Project 6 – Game of Life MPI Implementation

Due: 12/9/2019 11:59:59 pm

Goal. In this project, you will be parallelizing Conway's Game of Life in C using the OpenMPI library.

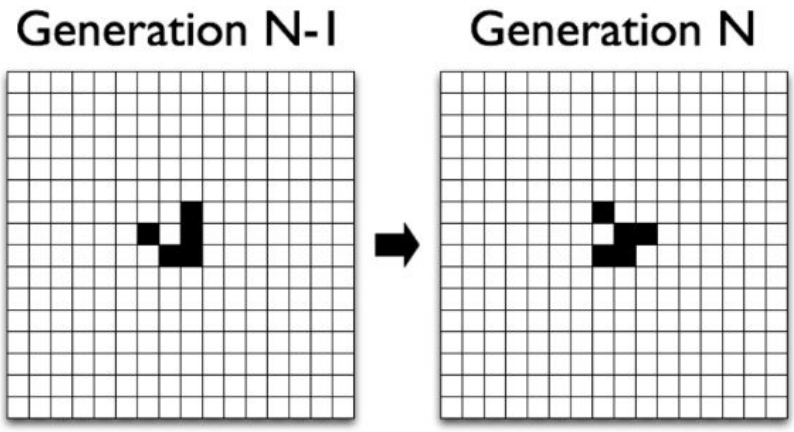
Getting started. We have provided you with many files to the program, which we will describe below:

- A serial (sequential) version of the program called `src/life_seq.c`. You should take a look at how this program implements the game. Also, the serial version will be used to generate the canonical output for the program. You should feed the input files into this program in order to see what the correct output is.
- The starter code for the program called `src/life_mpi.c` **This is the only file you will be modifying in this project.**
- A sample MPI program that shows how message passing works called `src/sample_mpi.c`
- `run/install_mpi.sh` which will install MPI on your virtual machine. **Make sure you are using a Mac or the course VM from project 5. DO NOT use Windows.** If you are using a Mac with homebrew, you can run `brew install open-mpi`
- In order to compile all of the examples and the `src/life_mpi.c` programs, you should go to the `run/` directory. You can either use the `Makefile` or the `run_*.sh` scripts to build and run your programs.
- Visualization and random input generation scripts. These are called `tools/life_visualizer.py` and `tools/input_generator.py` respectively. There is an attached `tools/TOOLS_README.txt` document with these two files which explains how they function.
- For more information, please see the `README` file under the project directory.

How the game works. The Game of Life consists of an N by M matrix, where each cell can either be 0 (dead) or 1 (alive). The game will start with the cells corresponding to the (x, y) coordinates in the initial input file being set to 1 (alive). This is the "first generation" of the game. The game consists of constructing successive generations of the board based on certain rules which are described below:

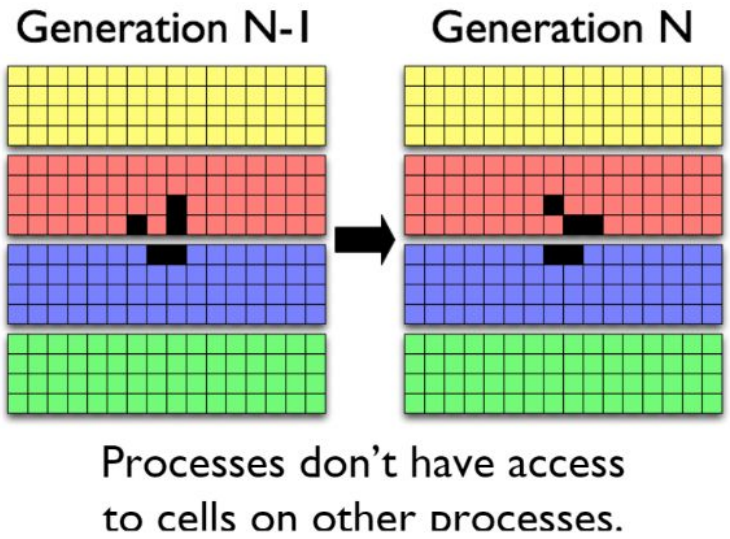
1. Cells with 0,1,4,5,6,7, or 8 neighbors die (0,1 of loneliness and 4-8 of overpopulation)
2. Cells with 2 or 3 neighbors survive to the next generation
3. An unoccupied cell with 3 neighbors becomes occupied in the next generation.

