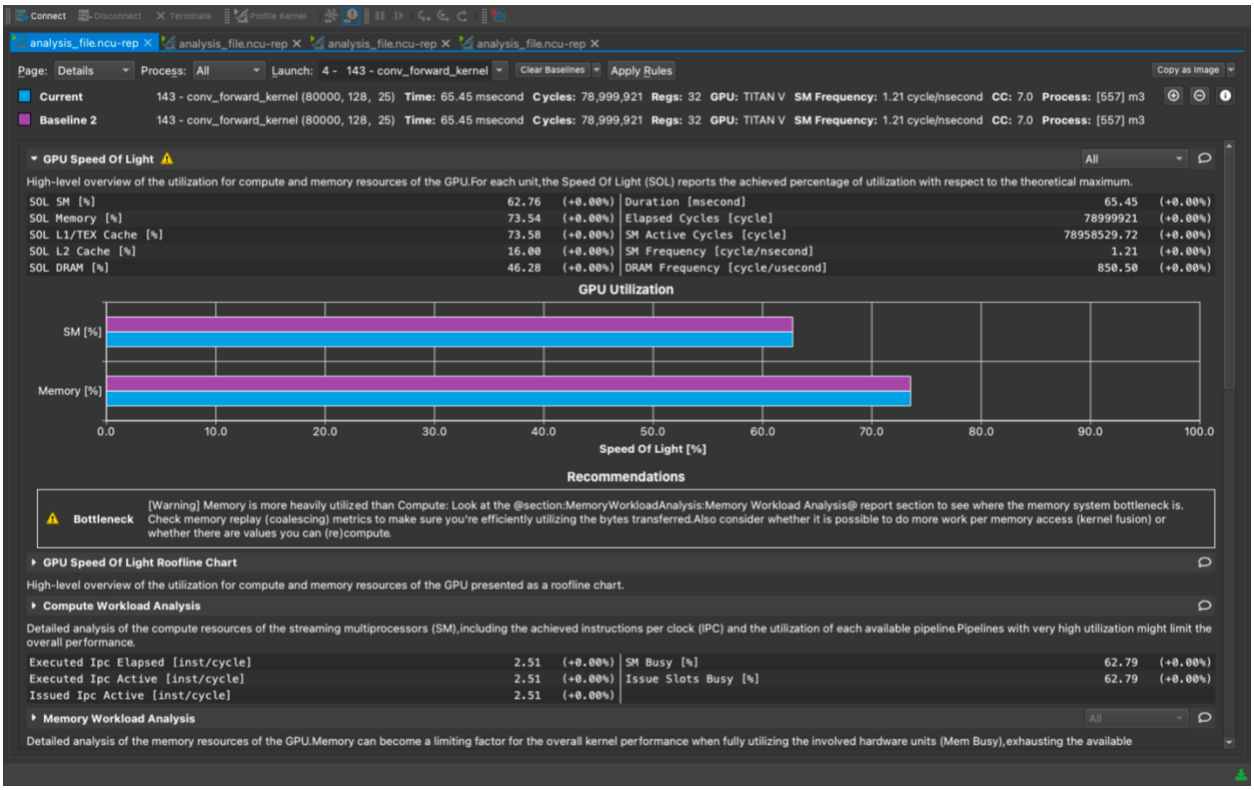
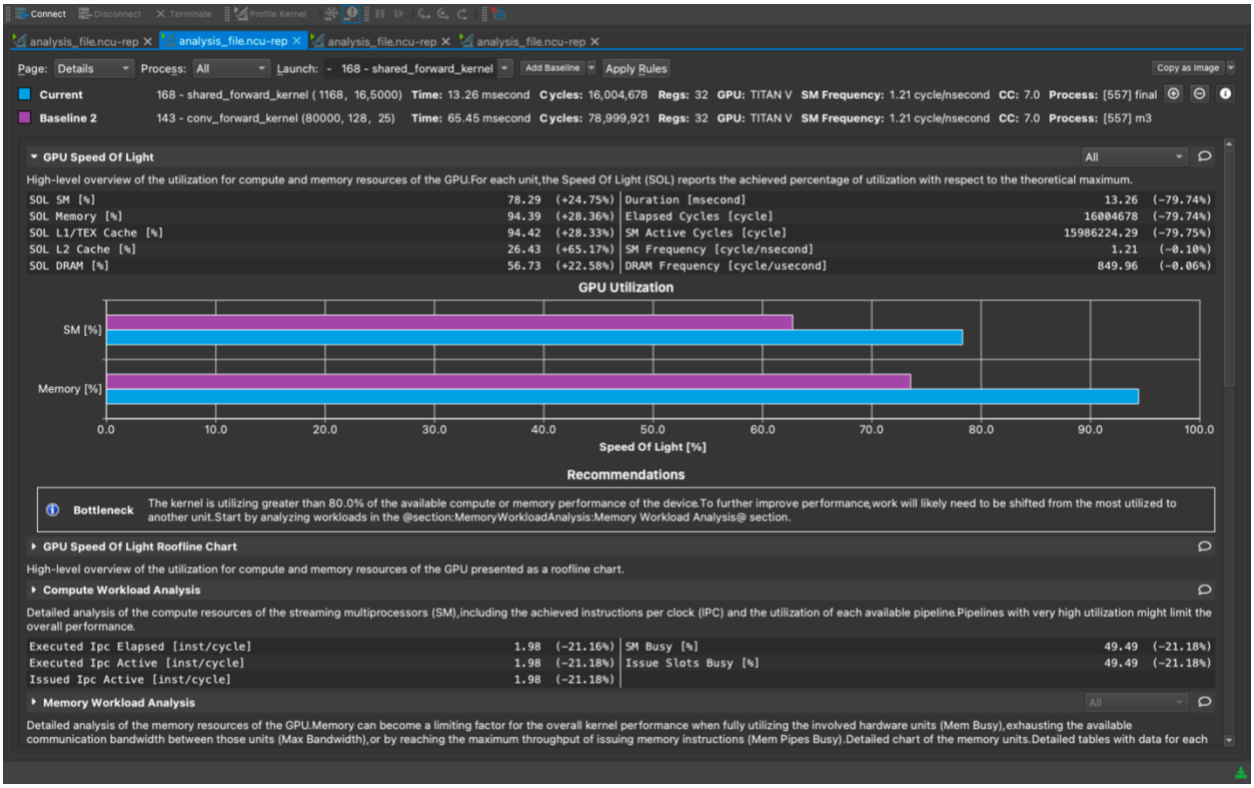
ECE 408 Final Project: Final Milestone

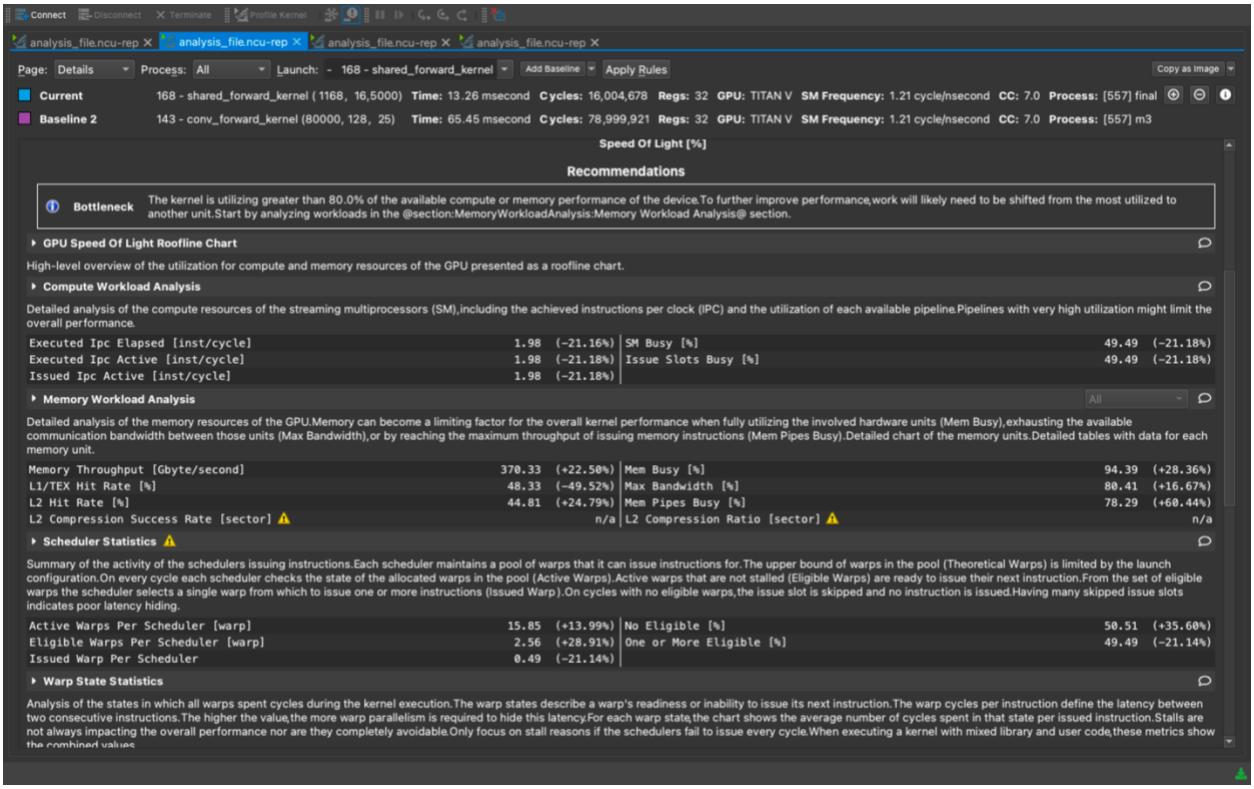
Vedant Vipul Jhaveri
NetID: vedantj2

I used the Kernel code from PM2 with a tile width of 8 as the baseline to check against all the mentioned optimizations. All tests gave an Accuracy of 0.8714 meaning there was no compromise on the result by the optimizations made by me.



First, I implemented the required shared memory matrix multiplication with GPU loop unrolling. Despite being a challenge to understand how to unroll for GPU works I was able to figure it out. I have used a Tile width of 16 for this kernel and the reasoning is I found it to work the best by using sweeping parameters mentioned later for Extra credit. As you can see from the figure below, we use more memory, by about 28.36%, and SMs but the benefit far outweighs the higher