Subject: CAAM 420/520 – Homework 2
Date Posted: April 29, 2023
Date Due: 11:59pm, May 5, 2023 via Canvas

---

**Maximum number of points available:**

**Notes:**

- You cannot get more points back than you lost previously in the semester. We will stop grading once/if you hit your limit.
- You can attempt as many or as few problems as you like and you can pick and choose which problems to do. However, you must **order the problems you attempt, and label them by the numbering/labels used here** (e.g. "Problem 2.b").
- **This assignment cannot be submitted late.**

## Problem 1 Parallel Speed-Up *(30 pts)*

Consider the following parallel speed-up results from the posted solution for the 520 version of HW2 shown in Figure 1. The data for this plot is given in the auxiliary file `hw2Timings.xlsx`.

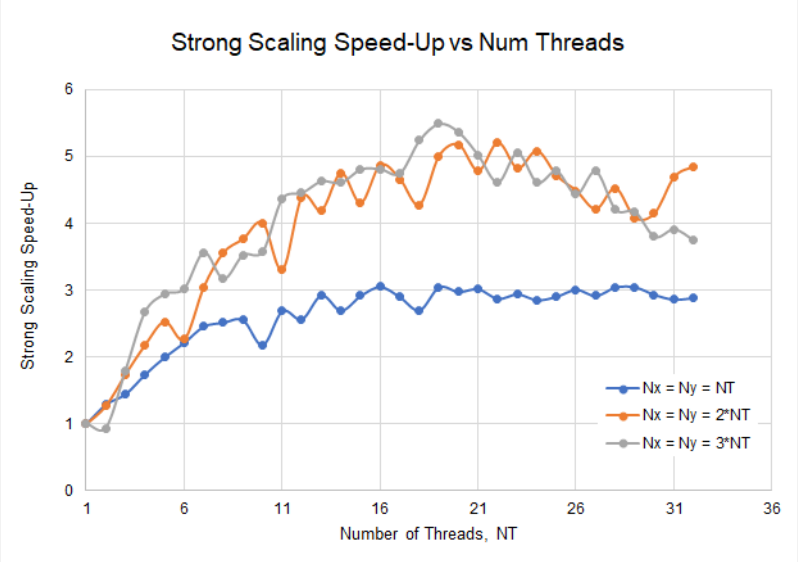


Figure 1. Parallel speed-up for the solutions to Problem 3 of HW2 for various number of threads and blocks.

These timings were run on a laptop with 4 cores and maximum of 8 threads. Note the number of threads reported in the laptop's specs refers to the number of threads that can be active at once/the computer can actually actuate. A program can ask for more threads (and thus oversubscribe the systems resources) but no more than 8 can actually be active at once.

(a) *(5pts)* Recall the formula for strong scaling speed-up can be expressed as

$$S(N_T) = \frac{1}{(1-p) + \frac{p}{N_T}}$$

where $p$ is the fraction of the program that is parallelizable. For the case where $N_x = N_y = N_T$ (the blue data), what is an approximate value of $p$ based on the data?

(b) *(5pt)* Using only the data for $N_T < 24$, what is an approximate value of $p$ for the $N_x = N_y = 2N_T$ case?

(c) *(5pt)* When the program uses more than approximately 24 threads (keeping in mind the computer can only actuate 8) the performance of the setups where $Nx, Ny > N_T$ degrades while the case where $N_x = N_y = N_T$ remains approximately asymptotic. Why might this be the case?

*Hint:* Consider the number of blocks in the fully spun up region.

(d) *(5pts)* The laptop the timings came from can only actuate 8 threads at once. As a result, some of the runs are in a sense equivalent:

- $N_T = 8, N_x = N + y = 16$ vs $N_T = N_x = N_y = 16$
- $N_T = 8, N_x = N + y = 24$ vs $N_T = N_x = N_y = 24$.

In both cases the $N_T = 8$ scenario was faster. Assuming that relationship was not due to timing error, what may be causing the time difference?

(e) *(5pts)* The parallel fraction $p$ is a parameter of both the problem and program. The timings we used to derive it were for the function `wavefront520` only, not the entire the program. All of the interior of `wavefront520` was in a parallel region. Why isn't $p = 1$? What could you change about the problem to change $p$?

*Hint:* There are things every thread does, like declare variables, find its thread ID, etc. This work has to be done by every thread, so the time it takes to do it remains constant no matter how many threads are used.

(f) *(5pts)* How does the phenomena enforcing $p < 1$ in a fully parallel region affect a program's ability to achieve it theoretical strong scaling limit?

**Problem 2 Domain Decompositione** *(30 pts)* Suppose you are working on a $n_i \times n_j$ FD problem that **requires a halo exchange**. You would like to parallelize this problem using MPI on a computing cluster. Suppose you are working on a computing node with three processors with the connections and connection speeds given in Figure 2.

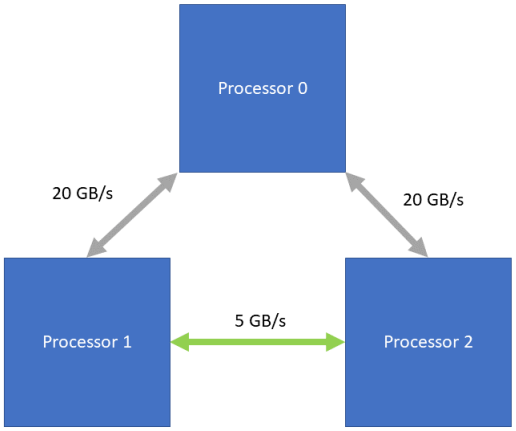


Figure 2. Example processor cluster for use in Problem 2.

(a) *(5pts)* Suppose the FD problem is the time-dependent (spatially) centered-ref problem shown in class (2D advection) where nodes can be updated independently of one another. How many blocks ($N_i \times N_j$) should you split the domain into (provide a picture) and why? Assume communication is extremely expensive.

(b) *(5pts)* Using your answer to part (2a), what processor would you assign to each block and why? Assume explicit (Dirichlet) boundary conditions are given on all edges of the domain.

(c) *(5pts)* What information would need to be sent and received in the halo exchange for part (2a)? Assume periodic boundary conditions (periodic boundary conditions yield a halo exchange like what we did in HW4, where you wrap around to find your neighbor so long as $N_i, N_j > 1$).

(d) *(5pts)* Suppose the FD problem is the backwards-ref problem from HW2 where nodes need to be updated in order using a wavefront approach. Now how many blocks ($N_i \times N_j$) would you split the domain into (provide a picture)? Assume communication is extremely expensive (note that synchronization in MPI is a form of communication).

(e) *(5pts)* Using your answer to part (2d), what block would you assign to each processor and why?

(f) *(5pts)* What information would need to be sent and received in the halo exchange for part (2d)? Assume boundary conditions are given for nodes with $i = 0$ and $j = 0$.