GPU Computing

Exercise Sheet 03

Maximilian Richter, Jan Kränzke, Markus Everling,
Nicolas Schledorn, Olga Sergeyeva, Florian Feltz

November 11, 2024

1 Global Memory - cudaMemcpy

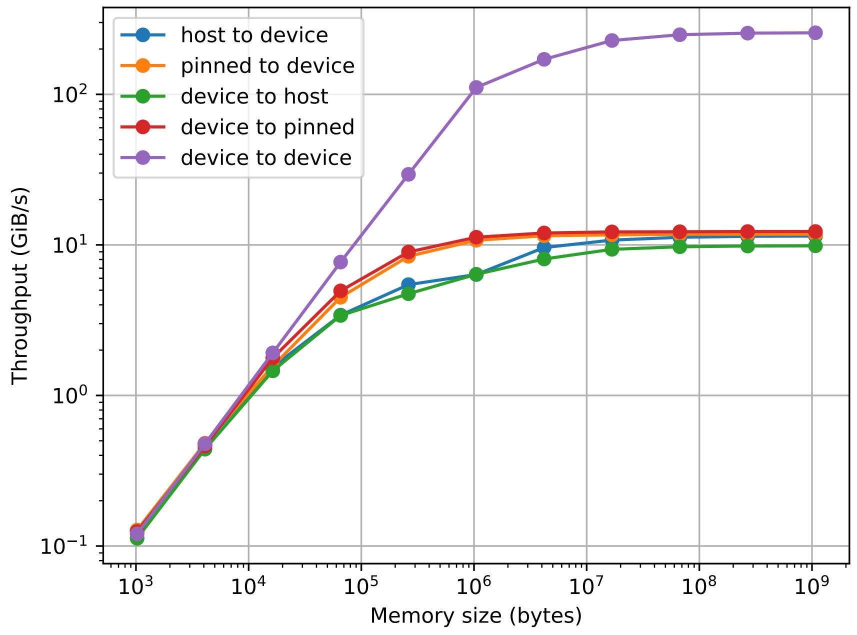

Figure 1: Throughput of different memory allocation functions and directions.

It is evident from the graph, that for small memory sizes, all copy operations are dominated by their launch time. For large memory sizes, they instead are limited by their respective memory throughput. This throughput is lowest for pageable host memory, slightly higher for pinned host memory, and significantly higher for device to device copies. This is because all copies between host and device are limited by the PCIe bandwidth, which in our case is roughly 16 GB/s, whereas device to device copies are limited by VRAM bandwidth, which seemed to be ∼280 GB/s on the cluster.

2 Global Memory – coalesced thread access

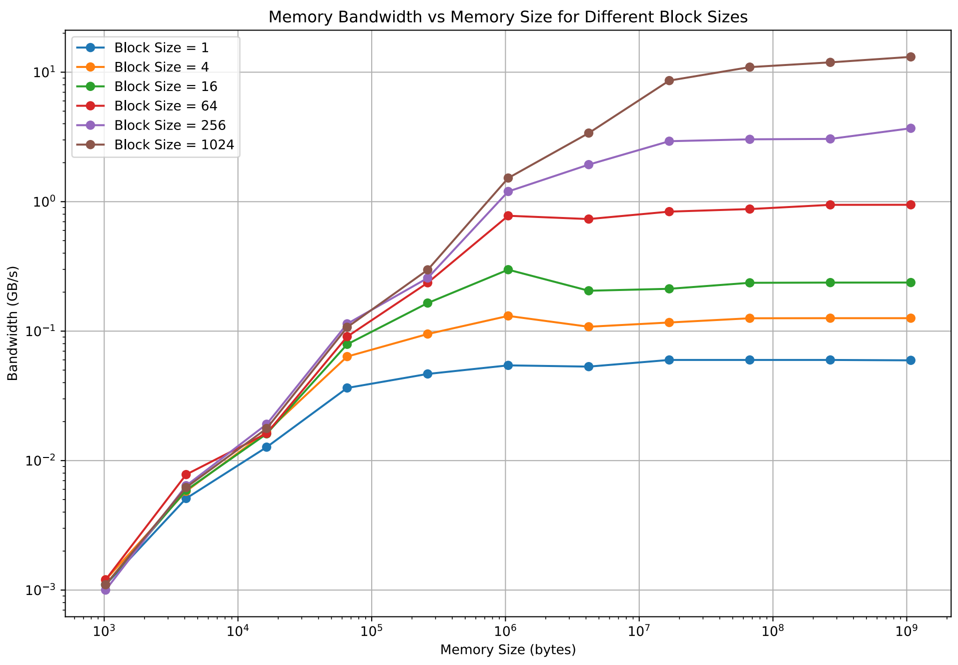

Figure 2: Throughput of coalesced memory accesses for different block and memory sizes

