UPPSALA
UNIVERSITET

Accelerator-Based Programming - 1TD055

ASSIGNMENT 1: USING THE CPU AND THE GPU

Jyong-Jhih Lin

September 18, 2022

0 Hardware information

snowy CPU:

```
Architecture:          x86_64
CPU op-mode(s):        32-bit, 64-bit
Byte Order:            Little Endian
CPU(s):                16
On-line CPU(s) list:   0-15
Thread(s) per core:    1
Core(s) per socket:    8
Socket(s):             2
NUMA node(s):          2
Vendor ID:             GenuineIntel
CPU family:            6
Model:                 45
Model name:            Intel(R) Xeon(R) CPU E5-2660 0 @ 2.20GHz
Stepping:              7
CPU MHz:               1200.000
CPU max MHz:           2200.0000
CPU min MHz:           1200.0000
BogoMIPS:              4388.80
Virtualization:        VT-x
L1d cache:             32K
L1i cache:             32K
L2 cache:              256K
L3 cache:              20480K
NUMA node0 CPU(s):     0-7
NUMA node1 CPU(s):     8-15
Flags:                 fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 clflush dts acpi mmx fxsr sse sse2 ss ht tm pbe syscall nx pdpe1gb rdtscp lm constant_tsc arch_perfmon pebs bts rep_good nopl xtopology nonstop_tsc aperfmperf eagerfpu pni pclmulqdq dtes64 monitor ds_cpl vmx smx est tm2 ssse3 cx16 xtpr pdcm pcid dca sse4_1 sse4_2 x2apic popcnt tsc_deadline_timer aes xsave avx lahf_lm epb ssbd ibrs ibpb stibp tpr_shadow vnmi flexpriority ept vpid xsaveopt dtherm ida arat pln pts md_clear spec_ctrl intel_stibp flush_l1d
```

snowy memory:

```
Handle 0x1100, DMI type 17, 40 bytes
Memory Device
Array Handle: 0x1000
Error Information Handle: Not Provided
Total Width: 72 bits
Data Width: 64 bits
Size: 32 GB
Form Factor: DIMM
Set: None
Locator: PROC 1 DIMM 1
Bank Locator: Not Specified
Type: DDR3
Type Detail: Synchronous LRDIMM
Speed: 1333 MT/s
Manufacturer: HP
Serial Number: Not Specified
Asset Tag: Not Specified
Part Number: 647654-081
Rank: 4
Configured Memory Speed: 1333 MT/s
Minimum Voltage: 1.35 V
Maximum Voltage: 1.5 V
Configured Voltage: 1.35 V

#####
DDR3-1333 4-channel memory total bandwidth = 1333e6(T/s) * 64(bits) * 4(channels) / 8e9(GBytes/s)
= 42.656(GBytes/s)
```

nvidia T4:

```
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 515.65.01    Driver Version: 515.65.01    CUDA Version: 11.7     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|                               |                      |               MIG M. |
|===============================+======================+======================|
|   0  Tesla T4            On   | 00000000:08:00.0 Off |                    0 |
| N/A   30C    P8    14W /  70W |      2MiB / 15360MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                                  |
|  GPU   GI   CI        PID   Type   Process name                  GPU Memory |
|        ID   ID                                                   Usage      |
|=============================================================================|
|  No running processes found                                                 |
+-----------------------------------------------------------------------------+
```

DELL Precision 7760 CPU:

```
Architecture:                    x86_64
CPU op-mode(s):                  32-bit, 64-bit
Byte Order:                      Little Endian
Address sizes:                   39 bits physical, 48 bits virtual
CPU(s):                          12
On-line CPU(s) list:             0-11
Thread(s) per core:              2
Core(s) per socket:              6
Socket(s):                       1
NUMA node(s):                    1
Vendor ID:                       GenuineIntel
CPU family:                      6
Model:                           141
Model name:                      Intel(R) Xeon(R) W-11855M CPU @ 3.20GHz
Stepping:                        1
CPU MHz:                         3200.000
CPU max MHz:                     4900.0000
CPU min MHz:                     800.0000
BogoMIPS:                        6374.40
Virtualization:                  VT-x
L1d cache:                       288 KiB
L1i cache:                       192 KiB
L2 cache:                        7.5 MiB
L3 cache:                        18 MiB
NUMA node0 CPU(s):               0-11
Vulnerability Itlb multihit:     Not affected
Vulnerability L1tf:              Not affected
Vulnerability Mds:               Not affected
Vulnerability Meltdown:          Not affected
Vulnerability Mmio stale data:   Not affected
Vulnerability Spec store bypass: Mitigation; Speculative Store Bypass disabled via prctl and seccomp
Vulnerability Spectre v1:        Mitigation; usercopy/swapgs barriers and __user pointer sanitization
Vulnerability Spectre v2:        Mitigation; Enhanced IBRS, IBPB conditional, RSB filling
Vulnerability Srbds:             Not affected
Vulnerability Tsx async abort:   Not affected
Flags:                           fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36
     clflush dts acpi mmx fxsr sse sse2 ss ht tm pbe syscall nx pdpe1gb rdtscp lm constant_tsc art
     arch_perfmon pebs bts rep_good nopl xtopology nonstop_tsc cpuid aperfmperf tsc_known_freq pni pclmulqdq
      dtes64 monitor ds_cpl vmx smx est tm2 ssse3 sdbg fma cx16 xtpr pdcm pcid sse4_1 sse4_2 x2apic movbe
     popcnt tsc_deadline_timer aes xsave avx f16c rdrand lahf_lm abm 3dnowprefetch cpuid_fault epb cat_l2
     invpcid_single cdp_l2 ssbd ibrs ibpb stibp ibrs_enhanced tpr_shadow vnmi flexpriority ept vpid ept_ad
     fsgsbase tsc_adjust bmi1 avx2 smep bmi2 erms invpcid rdt_a avx512f avx512dq rdseed adx smap avx512ifma
     clflushopt clwb intel_pt avx512cd sha_ni avx512bw avx512vl xsaveopt xsavec xgetbv1 xsaves
     split_lock_detect dtherm ida arat pln pts hwp hwp_notify hwp_act_window hwp_epp hwp_pkg_req avx512vbmi
     umip pku ospke avx512_vbmi2 gfni vaes vpclmulqdq avx512_vnni avx512_bitalg tme avx512_vpopcntdq rdpid
     movdiri movdir64b fsrm avx512_vp2intersect md_clear flush_l1d arch_capabilities
```