memory usage since it runs about 79.74% faster than the baseline and uses about 79.74% less overall cycles and active SM cycles. Due to larger processing powers required we had to launch the kernel twice hence there was less processing power needed per kernel, but it overall still beats the baseline. The main bottleneck is the number of threads that can be processed for 10k images since each kernel launch makes us allocate and free memory for x_unrolled and the kernel launch itself takes a standard minimum time. There is a much better L2 cache hit rate which is usually more import. The shared memory matrix multiply is faster than convolution mainly because of lower number of for loops resulting in a lower time complexity of processing.




Connect
Disconnect
Terminate
Profile Kernel
analysis_file.ncu-rep
analysis_file.ncu-rep
analysis_file.ncu-rep
analysis_file.ncu-rep
Page: Details
Process: All
Launch: - 168 - shared_forward_kernel
Add Baseline
Apply Rules
Copy as image
Current 168 - shared_forward_kernel (1168, 16,5000) Time: 13.26 msecond Cycles: 16,004,678 Regs: 32 GPU: TITAN V SM Frequency: 1.21 cycle/nsecond CC: 7.0 Process: [557] final
Baseline 2 143 - conv_forward_kernel (80000, 128, 25) Time: 65.45 msecond Cycles: 78,999,921 Regs: 32 GPU: TITAN V SM Frequency: 1.21 cycle/nsecond CC: 7.0 Process: [557] m3
Speed Of Light [%]
Recommendations
Bottleneck
The kernel is utilizing greater than 80.0% of the available compute or memory performance of the device.To further improve performance,work will likely need to be shifted from the most utilized to another unit.Start by analyzing workloads in the @section:MemoryWorkloadAnalysis:Memory Workload Analysis@ section.
GPU Speed Of Light Roofline Chart
High-level overview of the utilization for compute and memory resources of the GPU presented as a roofline chart.
