Homework 6: Report

Dante Buhl

May 30, 2024

Question 1: Latency in MPI Communication

After modifying the ping pong code from Hw4 and running message size tests for double precision real arrays, several plots are produced; one for my personal computer, one for hummingbird, and one for lux. The message sizes ranged from 1 real to 10000 reals, and each message size was repeated 200 times in order to find a significant average for the startup time and the time per byte. For each machine used, there is a slightly different plot found, however, the general trend is regular. There is some initial slope which is fairly linear, for messages below 500-1000 reals (varies for machine), and then after that the latency experiences a jump discontinuity or slope change and then stays on this slope for the rest of the data sizes with very minor deviations. Figures at the end.

Question 2: Finite Difference Stencil

For this problem, the only part that changes is that we decompose the domain in to pencils instead of planes. This changes the communication structure but doesn't force us to change the computation time. We see that instead of communicating NN_z cells in 2 directions we communicate N_z cells in 4 directions. This means that the communication startup will impact the execution time more than before, but it also allows the isoefficiency to be $O(P)$.

2.1 Execution Time

$$T_{2D\_FD} = \frac{T_{comm} + T_{comp}}{P}$$

$$T_{comp} = t_c N^2 N_z, \quad T_{comm} = 4P(t_s + 2t_w N_z)$$

$$T_{2D\_FD} = \frac{t_c N^2 N_z}{P} + 4t_s + 8t_w N_z$$

2.2 Efficiency

$$E = \frac{T_1}{PT_{2D\_FD}} = \frac{t_c N^2 N_z}{t_c N^2 N_z + 4Pt_s + 8t_w N_z P}$$

2.3 Isoefficiency

$$E = C \implies t_c N^2 N_z = E(t_c N^2 N_z + 4Pt_s + 8t_w N_z P)$$

$$N^2 \sim P \implies t_c N_z = E(t_c N_z + 4t_s + 8t_w N_z)$$

$$\text{Isoefficiency} \sim O(P)$$

Obviously this isoefficiency is better than the one presented in the lecture note. We see that though we my communicate more, we see that we only have to scale the domain by the square root of the number of processors implying better isoefficiency.

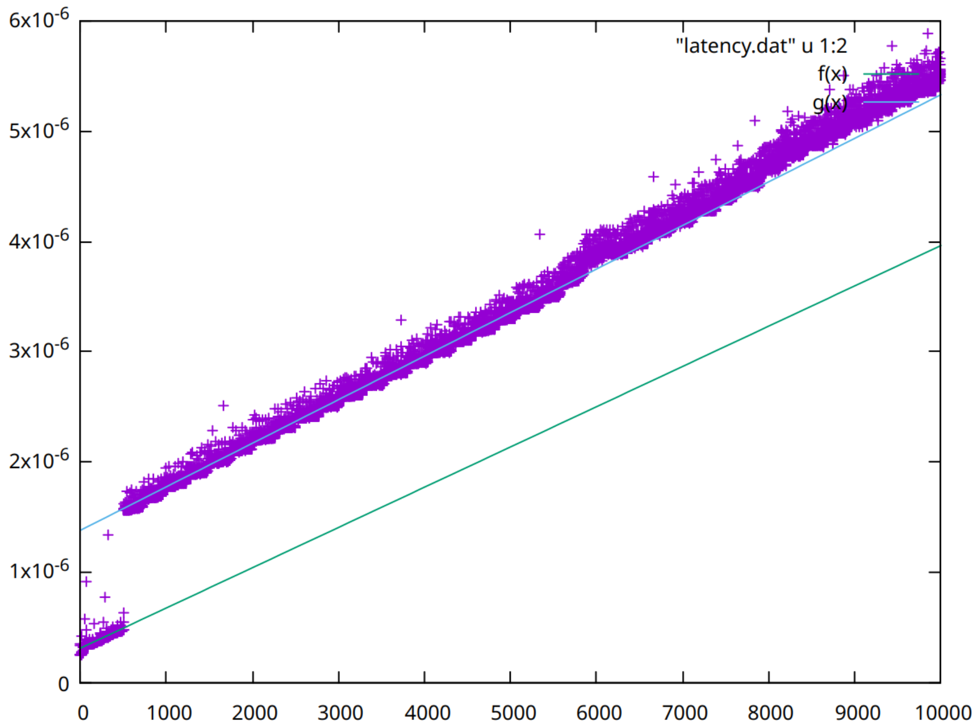

Figure 1: Latency Plot on my desktop, $t_s = 3.17 \cdot 10^{-7}$, $t_w = 3.64 \cdot 10^{-10}$