DELL Precision 7760 memory:

```
Handle 0x1100, DMI type 17, 92 bytes
Memory Device
    Array Handle: 0x1000
    Error Information Handle: Not Provided
    Total Width: 72 bits
    Data Width: 64 bits
    Size: 32 GB
    Form Factor: SODIMM
    Set: None
    Locator: DIMM C
    Bank Locator: BANK 0
    Type: DDR4
    Type Detail: Synchronous
    Speed: 2933 MT/s
    Manufacturer: 01980000802C
    Serial Number: 97B0B609
    Asset Tag: 04212100
    Part Number: 9965657-029.A00G
    Rank: 2
    Configured Memory Speed: 2933 MT/s
    Minimum Voltage: Unknown
    Maximum Voltage: Unknown
    Configured Voltage: 1.2 V
    Memory Technology: DRAM
    Memory Operating Mode Capability: Volatile memory
    Firmware Version: Not Specified
    Module Manufacturer ID: Bank 2, Hex 0x98
    Module Product ID: Unknown
    Memory Subsystem Controller Manufacturer ID: Unknown
    Memory Subsystem Controller Product ID: Unknown
    Non-Volatile Size: None
    Volatile Size: 32 GB
    Cache Size: None
    Logical Size: None

#####
DDR4-2933 2-channel memory total bandwidth = 2933e6(T/s) * 64(bits) * 2(channels) / 8e9(GBytes/s)
= 46.928(GBytes/s)
```

nvidia A3000:

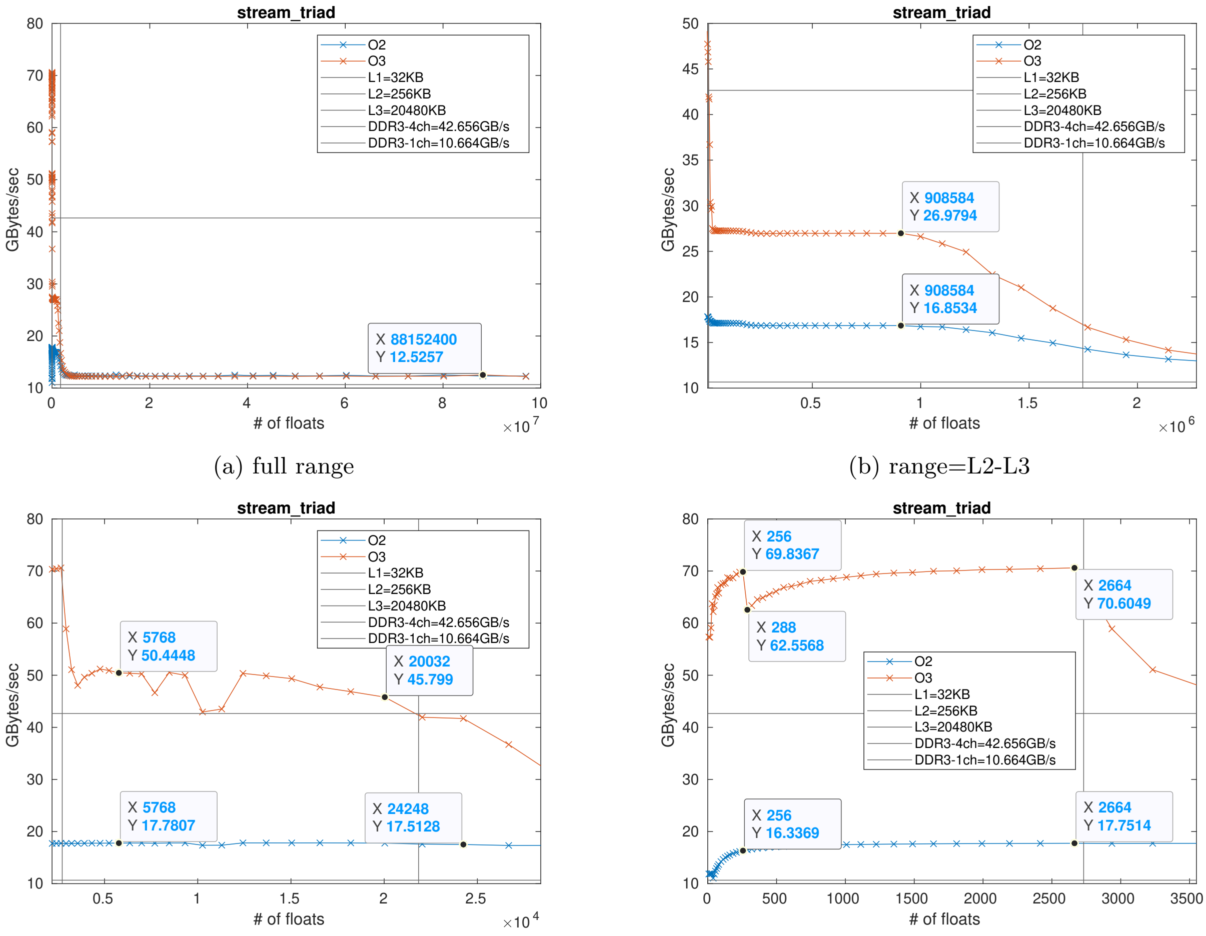
```
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 510.47.03    Driver Version: 510.47.03    CUDA Version: 11.6     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|                               |                      |               MIG M. |
|===============================+======================+======================|
|   0  NVIDIA RTX A300...  Off  | 00000000:01:00.0  On |                  N/A |
| N/A   58C    P0    36W /  N/A |   1606MiB /  6144MiB |    100%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                                  |
|  GPU   GI   CI        PID   Type   Process name                  GPU Memory |
|        ID   ID                                                   Usage      |
|=============================================================================|
|    0   N/A  N/A      3964      G   /usr/lib/xorg/Xorg                109MiB |
|    0   N/A  N/A     12824      G   /usr/lib/xorg/Xorg                603MiB |
|    0   N/A  N/A     12940      G   /usr/bin/gnome-shell              271MiB |
|    0   N/A  N/A     30513      G   ...308337019390783085,131072      481MiB |
|    0   N/A  N/A    215156      G   ...R2021a/bin/glnxa64/MATLAB        3MiB |
|    0   N/A  N/A    216972      G   ...GL_KHR_blend_equation_adv        5MiB |
|    0   N/A  N/A    284196      C   ./stream_triad_cuda               113MiB |
+-----------------------------------------------------------------------------+
```

