Profiling/Optimizing GPU Offload Application using Intel Analyzer Tools

Cory Levels – Technical Consulting Engineer
Xiao Zhu– Technical Consulting Engineer
Rupak Roy– Technical Consulting Engineer

Agenda

1. **Introduction**
   Overview of Analysis Tools
2. **Intel® Advisor**
   GPU Offload Modeling, GPU Roofline Analysis
3. **VTune™ Profiler**
   Work Scheduling Balancing, Data Transfer, GPU Occupancy
4. **Recommended Workflow**
   When are Advisor and VTune most helpful?
5. **Q&A**

Introduction

Overview of Analysis Tools

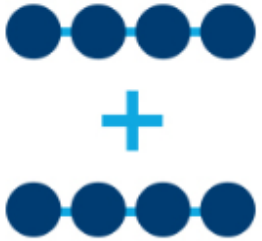
Intel® Advisor

Offload Advisor

Design your code for efficient GPU offload, even before you have the hardware

Automated Roofline Analysis

See performance headroom against hardware limitations. Get insights for an effective optimization roadmap.



Vectorization Optimization

Enable more vector parallelism and improve its efficiency

Thread Prototyping

Model, tune and test multiple threading designs

Build Heterogeneous Algorithms

Create and analyze data flow and dependency computation graphs

Learn More: software.intel.com/advisor

Optimize Performance

Intel® VTune™ Profiler

Get the Right Data to Find Bottlenecks

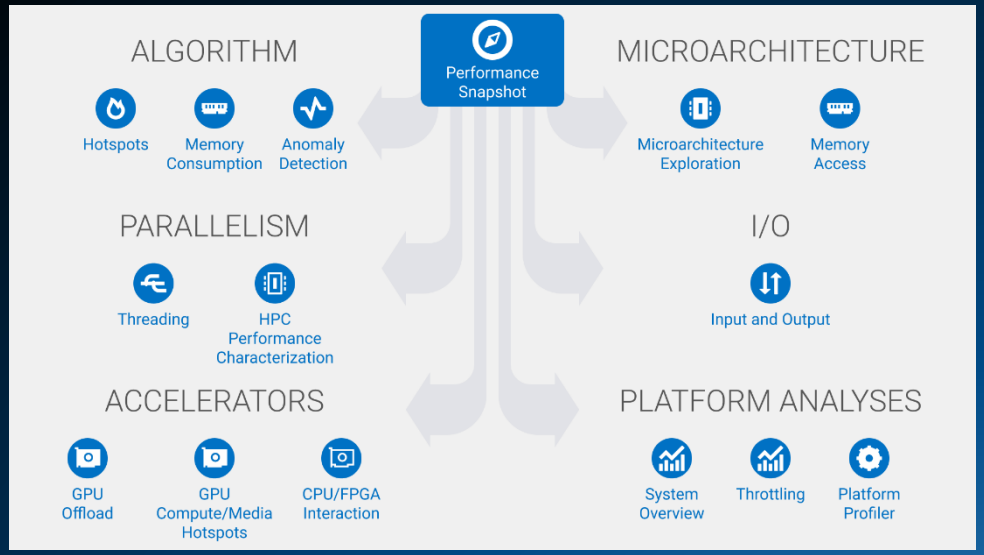
- A suite of profiling for CPU, GPU, FPGA, threading, memory, cache, storage, offload, power…
- Application or system-wide analysis
- DPC++, C, C++, Fortran, Python*, Go*, Java*, or a mix
- Linux, Windows, FreeBSD, Android, Yocto and more
- Containers and VMs

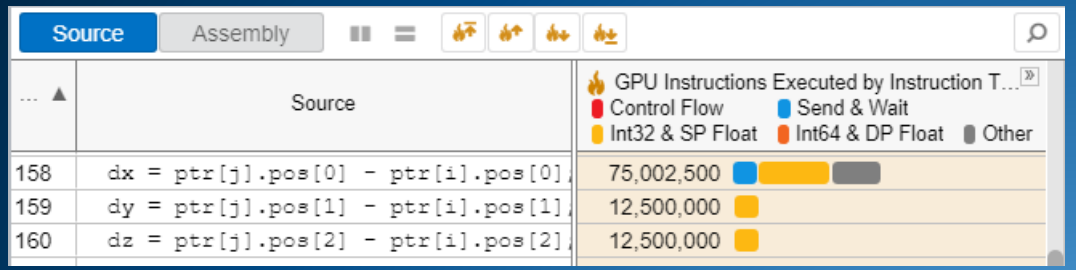
Analyze Data Faster

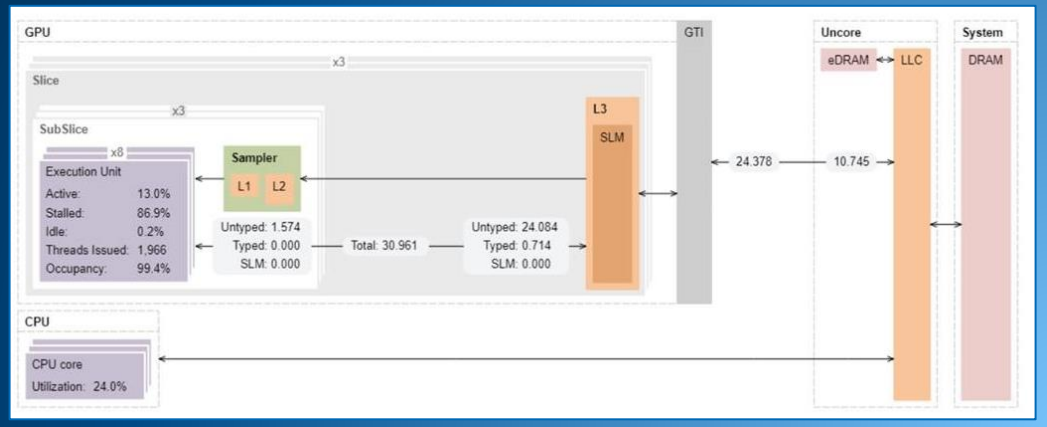
- Collect data HW/SW sampling and tracing w/o re-compilation
- See results on your source, in architecture diagrams, as a histogram, on a timeline…
- Filter and organize data to find answers

Work Your Way

- User interface or command line
- Profile locally and remotely
- GUI (desktop or web) or command line

