CS348K: Visual Computing Systems
Class/Reading Response Template

Reminder: please make sure the PDF you submit to Gradescope DOES NOT have your name on it. We will concatenate all responses and give everyone in the class a PDF of all responses.

Part 1: Top N takeaways from discussions in the last class. Note: this part of the response is unrelated to the current reading, but should pertain to the discussion of the prior reading in class (or just discussion in the class in general, if there was no reading):

- What was the most surprising/interesting thing you learned?
    - We discussed about an online image generator from Apache. This web app only allows a user to generate 4 picture at a time because of computational resources. To speed up model inference, the industry split the work into two stages and all things of the first stage is already computed and stored. Maybe the first stage is some common computation and the second stage is fine-tuning the result with prompts. (I think so but I really can not recall it clearly).
- Is there anything you feel passionate about (agreed with, disagreed with?) that you want to react to?
    - I like the parallelism part (CS149 content).
- Did class cause you to do any additional reading on your own? If so, what did you learn?
    - I read some about Nvidia's Ampere architecture. I think the most important thing is it supports TF32 and speeds single-precision work. TF32 uses the same 10-bit mantissa as the half-precision math, and adopts the same 8-bit exponent as FP32 so it can support the same numeric range. It supports quantization aware training, and helps to accelerate deep learning model training and inference.
- Major takeaways in general?
    - Two common ideas of parallelism are SIMD and multithreading.

Part 2: Answers/reactions to instructor's specific prompts for this reading. (Please see course website for prompts).

What is the basic processor building block?
The Execution Unit (EU)

How many times is this block replicated for additional parallelism?
3*8=24

How many hardware threads does it support?
7

What width of SIMD instructions are executed by those threads? Are there different widths supported? Why is this the case?
The width of SIMD are 1, 2, 4, 8, 16, 32.
No.

I think there are two reasons. First, for power of 2 widths, the addresses of vectors can be computed efficiently with bit shifting. Second, the elements of the vector can be loaded into a single block of cache, thus avoid false sharing and improve performance.

Does the core have superscalar execution capabilities?
Yes. An EU can handle maximum 4 instructions because it has 2 floating point ALUs, and one ALU can complete simultaneous add and multiply.

Consider your favorite data-parallel programming language, such as GLSL/HLSL shading languages from graphics, CUDA, OpenCL, ISPC, NumPy, TensorFlow, or just an OpenMP #pragma parallel for. Can you think through how an "embarrassingly parallel" for loop can be mapped to this architecture? (You don't need to write this down in your writeup, but you could if you wish.)

For example, use ISPC and launch 8 tasks (instead of 7).
Use CUDA and divide into 8 blocks.

Compare Nvidia GPU and Intel GPU?
The Nvidia V100 is based on the Volta architecture, while the Intel Gen9 is based on the Skylake architecture.

Nvidia GPU focus on deep learning and other high performance computing applications while Intel GPU can be used in a wide range of computing applications.

Nvidia GPU consists of SMs, which contains CUDA cores (for floating-point computations) and Tensor Cores (for matrix and tensor operations). Intel GPU consists of EUs, which uses Hyper-Threading.

Nvidia GPU has less control units like fetch decode, while Intel GPU has more of them.

Part 3: [Optional] Questions I'd like to have specifically addressed via in class. (We also encourage you to just post these questions on Ed immediately so anyone can answer!)