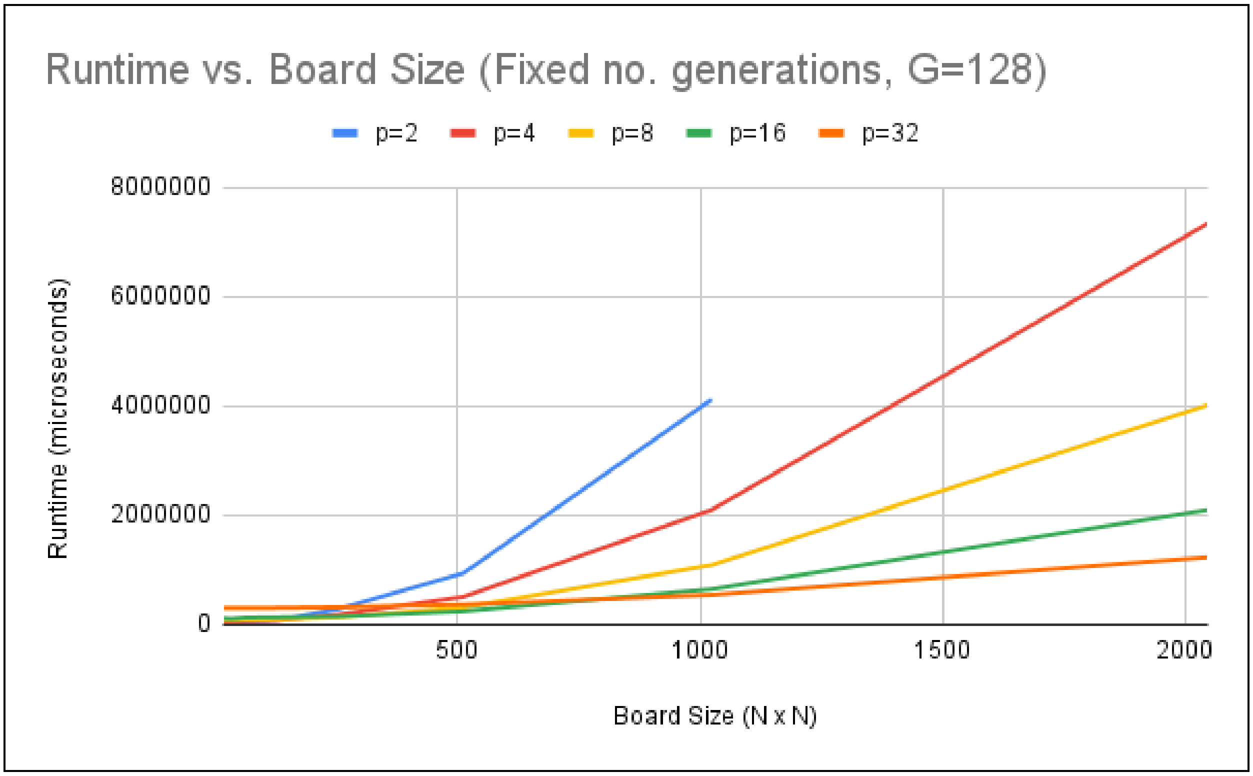
Ethan Vincent

CptS 411 HW 3 - Game of Life

October 13th, 2022

Runtime curves (fixed number of generations, G=128)

