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 grid of cells, all of which are dead or alive, and whose state is determined by state of its direct neighbors. Players of Life can gain a stronger visual understanding of evolutionary processes as well as [idk something annoying about what feels like chaos in a perfectly deterministic system? not sure what the "purpose" here is]

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 whenever the user clicks the board, and the pausing, restarting, and quitting of the simulation whenever the user inputs the corresponding keystroke [wording is not a thing yet].

The algorithm used to advance a Life simulation from one iteration to the next naturally lends itself to an embarrassingly parallel algorithm. 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 in this board.

Regarding the more general mechanics of our project, we will use an `SDL_surface` to create a board, which will be populated from a random starting configuration, one of some set of preset starting configurations, or by the user clicking squares on the board. The simulation's updates will run as in Conway's original description, but we will also allow the user to interact with the simulation by toggling cells on which they click as well as by pausing, restarting, or ending the simulation by means of certain keystrokes.

Implementation – Timeline

Implementation – Potential Challenges

It's possible that we may struggle with processing user input by means of the mouse, as none of us have worked directly with this [which one? "interface"?] in the past. In this case we will allow for a larger set of keyboard – some of which will require the GPU, and all of which will need to be scheduled.

Implementation – Potential Additions

- gradations of color indicating how "old" a cell is
- variations of the topology of the board (eg. on a torus)
- use ability to add certain shapes (eg. "gliders") as opposed to single cells
- genetic algorithms

Figures??

Citations?

Grading Criteria (for reference)

- Is the proposal written clearly using correct grammar and spelling?
- Does the proposal adequately describe the purpose of the proposed work?
- Does the proposed work span at least two key areas of this course?
- Is the proposed implementation plan sufficiently detailed?

Citations?