Compute Workload Analysis
Detailed analysis of the compute resources of the streaming multiprocessors (SM),including the achieved instructions per clock (IPC) and the utilization of each available pipeline.Pipelines with very high utilization might limit the overall performance.
Executed Ipc Elapsed [inst/cycle] 1.98 (-21.16%)
Executed Ipc Active [inst/cycle] 1.98 (-21.18%)
Issued Ipc Active [inst/cycle] 1.98 (-21.18%)
SM Busy [%] 49.49 (-21.18%)
Issue Slots Busy [%] 49.49 (-21.18%)
Memory Workload Analysis
All
Detailed analysis of the memory resources of the GPU.Memory can become a limiting factor for the overall kernel performance when fully utilizing the involved hardware units (Mem Busy),exhausting the available communication bandwidth between those units (Max Bandwidth),or by reaching the maximum throughput of issuing memory instructions (Mem Pipes Busy).Detailed chart of the memory units.Detailed tables with data for each memory unit.
Memory Throughput [Gbyte/second] 370.33 (+22.50%)
L1/TEX Hit Rate [%] 48.33 (-49.52%)
L2 Hit Rate [%] 44.81 (+24.79%)
L2 Compression Success Rate [sector] n/a
Mem Busy [%] 94.39 (+28.36%)
Max Bandwidth [%] 80.41 (+16.67%)
Mem Pipes Busy [%] 78.29 (+60.44%)
L2 Compression Ratio [sector] n/a
Scheduler Statistics
Summary of the activity of the schedulers issuing instructions.Each scheduler maintains a pool of warps that it can issue instructions for.The upper bound of warps in the pool (Theoretical Warps) is limited by the launch configuration.On every cycle each scheduler checks the state of the allocated warps in the pool (Active Warps).Active warps that are not stalled (Eligible Warps) are ready to issue their next instruction.From the set of eligible warps the scheduler selects a single warp from which to issue one or more instructions (Issued Warp).On cycles with no eligible warps,the issue slot is skipped and no instruction is issued.Having many skipped issue slots indicates poor latency hiding.
Active Warps Per Scheduler [warp] 15.85 (+13.99%)
Eligible Warps Per Scheduler [warp] 2.56 (+28.91%)
Issued Warp Per Scheduler 0.49 (-21.14%)
No Eligible [%] 50.51 (+35.60%)
One or More Eligible [%] 49.49 (-21.14%)
Warp State Statistics
Analysis of the states in which all warps spent cycles during the kernel execution.The warp states describe a warp's readiness or inability to issue its next instruction.The warp cycles per instruction define the latency between two consecutive instructions.The higher the value,the more warp parallelism is required to hide this latency.For each warp state,the chart shows the average number of cycles spent in that state per issued instruction.Stalls are not always impacting the overall performance nor are they completely avoidable.Only focus on stall reasons if the schedulers fail to issue every cycle.When executing a kernel with mixed library and user code,these metrics show the combined values

Extra Credit:

1. On top of the final milestone optimization, I decided to put k in constant memory to reduce the memory usage which it greatly did. While the final milestone optimization used 28% more memory than the baseline, this uses 56.4% LESS memory than the baseline which is shocking at first but with huge batch sizes and tiling for better memory performance storing k in constant memory proved to be a great improvement memory wise. Both kernels combined took ever so slightly more time than the baseline but the better memory utilization makes this optimization a win for us.

