Selects the backtrace method to use while sampling. The option dwarf uses DWARF's CFI (Call Frame Information).
--force-overwrite	if true, overwrites the existing results
-o	sets the output (qdrep) filename

```
mozhgank@prm-dgx-32:~/Code/openacc-training-materials/labs/module4/English/C/solutions/parallel$ nsys profile -t nvtx --stats=true -b dwarf --force-overwrite true -o laplace-seq ./laplace-seq
Collecting data...
Jacobi relaxation Calculation: 4096 x 4096 mesh
    0, 0.250000
  100, 0.002397
  200, 0.001204
  300, 0.000804
  400, 0.000603
  500, 0.000483
  600, 0.000403
  700, 0.000345
  800, 0.000302
  900, 0.000269
 total: 55.754501 s
Processing events...
Capturing symbol files...
Saving intermediate "/home/mozhgank/Code/openacc-training-materials/labs/module4/English/C/solutions/parallel/laplace-seq.qdstrm" file to disk...

Importing [===============================================================100%]
Saved report file to "/home/mozhgank/Code/openacc-training-materials/labs/module4/English/C/solutions/parallel/laplace-seq.qdrep"
Exporting 70802 events: [=================================================100%]

Exported successfully to
/home/mozhgank/Code/openacc-training-materials/labs/module4/English/C/solutions/parallel/laplace-seq.sqlite

Generating NVTX Push-Pop Range Statistics...
NVTX Push-Pop Range Statistics (nanoseconds)

Time(%)      Total Time   Instances         Average         Minimum         Maximum  Range
-------  --------------  ----------  --------------  --------------  --------------  --------
   49.9     55754497966           1   55754497966.0     55754497966     55754497966  while
   26.5     29577817696        1000      29577817.7        29092956        65008545  calc
   23.4     26163892482        1000      26163892.5        25761418        60129514  swap
    0.1       137489808           1     137489808.0       137489808       137489808  init
```

NVTX range statistics

"calc" region (calcNext function) takes 26.6%
"swap" region (swap function) takes 23.4% of total execution time

Open laplace-seq.qdrep with Nsight System GUI to view the timeline



USER ANNOTATIONS APIS FOR CPU & GPU
NVTX, OPENGL, VULKAN, AND DIRECT3D PERFORMANCE MARKERS

EXAMPLE: VISUAL MOLECULAR DYNAMICS (VMD) ALGORITHMS VISUALIZED WITH NVTX ON CPU

REFERENCES

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

https://developer.nvidia.com/hpc-sdk


THANK YOU
NVIDIA