Lab 3 Matrix Multiplication

Bryce Hills - 862073105

1. On Bender, compare the execution time of a 256 x 256 square matrix multiplication compared to a 1024 x 64 and 64 x 1024 rectangular matrix multiply. All input matrices have 65k entries. What do you observe? Which is faster? Can you explain the observed behavior? Tip: You may want to comment out the `verify()` function in `main.cu` when timing this question.
    a. Execution time for 256 x 256 = 0.000121s
    b. Execution time for 1024 x 64 * 64 x 1024= 0.000196s
    c. The execution time for the 256 x 256 square matrix was faster than the rectangular matrices. This is likely because there will be less warp divergence when handling boundary conditions for a square matrix rather than rectangular matrices. It is also more likely that a square tile will more accurately match a square matrix and thus have less total divergence.

2. Conceptual Question: For a 64 square *tiled* matrix multiplication, how many times is each element of the input matrices loaded from global memory? Assume 16x16 tiles.
    a. In the tiled example, each element of data will be loaded by 64/16 = 4 blocks. Therefore each element is loaded **4 times**.

3. Conceptual Question: For a 64 square *non-tiled* matrix multiplication, how many times is each element of the input matrices loaded from global memory?
    a. The non tiled example, we will have 16 * 16 * 64 = 16384 total loads from global memory, which means each element gets loaded **64 times**.

4. GPGPU-Sim related question: In this part, we will compare the execution of a 128x128 square tiled matrix multiplication across different tile sizes. Run `./sgemm-tiled 128` in GPGPU-Sim with `TILE_SIZE` of 8, 16 (default), and 32. Fill the following table:

Tile size	**8**	**16**	**32**	**Note**
gpu_tot_sim_cycle	43,433	27,727	57,904	Total cycles

	gpu_tot_ipc	420.6	460	390	Instruction per cycle
	gpgpu_n_load_insn	524,288	262,144	131,072	Total loads to global memory
	gpgpu_n_store_insn	16,384	16,384	16,384	Total stores to global memory
	gpgpu_n_shmem_insn	4,718592	4,456,448	4,325,376	Total accesses to shared memory

5. Which tile size resulted in the least number of accesses to global memory? Which tile size resulted in the most number of accesses to global memory? What is the reasoning behind this observation?
    a. Tile size 32 had the least number of accesses to GM.
    b. Tile size 8 had the greatest number of accesses to GM.
    c. Since the size of the amount of shared memory is larger for 32 x 32, then we would less often need to access global memory since it provides more coverage over the input matrix and requires less total blocks. The inverse is true for the 8x8 tile size. Since shared memory is smaller, we won't be able to cache as much and thus have to access global memory more frequently.

6. Which tile size performed the fastest, which tile size performed the slowest? Why do you think that is?
    a. 16x16 gives us 256 threads per block which is the optimal thread amount in this case, since there are thread and block limitations by the SM. We are underutilizing blocks in the case of 32 x 32 and underutilizing the number of threads in the 8 x 8 case. 16x16 utilizes the optimal amount of threads and blocks for the SM, so it requires the least amount of total cycles and gives the most instructions per cycle.

Tile size 8:

```
gpu_sim_cycle = 43433
gpu_sim_insn = 18268160
gpu_ipc =     420.6055
gpu_tot_sim_cycle = 43433
gpu_tot_sim_insn = 18268160
gpu_tot_ipc =     420.6055
gpu_tot_issued_cta = 0
gpu_stall_dramfull = 26125
gpu_stall_icnt2sh    = 101591
gpu_total_sim_rate=961482
```

```
gpgpu_n_load_insn  = 524288
gpgpu_n_store_insn = 16384
gpgpu_n_shmem_insn = 4718592
```

Tile Size 16:

```
gpu_sim_cycle = 27727
gpu_sim_insn = 12763136
gpu_ipc =     460.3144
gpu_tot_sim_cycle = 27727
gpu_tot_sim_insn = 12763136
gpu_tot_ipc =     460.3144
gpu_tot_issued_cta = 0
gpu_stall_dramfull = 19792
gpu_stall_icnt2sh    = 61626
gpu_total_sim_rate=981779
```

```
gpgpu_n_load_insn  = 262144
gpgpu_n_store_insn = 16384
gpgpu_n_shmem_insn = 4456448
```

Tile Size 32:

```
gpu_sim_cycle = 57904
gpu_sim_insn = 22593536
gpu_ipc =     390.1895
gpu_tot_sim_cycle = 57904
gpu_tot_sim_insn = 22593536
gpu_tot_ipc =     390.1895
gpu_tot_issued_cta = 0
gpu_stall_dramfull = 7546
gpu_stall_icnt2sh    = 20818
```

```
gpgpu_n_load_insn  = 131072
gpgpu_n_store_insn = 16384
gpgpu_n_shmem_insn = 4325376
```