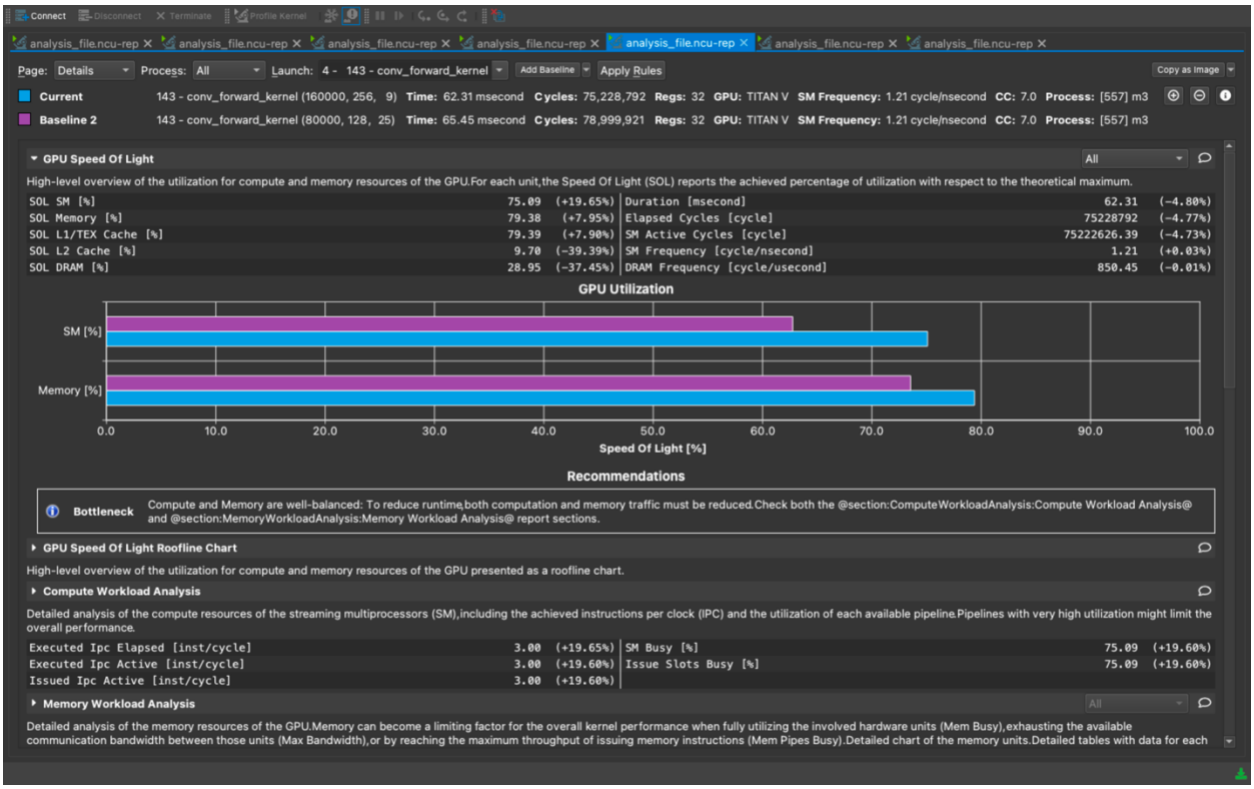
2. Restrict and loop unroll ever so slightly improved the time running time over x_unroll + matrix multiply and used slightly more memory. If runtime and calculation time is of utmost importance, then this is a decent optimization otherwise x_unroll + matrix multiply is more than enough to give the needed boost.

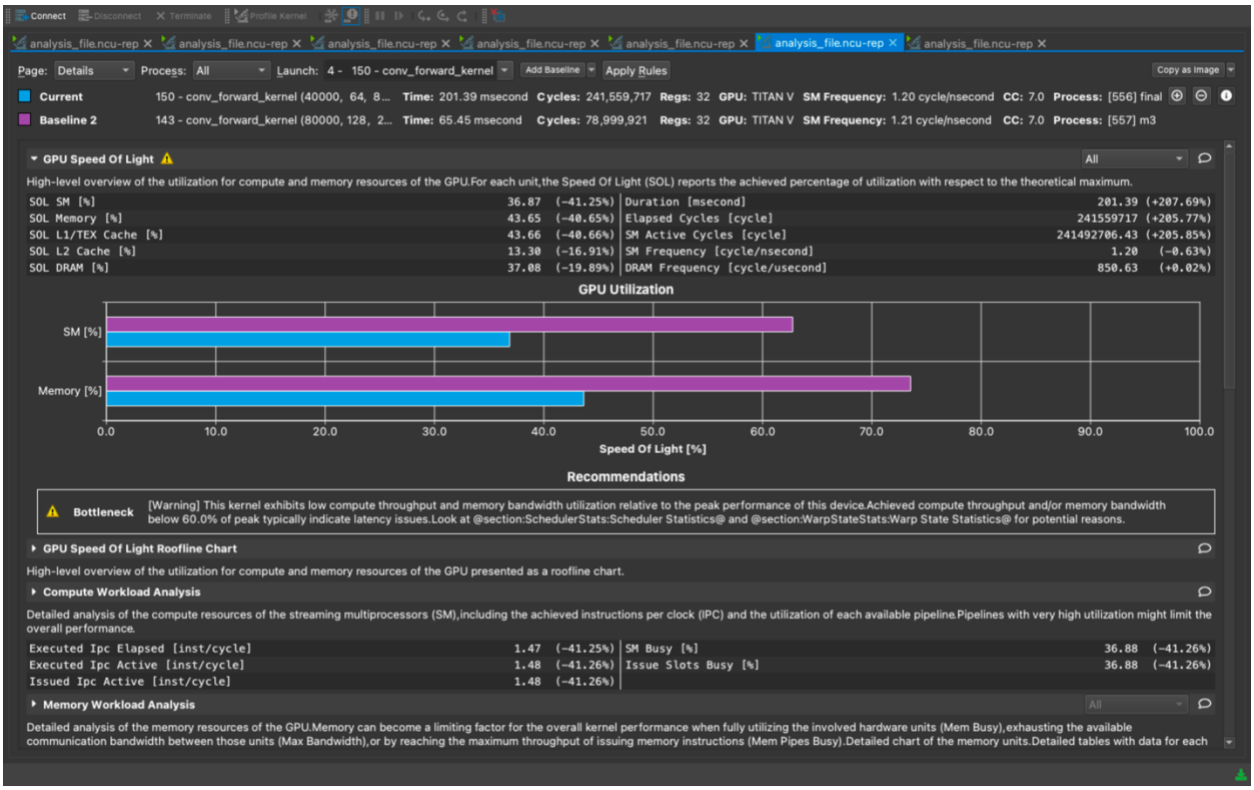
3. By sweeping parameters I changed the tile width which changed the block sizes and how the computation for the convolution proceeds. The baseline shown above on Page 1 implements the basic convolution with a Tile Width of 8. Now I will post the results for tile widths of 16, 4, and 32 respectively to show the difference in performance and why on page 2 and my final milestone implementation onward I used tile width of 16.

   For a very small increase in memory usage compared to the baseline, 7.9%, we get a better utilization of the DRAM as it reduces by 37.35% and an even better utilization of the L2 cache by about 39.39% less usage compared to the baseline with a tile width of 8. To put a cherry on top, we also get a faster runtime by about 4.8%.

   The time taken increases by 207% for a tile width of 4 which is enough to reject this tile width.

   For tile width of 32 we use about 22% less memory compared to baseline but the runtime is slower by 33% which is not a good enough tradeoff hence the best parameter by sweeping was tile width of 16.


Connect
Disconnect
Terminate
Profile Kernel
analysis_file.ncu-rep
Page: Details
Process: All
Launch: 4 - 143 - conv_forward_kernel
Add Baseline
Apply Rules
Copy as Image
Current 143 - conv_forward_kernel (160000, 256, 9) Time: 62.31 msecond Cycles: 75,228,792 Regs: 32 GPU: TITAN V SM Frequency: 1.21 cycle/nsecond CC: 7.0 Process: [557] m3
Baseline 2 143 - conv_forward_kernel (80000, 128, 25) Time: 65.45 msecond Cycles: 78,999,921 Regs: 32 GPU: TITAN V SM Frequency: 1.21 cycle/nsecond CC: 7.0 Process: [557] m3
GPU Speed Of Light
High-level overview of the utilization for compute and memory resources of the GPU.For each unit,the Speed Of Light (SOL) reports the achieved percentage of utilization with respect to the theoretical maximum.