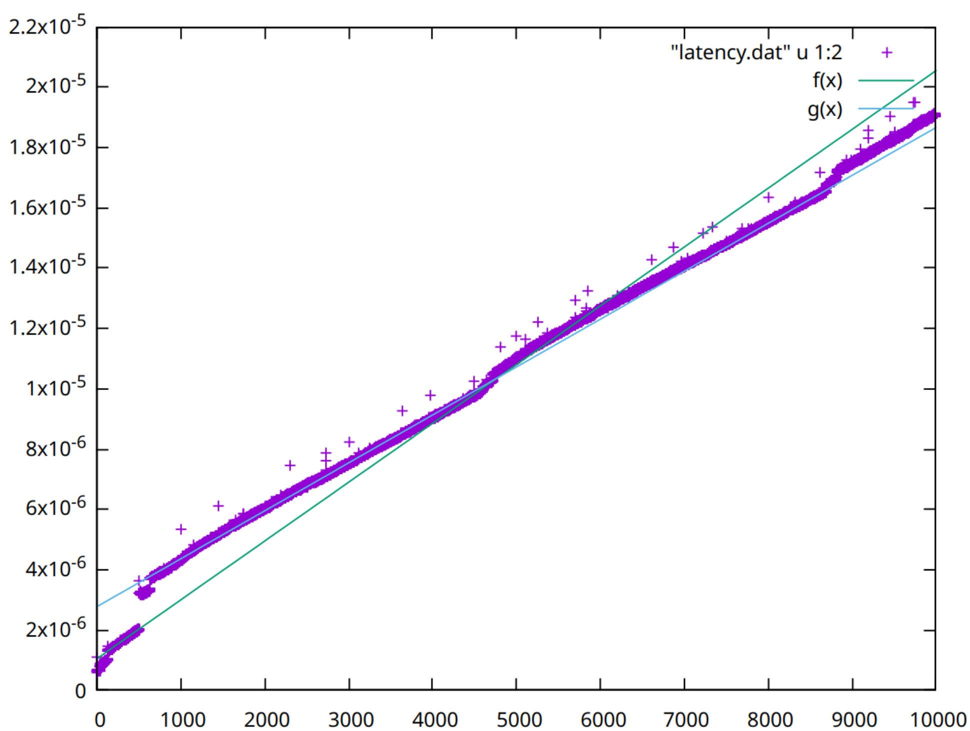

Figure 2: Latency Plot on Hummingbird, $t_s = 1.05 \cdot 10^{-6}$, $t_w = 1.58826 \cdot 10^{-9}$

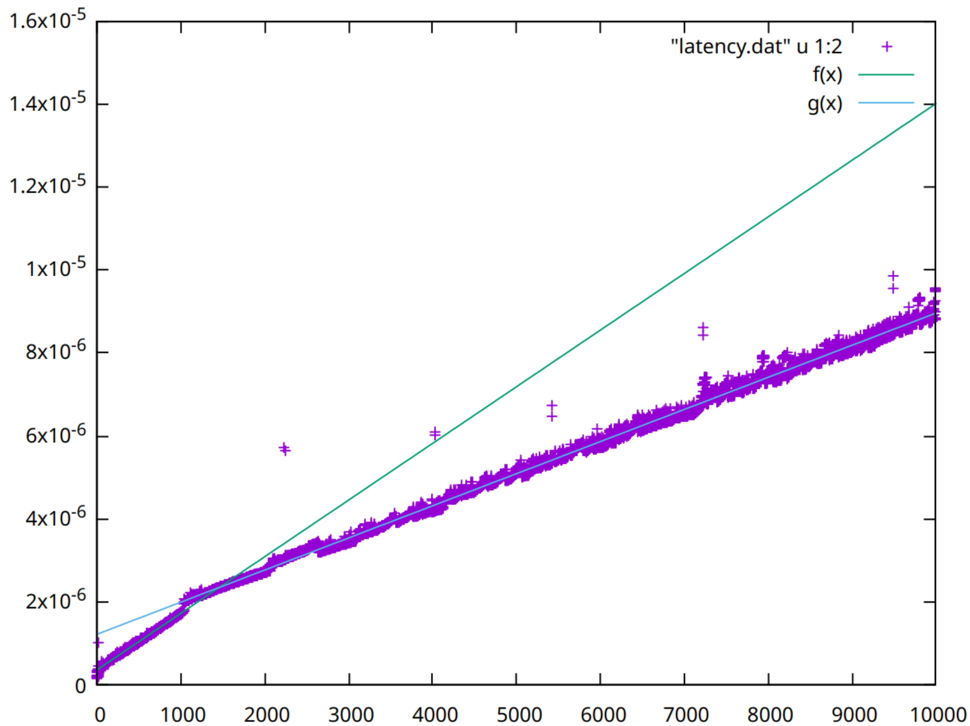

Figure 3: Latency Plot on Lux, $t_s = 3.70 \cdot 10^{-7}$, $t_w = 7.73 \cdot 10^{-10}$