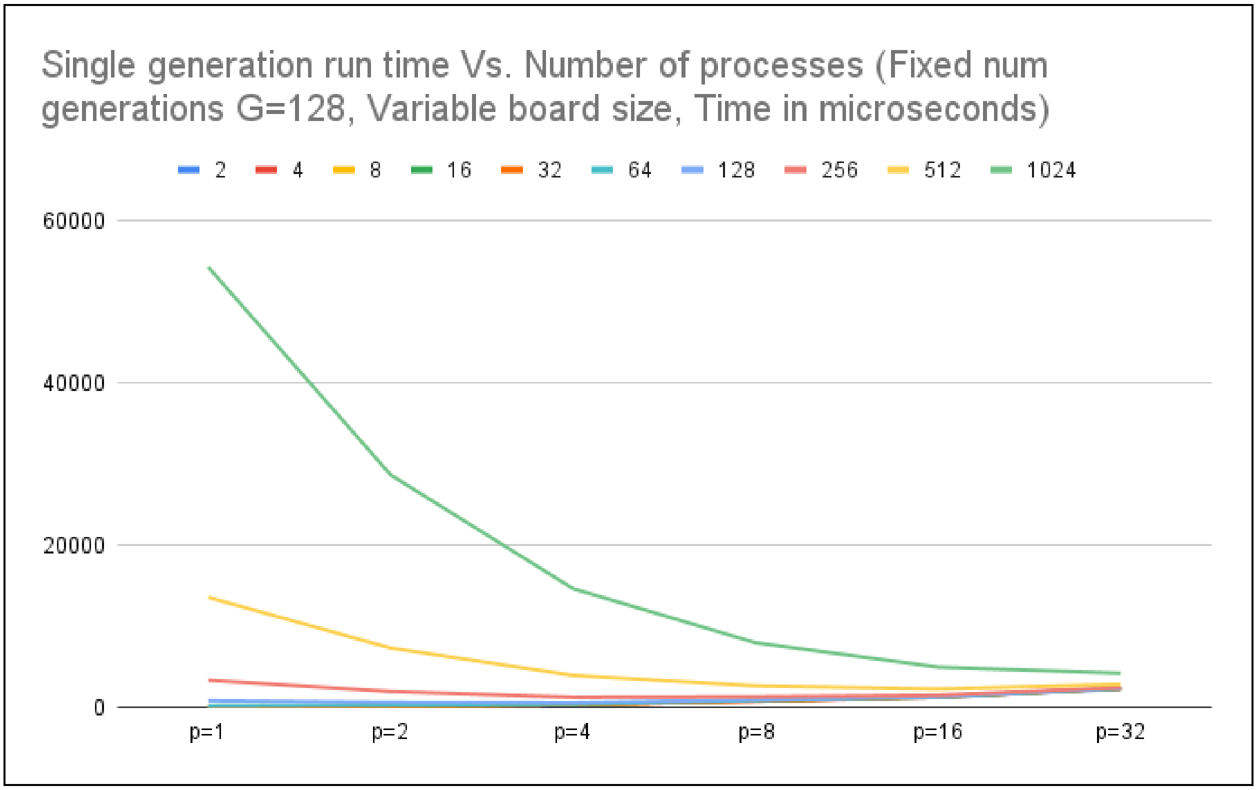


g1 *Because my program used static memory allocation, the stack could not afford to create the requisite arrays in the case of p=2,N=2048*

avg single gen time	2	4	8	16	32	64	128	256	512	1024
p=1	0.583	1.454	3.972	13.942	53.536	212.198	848.056	3370.478	13590.774	54320.311
p=2	220.302	251.537	253.925	236.292	210.742	329.552	641.467	1984.99	7346.89	28684.5

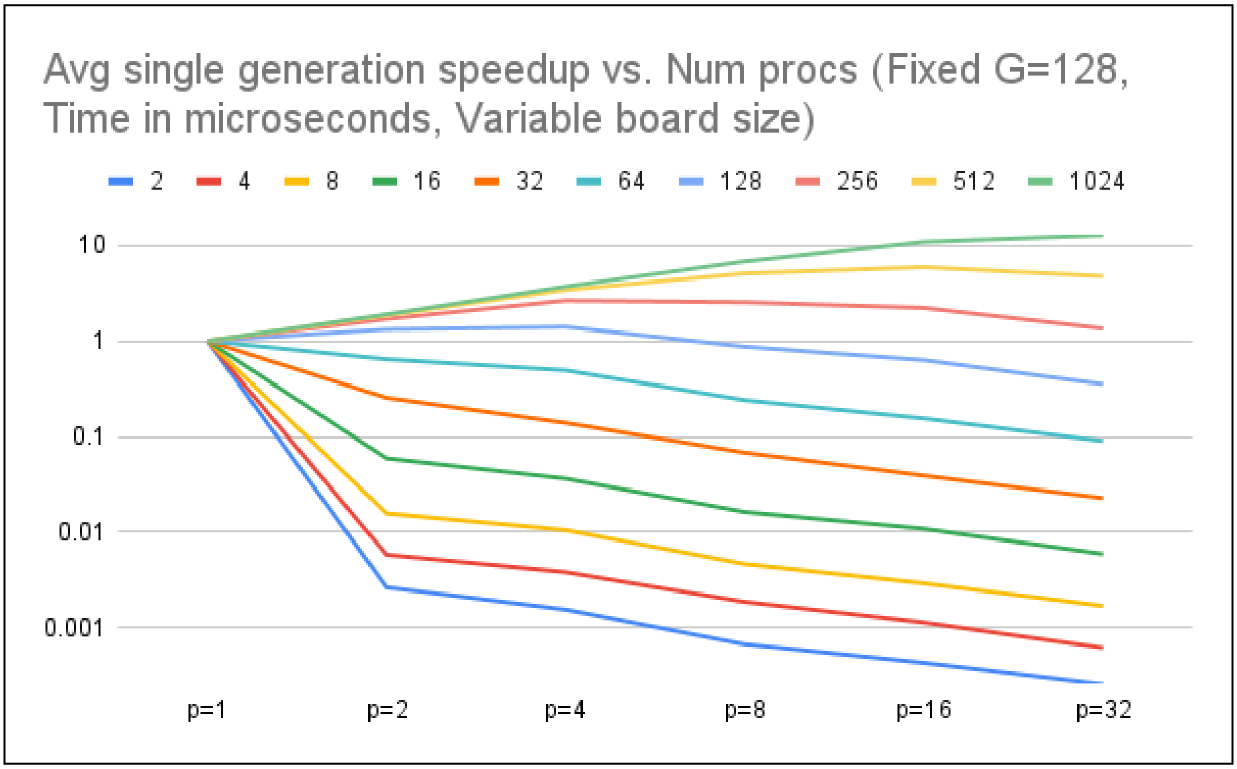
								4	2	6
p=4	378.248	381.583	378.857	384.257	386.114	431.483	603.952	1275.977	3974.847	14656.762
p=8	868.386	781.842	859.082	861.08	790.594	883.93	972.465	1330.574	2674.446	7997.938
p=16	1363.673	1285.395	1361.551	1288.378	1373.652	1373.78	1345.756	1521.525	2300.449	4993.202
p=32	2278.997	2356.752	2355.158	2367.685	2365.672	2369.469	2400.353	2493.295	2862.73	4243.325

