ECE454 Lab 5 Report

Team members:
Yeqi Shi #1000274277
Yi Fan Shao #1000084151

Average runtime of unoptimized version over 5 iterations: 114.42 seconds
Average runtime of optimized version over 5 iterations: 3.76 seconds
Speed up factor: 114.42/3.76 = 30

We optimized the program by utilizing all 8 cores on the ug computers. The board is divided into as many pieces as there are threads (8 in this case) and a worker function is issued to each thread to process the piece of board issued to them. We also changed the cell encoding by utilizing 5 out of 8 bits of a char variable instead of 1 out of 8 bits. The first 4 bits are used to record the number of live neighbours since there can be a maximum of 8 neighbours. The 5th bit is used to store the live/dead status of the current cell. By storing neighbour infos in the cell itself, we can avoid scanning every neighbour to determine the cell's status.

Since cells near the boundary of assigned board pieces can have neighbours across the boundary, worker threads can lock (or attempt to lock) the boundary when they are near the two columns near the boundary, thus preventing race conditions writing to the output board. We got our program runtime down to 10 seconds with this method while ensuring correct output. To further reduce the runtime, locking and unlocking functions are only put around codes that actually write to the outboard, and all conditional checks are outside the lock. This can be done because the thread checks the condition from the input board, which will be read-only, and writes to the output board. By shortening the critical section we achieved an additional 3x speedup of the program.

While we could have further improved the program via loop unrolling, constant propagation, or loop invariant code motion, we have decided not to as it will drastically decrease the readability of the code while offering very little performance improvements in return.