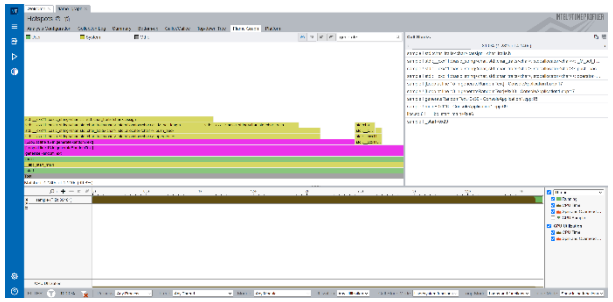






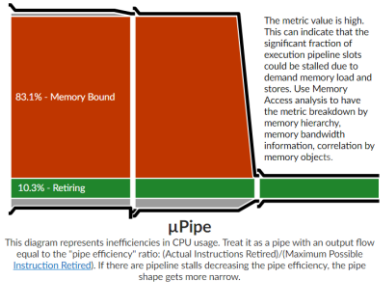
Rich Set of Profiling Capabilities

Intel® VTune™ Profiler



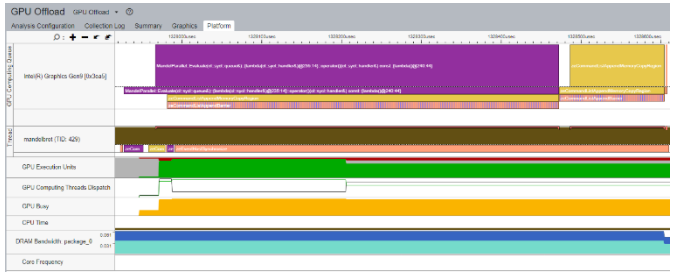
Algorithm Optimization

- ✓ Hotspots
- ✓ Anomaly Detection
- ✓ Memory Consumption



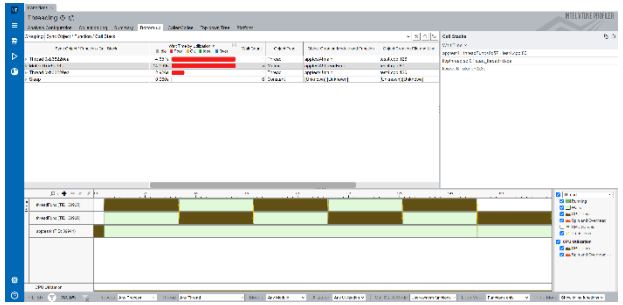
Microarch.&Memory Bottlenecks

- ✓ Microarchitecture Exploration
- ✓ Memory Access



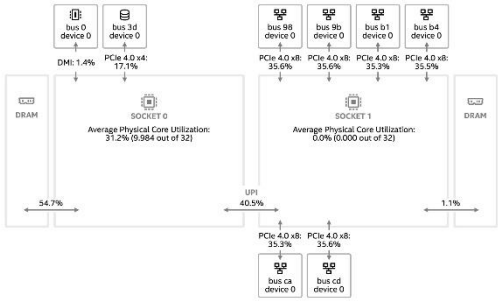
Accelerators / xPU

- ✓ GPU Offload
- ✓ GPU Compute / Media Hotspots
- ✓ CPU/FPGA Interaction



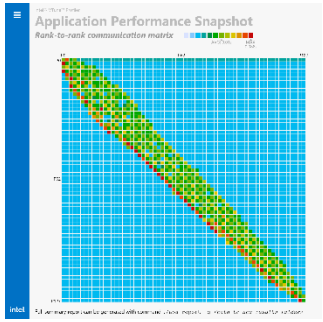
Parallelism

- ✓ Threading
- ✓ HPC Performance Characterization



Platform & I/O

- ✓ Input and Output
- ✓ System Overview
- ✓ Platform Profiler



Multi-Node

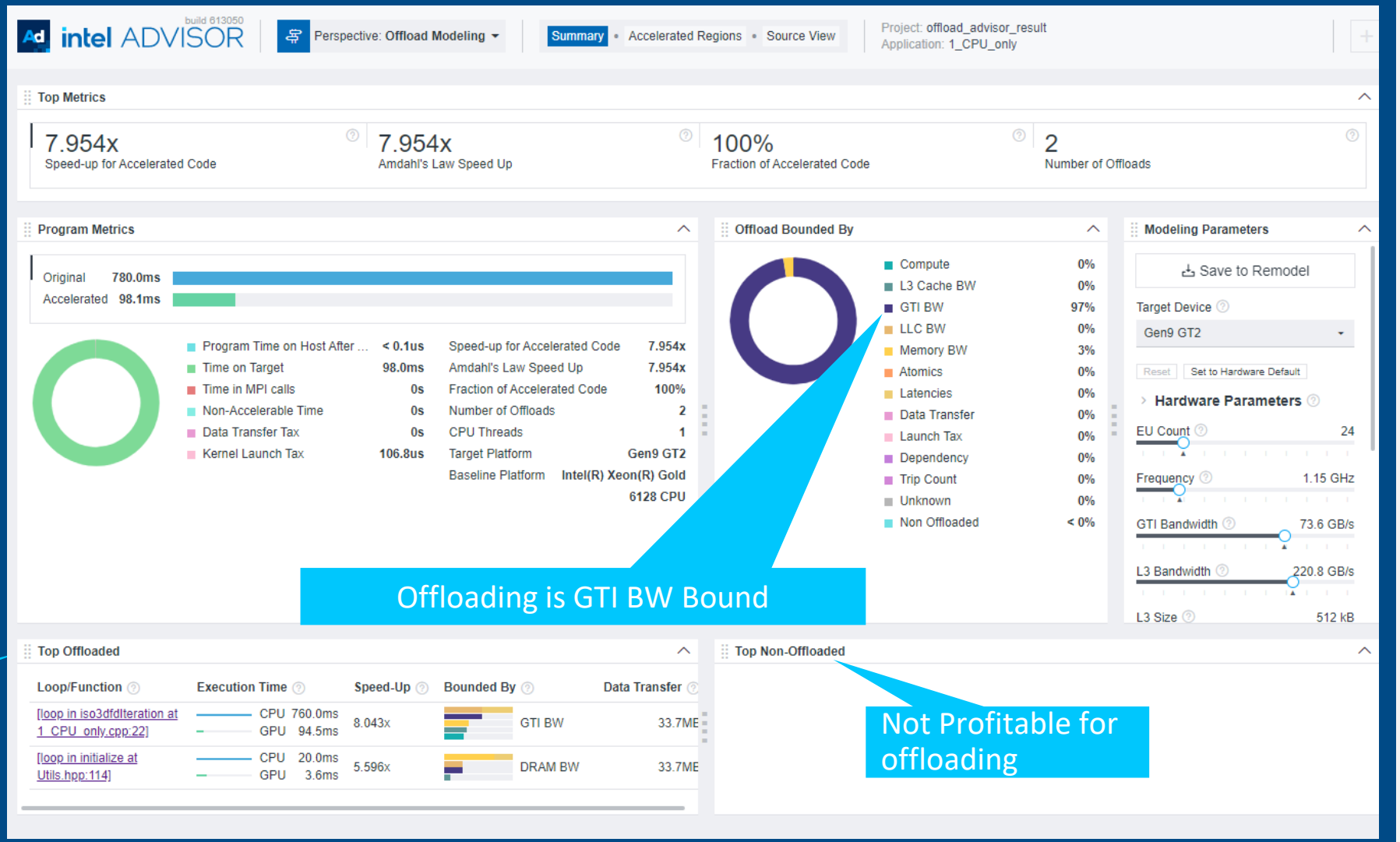
- ✓ Application Performance Snapshot

Intel® Advisor

Offload Advisor

Intel® Advisor

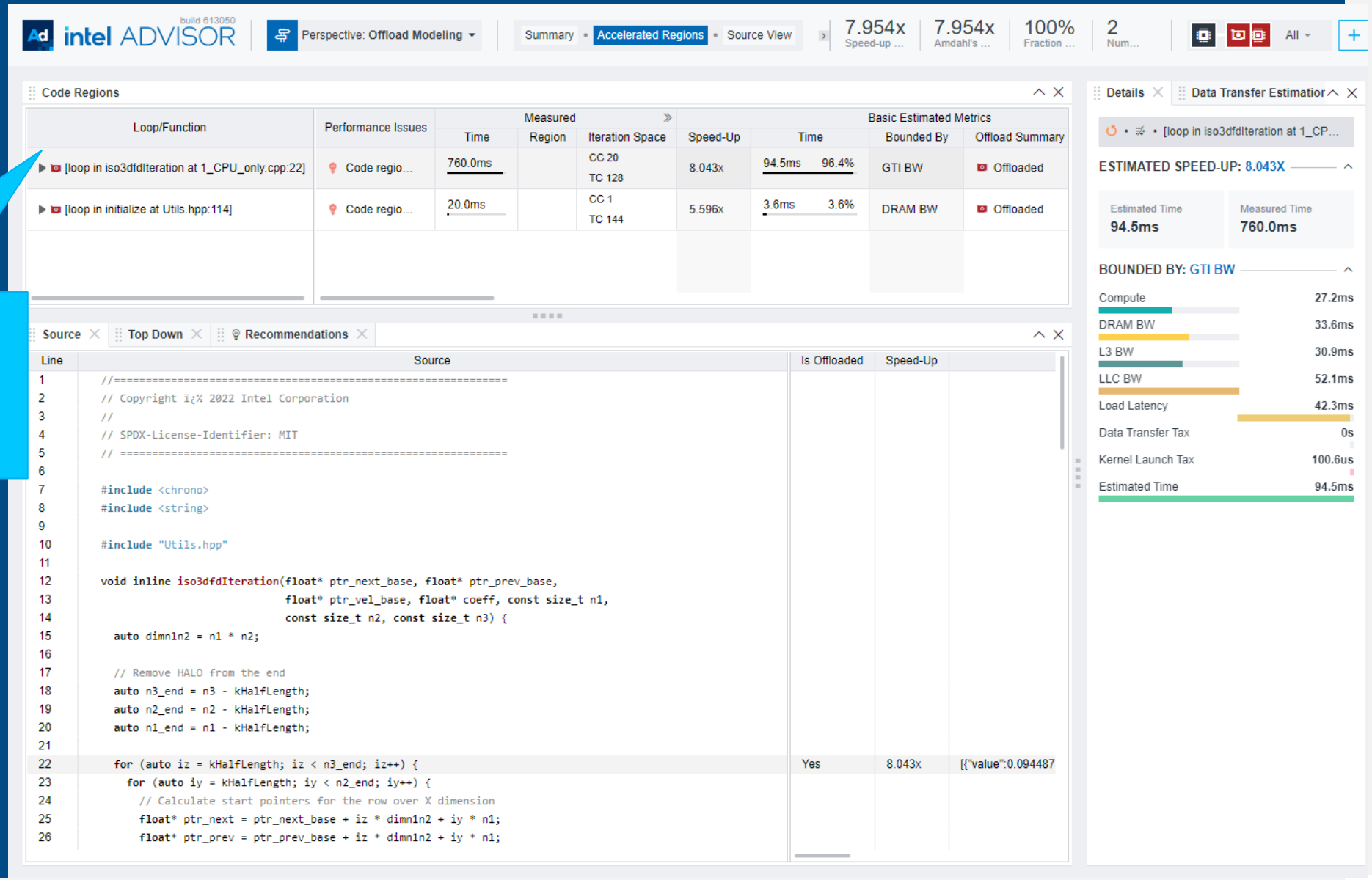
- Run on CPU or GPU – Predict for GPU
- Helps to define which section of the code should run on given accelerator
- Provides performance projection on accelerators



In-Depth Analysis of Top Offload Regions

Intel® Advisor

Loop at 1_CPU_only.cpp:22 is recommended for offloading
- GTI BW Bound
- Estimated to run on GPU in 94.5ms

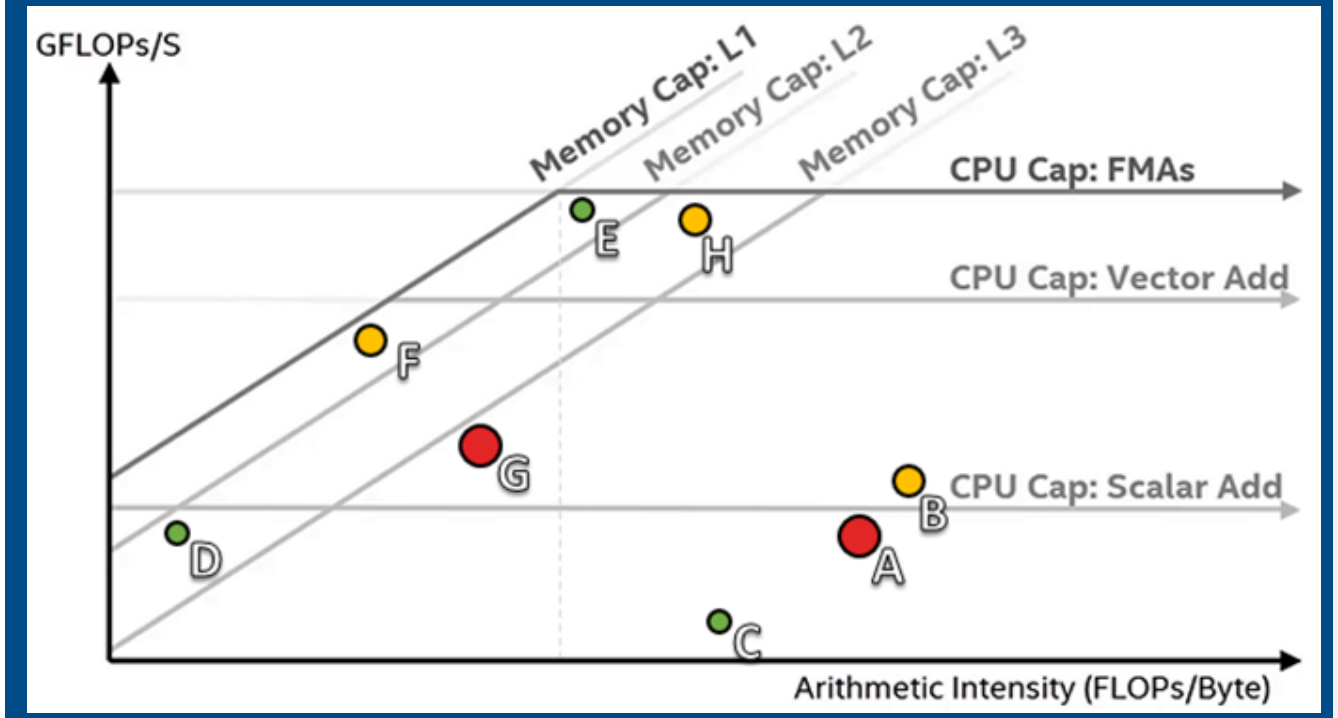


Identifying Good Optimization Candidates

Focus optimization effort where it makes the most difference

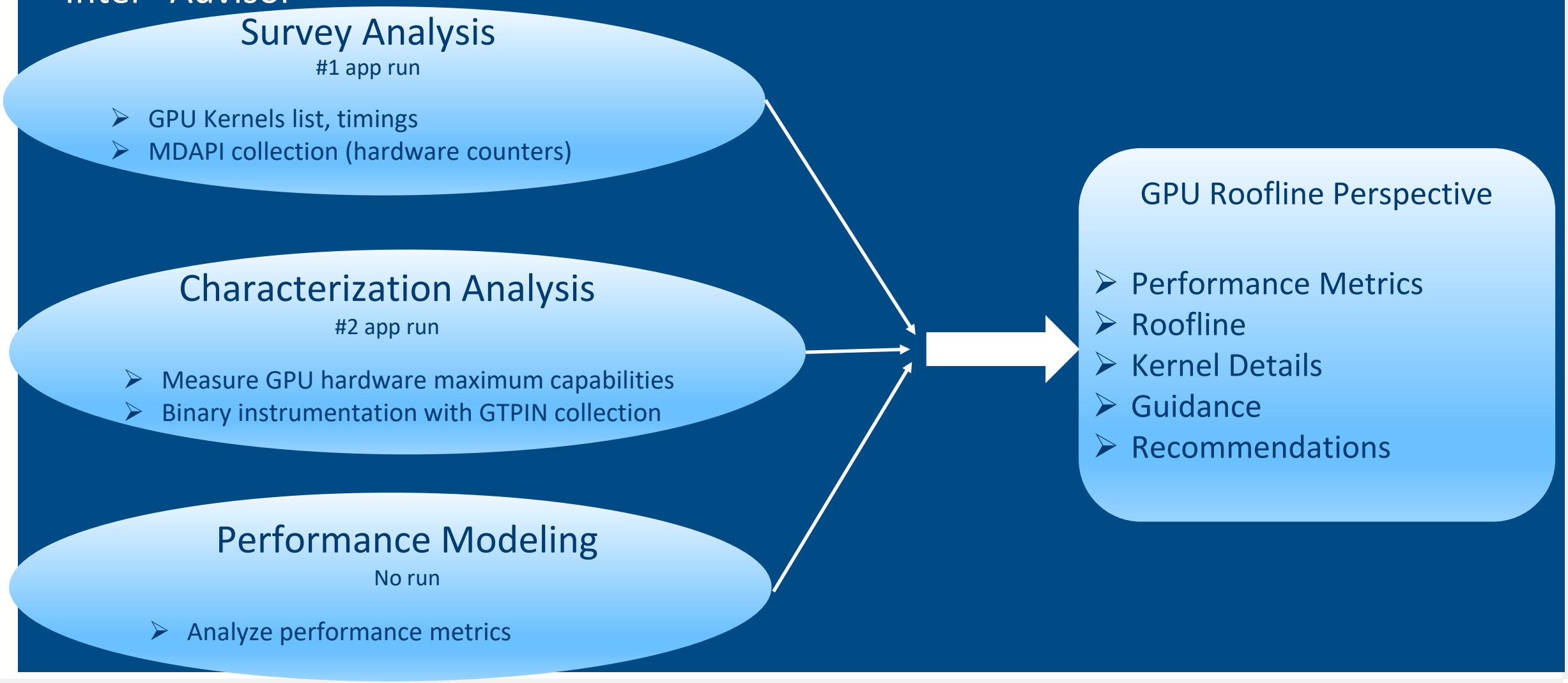
- Large, red loops have the most impact
- Loops far from the upper roofs have more room to improve