1 Task 1

Done.

2 Task 2



(a) full range

(b) range=L2-L3

(c) range=L1-L2

(d) range=0-L1

Figure 1: x-axis=array length N

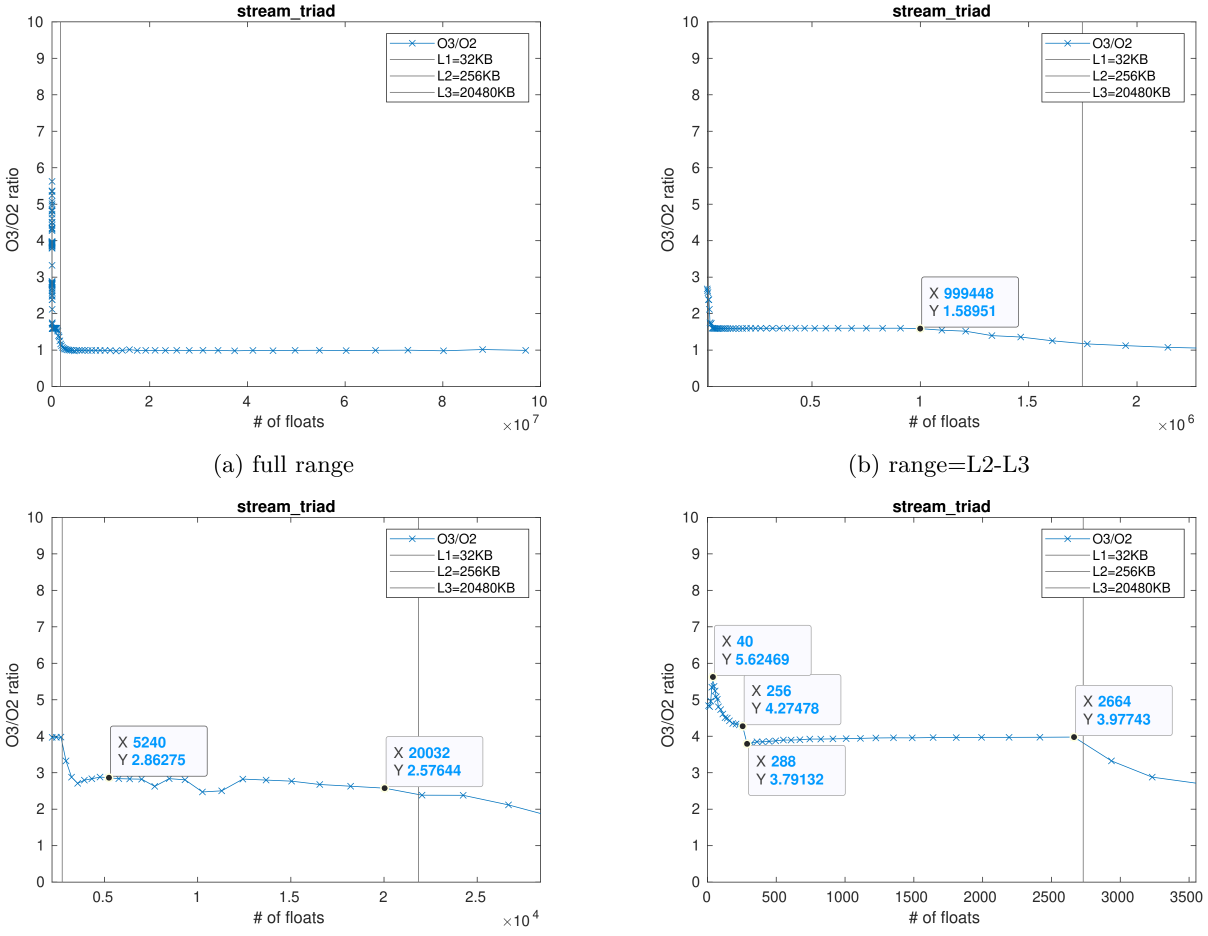
The vertical line L1, L2, and L3 are drew at the x position when array size N*3 = cache size. Therefore they are used to indicate in which memory region the data points are.

