Warsaw University of Technology
Faculty of Electronics and Information Technology

Intelligent Information Systems

Conway's Game Of Life

Final report

Author:
Maciej Suchecki

Lecturer:
Prof. Mieczysław Muraszkiewicz

8 lutego 2015

1 Introduction and motivation

This document contains a final report for a project developed by author during his participation in the Intelligent Information Systems lecture. The aim of the project was to implement an application that is able to run cellular automaton simulations, in particular Conway's Game of Life. The choice of this particular project was supported by a deep interest in the mentioned topic. The following chapters contain brief description of the Game of Life, implementation details, instructions for using the application, as well as further development ideas and the conclusion.

2 Conway's Game of Life

2.1 Description

The Game of Life is a cellular automaton introduced by the British mathematician John Horton Conway in 1970. It is a zero-player game, meaning that its evolution is determined by its initial state, requiring no further input. The user interacts with the Game of Life by creating an initial configuration and observing how it evolves. The game gained a lot of attention over the years, which is confirmed by the existence of a large number of websites devoted to this topic, such as the *LifeWiki*[1].

2.2 Rules

The universe of the Game of Life is an infinite two-dimensional grid of square cells, each of which is in one of two possible states: *alive* or *dead*. Every cell interacts with its eight neighbours. At each step, the following transitions may occur:

- Any live cell with fewer than two live neighbours dies, as if caused by under-population.
- Any live cell with two or three live neighbours lives on to the next generation.
- Any live cell with more than three live neighbours dies, as if by overcrowding.
- Any dead cell with exactly three live neighbours becomes a live cell, as if by reproduction.

The initial pattern is inputed by the user. The first generation is created by applying the above rules simultaneously to every cell in the seed -— births and deaths occur simultaneously, and the discrete moment at which this happens is sometimes called a tick. The rules continue to be applied repeatedly to create further generations.

In authors opinion, the most exciting and surprising thing about the Game Of Life is the fact, that despite its very basic rules, the simulation tends to create very complicated (sometimes infinite) systems and patterns. That is also probably the reason for its popularity over the years. Many scientists (mostly mathematicians) are fascinated by the game, devoting themselves to its further research – e.g. finding new interesting patterns – like described in the following chapter.

[1] `http://www.conwaylife.com/wiki/Main_Page`

2.3 Patterns

During Game of Life research process, a number of categories of patterns were discovered. Some of them have their representations in the application's predefined patterns, described in the following chapters.

2.3.1 Agars

An agar is a pattern that can tile the Life universe periodically both in space and in time.

2.3.2 Gardens of Eden

A Garden of Eden is a pattern that has no parents and thus can only occur in generation 0.

2.3.3 Guns

A gun is a stationary pattern that emits spaceships (or rakes) repeatedly forever.

2.3.4 Methuselahs

A methuselah is a pattern that takes a large number of generations in order to stabilize (known as its lifespan) and becomes much larger than its initial configuration at some point during its evolution.

2.3.5 Oscillators

An oscillator is a pattern that is a predecessor of itself. That is, it is a pattern that repeats itself after a fixed number of generations (known as its period).

2.3.6 Puffers

A puffer is a pattern that moves like a spaceship but leaves debris behind as it moves.

2.3.7 Sawtooths

A sawtooth is a finite pattern whose population grows without bound but does not tend to infinity. In other words, it is a pattern with population that reaches new heights infinitely often, but also infinitely often drops below some fixed value.

2.3.8 Spaceships

A spaceship is a finite pattern that reappears (without additions or losses) after a fixed number of generations displaced by a non-zero amount.

2.3.9 Still lifes

A still life is a pattern that does not change from one generation to the next, and thus is a parent of itself.

2.3.10 Wicks

A wick is a static or oscillating linearly repeating pattern that can be made to burn at one end.

3 Implementation

This particular implementation was done in JavaScript. This is due to author's desire to publish the project on the Internet after its completion. The JavaScript code is embedded into a small HTML website. The website contains a HTML5 canvas element, along with the buttons. The JavaScript part is drawing the cells using the canvas element.

Due to selected programming language and implementation type, the application (after publishing on a web server) will be able to be run on any kind of device which has a modern web browser. The website containing the application is build with RWD[2] paradigm, which means that the screen size is adjusted automatically to the screen size of the device. Furthermore, the canvas element also adjusts its size automatically, stretching the cell universe size to full screen, thus allowing for displaying as many cells as possible. User also is able to select one of the 3 cell sizes (10 by 10 pixels, 20 by 20 pixels and 40 by 40 pixels). This allows the user to customize grid size, readability and the application speed to own needs.

4 Launching and using the program

The application could be simply launched by opening the *index.html* file in web browser. The application was tested on *Google Chrome* and *Mozilla Firefox*, but it should run correctly on any modern web browser. When the browser window is opened, user is able to do the following tasks:

- load one of the predefined schemes:
  - generate random scheme,
  - generate simple oscillators (traffic light or pulsar),
  - generate simple spaceships (glider or lightweight spaceship),
  - generate simple gun (gosper glider gun),
  - generate a methuselah (acorn. diehard or r-pentomino).
- clear the board,
- customize cell size,
- run the simulation,
- generate only one step of the simulation.

All of the above tasks may be done using blue buttons on the bottom of the screen. Furthermore, the user is able to switch state (alive/death) of any cell on the grid by simply clicking on it using mouse. The simulation needs to be stopped for the mentioned operation. This allows for almost infinite possibilities of game testing.

The grid is of course finite, whereas in principle, the Life field is infinite. This leads to problems when the active area encroaches on the border of the array. Due to this, the left and right edges of the field are stitched together, and the top and bottom edges also, yielding a toroidal array.

[2] Rseponsive Web Design

5 Further development

Despite the fact that the author is rather satisfied with the application, he plans to further improve the application by performing the following steps in the future:

- increase the performance by implementing more complicated algorithm, such as List Life[3],
- allow for customising the parameters, such as iteration timeout, colors etc.,
- implement grid state import and export functionalities,
- add links to the informations about the author and the game itself,
- add more predefined patterns.

6 Conclusion

To sum up, the project was a great pleasure for the author, due to his deep interest in the simulation itself. Furthermore, development of the application was a great opportunity to further study the Game of Life and improve javaScript development skills. Of course there are some flaws present in the application, however the author wants to continue the development in the future.

[3] `http://dotat.at/prog/life/life.html`