Alperen Degirmenci
CS 205 - HW 3
10/26/12

Problem 1

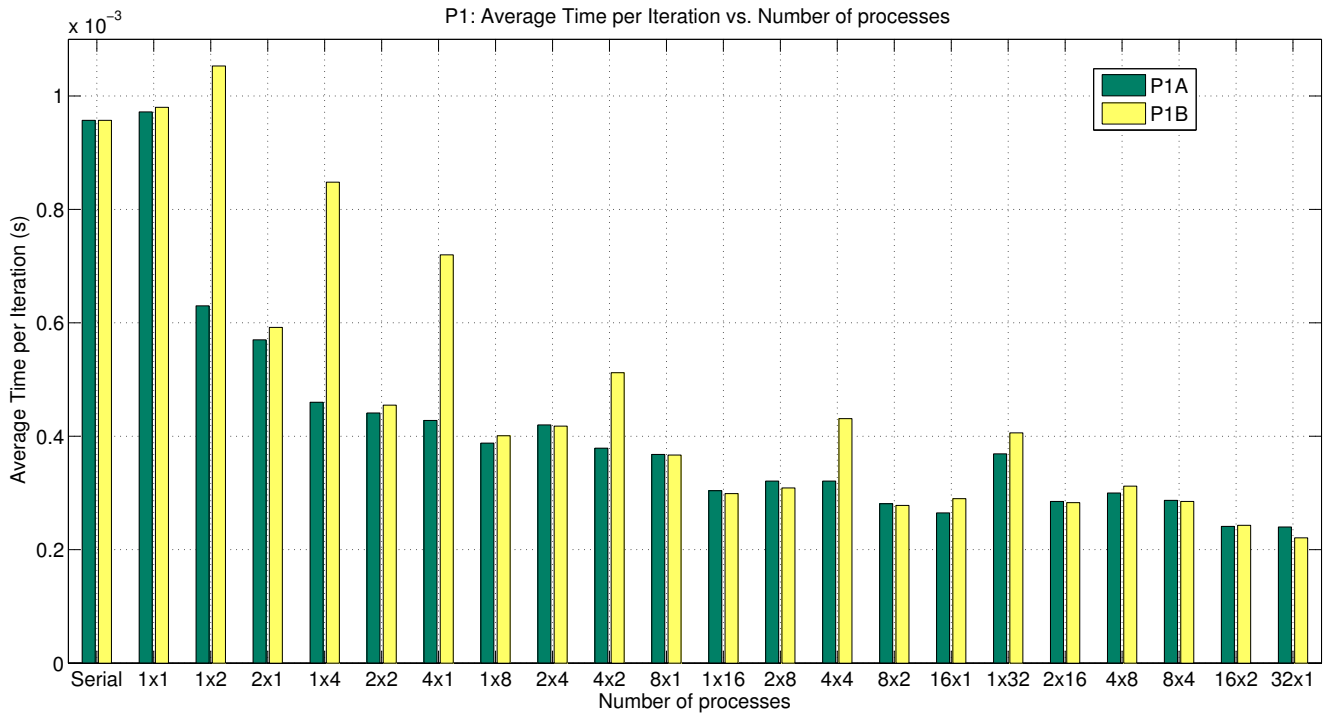

Figure 1: Average Time per Iteration of the Domain Decomposition Algorithm using implementation P1A and P1B.

In Figure 1a we see that the time it takes to run one iteration of the domain decompoisiton algorithm decreases as we increase the number of processes. This speedup is more pronounced in some cases than others. P1A seems to perform better than P2A when $P_x = 1$. This difference in performance becomes less prononuced as we increase the number of processes, since Sendrecv slows down the computation due to its blocking nature whereas Isend/Irecv lets the process continue with its computation before communications are done (I probably haven't fully exploited this useful property; I could have applied the stencil to an inner block (where we already have the data from the previous iteration) while waiting for the ghost cells to update).

Speedup diminishes as we increase the number of processes, despite the fact that the load per process is decreasing. This suggests that communications is slowing down the computation. This also shows that the problem doesn't exhibit weak scaling, since otherwise, as we decreased the number of computations per process, speedup would have kept increasing instead of diminishing. The cost of communication between processes is going to overwhelm the entire computation, and assuming we kept the load per process constant, we would see a decrease in efficiency due to the increased communication time.