SOL SM [%] 75.09 (+19.65%)
SOL Memory [%] 79.38 (+7.95%)
SOL L1/TEX Cache [%] 79.39 (+7.90%)
SOL L2 Cache [%] 9.70 (-39.39%)
SOL DRAM [%] 28.95 (-37.45%)
Duration [msecond] 62.31 (-4.80%)
Elapsed Cycles [cycle] 75228792 (-4.77%)
SM Active Cycles [cycle] 75222626.39 (-4.73%)
SM Frequency [cycle/nsecond] 1.21 (+0.03%)
DRAM Frequency [cycle/usecond] 850.45 (-0.01%)
GPU Utilization
SM [%]
Memory [%]
0.0
10.0
20.0
30.0
40.0
50.0
60.0
70.0
80.0
90.0
100.0
Speed Of Light [%]
Recommendations
Bottleneck
Compute and Memory are well-balanced: To reduce runtime,both computation and memory traffic must be reduced.Check both the @section:ComputeWorkloadAnalysis:Compute Workload Analysis@ and @section:MemoryWorkloadAnalysis:Memory Workload Analysis@ report sections.
GPU Speed Of Light Roofline Chart
High-level overview of the utilization for compute and memory resources of the GPU presented as a roofline chart.
Compute Workload Analysis
Detailed analysis of the compute resources of the streaming multiprocessors (SM),including the achieved instructions per clock (IPC) and the utilization of each available pipeline.Pipelines with very high utilization might limit the overall performance.
Executed Ipc Elapsed [inst/cycle] 3.00 (+19.65%)
Executed Ipc Active [inst/cycle] 3.00 (+19.60%)
Issued Ipc Active [inst/cycle] 3.00 (+19.60%)
SM Busy [%] 75.09 (+19.60%)
Issue Slots Busy [%] 75.09 (+19.60%)
Memory Workload Analysis
Detailed analysis of the memory resources of the GPU.Memory can become a limiting factor for the overall kernel performance when fully utilizing the involved hardware units (Mem Busy),exhausting the available communication bandwidth between those units (Max Bandwidth),or by reaching the maximum throughput of issuing memory instructions (Mem Pipes Busy).Detailed chart of the memory units.Detailed tables with data for each


Connect
Disconnect
Terminate
Profile Kernel
analysis_file.ncu-rep
Page: Details
Process: All
Launch: 4 - 150 - conv_forward_kernel
Add Baseline
Apply Rules
Copy as Image
Current 150 - conv_forward_kernel (40000, 64, 8... Time: 201.39 msecond Cycles: 241,559,717 Regs: 32 GPU: TITAN V SM Frequency: 1.20 cycle/nsecond CC: 7.0 Process: [556] final
Baseline 2 143 - conv_forward_kernel (80000, 128, 2... Time: 65.45 msecond Cycles: 78,999,921 Regs: 32 GPU: TITAN V SM Frequency: 1.21 cycle/nsecond CC: 7.0 Process: [557] m3
GPU Speed Of Light
High-level overview of the utilization for compute and memory resources of the GPU.For each unit,the Speed Of Light (SOL) reports the achieved percentage of utilization with respect to the theoretical maximum.
SOL SM [%] 36.87 (-41.25%)
SOL Memory [%] 43.65 (-40.65%)
SOL L1/TEX Cache [%] 43.66 (-40.66%)
SOL L2 Cache [%] 13.30 (-16.91%)
SOL DRAM [%] 37.08 (-19.89%)
Duration [msecond] 201.39 (+207.69%)
Elapsed Cycles [cycle] 241559717 (+205.77%)
SM Active Cycles [cycle] 241492706.43 (+205.85%)
SM Frequency [cycle/nsecond] 1.20 (-0.63%)
DRAM Frequency [cycle/usecond] 850.63 (+0.02%)
GPU Utilization
SM [%]
Memory [%]
0.0
10.0
20.0
30.0
40.0
50.0
60.0
70.0
80.0
90.0
100.0
Speed Of Light [%]
Recommendations
Bottleneck
[Warning] This kernel exhibits low compute throughput and memory bandwidth utilization relative to the peak performance of this device.Achieved compute throughput and/or memory bandwidth below 60.0% of peak typically indicate latency issues.Look at @section:SchedulerStats:Scheduler Statistics@ and @section:WarpStateStats:Warp State Statistics@ for potential reasons.
GPU Speed Of Light Roofline Chart
High-level overview of the utilization for compute and memory resources of the GPU presented as a roofline chart.
