3 GPUs and SIMD [90 points]

We define the *SIMD utilization* of a program run on a GPU as the fraction of SIMD lanes that are kept busy with *active threads* during the run of a program. As we saw in lecture and practice exercises, the SIMD utilization of a program is computed across the *complete run* of the program.

The following code segment is run on a GPU. Each thread executes **a single iteration** of the shown loop. Assume that the data values of the arrays A, B, and C are already in vector registers, so there are no loads and stores in this program. (Hint: Notice that there are 6 instructions in each thread.) A warp in the GPU consists of 32 threads, and there are 32 SIMD lanes in the GPU. Please assume that all values in arrays B and C have magnitudes less than 10 (i.e., $|B[i]| < 10$ and $|C[i]| < 10$, for all i).

```
for (i = 0; i < 1008; i++) {
  A[i] = B[i] * C[i];
  if (A[i] < 0) {
    C[i] = A[i] * B[i];
    if (C[i] < 0) {
      A[i] = A[i] + 1;
    }
    A[i] = A[i] - 2;
  }
}
```

Please answer the following six questions.

(a) [10 points] How many warps does it take to execute this program?

32 warps

Explanation: number of warps = ⌈ (number of elements) / (warp size) ⌉ = ⌈1008/32⌉ = 32 warps

(b) [10 points] What is the *maximum* possible SIMD utilization of this program?

0.984375

Explanation: We have 31 fully-utilized warps and one warp with only 16 active threads. In case with no branch divergence, the SIMD utilization would be: $(32 \times 31 + 16)/(32 \times 32) = 0.984375$

(c) [20 points] Please describe what needs to be true about arrays B and C to reach the *maximum* possible SIMD utilization asked in part (b). (Please cover all possible cases in your answer)

For each 32 consecutive elements: ((B[i]==0 || C[i]==0) || (B[i] < 0 & C[i] < 0) || (B[i] > 0 & C[i] > 0)) || ((B[i] < 0 & C[i] > 0) || (B[i] > 0 & C[i] < 0))

Explanation: We can have two possibilities:
(1) No threads inside a warp take the first "if". It is possible if for 32 consecutive elements, the following condition is true:
(B[i]==0 || C[i]==0) || (B[i] < 0 & C[i] < 0) || (B[i] > 0 & C[i] > 0)

(2) All threads inside a warp take the first "if". It is possible if for 32 consecutive elements, the following condition is true:
(B[i] < 0 & C[i] > 0) || (B[i] > 0 & C[i] < 0)
This condition also ensures that for the second "if", all threads take the branch or no threads take the branch.
Finally, for every 32 consecutive elements, we should have one of the mentioned possibilities. For the warp with 16 active threads, we should have one of mentioned possibilities for every 16 consecutive elements

(d) [10 points] What is the *minimum* possible SIMD utilization of this program?

((32+32+4)×31+16+16+4)/(32×6×32)=0.3489

Explanation: The first two lines must be executed by every thread in a warp. The minimum utilization results when a single thread from each warp passes both conditions, and every other thread fails to meet the condition on the first "if". The thread per warp that meets both conditions, executes four instructions.

(e) [20 points] Please describe what needs to be true about arrays B and C to reach the *minimum* possible SIMD utilization asked in part (d). (Please cover all possible cases in your answer)

Exactly 1 of every 32 consecutive elements in arrays B and C should have this condition:
B[i] > 0 & C[i] < 0

Explanation: Only one thread in a warp should pass the first condition. Therefore, exactly 1 of every 32 consecutive elements in arrays B and C should have the following condition:
(B[i] < 0 & C[i] > 0) || (B[i] > 0 & C[i] < 0)
However, the thread in a warp which passes the first condition should also pass the second condition. Therefore, the only condition which ensures the minimum possible SIMD utilization is: exactly 1 of every 32 consecutive elements in arrays B and C should have this condition: B[i] > 0 & C[i] < 0. For the warp with 16 active threads, this condition should be true for exactly 1 of the 16 elements.

(f) [20 points] Now consider a GPU that employs *Dynamic Warp Formation (DWF)* to improve the SIMD utilization. As we discussed in the class, DWF dynamically merges threads executing the same instruction (after branch divergence). What is the maximum achievable SIMD utilization using DWF? Explain your answer (Hint: The *maximum* SIMD utilization can happen under the conditions you found in part (e)).

$((32+32)\times31+16+16+32\times4)/((32+32)\times32+32\times4) = 0.985$

Explanation: DWF can ideally merge 32 warps with only one active threads in a single warp. This could be happen if none of the threads inside the warp have overlaps. If it happens, we will run 31 fully utilized warps and one warp with 16 active threads for the first and second instructions. Then, we will have only one fully-utilized warp which executes the remaining 4 instructions.