The sub-figure 1a shows that when the array size is large and stored in the DRAM, the O2 and O3 program performances are bounded by the DRAM bandwidth which is approximately 12.5 GB/s in my test. However, I cannot explain why the memory throughput is much lower than the DDR3-4ch 42.6 GB/s and slightly above the DDR3-1ch 10.6 GB/s. It might be something to do with the single core, cache lines, and data transferring from the DRAM but I cannot provide a solid explanation right now.

The sub-figure 1b shows that when the array size is within the L3 cache size, the O2 program is bounded by its own calculation performance which is approximately 16.8 GB/s and the O3 program is bounded by the L3 bandwidth which is approximately 26.9 GB/s. However, I cannot explain why there is a slow curve drop starting at N=1e6 in both O2 and O3 programs. My hypothesis is that the L3 cache is shared with all cores, therefore there are some L3 cache already occupied by other programs.

The sub-figure 1c shows that when the array size is within the L2 cache size, the O2 program is bounded by its own calculation performance which is approximately 17.7 GB/s, which is very close to the result in the sub-figure 1b, and the O3 program is bounded by the L2 bandwidth which is approximately 50.4 GB/s. However, I cannot explain why there is a slow curve drop starting at N=1.5e4 in the O3 programs. My hypothesis is that the CPU frequency is dynamic and drops when it is SIMD instruction. Also, the heat accumulated from the long execution time would cause the CPU frequency drop.

The sub-figure 1d shows that when the array size is within the L1 cache size, the O2 program is bounded by its own calculation performance which is approximately 17.7 GB/s, which is very consistent with the result in the sub-figure 1b and 1c. The O2 program reaches its maximum performance approximately at N=256. The O3 program reaches its maximum performance approximately 70 GB/s. However, due to the lack of the detailed runtime information such as the SIMD execution length, the CPU frequency, and etc., I cannot determine the O3 program is bounded by its calculation performance or L1 bandwidth. There is a obvious performance drop between N=256 and N=288 for the O3 program, which I cannot explain it either. My guess would be the transition between the register memory to the L1 cache memory. The performance drops the most when the data just exceeds the register size because the latency of L1 cache affects the most. When the data is close to the L1 size, the L1 latency would be compensated by the high bandwidth of the L1 cache. It could be why it is able to reach the same performance as within the register region.



(a) full range

(b) range=L2-L3

(c) range=L1-L2

(d) range=0-L1

Figure 2: x-axis=array length N

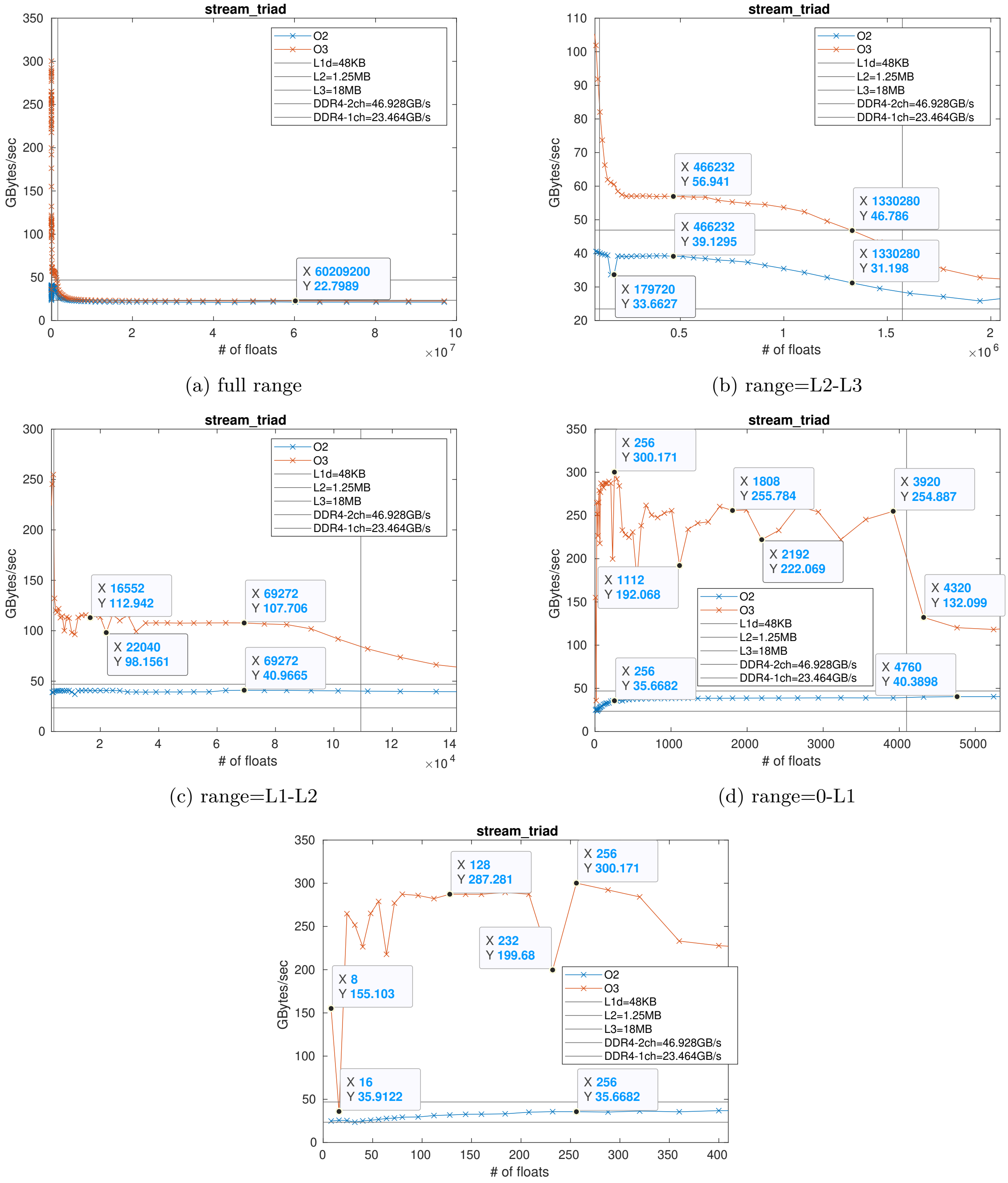
The sub-figure 2a shows that the O2 and O3 program performances are the same in the DRAM region.