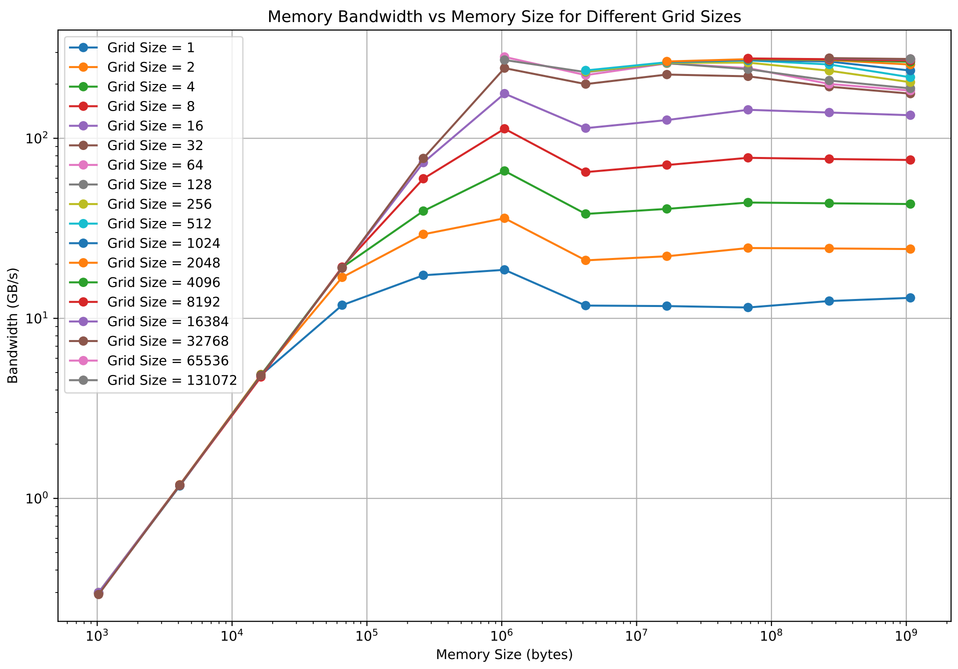

Figure 3: Throughput of coalesced memory accesses for different memory and grid sizes

From the graphs, it can be seen that higher block and grid sizes lead to significant increases of memory bandwidth. Furthermore, the bandwidth drops once the memory size passes the cache size. After that, the maximum bandwidth achievable reaches saturation which happens due to RAM bandwidth limitations. The highest bandwidth is observed when each thread only copies one element of the array, i.e. when the grid size is maximal.

3 Global Memory – thread access with stride

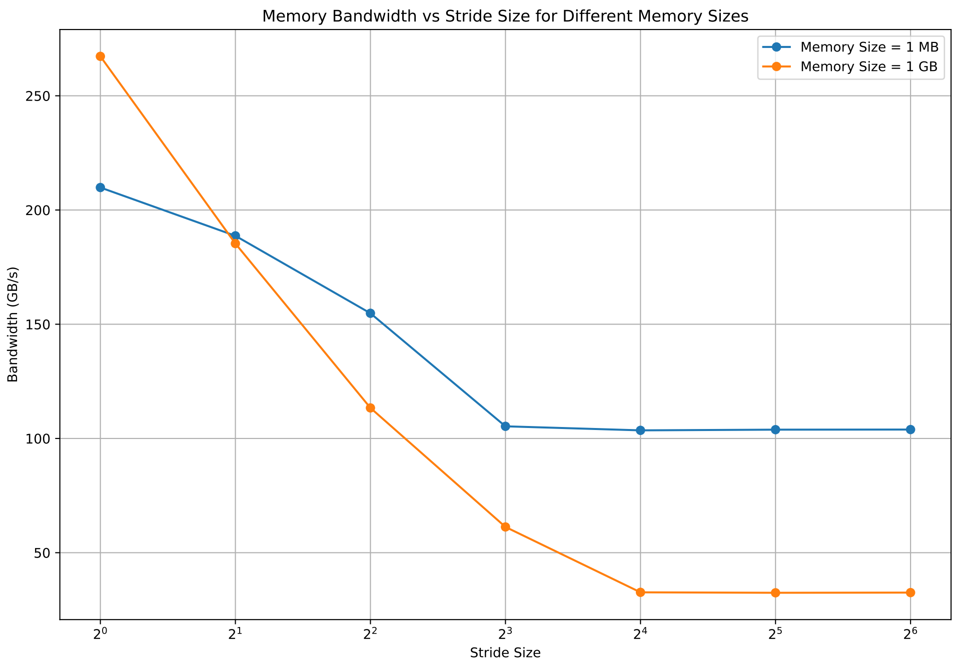

Figure 4: Throughputs over strides for 1 MB and 1 GB total sizes

From the graphs, it is evident that higher strides lead to lower memory throughput as the loads cannot be coalesced anymore. The effect is much more drastic for 1 GB copies than for 1 MB copies, as the data set doesn't fit into cache anymore and every single thread has to fetch its own cache line. As expected, for a stride of 1 the performance matches that of a memcpy. The throughput plateaus at a stride of 16, which suggests that the cache line size of the device on the cluster is $4 \cdot 16 = 64$ bytes.