Additional roofs can be plotted for specific computation types or cache levels



GPU Analysis Flow

Intel® Advisor



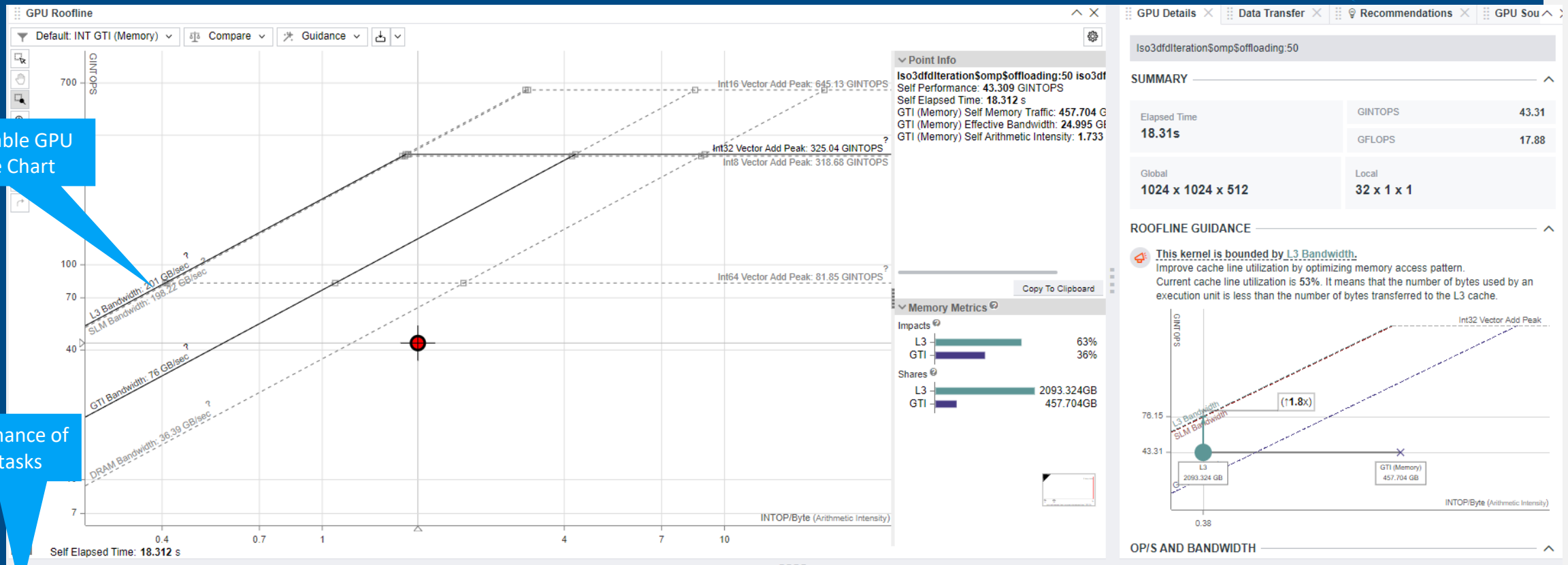
GPU Roofline Chart

Intel® Advisor

View GPU Details, Data Transfer, GPU Source ans Assembly info

Customizable GPU Roofline Chart

GPU performance of compute tasks



GPU Kernels

Kernel	Elapsed Time	Performance Issues	GPU Compute Performance				Work Size		Kernel Type	Data Transferred		Kernel Details	
			FLOAT Operations		INT Operations		Global	Local		Total Size	Total Time	Average Time	SIMD Width
Iso3dfdIterationompoffloading:50	18.312s	Private memory usage is present	327.491 GOp 17.884 GOp/s	AI 0.716	793.059 GOp 43.309 GOp/s	AI 1.733	1024 x 1024 x 512	32 x 1 x 1	Compute		0s	1.831s	32
zeCommandListAppendMemoryCopy	0.000s		0.000 GOp 0.000 GOp/s	AI 0.000	0.000 GOp 0.000 GOp/s	AI 0.000			Host-to-De...	5.1 KB		0.000s	
[Outside any task]	31.765s		0.000 GOp 0.000 GOp/s	AI 0.000	0.000 GOp 0.000 GOp/s	AI 0.000			[Unknown]		1.021s	0s	

Profiling/Optimizing GPU Offload Application using Intel® VTune™ Profiler

Rupak Roy– Technical Consulting Engineer

Xiao Zhu– Technical Consulting Engineer

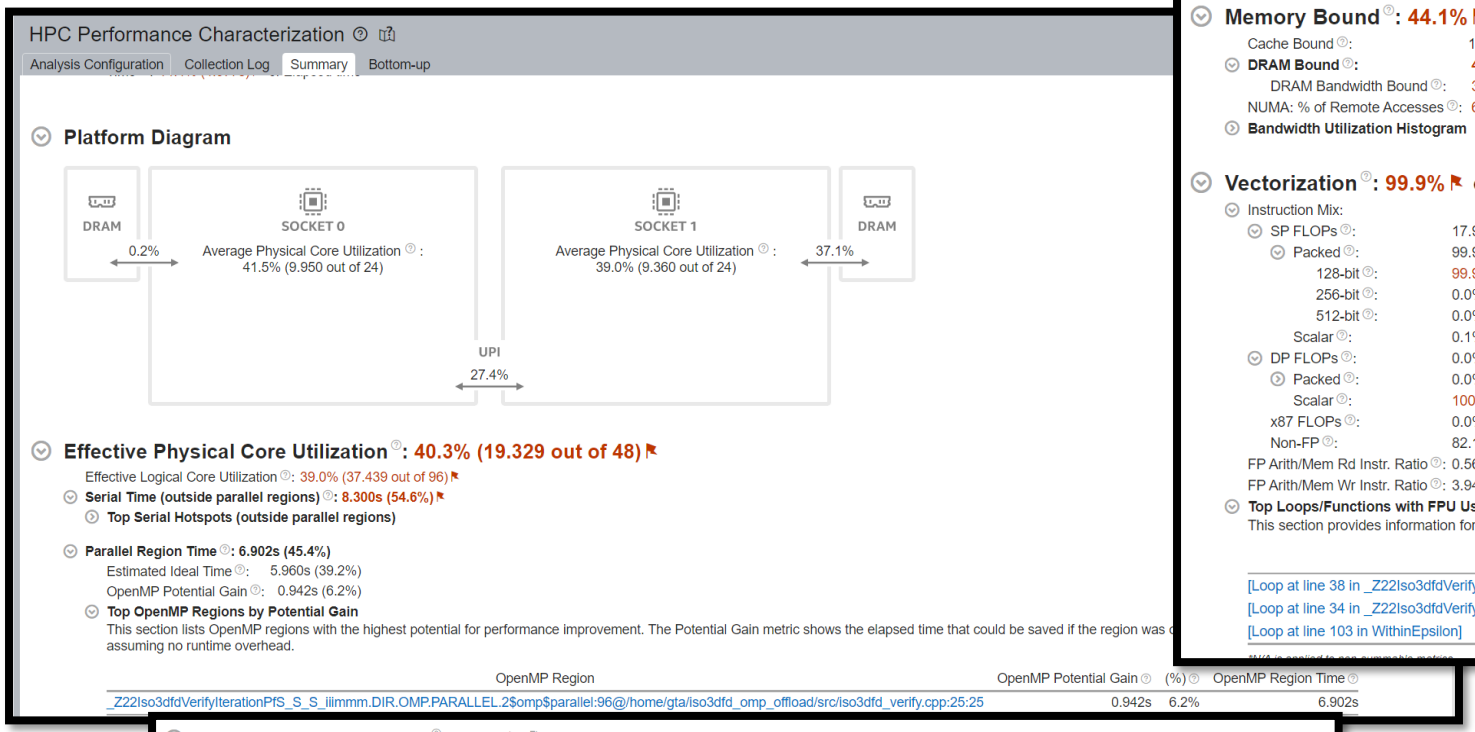
Cory Levels – Technical Consulting Engineer

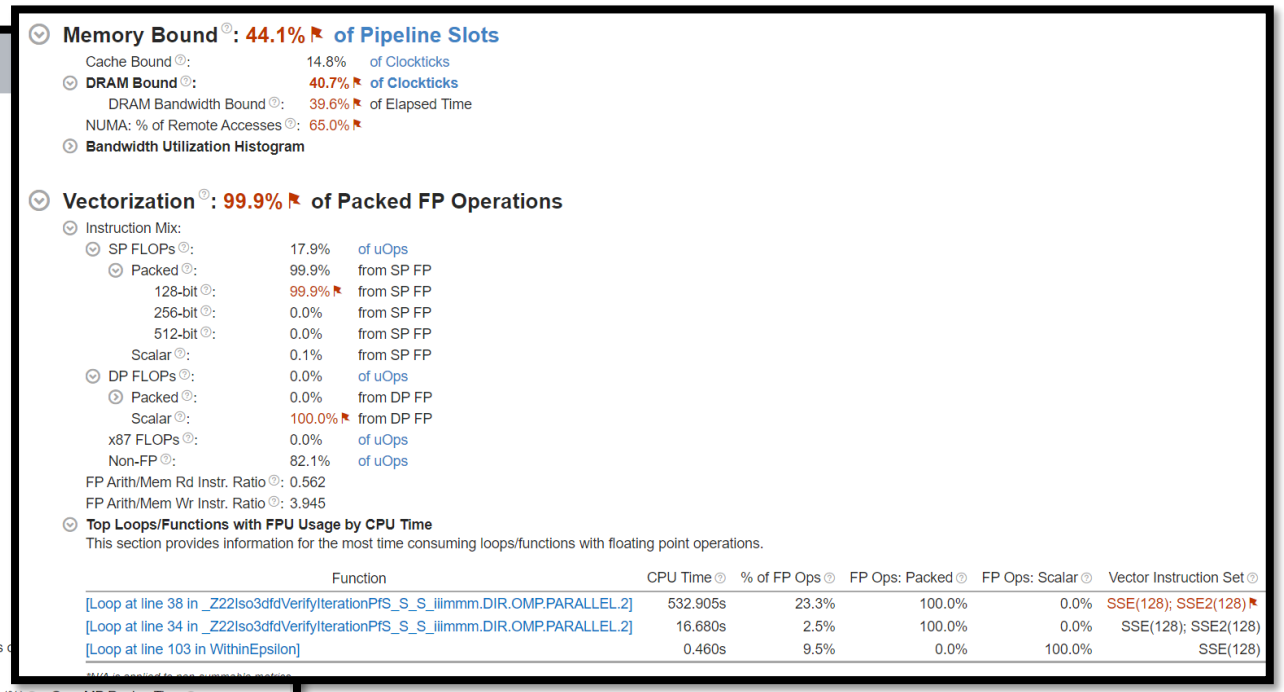
Agenda

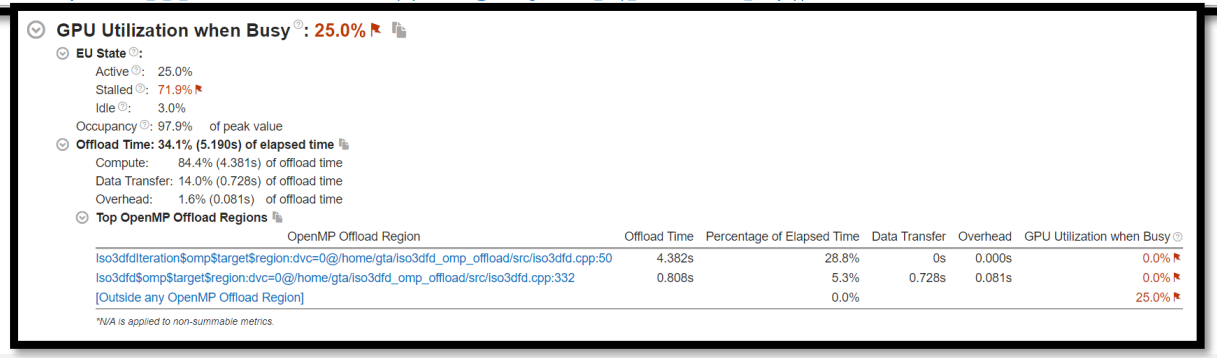
- GPU Profiling Overview
- GPU Performance Issues
- OpenMP Offload Case Study
- Conclusion