Runtime curves of avg single generation vs. number of procs - fixed number of generations (G=128), variable board size



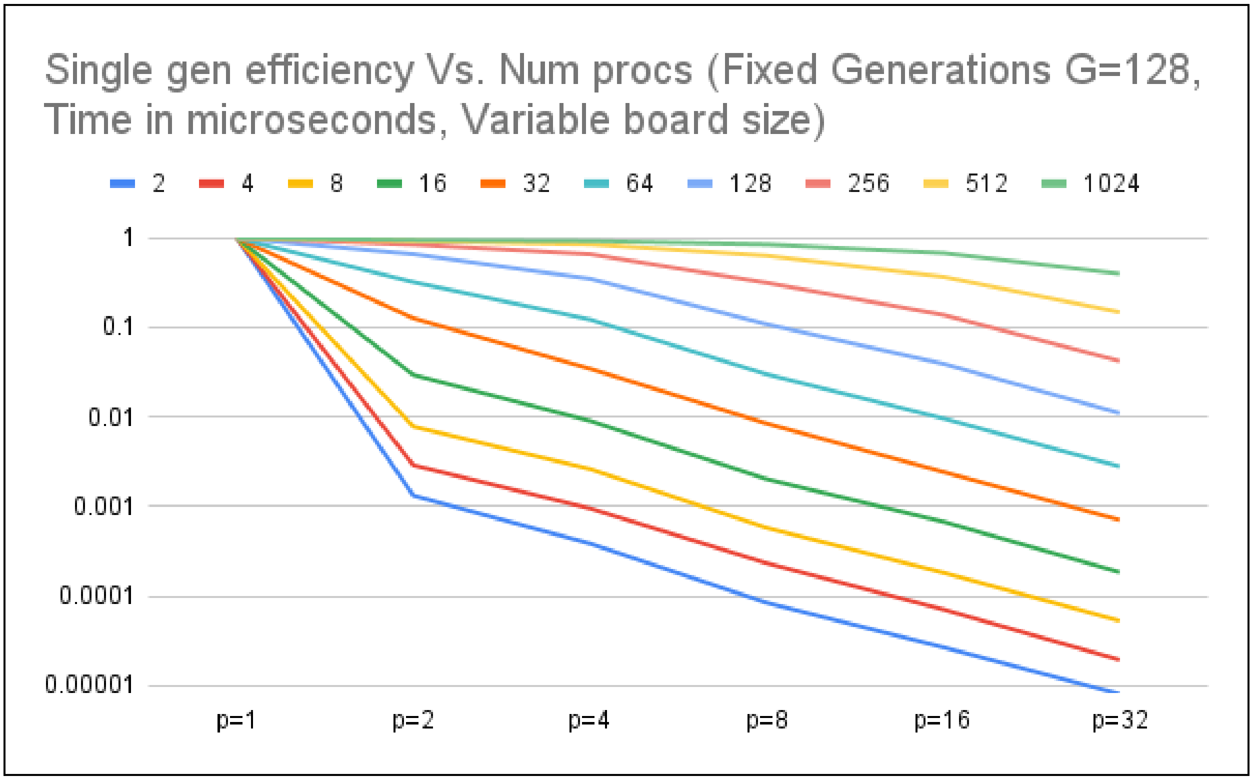
g2

Speedup curves avg single generation vs. number of procs - fixed number of generations (G=128), variable board size



