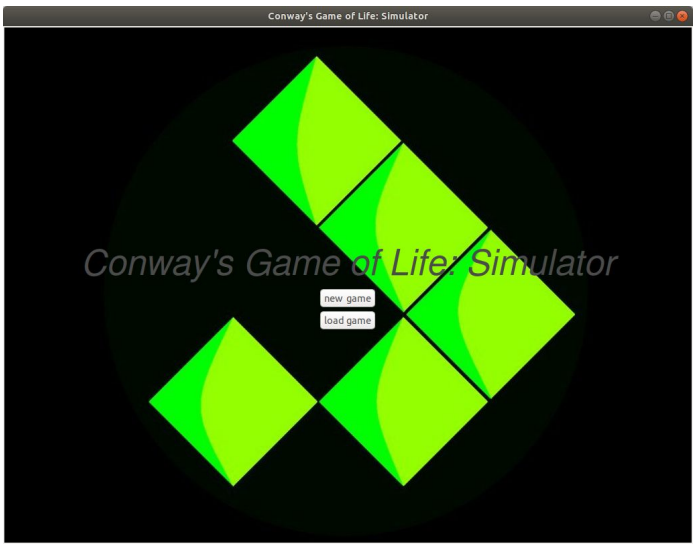
CS-154 PROJECT - REPORT

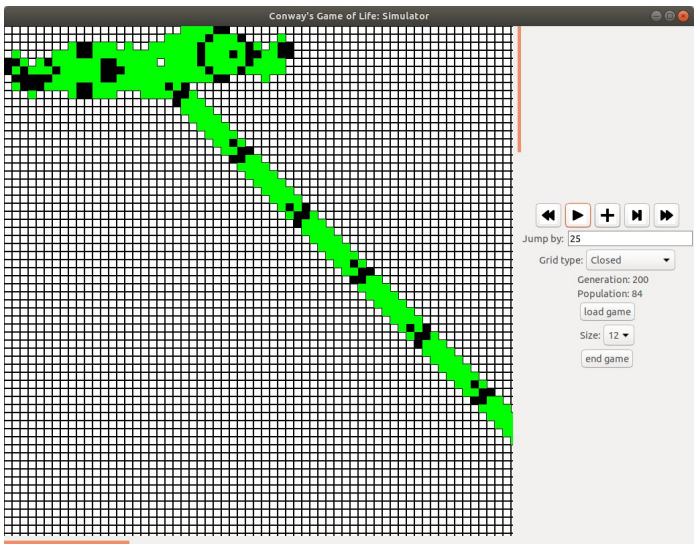
Conway’s Game of Life

R Nikhil Reddy
170050096
rnr1410@gmaiil.com

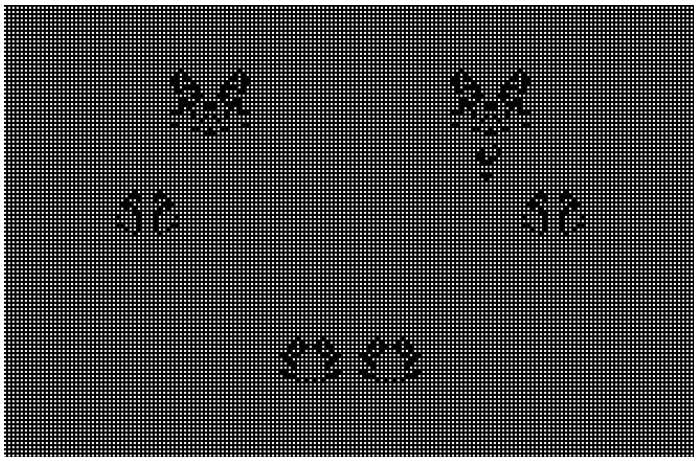
Poorvi R Hebbar
170050096
rpoorvihebbar@gmail.com



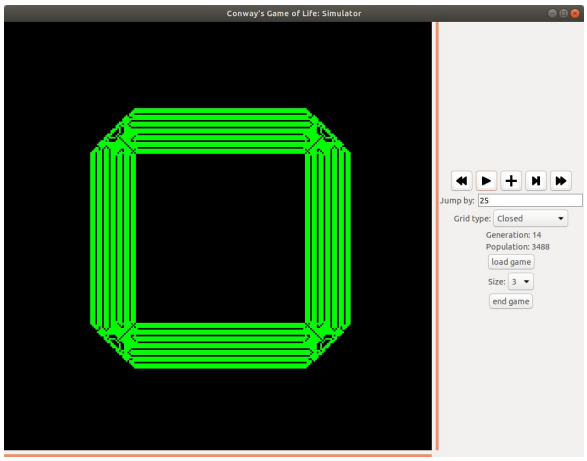
THE START FRAME



GOSPER-GLIDER GUN



PUFFER



SQUARE DESIGN

OBJECTIVE

- The Game of Life is a cellular automaton devised by the British mathematician John Horton Conway in 1970.
- The evolution of this game depends only on its initial state, Given any initial state of live and dead cells future state of the cells will be pre decided. thereby making it more of a simulation than a game.
- Using this concept, we made a simulator with various interesting examples and options.