Intel® VTune™ Profiler

HPC Performance Characterization







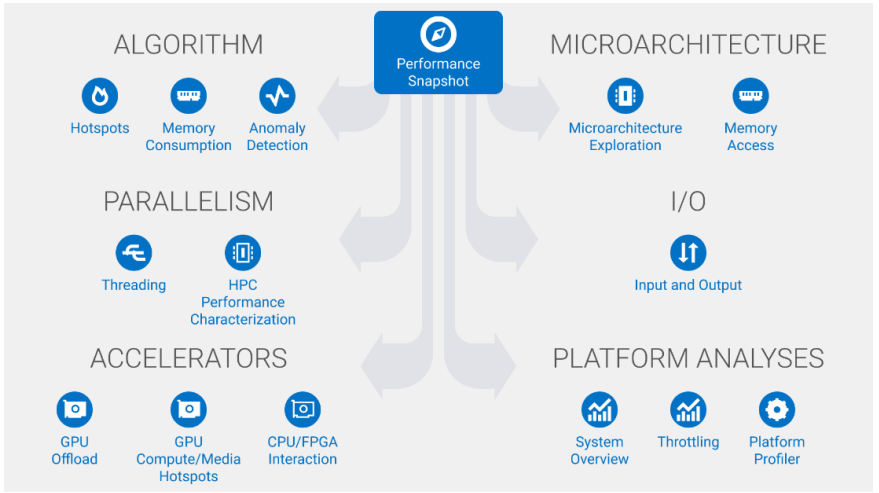
A starting point for performance optimization

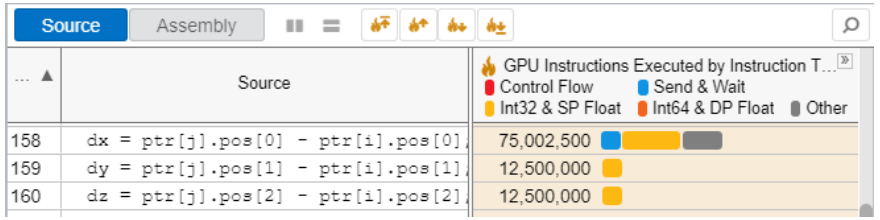
- CPU/GPU usage, Memory efficiency, and Floating-point utilization

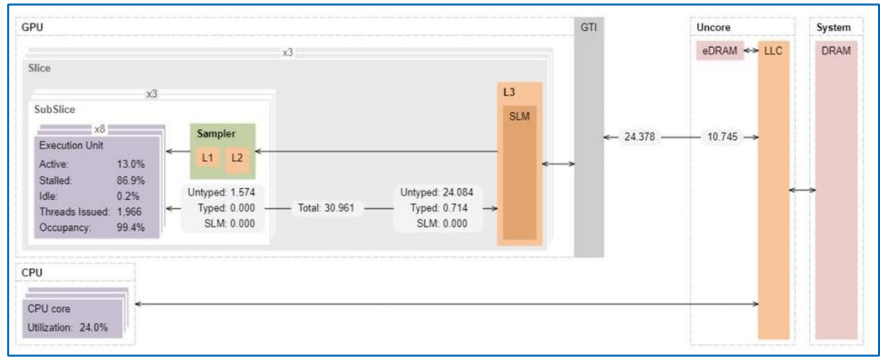
GPU Performance Problems

Addressing performance issues with dynamic analysis tools

- Work Distribution
- Data transfer
- GPU occupancy
- Memory access
- Kernel inefficiencies
- Non-scaling implementations
- …



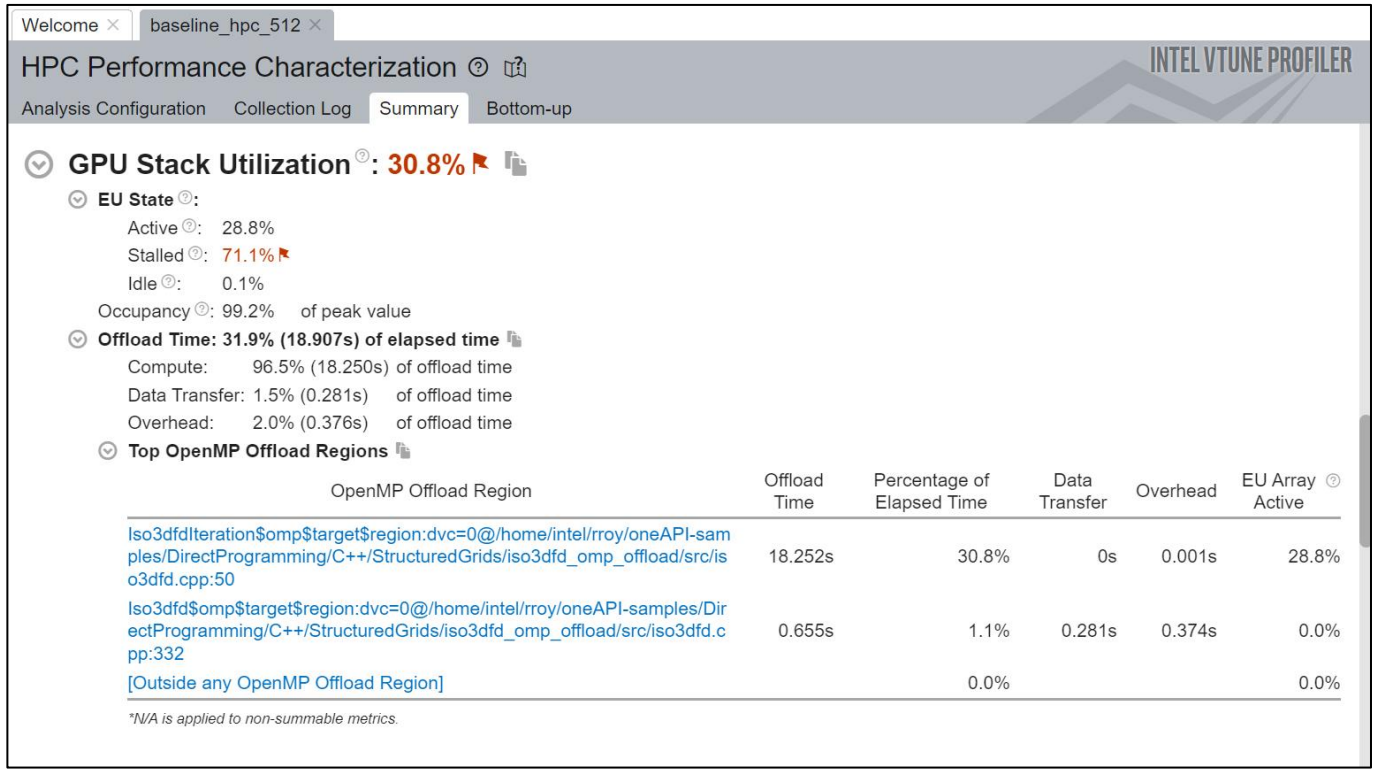




Work Distribution

Work distribution among computing resources

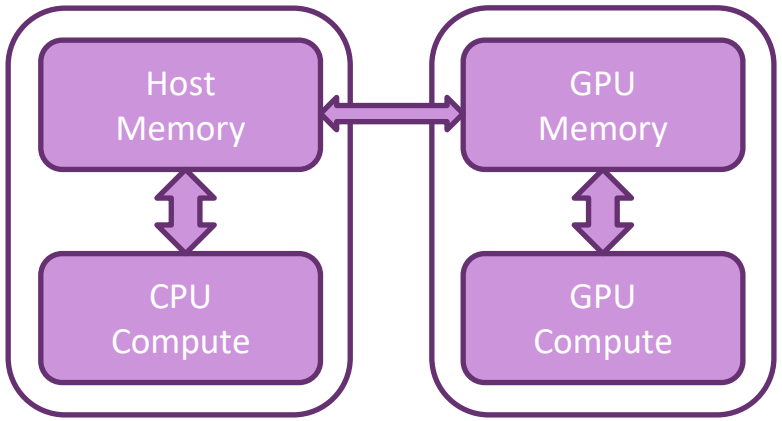
- CPU or GPU bound?
- GPU Utilization for OpenMP regions
- EU/XVEs efficiency (Active, Stalled, Idle)
- Offload Time characterization
  - Compute
  - Data Transfer
  - Overhead



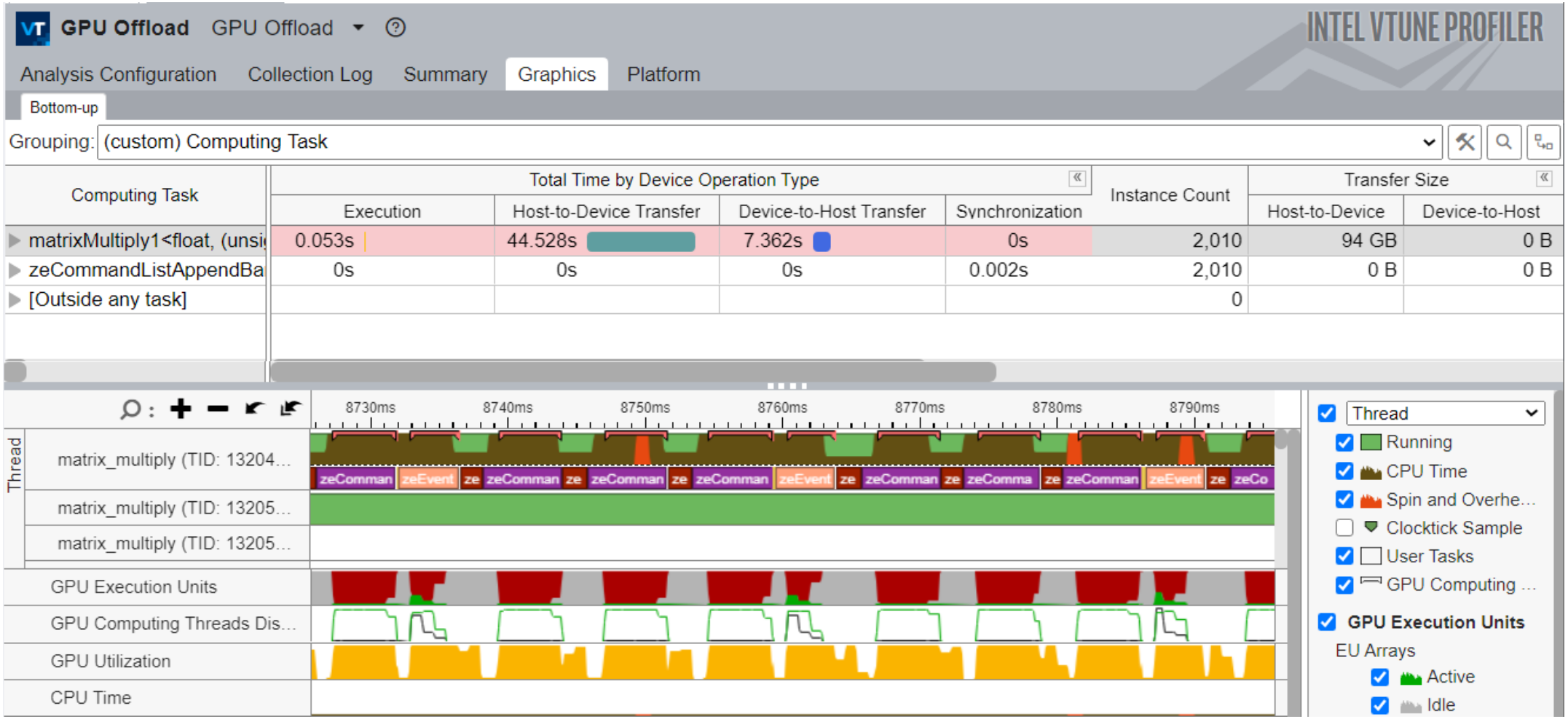
Host and GPU Data Transferring

A commonly known problem of host-to-device transfer performance

- Data transfer time
- Amount of transferred data
- Transfer direction
- Execution time



