Next, we examine the larger case of 1024 by 1024, still with 32 squares per agent. Now we see a much larger benefit from parallelization. Runtime decreased from 23.1 seconds to a minimum of 8.91 seconds with 16 threads. Interestingly, we still see an uptick in runtime when attempting to use 64 threads.

The largest board size we examined was 4096 by 4096. With 32 squares per agent, in serial this is vastly more difficult than either previous run, with a runtime of over 8 minutes compared to the 23 second runtime of 1024 by 1024. Parallelization achieved its largest benefit here, decreasing runtime to a minimum of 161 seconds with 16 threads. Again, however, there is a slight increase in runtime when running with 64 threads.

The next test case we examined was decreasing the density to only 64 squares per agent. We compared the case of a board size of 1024 x 1024. The results matched our hypothesis, as serial ran in about a quarter of the time. The minimum time was 2.9 seconds in this case with again 16 threads, as opposed to 8.9 seconds with a denser board.

Similarly, we again further decreased density to 128 squares per agent. This time, however, we compared the case of a board size of 4096 by 4096. Run time again was vastly decreased, this time by a factor of six. Run time decreased from 500 seconds to only 81.7 seconds in serial. With threads, it decreased to even 23.0 seconds with 16 threads.

Our results tended to follow a trend of decreasing runtime towards 16 threads, but then a slight increase with 64 threads. Sometimes this increase was more apparent, especially in the smaller board cases. In general, parallel efficiency was at its worst with 64 threads. This is likely due to the collisions that can occur during movement. More threads means more agents of different threads interacting with each other, which can cause “overlap” of movement. This results in the movement method being repeated multiple times, and an increase in runtime. The larger the board, the less likely this is to happen so there is less of an increase with 64 threads.

The larger difference was actually found in density. Since our method of implementing parallelization mostly was done by grouping agents to each thread, such as in movement and updating, an increase in the number of agents vastly increased runtime. This also has the effect of increasing the number of collisions, again increasing runtime.