RULES

A Cell with

- Less than 2 neighbours dies due to underpopulation
- More than 3 neighbours dies due to overpopulation
- Equal to 3 neighbours lives in the next generation
- Remains the same otherwise.

PROGRAM STRUCTURE

We just finally made a single rkt file “application.rkt” so that we could set the state variables in different functions and hence make the overall procedure faster.
The program can be divided into the following parts:

- The GUI part .
- The parsing part - containing code to convert RLE files into good grids.
- The conway processing part .

FEATURES

- Load game: Games can be loaded from a variety number of
- Control Panel: Controlling features like Play/Pause, Fast and Slow Forwarding, Next State and Jump by any number of generations.
- Grid type: Ability to change grid-type between Closed (Toroidal) and Bounded Open grids.
- Cell targeted toggle: Clicking on a cell toggles it’s state.
- Cell size: Cell size can be changed in the right panel,.
- Scrolling feautures

SAMPLE INPUTS AND OUTPUTS

- Selected a grid, we can see the simulation of the grid or in other words see how the grid changes with respect to time following the given rules.
- INPUT : file which contains the information (like the rle) of the corresponding grid.
- OUTPUT : The simulation or the changing grid with respect to time. (which is actually very interesting as we can even play with the frame rate and the state of the grid).

DISCUSSION

- Making functions like grid-apply which basically applies mapping on grid and using it to calculate the number of neighbours, which made our processing code run 10-20% faster than before.
- Using the concept of swapping parts of an image instead of drawing the image as a whole again for the canvas% class in simulator, which made rendering runtime faster than grid processing.
- Using the canvas' drawing context to efficiently draw images instead of the previous bitmap updating, thus decreasing render time.

DESIGN OF THE PROGRAM

The program begins with a start frame , in which the user is expected to select a folder and a grid that corresponds to the selected folder from the drop down menus that appear on clicking on the tabs on the frame.2. Once the grid is selected, the simulation starts when you press the play button, you can also change the grid type that is keep it bounded or unbounded.3. Also you can change the the simulation and also its rate by toggling cells, and by changing the rate at which the frame changes.

BASIC INTERFACE

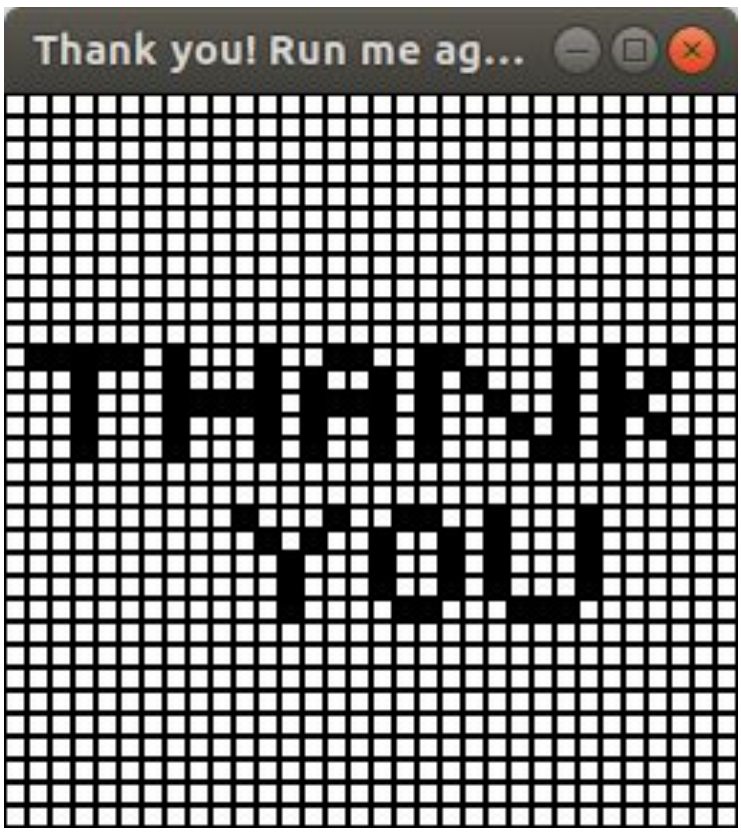
- requires racket/gui
- The racket/gui language combines all bindings of the racket language and the racket/gui/bas e and racket/draw modules.

