CSC-213: PROF. CURTSINGER
FINAL PROJECT PROPOSAL

ANNA BLINDERMAN, DAVID KRAEMER, ZACHARY SEGALL

Purpose and Impact

We will implement a variation of Conway's Game of Life – a cellular automaton simulation created by the mathematician John Conway. Life is simply a series of iterations of a rendering of a grid of cells, all of which are either dead or alive. Each cell's state is determined by the states of its direct neighbors.

In general, Life provides a basis for a visual understanding of certain evolutionary processes. Further, the GPU parallelization present in our implementation will make clear the importance of optimization as it relates to the speed of the simulation, and thus the role of well-written code in fields such as scientific computing.

Course Topics Spanned

The variety of simultaneous events that comprise Life call for virtualization of the CPU, while concurrency on the GPU will be necessary in order to carry out the constant intensive calculations needed to run the simulation. Thus, our implementation of the Game of Life will build off of our scheduler from the Worm lab and our stencil-and-update patterns from the Cartoonify lab.

Implementation – Major Components

Our scheduler will need to handle the order and timing of the execution of various events integral to our simulation. For some examples, we must update the cells on the board in order to advance the simulation to the next iteration at regular intervals, add cells to the board in locations specified by user input, and pause, restart, or quit the simulation whenever the user inputs the appropriate keystroke. We will model our scheduler off of the one implemented by Zachary and Mattori for the Worm lab.

The algorithm used to advance a Life simulation from one iteration to the next naturally lends itself to an "embarrassingly parallel" implementation. Specifically, we will use a stencil pattern since a cell's next state depends on the states of its eight immediate neighbors and we can process whether or not a cell survives independent of this determination for any other cell on this board.

Regarding the more general mechanics of our project, we will use an `SDL_surface` to create a board as in the Galaxy lab. The cells on the board will be initially populated from either a random starting configuration, one of some set of preset starting configurations, or by the user clicking squares on the board. We will have a struct to represent the state of the board and will use a bitmap for the visualization of the board. Because our internal representation will be separate from our visualization, we will need to translate from the internal representation (the struct) to the visualization (the bitmap) at each interval of time. Further, this separation will allow us to more easily manipulate and update the representation and then simply update the visualization afterwards using Conway's original algorithm.

Implementation – Timeline

Date	Objective
Monday, April 24th	have a functioning graphics system and ability to take user input
Friday, April 28th	create a set of initial scenarios and capacity for randomization
Monday, May 1st	implement Conway's update pattern on GPU
Friday, May 5th	tests and evaluation
Monday, May 8th	extra features if done, otherwise `malloc`'d time for unforeseen issues

In the case that we finish early, we will try to implement some subset of the following:

- gradations of color indicating how "old" a cell is
- variations of the topology of the board (eg. a torus)
- ability to add certain shapes (eg. "gliders") as opposed to just single cells
- genetic algorithms
- ability to read in any initial board configuration from a file

Implementation – Potential Challenges

A challenge will likely arise from our attempt to write a lower-dimension game board on top of a high-dimension pixel grid, as doing so will introduce a large class of edge cases to consider.

It's also possible that we will struggle with processing user input by means of the mouse, as none of us have worked directly with the `SDL` API in the past. In this case, we will allow for a larger set of keyboard commands – some of which will require the GPU, and all of which will still need to be scheduled.

Citations

https://en.wikipedia.org/wiki/Conway%27s_Game_of_Life