The sub-figure 2b shows that the O3/O2 performance ratio is approximately 1.6 in the L3 cache region. However, there is a slow curve drop starting at N=1e6, which is discussed above.

The sub-figure 2c shows that the O3/O2 performance ratio is approximately 2.8 in the L2 cache region. The same slow curve drop at N=1.5e4 due to the O3 program performance drop.

The sub-figure 2d shows that the O3/O2 performance ratio is approximately 3.9 in the L1 cache region. The same curve drop between N=256 and N=288. In the very small N region, the O3/O2 ratio reaches the maximum approximately 5.6 at N=40 and then slowly decreases until N=256. Sadly, I cannot give a theoretical explanation for the behavior of the ratio in this plot. My hypothesis for the drop at N=256 is because of the memory region from the registers to the L1. My hypothesis for the maximum speed-up ration not equaling to 8(256/32, for AVX256) and the following drop is the dynamic CPU frequency adjustment, which is discussed above.

3 Task 3



(a) full range

(b) range=L2-L3

(c) range=L1-L2

(d) range=0-L1

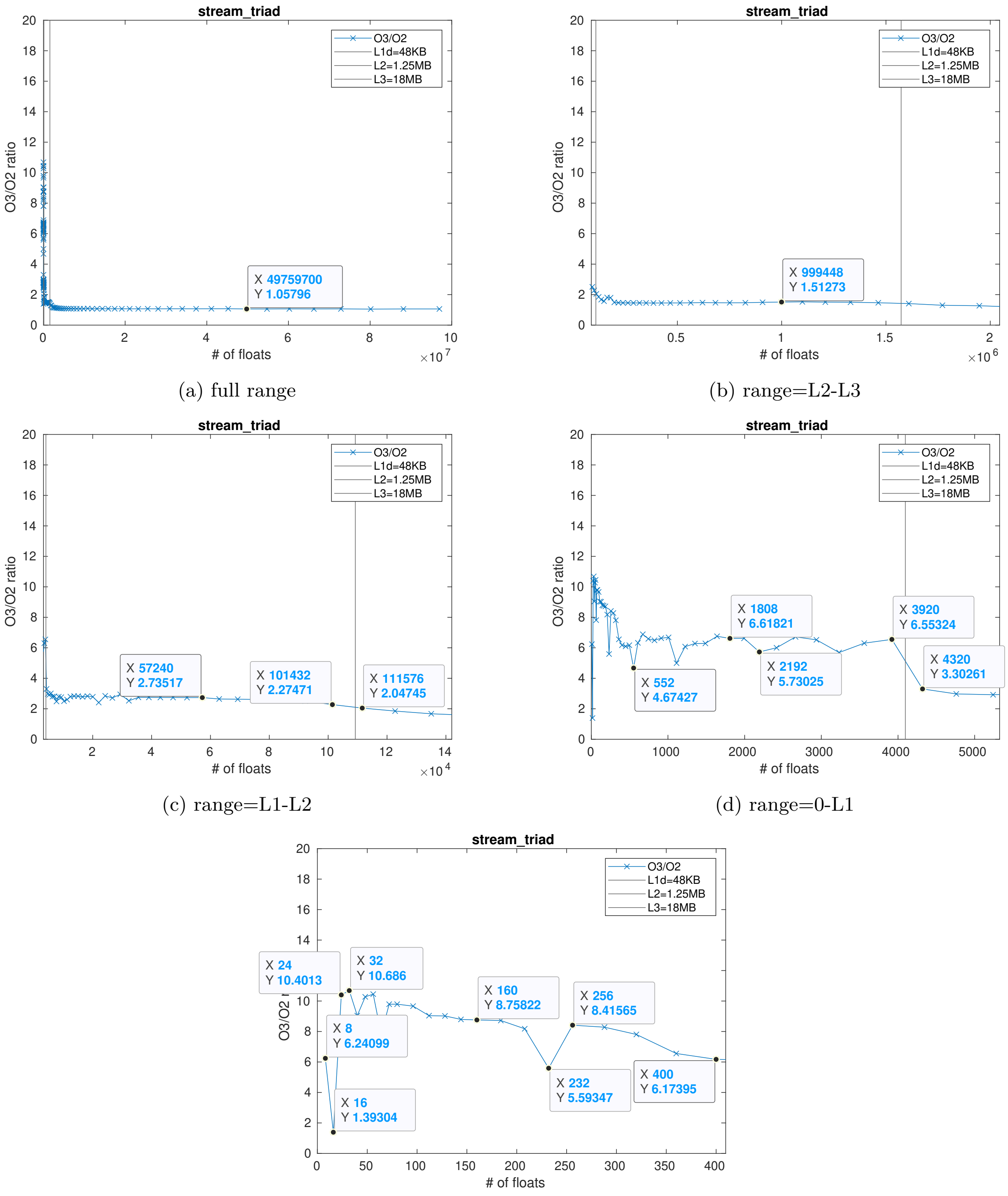
(e) range=0-L1/10

Figure 3: x-axis=array length N

The performance is definitely improved. The following discussion would be focused on the differences between task-3 and task-2.

The sub-figure 3a shows that the O2 and O3 program performances in the DRAM region reach the memory bandwidth bound which is DDR4-1ch 23.464 GB/s. It is more reasonable but I cannot explain why it is single channel bandwidth, not dual channel bandwidth.

