Midterm, Parallel Programming (EN 620.320/420/620), Fall 2020

October 19, 2020 @ 8 am – October 21, 2020 @ 10 am.

This is an untimed exam that will be available for 50 hours. It is expected to take 2 hours to complete. You may use any static online or written resources to research the topics raised in the questions. The answers that you give must be **solely your own**, written and prepared by you individually. You may not discuss your answers to the exam with anyone, including other students in the class, past students, and in online forums.

Please be careful to answer all parts of all questions. Each question mark indicates that a response is required. Look for directives such as *Explain*, *Define*, or *Give* and follow these instructions. Please keep your answers brief. Answers that include extraneous facts and irrelevant details will lose credit even if the correct answer is included in the response.

The assignment should be submitted to GradeScope within the specified time.

1. (*25 pts*) Amdahl's law forwards and backwards.

   (a) (*7 pts*) You run a profiler on the following program and determine that it spends 95% of it's time inside the for loop. What is the maximum possible speedup according to Amdahl's law? Show your work, i.e. how you came to this result starting with the equation for Amdahl's law.

   ```
   //some code outside the loop
   do_outside_thing();

   for (i=0; i<n; i++)
   {
     do_loop_thing();
   }
   ```

   (b) (*7 pts*) You parallelize the for using OpenMP and run it on 8 cores and witness at 4 times speedup. Estimate the Amdahl number (p) that you realized in practice? Again, show your work.

   ```
   //some code outside the loop
   do_outside_thing();

   #pragma openmp parallel for
   for (i=0; i<n; i++)
   {
     do_loop_thing();
   }
   ```

   (c) (*11 pts*) Measurements of your of parallelized version indicates that the loop iterations in each thread all took 133 ms. How long did the thread creation and join code injected by OpenMP take? You should assume that these are startup overheads charged to the unoptimized (serial) part of the computation. You should also assume that the loops have no interference and no skew. Again, show your work.

2. (*35 pts*) Reduction is a fundamental primitive in parallel programming frameworks You are going to write three different implementations of a reduction using different principles. I have given an unsafe implementation at `https://parallel.cs.jhu.edu/midterm/ReduceUnsafe.java`. You should modify this in parts b, c, and d and turn in a 'diff'. For example, the reference answer for part b ran:

```
diff ReduceUnsafe.java ReduceSynch.java
```

This returned 4 clauses that changed 4 lines of the original file and added two new lines to the subsequent file. You should change the file name and class name from ReduceUnsafe to something else. Try and keep the diffs as compact as possible.

   (a) (*5 pts*) Why is ReduceUnsafe.java unsafe, i.e. not thread safe?

   (b) (*5 pts*) Modify the program to update a shared variable in a synchronized block. Turn in the diff output only.

   (c) (*5 pts*) Modify the program to update a shared AtomicInteger. Turn in the diff output only.

   (d) (*15 pts*) Decompose the reduction into operations against a thread local variable and sum the thread local variables after all threads complete. Turn in the diff output only.

   (e) (*5 pts*) When summing a large number of variables so that each thread takes a non-trivial amount of time, which implementation would you expect to be the most efficient and why?

3. (*12 pts*) Consider a NUMA system with 2 processors (P1 and P2) each with 2 cores (C1 and C2). The system has the following cache latencies:

level	**latency**	**sharing**
L1	1 cycles	private to each core
L2	10 cycles	private to each core
L3	25 cycles	shared by cores on the same processor
RAM	135 cycles	shared by all processors

The system cache is *strictly inclusive*. The L1 cache is *write back*. You may further assume that the hardware cache consistency protocol has zero overhead. Answer the following questions and give a one sentence answer that explains why.

   (a) (*3 pts*) P1, C1 has read a memory location X (assume that the clean value is cached in L1). P1, C2 reads the memory location X. What is the read latency for P1, C2? Why?

   (b) (*3 pts*) P1, C1 has written a memory location X (assume that the dirty value is cached in L1). P1, C2 reads the memory location X. What is the read latency for P1, C2? Why?

   (c) (*3 pts*) P1, C1 has read a memory location X (assume that the clean value is cached in L1). P2, C1 reads the memory location X. What is the read latency for P2, C1? Why?

   (d) (*3 pts*) P1, C1 has written a memory location X (assume that the dirty value is cached in L1). P2, C1 reads the memory location X. What is the read latency for P2, C1? Why?