Graphics View of GPU Offload

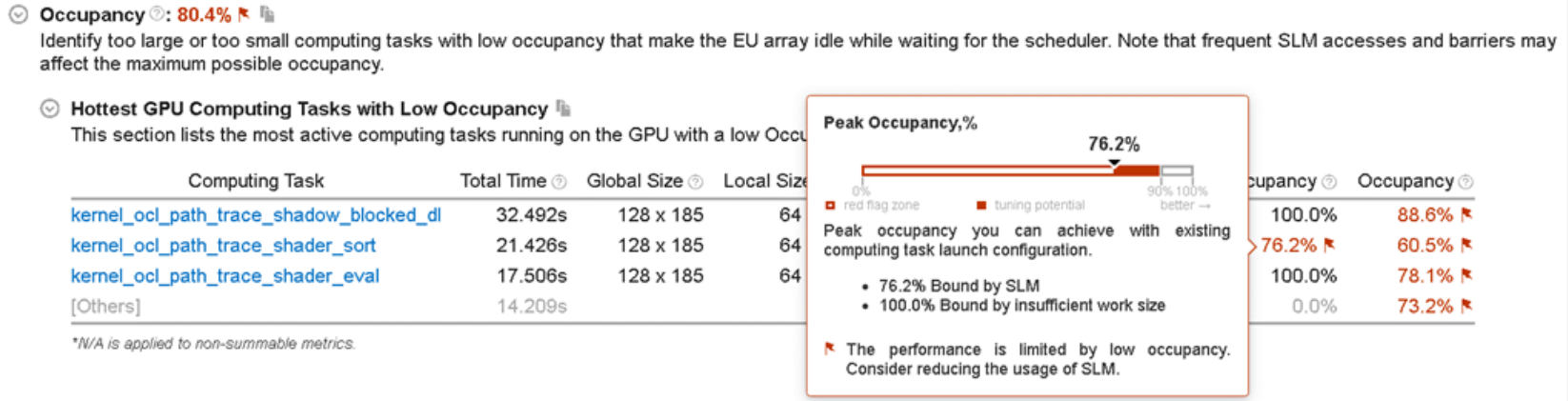


Achieving High XVE Threads Occupancy

Occupancy analysis helps identifying ground problems with work mapping

- Detecting workgroups by global and local sizes
- SIMD Width
- Barriers usage
- Tiny/huge kernels

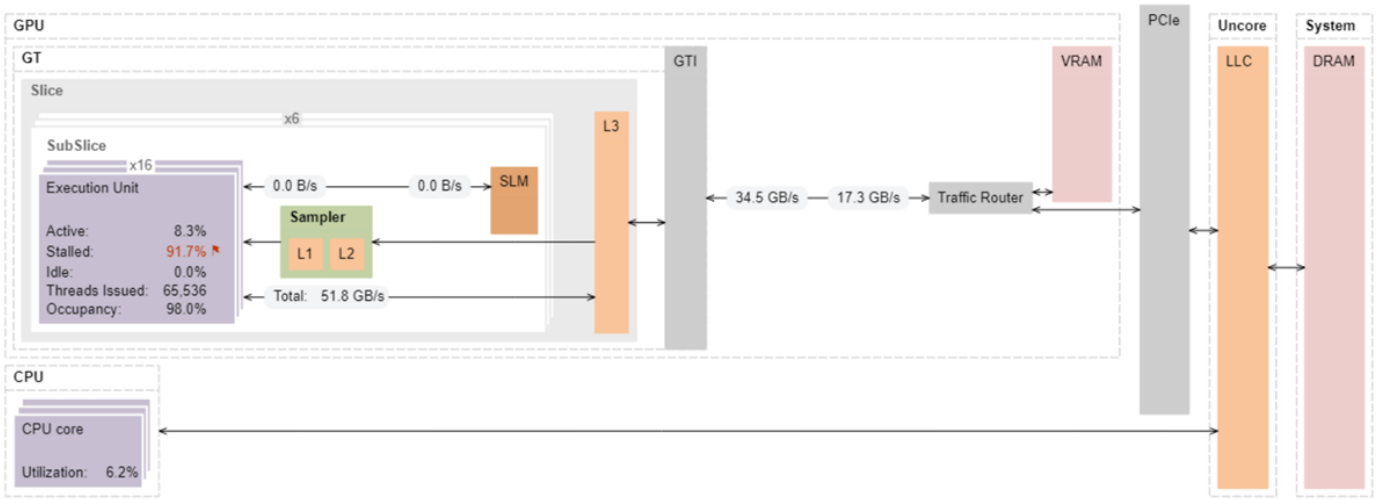
  scheduling issues



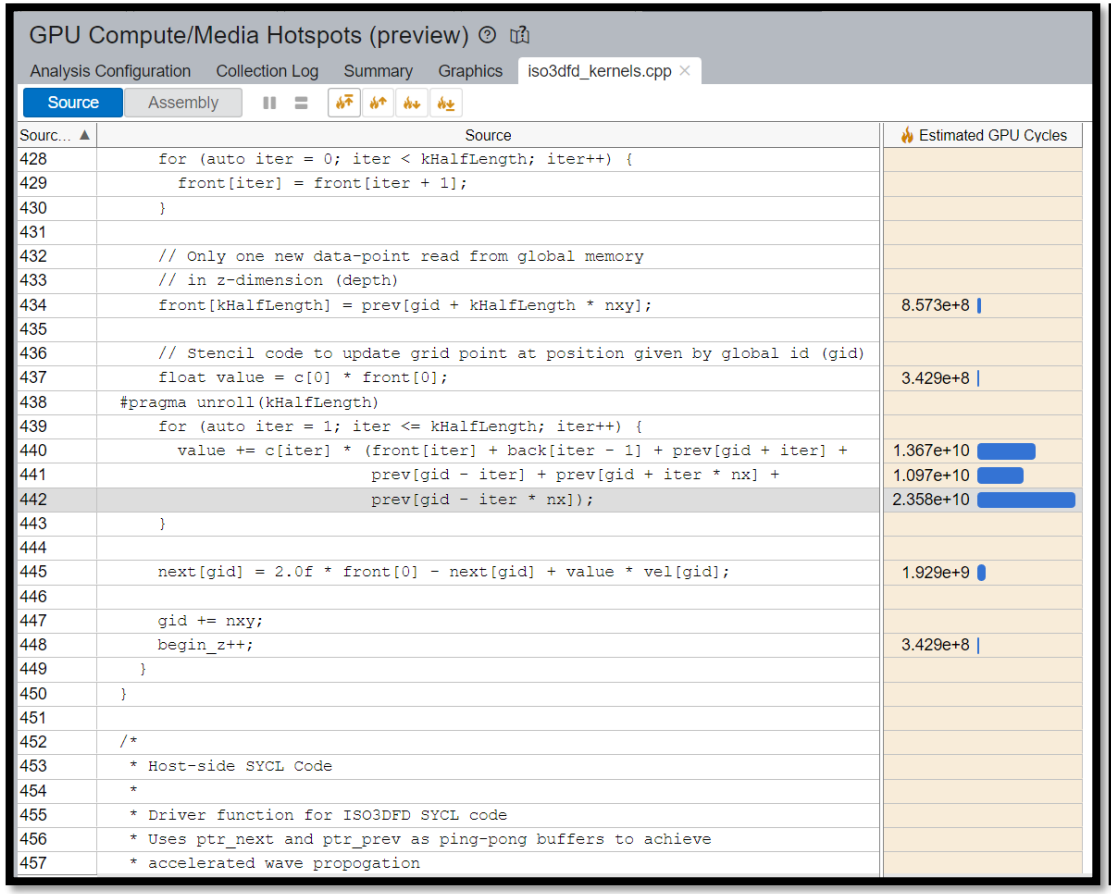
Memory Access problems

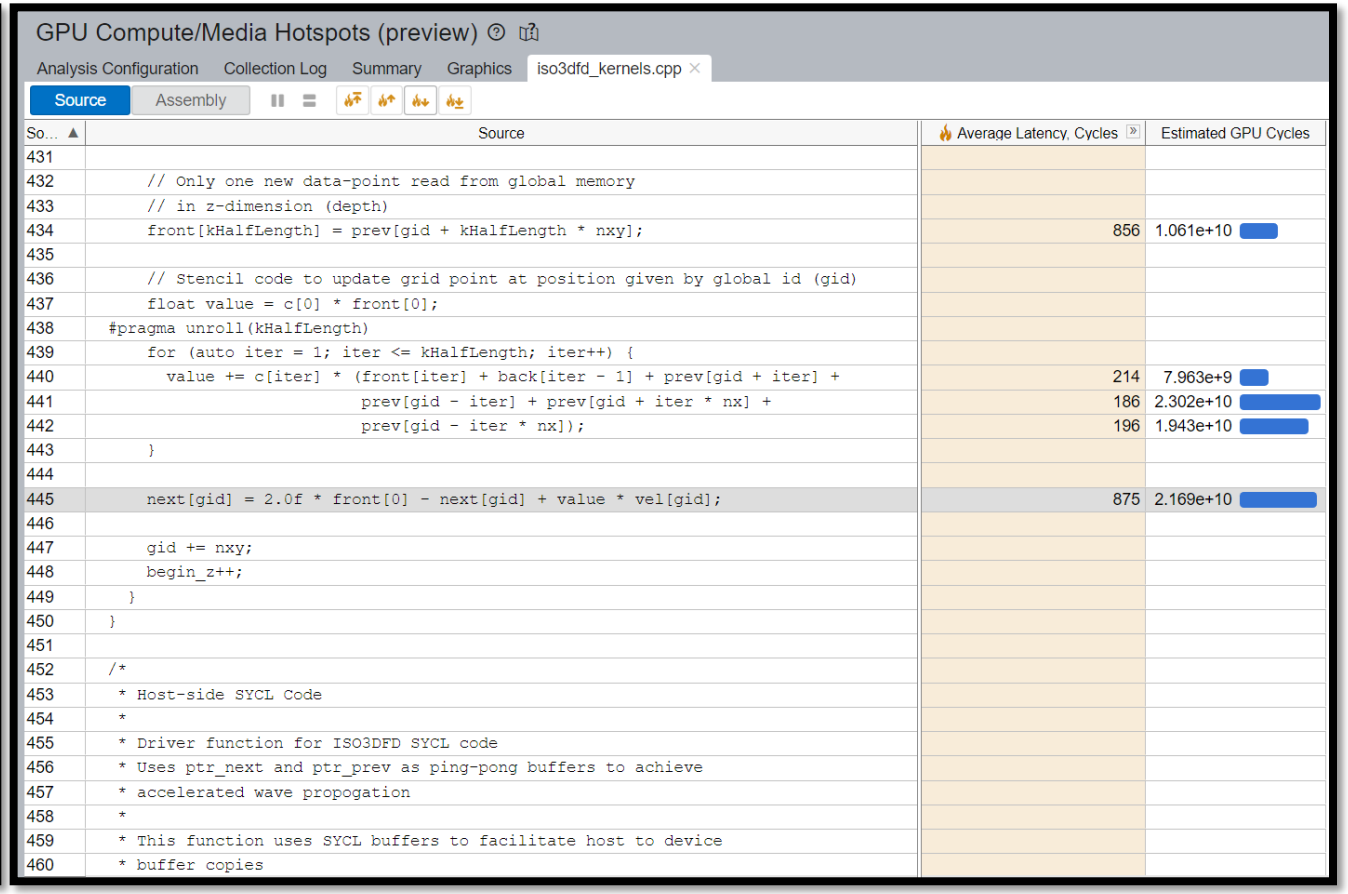
- Global memory access penalty
- Cache memory resource limit

