CSCI-UA.0480-051: Parallel Computing
Midterm Exam (March 14th, 2024)

Total: 100 points
Important Notes - READ BEFORE SOLVING THE EXAM

- If you perceive any ambiguity in any of the questions, state your assumptions clearly and solve the problem based on your assumptions. We will grade both your solutions and your assumptions.
- This exam is take-home.
- The exam is posted on Brightspace, at the beginning of the March 14th lecture (2pm EST).
- You have up to 24 hours to submit on Brightspace (i.e. till March 15th 2pm EST), in the same way as you submit an assignment. However, unlike assignments, you can only submit once.
- Your answers must be very focused. You may be penalized for giving wrong answers and for putting irrelevant information in your answers.
- Your answer sheet must be organized as follows:
  - The very first page of your answer must contain only:
    - Your Last Name
    - Your First Name
    - Your NetID
    - Copy and paste the honor code shown below.
  - In your answer sheet, answer one problem per page. The exam has ten main problems, each one must be answered in a separate page.
- This exam consists of 10 problems, with a total of 100 points.
- Your answers can be typed or written by hand (but with clear handwriting). It is up to you. But you must upload one pdf file containing all your answers.

Honor code (copy and paste to the first page of your exam)

- You may use the textbook, slides, the class recorded lectures, the information in the discussion forums of the class on Brightspace, and any notes you have. But you may not use the internet.
- You may NOT use communication tools to collaborate with other humans. This includes but is not limited to Google-Chat, Messenger, E-mail, etc.
- You cannot use LLMs such as chatGPT, Gemini, Bard, etc.
- Do not try to search for answers on the internet, it will show in your answer, and you will earn an immediate grade of 0.
- Anyone found sharing answers, communicating with another student, searching the internet, or using prohibited tools (as mentioned above) during the exam period will earn an immediate grade of 0.
- "I understand the ground rules and agree to abide by them. I will not share answers or assist another student during this exam, nor will I seek assistance from another student or attempt to view their answers."

Problem 1

a. [10] Suppose we have a superscalar core with four execution units. Will this core always achieve a speedup of 4x over a single-execution-unit core? Justify your answer in 1-2 lines.

b. [10] Explain the concept of context switching in the context of an operating system managing multiple processes. What are its performance implications?

c. [10] Describe a scenario where using threads instead of processes would be significantly more efficient. Explain why.

d. [6] What is the purpose of a cache coherence protocol in a multicore system? Why is it important?

Problem 2

Suppose we have the following DAG that represents different tasks and their dependencies:

(Insert DAG image showing a simple directed acyclic graph with at least 5 nodes and their dependencies. Example: Node A -¿ Node B, Node A -¿ Node C, Node B -¿ Node D, Node C -¿ D, Node D -¿ E)

The following table shows the execution time of each task if we execute it on a core of type X and if we execute it on core of type Y. Each CPU type is optimized for some type of operations.

(Insert Table with execution times for each task on CPU type X and CPU type Y. Example: Task — CPU X — CPU Y — A — 2ms — 5ms — B — 3ms — 2ms — C — 1ms — 4ms — D — 4ms — 3ms — E — 2ms — 1ms)

a. [10] What is the minimum number of CPUs of each type that we need to get the highest speedup over sequential execution on CPU of type X? Show which CPU will execute which task(s) and calculate the final speedup.

b. [10] Repeat the problem above but using CPU of type Y.

c. [10] Suppose we can add one more CPU of either type X or type Y. Which type would you choose and why? Justify your answer with a calculation of the expected speedup.

Problem 3

Consider a system with two processors P0 and P1. Both processors have access to a shared memory. Initially, a shared variable ‘counter‘ is set to 0.

```
// Code for P0
counter = counter + 1;

// Code for P1
counter = counter + 1;
```

a. [10] What are the possible final values of ‘counter‘ after both processors have executed their code? Explain why.

b. [10] Explain how a mutex lock can be used to prevent the problem in part (a). Illustrate with pseudocode.

c. [5] What is a race condition?

d. [5] Describe the concept of critical section.

Problem 4

Suppose that MPI_COMM_WORLD consists of four processes 0, 1, 2, and 3, and suppose the following code is executed after MPI has been initialized:

```
int x, y;
MPI_Status status;
if (my_rank == 0) {
  x = 10;
  MPI_Send(&x, 1, MPI_INT, 1, 1, MPI_COMM_WORLD);
} else if (my_rank == 1) {
  MPI_Recv(&x, 1, MPI_INT, 0, 1, MPI_COMM_WORLD, &status);
  y = x * 2;
  MPI_Send(&y, 1, MPI_INT, 2, 2, MPI_COMM_WORLD);
} else if (my_rank == 2) {
  MPI_Recv(&y, 1, MPI_INT, 1, 2, MPI_COMM_WORLD, &status);
  x = y + 5;
  MPI_Send(&x, 1, MPI_INT, 3, 3, MPI_COMM_WORLD);
} else { // my_rank == 3
```

```
  MPI_Recv(&x, 1, MPI_INT, 2, 3, MPI_COMM_WORLD, &status);
  printf("Final value of x: %d\n", x);
}
```

a. [10] What will be the final value of x printed by process 3?

b. [5] Explain the role of the 'MPI$_{S}tatus$'$structure$.

c. [5] What would happen if we removed the tag numbers (1, 2, 3) from the MPI$_{S}endandMPI_{R}ecvcalls$?

d. [5] What is a deadlock in the context of MPI communication? Give an example.

Problem 5

a. [10] Describe Amdahl's Law. What are its implications for the scalability of parallel programs?

b. [10] Explain the difference between strong scaling and weak scaling in parallel computing.

Problem 6

a. [10] What are the main differences between OpenMP and MPI? When would you choose one over the other?

b. [10] Explain how OpenMP directives are used to parallelize a loop. Give an example.

Problem 7

a. [10] Describe the concept of load balancing in parallel computing. Why is it important?

b. [10] Discuss techniques for improving load balancing in a parallel application.

Problem 8

a. [5] Explain the concept of false sharing in a cache.

b. [5] How can false sharing be avoided?

c. [5] What is data locality and why is it important for parallel performance?

d. [5] Explain the difference between spatial and temporal locality.

Problem 9

Consider a parallel program that sorts an array of numbers using a divide-and-conquer approach.

a. [10] Describe a parallel sorting algorithm suitable for this task (e.g., merge sort). Explain how it can be parallelized efficiently.

b. [10] Analyze the runtime complexity of your parallel sorting algorithm, considering both the sequential and parallel components. Discuss the impact of the number of processors on the overall performance.

Problem 10

a. [10] What are some common challenges and difficulties in debugging parallel programs?

b. [10] Discuss debugging tools and techniques that can be used to identify and fix errors in parallel code.