8 GPUs and SIMD [75 points]

We define the *SIMD utilization* of a program that runs on a GPU as the fraction of SIMD lanes that are kept busy with *active threads* during the run of a program.

The following code segments are run on a GPU. We assume that (1) `A` resides in memory and is shared by all threads, (2) `s` resides in a register and is private to each thread, and (3) the code segments are correct (i.e., do not think about any correctness issues when answering this question).

A warp in the GPU consists of 32 threads, and there are 32 SIMD lanes in the GPU. Each thread executes **a single iteration** of the outermost loop (with index `i`). Assume that the data values of the array `A` are already in vector registers so there are no memory loads and stores in this program. (Hint: Notice that there are 4 instructions in each iteration of the outermost loop of both code segments.)

```
s = 1;
for (i = 0; i < 1024; i++) {
    for (j = 0; j < 10; j++) { // Inst. 1
        if (i % (2 * s) == 0)  // Inst. 2
            A[i] += A[i + 1];  // Inst. 3
        s = s << 1;            // Inst. 4
    }
}
```

Code Segment 1

```
s = 512;
for (i = 0; i < 1024; i++) {
    for (j = 0; j < 10; j++) { // Inst. 1
        if (i < s)             // Inst. 2
            A[i] += A[i + s];  // Inst. 3
        s = s >> 1;            // Inst. 4
    }
}
```

Code Segment 2

Please answer the following questions.

(a) [5 points] How many warps does it take to execute these code segments?

32 warps.

Explanation:
The number of warps is calculated as:
$\#Warps = \lceil \frac{\#Total\_threads}{\#Warp\_size} \rceil$,

where
$\#Total\_threads = 1024 = 2^{10}$ (i.e., one thread per loop iteration),
and
$\#Warp\_size = 32 = 2^5$ (given).

Thus, the number of warps needed to run this program is:
$\#Warps = \lceil \frac{2^{10}}{2^5} \rceil = 2^5 = 32$.

(b) [10 points] What is the SIMD utilization of the first iteration of the inner loop (`j = 0`) for Code Segment 1? Show your work. (Hint: The warp scheduler does *not* issue instructions when no thread is active).

The utilization of the first iteration (`j = 0`) of Code Segment 1 is $\frac{7}{8}$.

Explanation:
Instructions 1, 2, and 4 are executed by all threads in Code Segment 1.

In Code Segment 1, `s = 1` during the first iteration. Thus, only even numbered threads fulfill the predicate of the `if` statement, and only half of the threads of each warp execute Instruction 3.
Code Segment 1, `j = 0`: $SIMD\_utilization = \frac{1024+1024+512+1024}{1024+1024+1024+1024} = \frac{7}{8}$.

(c) [10 points] What is the SIMD utilization of the first iteration of the inner loop (`j = 0`) for Code Segment 2? Show your work. (Hint: The warp scheduler does *not* issue instructions when no thread is active).

The utilization of the first iteration (`j = 0`) of Code Segment 2 is 100%.

Explanation:
Instructions 1, 2, and 4 are executed by all threads in Code Segment 2.

In Code Segment 2, `s = 512` during the first iteration. Thus, only threads with `i < 512` fulfill the predicate of the `if` statement, and all threads of only half of the warps execute Instruction 3.
Code Segment 2, `j = 0`: $SIMD\_utilization = \frac{1024+1024+512+1024}{1024+1024+512+1024} = \frac{7}{7} = 100\%$.

(d) [15 points] What is the SIMD utilization of any iteration of the inner loop (`0 <= j < 10`) for Code Segment 1? Show your work. (Hint: Derive an analytical expression, which may be piecewise).

As mentioned in part (b), Instructions 1, 2, and 4 are executed by all threads.

In Code Segment 1, with `0 <= j < 5`, all 32 warps are active, but the number of active threads per warp divides by half in each iteration. With `5 <= j < 10`, only one thread per warp is active, and the number of active warps divides by half in each iteration. As a result:

Code Segment 1, iteration `j`:

$$SIMD\_utilization = \begin{cases} \frac{3072+2^{(9-j)}}{4096}, & \text{if } 0 \leq j < 5 \\ \frac{3072+2^{(9-j)}}{3072+32*2^{(9-j)}}, & \text{if } 5 \leq j < 10 \end{cases} \quad (1)$$

(e) [15 points] What is the SIMD utilization of any iteration of the inner loop (`0 <= j < 10`) for Code Segment 2? Show your work. (Hint: Derive an analytical expression, which may be piecewise).

As mentioned in part (b), Instructions 1, 2, and 4 are executed by all threads.

In Code Segment 2, with `0 <= j < 5`, all 32 threads per warp are active, but the number of active warps divides by half in each iteration. With `5 <= j < 10`, only one warp is active, and the number of active threads divides by half in each iteration. As a result:

Code Segment 2, iteration `j`:

$$SIMD\_utilization = \begin{cases} \frac{3072+32*2^{(4-j)}}{3072+32*2^{(4-j)}} = 100\%, & \text{if } 0 \leq j < 5 \\ \frac{3072+2^{(9-j)}}{3072+32}, & \text{if } 5 \leq j < 10 \end{cases} \quad (2)$$

(f) [10 points] Is there any iteration (`0 <= j < 10`) where both code segments have the same utilization? Explain your reasoning.

Yes, with `j = 9` only one thread of only one warp is active, since only one thread (out of 1024) is needed to perform the last addition.

(g) [10 points] Which code is expected to run faster on a GPU? Explain your reasoning.

Code Segment 2 is faster because it has less intra-warp divergence, and thus higher SIMD utilization. In each iteration (except the last one), the number of warps that Code Segment 2 schedules is smaller than the number of warps that Code Segment 1 schedules. This results in fewer execution cycles.