- Which code is responsible for latency?
- Per Basic Block and latencies per individual instructions



Source level in-kernel profiling





Basic Block Latency

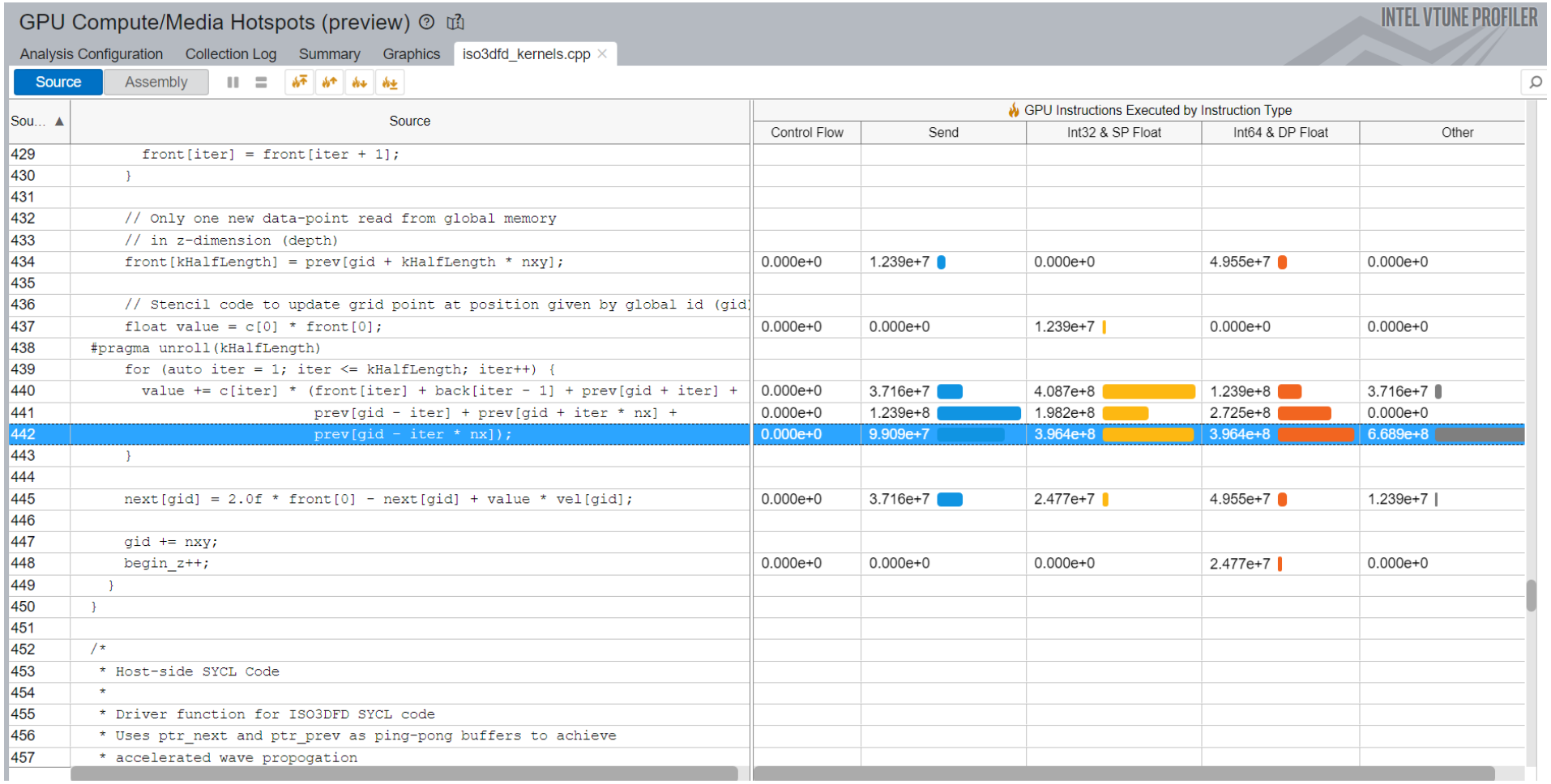
Memory Latency

Kernel code optimizations

Advanced code optimizations on kernel level

- Are FPUs and EM pipelines fully utilized?
- How are the systolic instructions used in AI application?
- Instructions counting profiles
- FPU and XMX pipeline Utilization

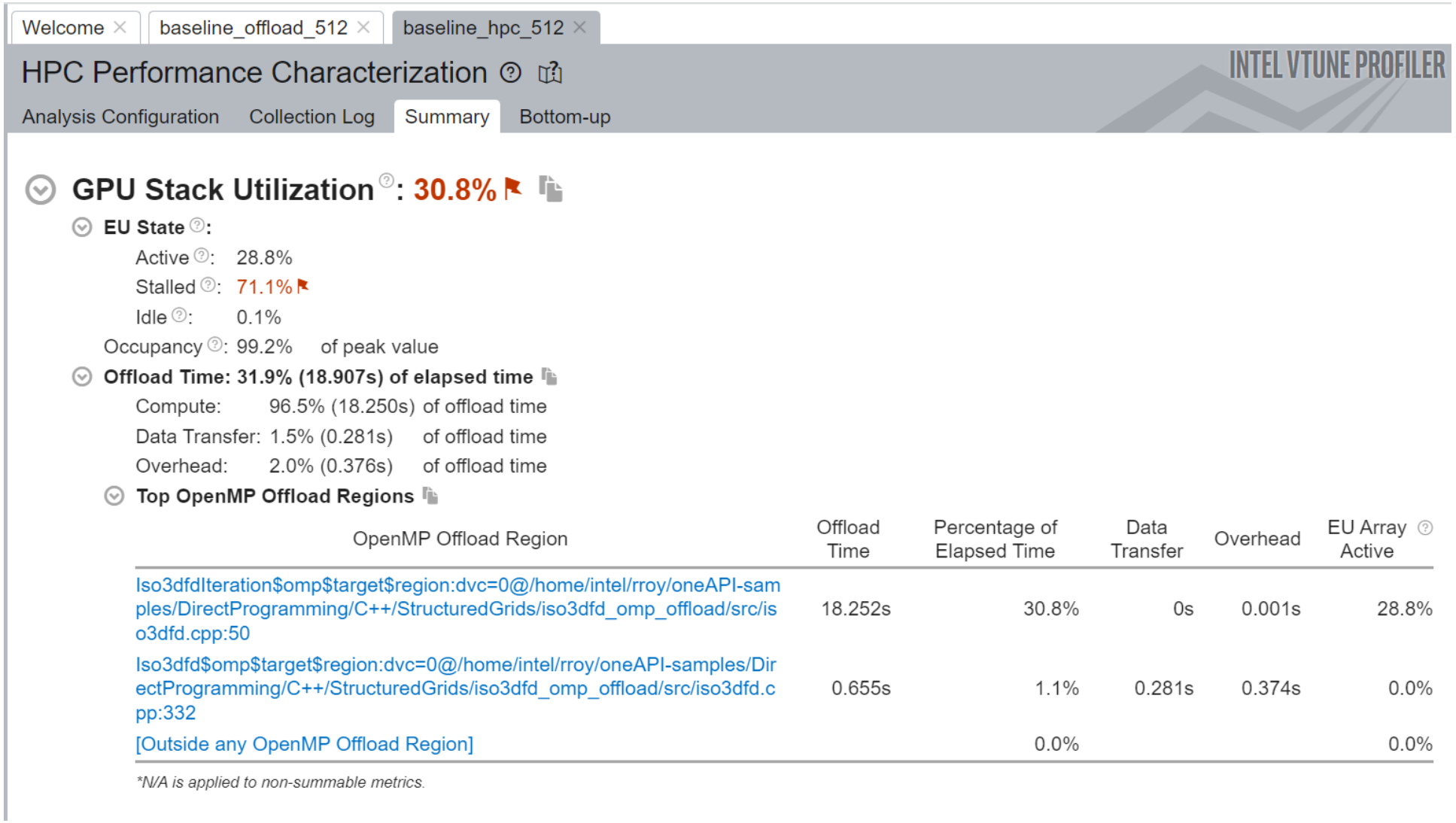
Source level in-kernel profiling



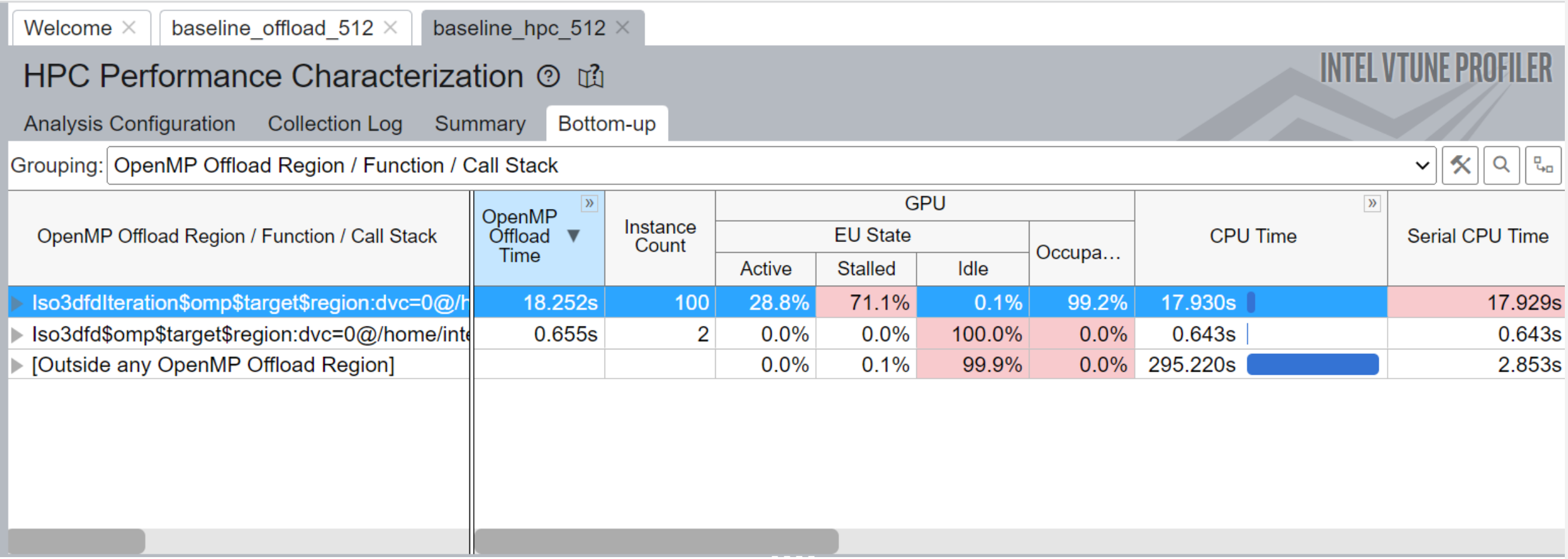
Case Study (OpenMP* Offload on GPU)

- Application Name: iso3dfd OpenMP Offload
- Profile the baseline version
- Detect the bottlenecks
- Make code changes based on the findings
- Compare baseline vs optimized