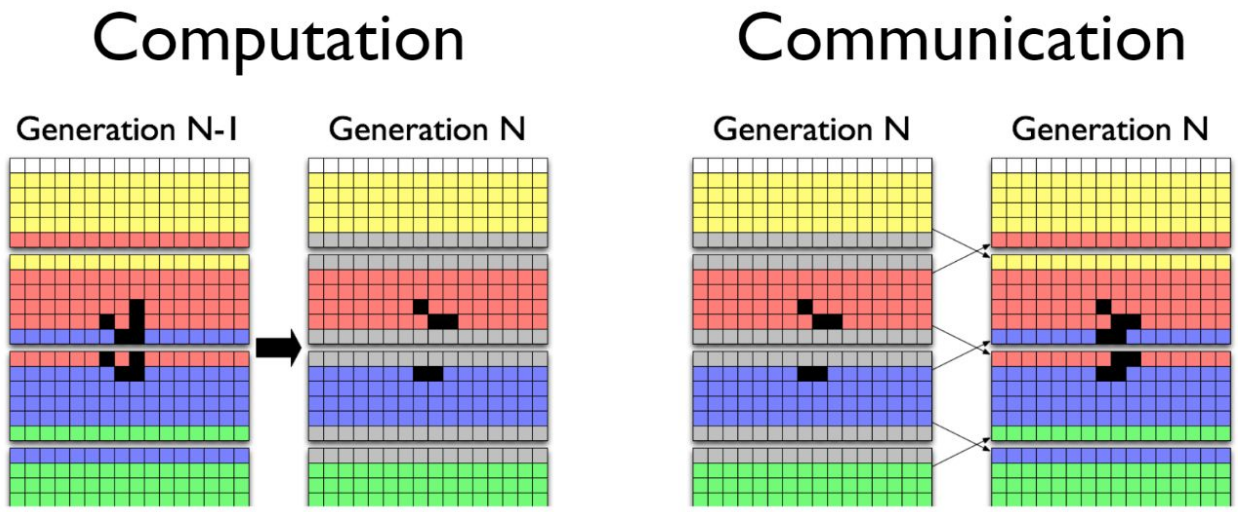
Here is an example graphic of one iteration of the game:



This is how the game will function. For more information on the Game of Life, check out the link at the beginning of the file.

Implementing the game using MPI. We have provided some starter code, which includes all of the initialization of the MPI functionality for the program, as well as the code for the rank 0 process. You will need to implement the code for the rank 1 to n processes (worker processes). The images below briefly illustrate how the MPI-parallelized game works.





Note. If you wish to implement the entire MPI-parallelized game of life instead of using the provided code, please feel free to do it in `life_mpi.c`.

Running the serial and MPI programs. First, go to the `run/` directory. Then, you can use either the `Makefile` or the `run_*.sh` files to run the programs.

Testing with the sample input/output. You can use the 4 sets of sample input/output files located in the `run/` directory to test your program.

Testing with the input generator tool. First, you can create random inputs for the program using the `input_generator.py` script. Next, you should run these input files either through the visualizer or the serial version in order to get the correct output for the final generation that you specified. Finally, run the MPI program with the input file and get the output from that.

Comparing the output files. We recommend using the `diff` command (or https://www.diffchecker.com/ if you prefer a graphical display) to verify that the output from your MPI program is the same as the output from the serial version.

Submission. Submit the `life_mpi.c` file to the submit server. It will be graded offline. And, since you can easily test your program locally, all tests on our side will be secret.