HW8.4. How To Parallelize

At this point, we have discussed 2 levels of parallelism:

- Parallelism of data (ex. SIMD instructions), which has low overhead, but is limited by a small set of possible operations.
- Parallelism of threads/processes (ex. OpenMP), which has high overhead (often on the order of thousands of normal operations), and generally requires splitting the problem into several independent problems to avoid data races, but is generally more versatile than SIMD.

For each of the following programming problems, determine the most appropriate parallelism method, or indicate that neither method of parallelism would be useful. If both methods of parallelism are useful, answer with "Thread-level Parallelism"

Q1.1: You want to compute the average number of white pixels per frame of a 3 minute video

- (a) Neither method of parallelism is useful.
- (b) Data-level Parallelism
- [✓ (c) Thread-level Parallelism] ✓

✓100%

Q1.2: You want to compute the bitwise XOR of two buffers, each of which is 512 bytes long.

- (a) Neither method of parallelism is useful.
- [✓ (b) Data-level Parallelism] ✓
- (c) Thread-level Parallelism

✓100%

Q1.3: You want to search a sorted list of a million integers for a specific value.

- [✓ (a) Neither method of parallelism is useful.] ✓
- (b) Data-level Parallelism
- (c) Thread-level Parallelism

✓100%

Q1.4: You want to run Snake on a 10,000 x 10,000 board with 100,000 active food pellets and 3 snakes.

- [✓ (a) Neither method of parallelism is useful.] ✓
- (b) Data-level Parallelism
- (c) Thread-level Parallelism

✓100%

Q1.5: You want to run `dot` on two vectors of length 100, with stride lengths 1 and 2.

- [✓ (a) Neither method of parallelism is useful.] ✓
- (b) Data-level Parallelism
- (c) Thread-level Parallelism

✓100%

Try a new variant

Correct answer

Q1.1: You want to compute the average number of white pixels per frame of a 3 minute video

Total points:	60/100
Score:	60%

Question

Value:	10
History:	10
Awarded points:	10/10

Report an error in this question

Attached files

No attached files

Attach a file

Attach text

(c) Thread-level Parallelism

1: We can run a thread for each frame separately, then add up the white pixels through a reduction.

Q1.2: You want to compute the bitwise XOR of two buffers, each of which is 512 bytes long.

(b) Data-level Parallelism

2: The size of this problem is sufficiently small that multithreading will likely take longer than just doing it normally. It fits well with data-level parallelism, though.

Q1.3: You want to search a sorted list of a million integers for a specific value.

(a) Neither method of parallelism is useful.

3: We *could* do this by splitting the data between multiple threads and using each thread to check each part of the array, but that will be way slower than a binary search. The binary search is not easy to parallelize, since we need to decide at each step what to do next, and we can't easily split the problem into smaller parts. Ultimately, a faster algorithm tends to take priority over parallelism.

Q1.4: You want to run Snake on a 10,000 x 10,000 board with 100,000 active food pellets and 3 snakes.

(a) Neither method of parallelism is useful.

4: Neither is beneficial since there are only 3 active snakes and the overhead for parallelization far outweighs the very marginal gains of parallelizing the movement of the 3 snakes. There's also no parallelization that can be done on the board size nor the pellets, since you don't actually do anything for each pellet or board square.

Q1.5: You want to run `dot` on two vectors of length 100, with stride lengths 1 and 2.

(a) Neither method of parallelism is useful.

5: Since the second vector isn't in consecutive memory addresses, we can't easily put things into SIMD registers. The problem is also small enough that OpenMP will likely just slow us down.

Submitted answer 3 **correct: 100%**

Submitted at 2022-11-09 03:38:28 (PST)

show