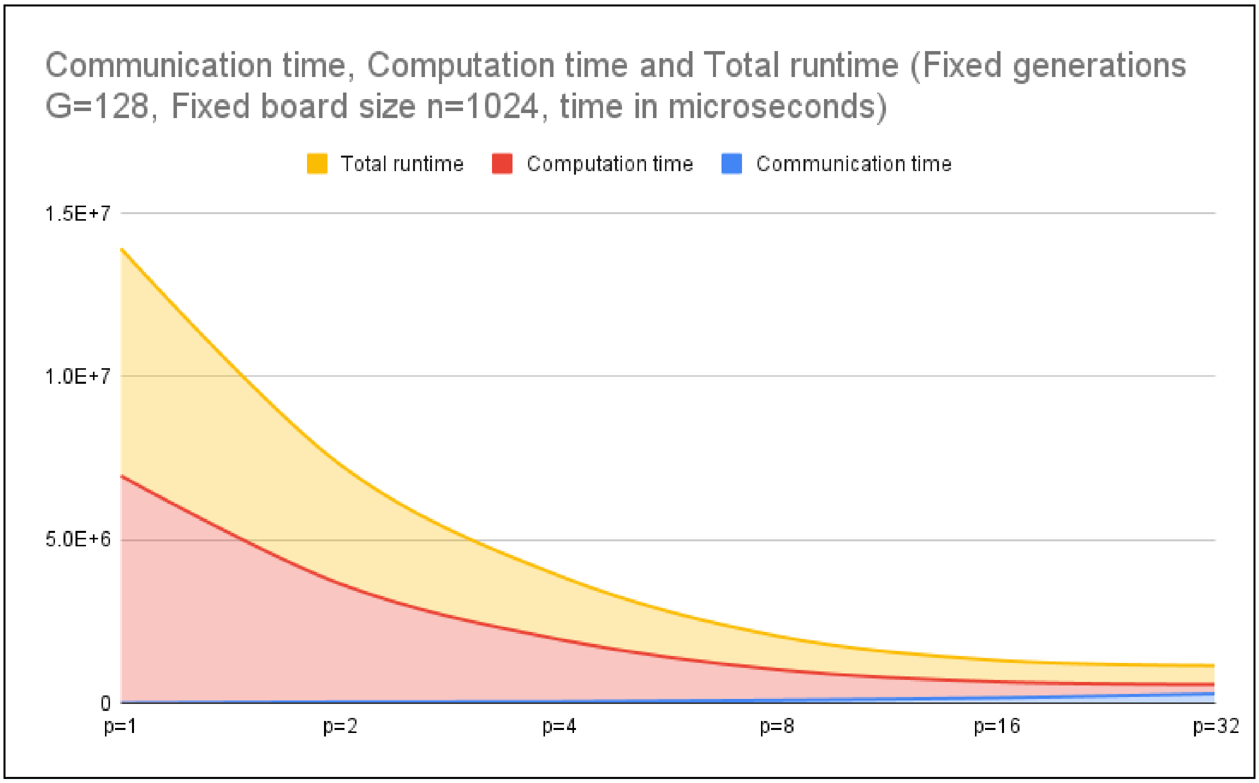
g3

Efficiency curves avg single generation vs. number of procs - fixed number of generations (G=128), variable board size



g4

Computation time and Communication contribution to total runtime (Fixed generations G=128, Fixed board size n=1024, Time in microseconds)



g5

Observations

In my opinion, my program did exhibit good-scaling behavior, I simply was not able to test it due to my static memory implementation. As the board size began to get large (≥ 1024), the speedup and efficiency of the program began to increase. I had quite a lot of communication overhead in my program, which I believe to be the cause for this initial, poor scaling. Sending 2 arrays of size n for each process at every iteration of the main loop is a poor use of resources when n is small and p is large. In the total runtime composition, we can see that at this board size and number of generations, communication time has not yet started to dominate the computation time. This is further evidence for my guess that increasing n would still yield better performance past the mark I measured.

In general, creating the arrays dynamically would have yielded better performance than using the stack. It would have made it possible to increase the board size further. I believe my program would exhibit more standard scaling metrics if more extensively measured.