The sub-figure 3c, 3d, and 3e shows that starting from the N=4e4; approximately half of the L2; there are periodic performance oscillations. I cannot explain what caused this phenomenon. There is also a strange performance drop especially at N=16.



(a) full range

(b) range=L2-L3

(c) range=L1-L2

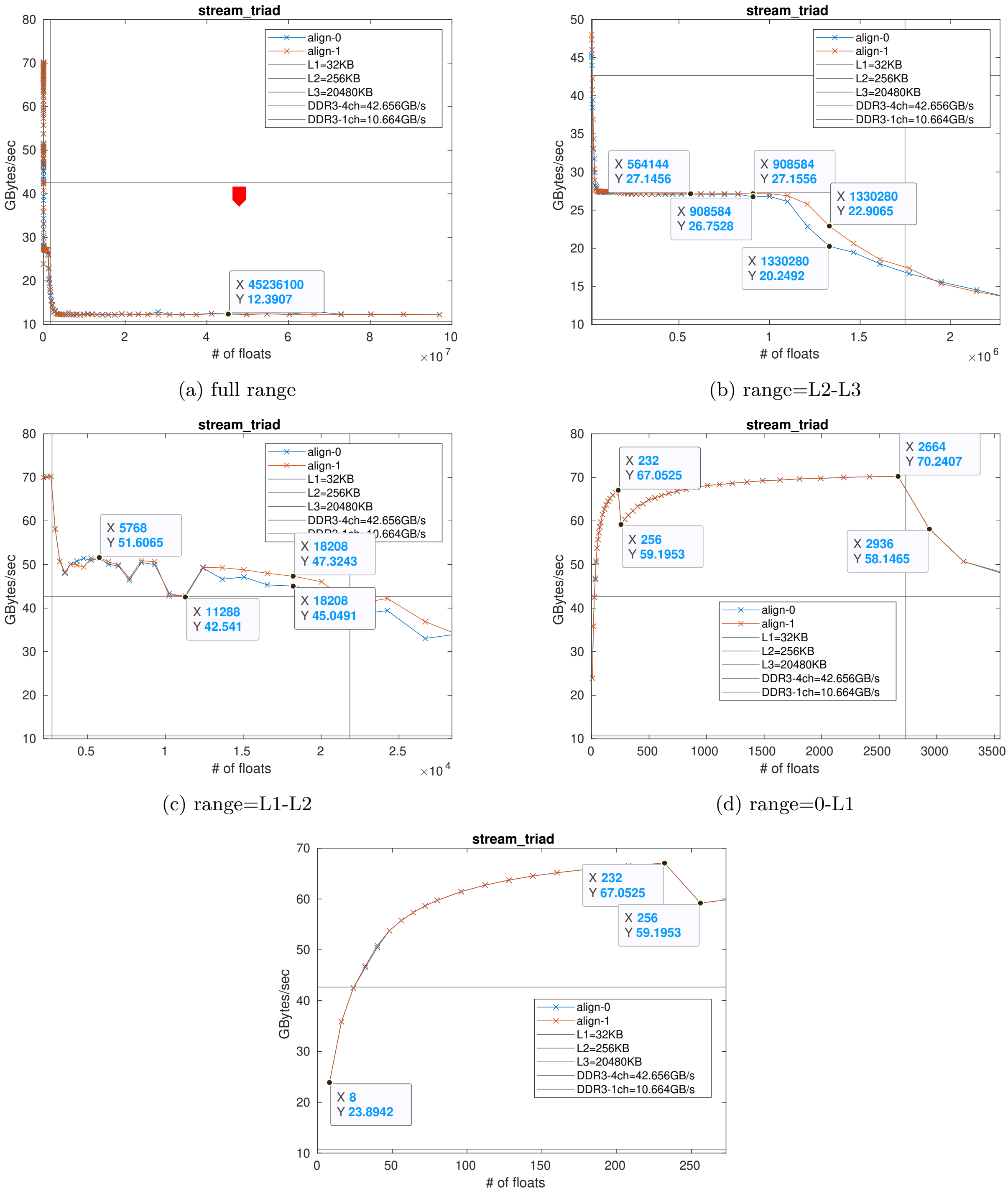
(d) range=0-L1

(e) range=0-L1/10

Figure 4: x-axis=array length N

The maximum O3/O2 ratio, which is caused by SIMD and other minor optimisation, is 10.6 achieved at N=32.