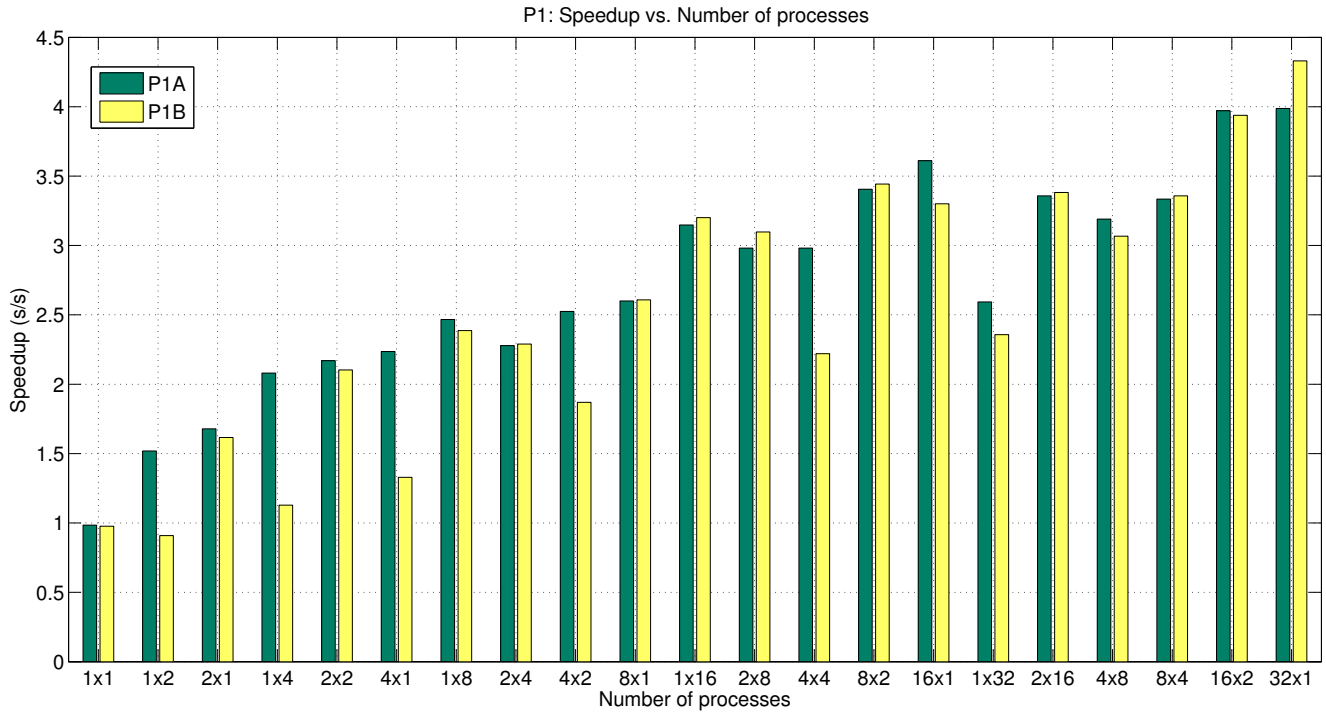

Figure 2: Speedup for P1A and P1B.

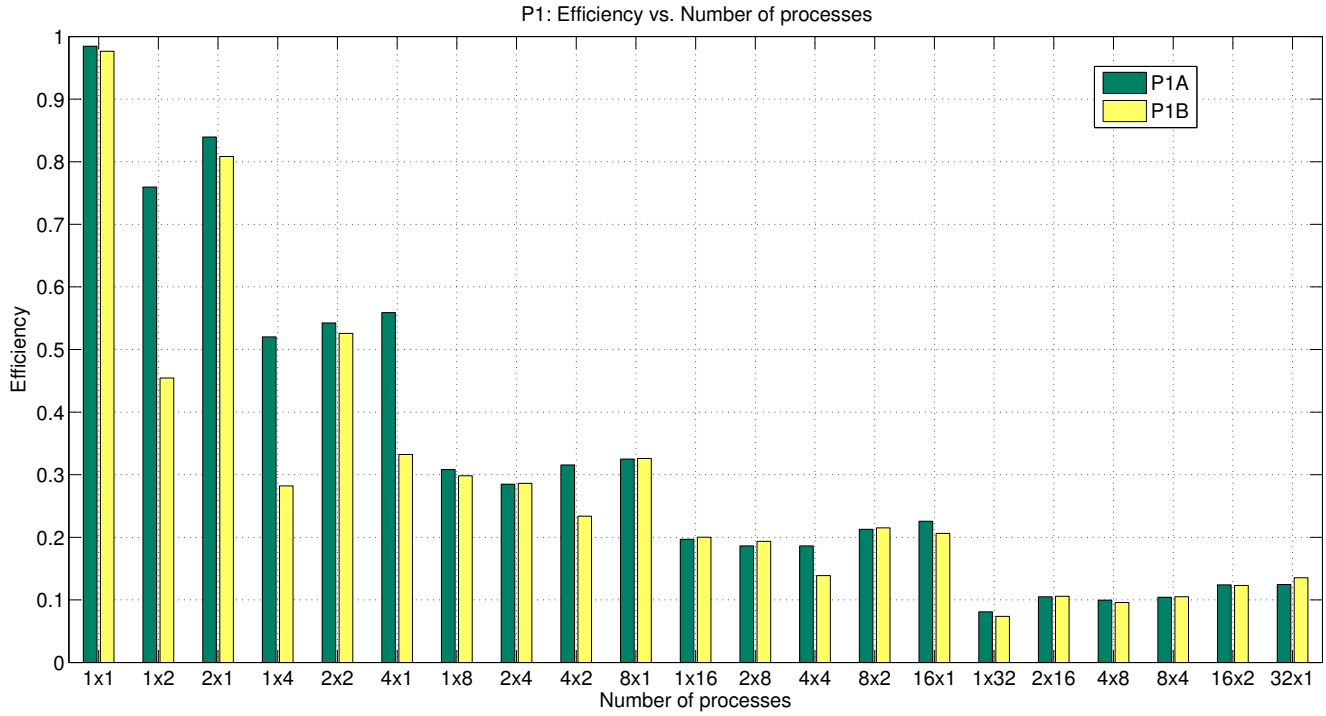

Figure 3: Efficiency of P1A and P1B.