Compute Workload Analysis
Detailed analysis of the compute resources of the streaming multiprocessors (SM),including the achieved instructions per clock (IPC) and the utilization of each available pipeline.Pipelines with very high utilization might limit the overall performance.
Executed Ipc Elapsed [inst/cycle] 1.47 (-41.25%)
Executed Ipc Active [inst/cycle] 1.48 (-41.26%)
Issued Ipc Active [inst/cycle] 1.48 (-41.26%)
SM Busy [%] 36.88 (-41.26%)
Issue Slots Busy [%] 36.88 (-41.26%)
Memory Workload Analysis
Detailed analysis of the memory resources of the GPU.Memory can become a limiting factor for the overall kernel performance when fully utilizing the involved hardware units (Mem Busy),exhausting the available communication bandwidth between those units (Max Bandwidth),or by reaching the maximum throughput of issuing memory instructions (Mem Pipes Busy).Detailed chart of the memory units.Detailed tables with data for each


Connect
Disconnect
Terminate
Profile Kernel
analysis_file.ncu-rep
Page: Details
Process: All
Launch: 4 - 150 - conv_forward_kernel
Add Baseline
Apply Rules
Copy as Image
Current 150 - conv_forward_kernel (320000, 512, ...) Time: 87.16 msecond Cycles: 104,557,969 Regs: 32 GPU: TITAN V SM Frequency: 1.20 cycle/nsecond CC: 7.0 Process: [556] final
Baseline 2 143 - conv_forward_kernel (80000, 128, 25) Time: 65.45 msecond Cycles: 78,999,921 Regs: 32 GPU: TITAN V SM Frequency: 1.21 cycle/nsecond CC: 7.0 Process: [557] m3
GPU Speed Of Light
All
High-level overview of the utilization for compute and memory resources of the GPU.For each unit,the Speed Of Light (SOL) reports the achieved percentage of utilization with respect to the theoretical maximum.
SOL SM [%] 72.63 (+15.73%)
SOL Memory [%] 57.34 (-22.02%)
SOL L1/TEX Cache [%] 57.29 (-22.13%)
SOL L2 Cache [%] 4.36 (-72.73%)
SOL DRAM [%] 13.20 (-71.47%)
Duration [msecond] 87.16 (+33.16%)
Elapsed Cycles [cycle] 104557969 (+32.35%)
SM Active Cycles [cycle] 104643512.29 (+32.53%)
SM Frequency [cycle/nsecond] 1.20 (-0.61%)
DRAM Frequency [cycle/usecond] 850.26 (-0.03%)
GPU Utilization
SM [%]
Memory [%]
0.0
10.0
20.0
30.0
40.0
50.0
60.0
70.0
80.0
90.0
100.0
Speed Of Light [%]
Recommendations
Bottleneck
[Warning] Compute is more heavily utilized than Memory: Look at the @section:ComputeWorkloadAnalysis:Compute Workload Analysis@ report section to see what the compute pipelines are spending their time doing.Also,consider whether any computation is redundant and could be reduced or moved to look-up tables.
GPU Speed Of Light Roofline Chart
High-level overview of the utilization for compute and memory resources of the GPU presented as a roofline chart.
Compute Workload Analysis
Detailed analysis of the compute resources of the streaming multiprocessors (SM),including the achieved instructions per clock (IPC) and the utilization of each available pipeline.Pipelines with very high utilization might limit the overall performance.
Executed Ipc Elapsed [inst/cycle] 2.91 (+15.73%)
Executed Ipc Active [inst/cycle] 2.90 (+15.58%)
Issued Ipc Active [inst/cycle] 2.90 (+15.58%)
SM Busy [%] 72.57 (+15.58%)
Issue Slots Busy [%] 72.57 (+15.58%)
Memory Workload Analysis
All
Detailed analysis of the memory resources of the GPU.Memory can become a limiting factor for the overall kernel performance when fully utilizing the involved hardware units (Mem Busy),exhausting the available communication bandwidth between those units (Max Bandwidth),or by reaching the maximum throughput of issuing memory instructions (Mem Pipes Busy).Detailed chart of the memory units.Detailed tables with data for each