4 Task 4



(a) full range

(b) range=L2-L3

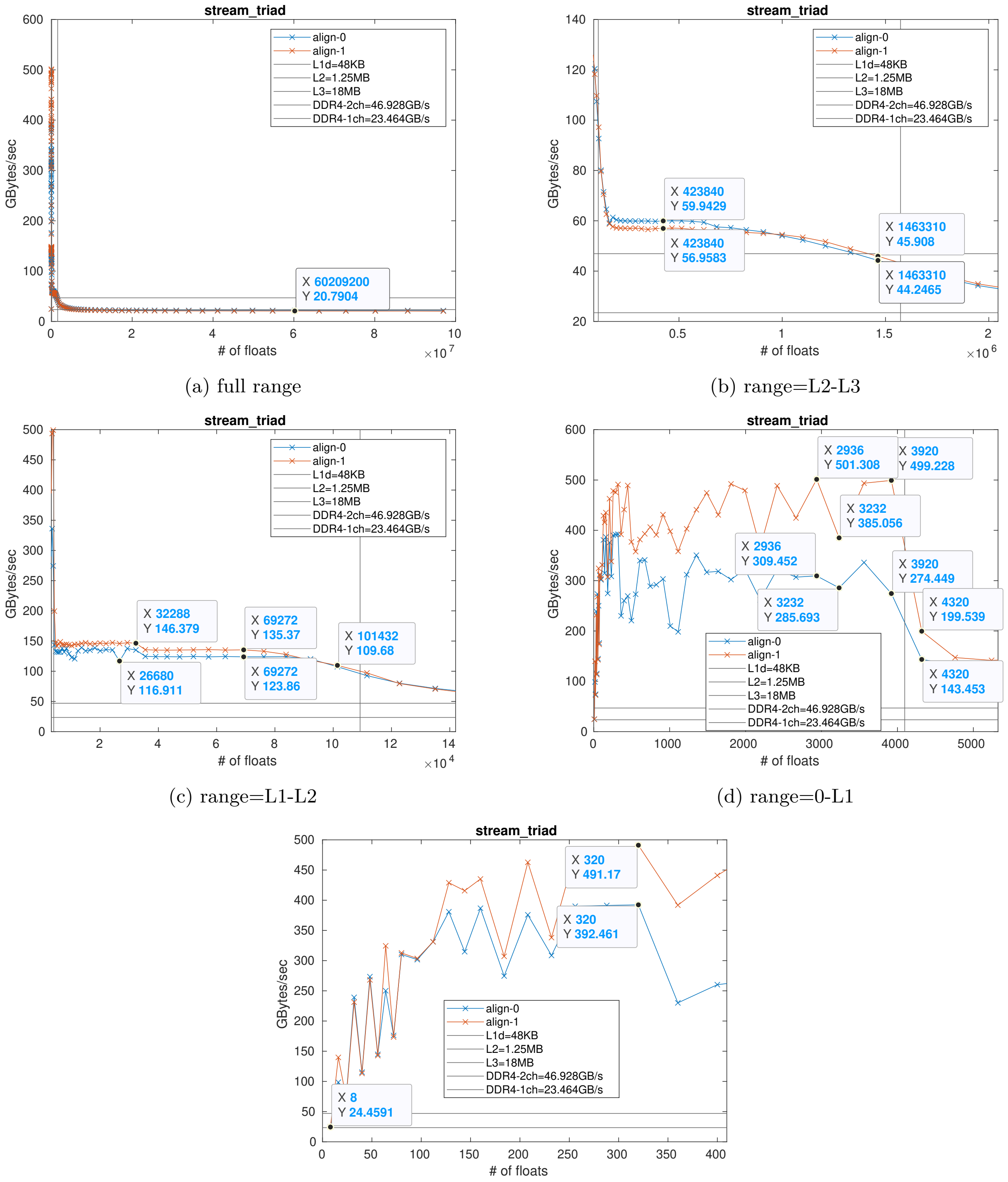
(c) range=L1-L2

(d) range=0-L1

(e) range=0-L1/10

Figure 5: E5-2660, x-axis=array length N

For the snowy CPU E5-2660, the figure 5 and 1 are very consistent. It means that the compiler could analyze and SIMD the "stream_triad" code very well. However, the memory alignment effect does not show in the task 4, even in the very small N region. The memory alignment effect to the performance should play an important role especially when the N is small and may lower its importance when the N is large. I cannot explain the figure 5e right now.



(a) full range

(b) range=L2-L3

(c) range=L1-L2

(d) range=0-L1

(e) range=0-L1/10

Figure 6: W-11855M, x-axis=array length N

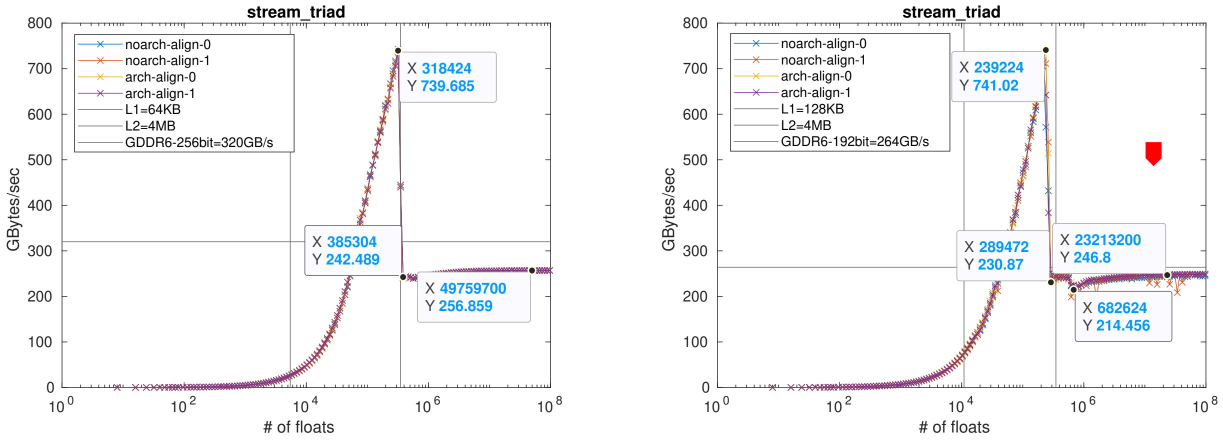
For the W-11855M CPU, the “stream_triad_SIMD” code achieves twice higher performance than the “stream_triad” O3 code. It means that the hand-crafted x86 instruction code(vectorization.h) is still better than the compiler generated O3 code. It is very reasonable and shows the limitation of compiler optimization.

The memory alignment has huge effect in this W-11855M CPU. In the figure 6b, in the approximately first one-third L3 region, the non-aligned code has even slightly higher performance than the aligned one. It is very odd and I cannot give a good explanation for it.

In figure 6c, it starts to show the performance difference due to the alignment. In the figure 6d, which is the L1 region, the aligned code could have maximum 5/3 times higher performance than the non-aligned one.