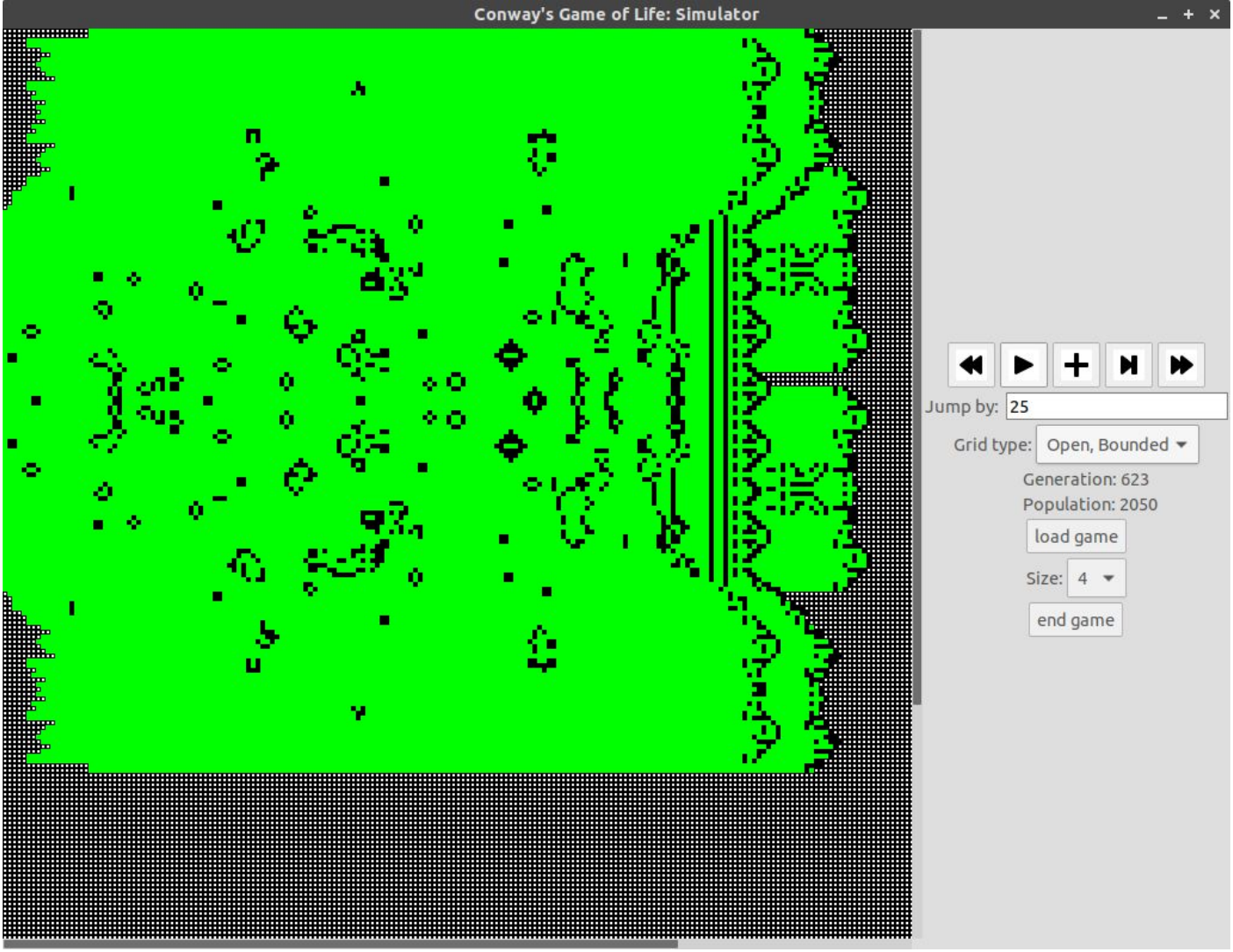
LIMITATIONS AND BUGS

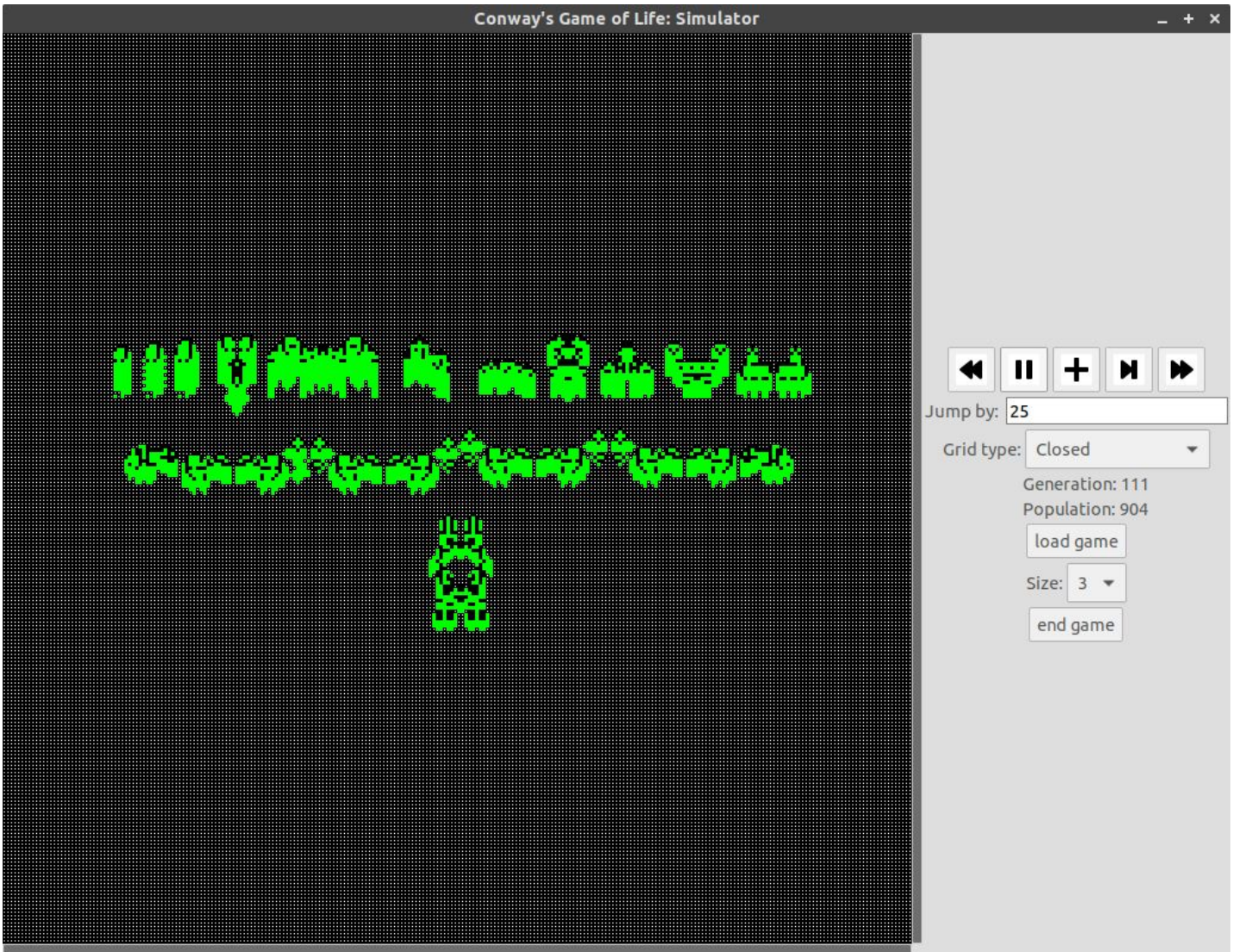
1. Jumping frames at a rapid rate or, in other words, moving forward a lot of number of frames at very fast rate is a bit of a problem as the program tends to freeze if the fast forward button is pressed multiple number of times.
2. Drawing a completely grid, i.e loading a different game takes time because the grid is now actually being completely rendered.
3. The case of Infinite open grid , where one can scroll through and see the entire variartion throughout the space could not be done. Therefore Instead of an infinte two-dimensional cell, we consider a finite rectangular grid with modifiable topological properties.
4. The scroll bars are a bit too slow.

CREDITS

Inspired from a conway video seen on youtube, we made a small credits presentation using the conway code we made, and used ti as our credits at the end of the game.




Conway's Game of Life: Simulator
Jump by: 25
Grid type: Open, Bounded
Generation: 623
Population: 2050
load game
Size: 4
end game


Conway's Game of Life: Simulator
Jump by: 25
Grid type: Closed
Generation: 111
Population: 904
load game
Size: 3
end game