HPC Characterization



HPC Characterization Bottom-up View for Baseline Version



Welcome | Welcome | v3_hotspots_512 | baseline_hotspots_512 | baseline_hpc_512

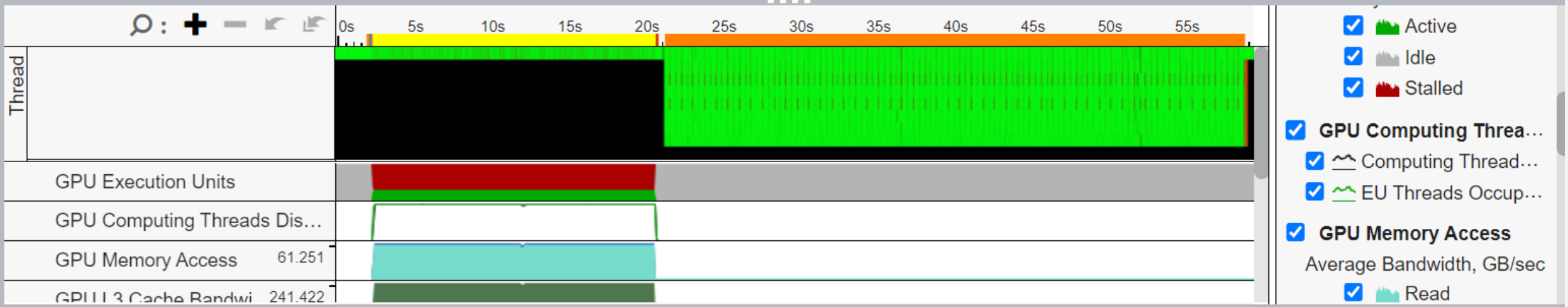
HPC Performance Characterization

INTEL VTUNE PROFILER

Analysis Configuration | Collection Log | Summary | Bottom-up

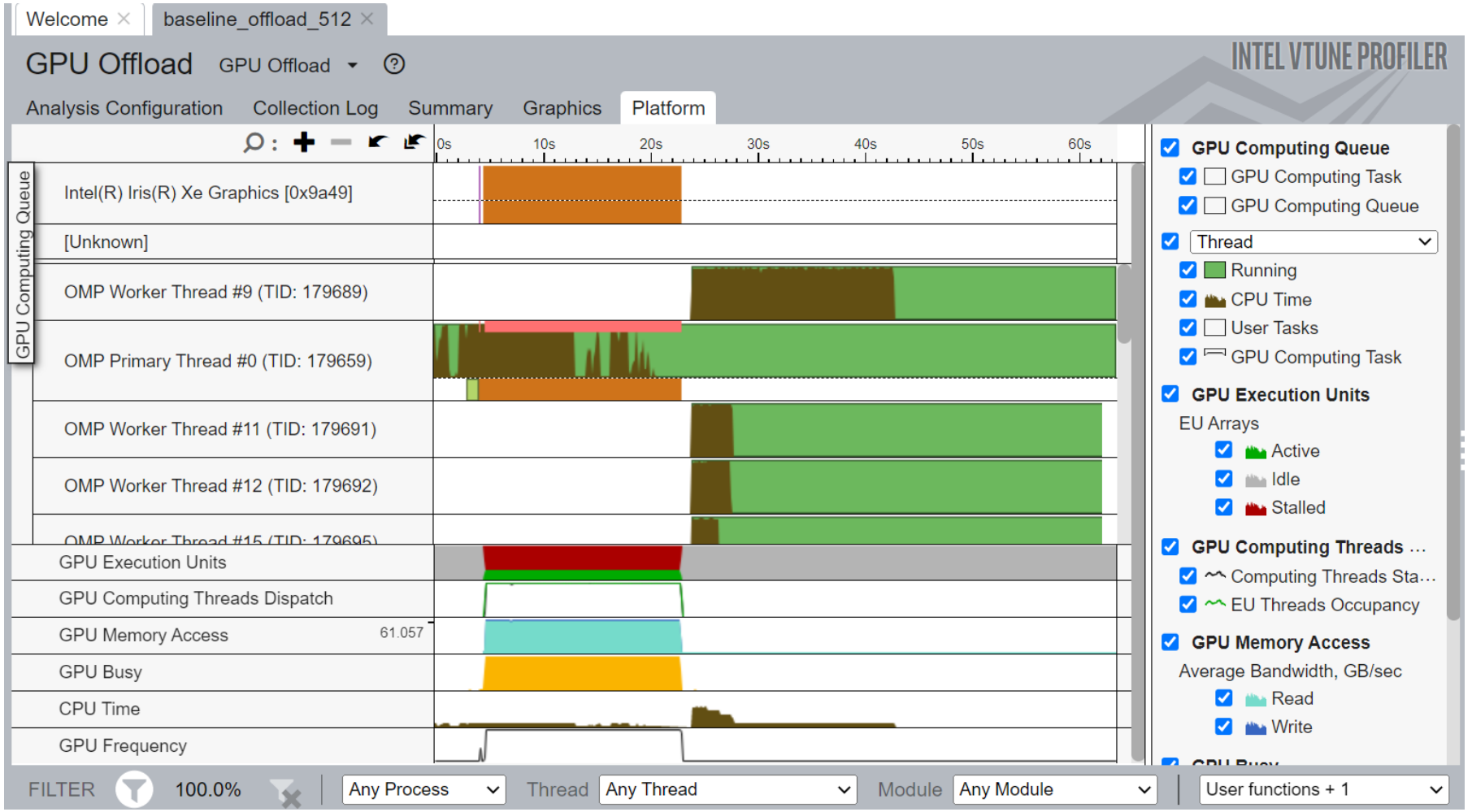
Grouping: Process / OpenMP Region / OpenMP Barrier-to-Barrier Segment / Function / Call Stack

Process / OpenMP Region / OpenMP Barrier-to-Barrier Segment / Function / Call Stack	Elapsed Time ▼	SP GFLOPS	Serial CPU Time	Memory Bound	OpenMP P…		
					Imbalance	Lock Contention	Creatic
iso3dfd	59.330s	13.415	21.425s	28.9%	0.816s	0s	
_Z22Iso3dfdVerifyIterationPfS_S_S_iiimmm	37.481s	21.222	0s	30.3%	0.816s	0s	
Iso3dfdVerifyIterationomploop_barrier_s	37.479s	21.222	0s	30.3%	0.816s	0s	
_INTERNALbae80b52::__kmp_for_stati		0.000	0s	0.0%			
__kmp_get_global_thread_id		0.000	0s	0.0%			
[Loop@0x167f7e in oa_buffer_check_u		0.000	0s	100.0%			
oa_buffer_check_unlocked		0.000	0s	0.0%			

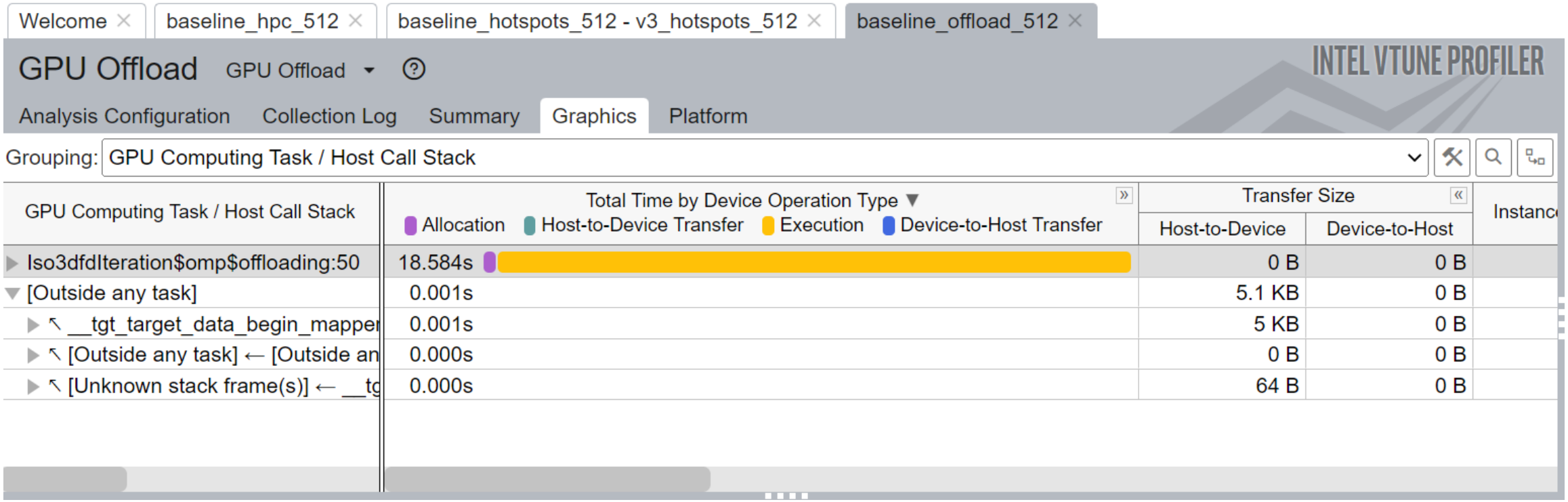


FILTER 100.0% | Any Process | Module: Any Module | User functions + 1 | Loops and function: | Show inline function:

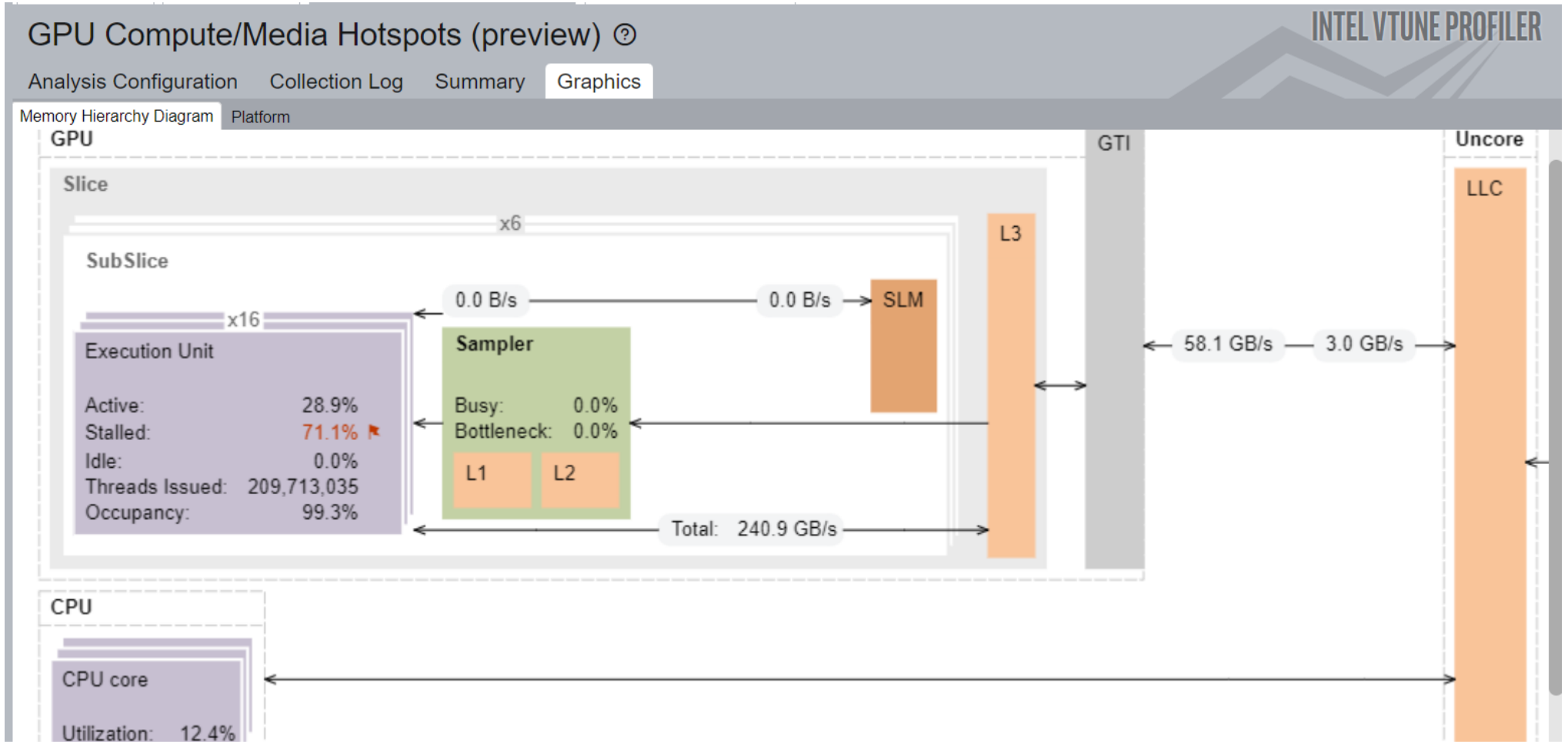
GPU-Offload Platform View for Baseline Version