However, in the figure 6e, which is the very small N region, the performance difference between the aligned and non-aligned programs vanishes. It contradicts my current knowledge and I have no explanation for it.

5 Task 5



(a) T4,full range

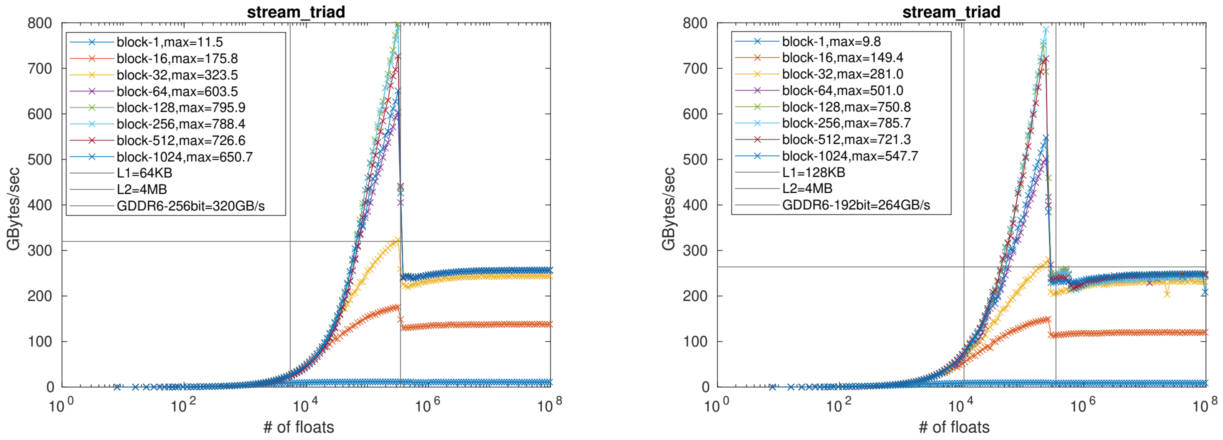
(b) A3000,full range

Figure 7: x-axis=array length N

The GPU performance is much higher than the CPU when the N is large and vice versa when the N is small because of the hardware architecture difference. The CPU is low latency, low memory bandwidth but the GPU is high latency and high memory bandwidth.

The T4 and A300 GPU both achieve the maximum performance 740 GB/s at the end of L2 cache. When the N is larger than the L2 cache, the performance immediately drops to the global GDDR memory bandwidth. However, the A3000 performance ideally matches the theoretical GDDR memory maximum bandwidth but the T4 performance is considerable lower than its theoretical bandwidth. The T4 and A3000 have the same performance and I suspect it is because they have the same L1 and L2 memory bandwidth, especially when they are both memory bandwidth bound in this code.

6 Task 6



(a) T4,full range

(b) A3000,full range

Figure 8: x-axis=array length N

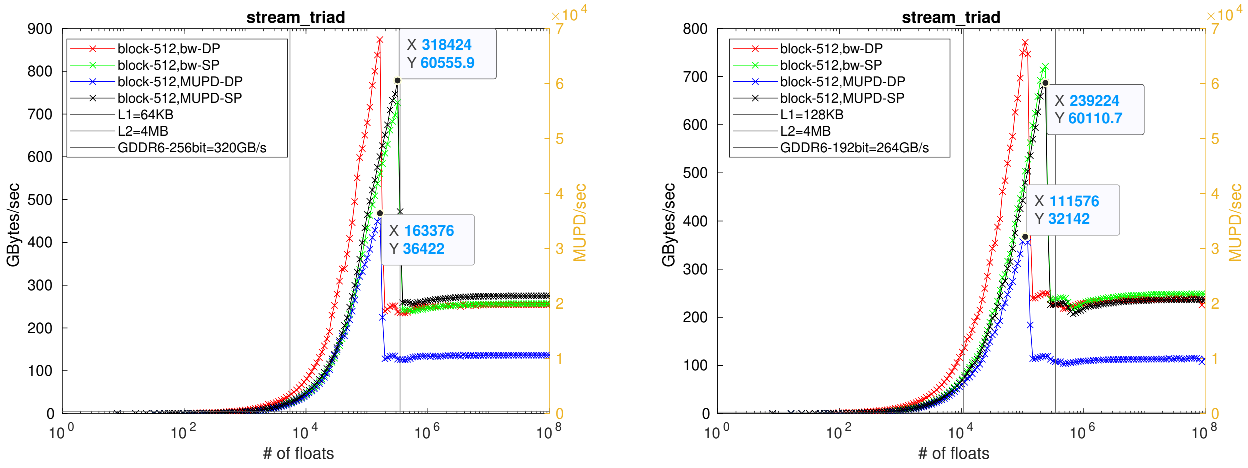
First of all, the maximum block size is 1024, therefore the experiment range is between 1 to 1024.

For the T4 GPU, the maximum performance is 796 GB/s at block-128. For the A3000 GPU, the maximum performance is 786 GB/s at block-256.

A general guideline for an appropriate block size is:

- The block number should be equal to or larger than the streaming multiprocessor number.
- The block size should be larger than and divisible by the warp size.
- The block size should contain multiple warps in order to provide enough warp context switching to hide the data transfer time.
- The memory occupancy should be calculated.

7 Task 7



(a) T4,full range

(b) A3000,full range

Figure 9: x-axis=array length N

Unlike the CPU architecture, the performance ratio of FP64 and FP32 are usually not 1:2 ratio but depends on its various models and architectures. In this task 7, because the performances are mainly memory bandwidth bound, the FP32 and FP64 performance are almost identical. However, the million updates per second(MUPD/s) is decided by the memory bandwidth and the variable size. When the variable size increases from FP32 to FP64, the MUPD/s is cut to half as the theoretical prediction.