4 Global Memory – thread access with offset

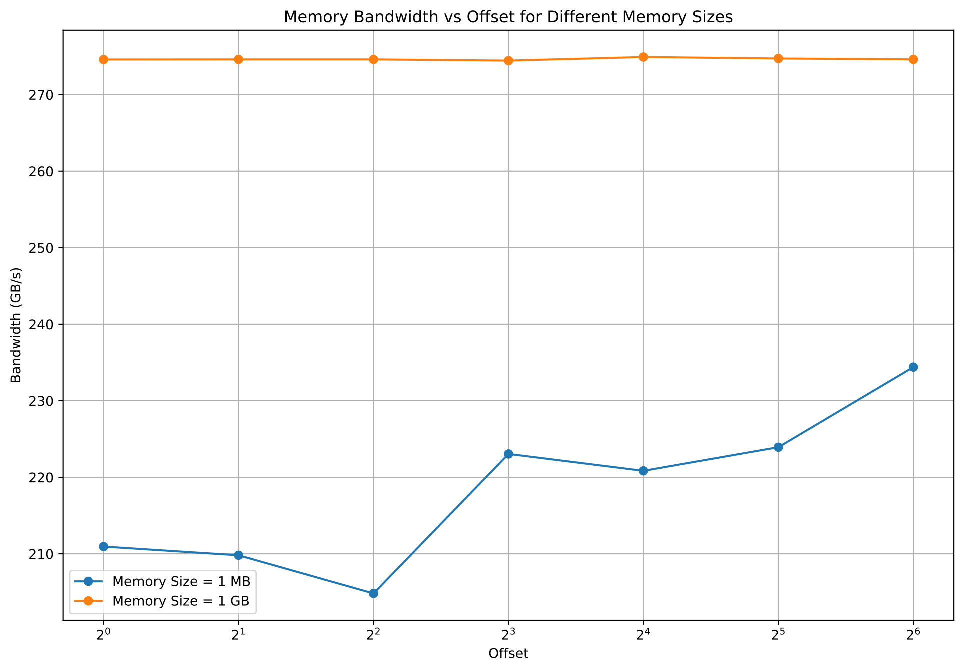

Figure 5: Throughputs over offsets for 1 MB and 1 GB total sizes

The graphs suggest that differing offsets do not influence the memory bandwidth of the kernel. This is expected, as the memory accesses per block are still contiguous and can thus be coalesced just as well as with non-offset memory accesses.

5 Willingness to present

- Global Memory – cudaMemcpy: **true**
- Global Memory – coalesced thread access: **true**
- Global Memory – thread access with stride: **true**
- Global Memory – thread access with offset: **true**

Appendix

Sometime in the evening:

```
JOBID  PARTITION       NAME       USER  ST          TIME   NODES  NODELIST(REASON)
32846  exercise-      ex3.sh     gpu11  PD          0:00       1  (Resources)
32847  exercise-      memCpy     gpu03  PD          0:00       1  (Nodes required for
job are DOWN, DRAINED or reserved for jobs in higher priority partitions)
32848  exercise-      run.py     gpu06  PD          0:00       1  (Priority)
32414  exercise-      memCpy     gpu05   R       3:23:14       1  csg-brook02
32602  exercise-      memCpy     gpu05   R       1:21:24       1  csg-brook02
32623  exercise-        bash     gpu08   R       1:12:32       1  csg-brook02
32666  exercise-      memCpy     gpu05   R         55:47       1  csg-brook02
32826  exercise-      memCpy     gpu05   R         23:46       1  csg-brook02
32837  exercise-        bash     gpu11   R         20:57       1  csg-brook02
32838  exercise-      memCpy     gpu05   R         20:56       1  csg-brook02
```

Half 2 at night:

```
JOBID  PARTITION       NAME       USER  ST          TIME   NODES  NODELIST(REASON)
32602  exercise-      memCpy     gpu05   R       2:59:21       1  csg-brook02
32623  exercise-        bash     gpu08   R       2:50:29       1  csg-brook02
32666  exercise-      memCpy     gpu05   R       2:33:44       1  csg-brook02
32826  exercise-      memCpy     gpu05   R       2:01:43       1  csg-brook02
32838  exercise-      memCpy     gpu05   R       1:58:53       1  csg-brook02
```

"Man sieht halt einfach ernsthaft wie gpu05 seit Stunden den Cluster belegt" - Wir um halb 2 Uhr morgens