GPU-Offload Graphics View for Baseline Version



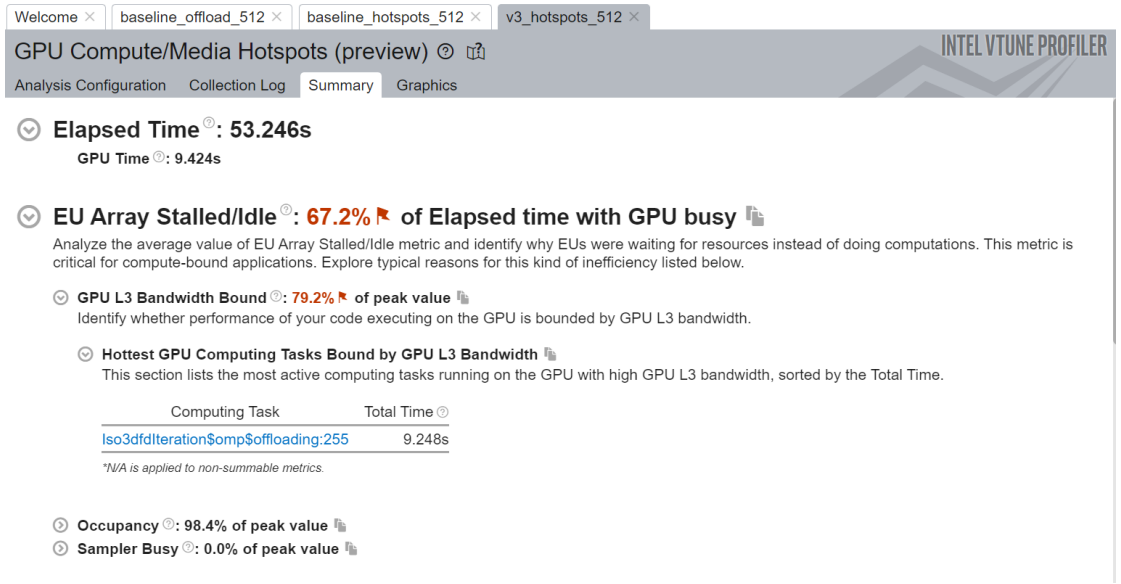
GPU-Hotspots Graphics View for Baseline Version

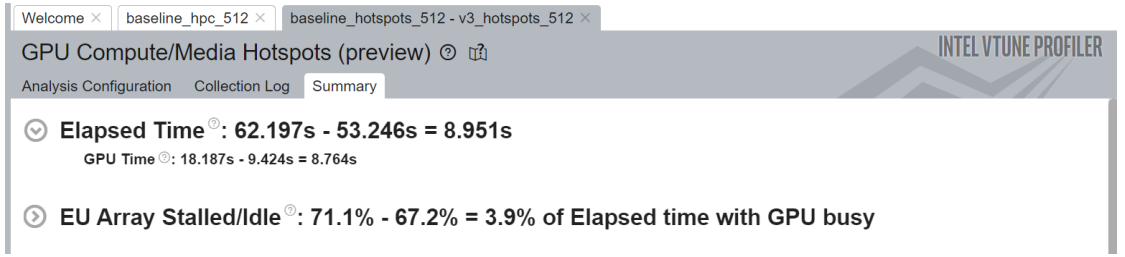


Code Optimization

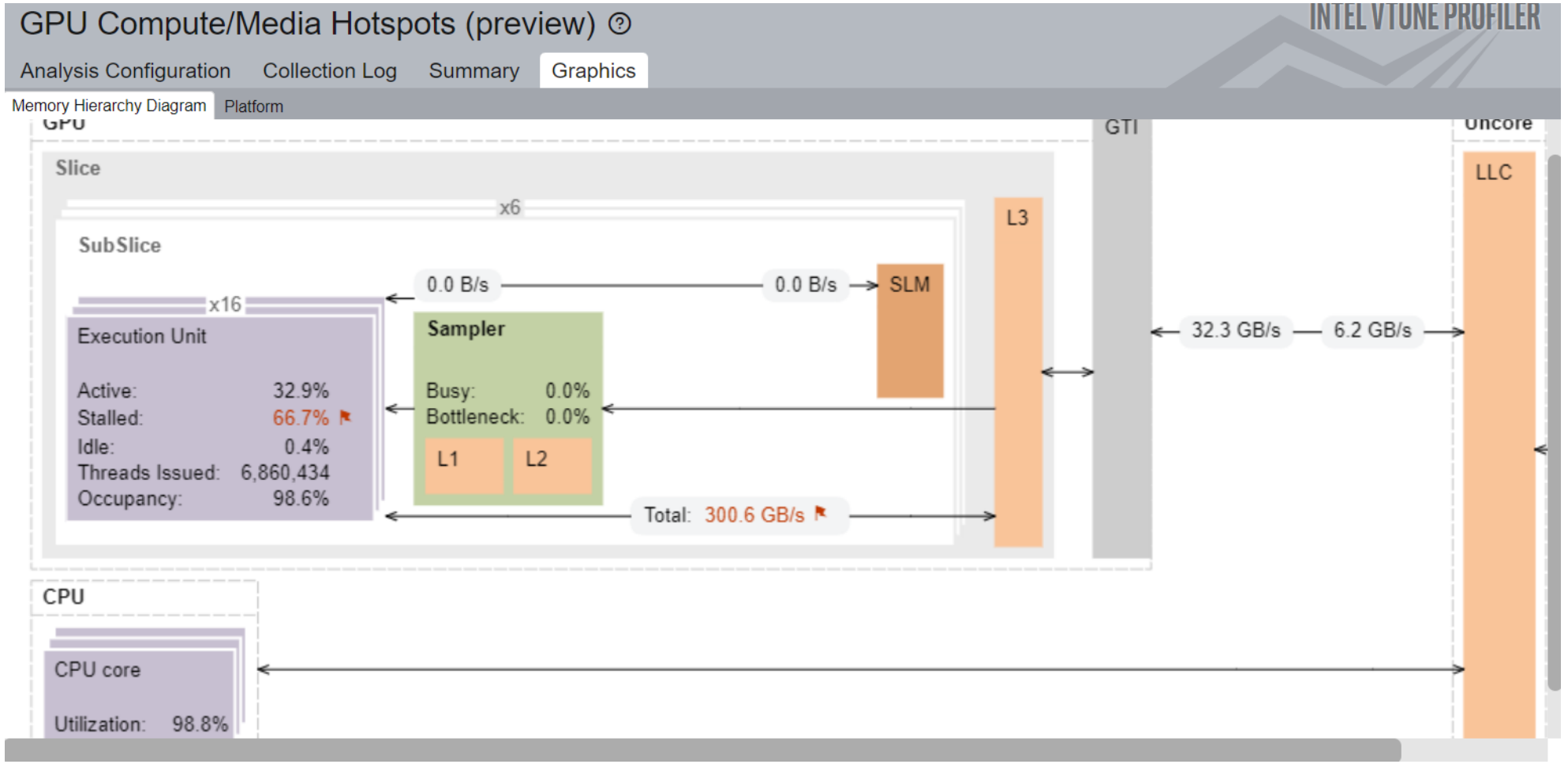
- Added ‘omp target teams distribute’ construct
- Distribute the work among different teams
- Store variables in registers for reuse
- Add OpenMP CPU Threads

GPU-Hotspots Summary View for Optimized Version



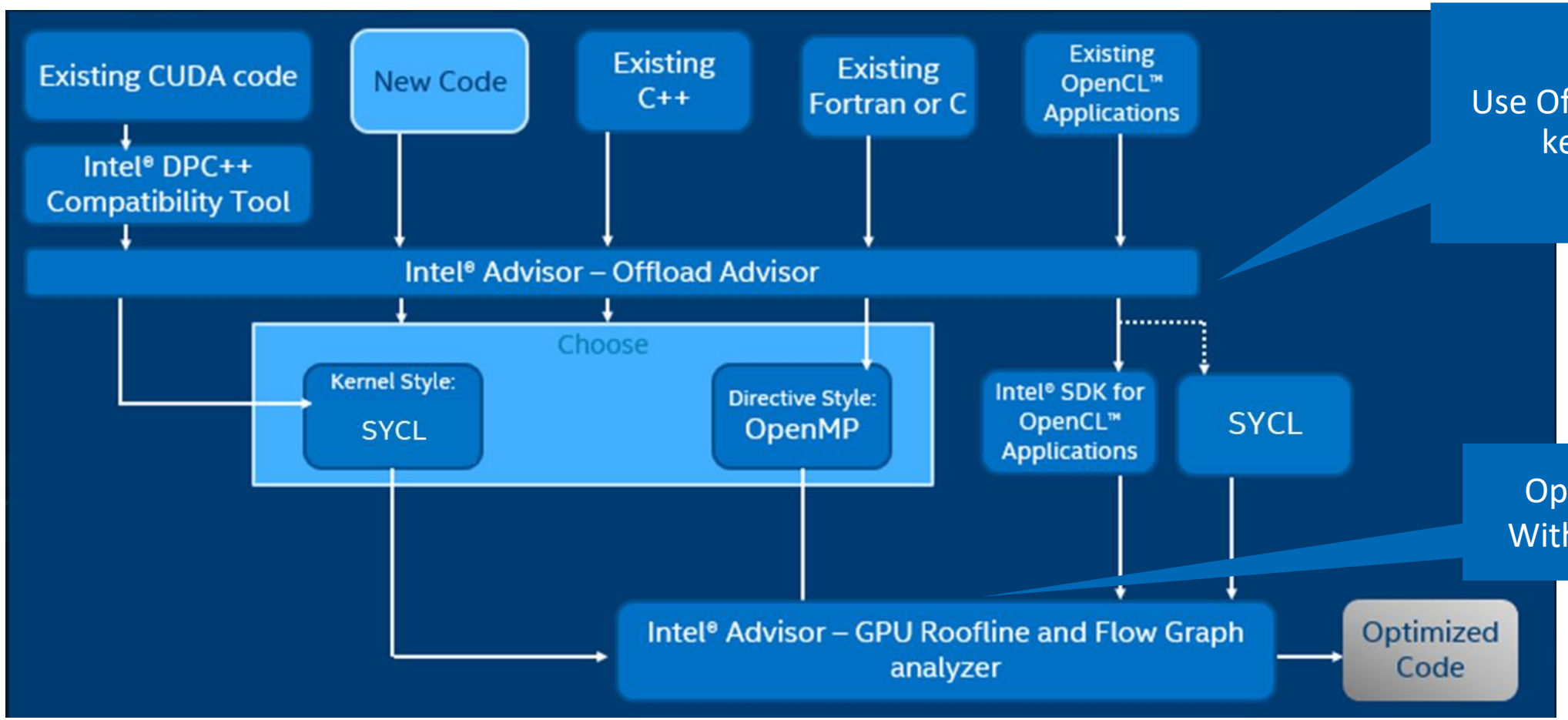


GPU-Hotspots Graphics View for Optimized Version



Recommended Workflow

Using Intel® Analyzers to increase performance



Use Offload Advisor to find kernels to offload

Optimize your kernels With Advisor and VTune


intel