Artificial Intelligence - Class Project

Optimizing longevity and robustness in Conway's Game of life

Rishi Baijal

Pulkit Manocha

Introduction

History

- In 1970, British mathematician John Conway conceptualized a cellular automaton based on von Neumann's ideas
- This was a hypothetical machine that could create copies of itself

Introduction

History

- The universe of this game consists of an infinite, two-dimensional and orthogonal grid of square cells.
- Each of these can be in one of the following possible states - alive or dead
- Each cell interacts with cells that can be horizontally, vertically or diagonally adjacent to it (also called neighbours)

Introduction

Rules

Neighbours can transform between states as per the following rules:

- Death by under-population: Any cell with fewer than two live neighbours dies.
- Death by overpopulation: Any live cell with more than three neighbours dies.
- Reproduction: Any dead cell with exactly three neighbours becomes alive
- Transition to the next generation: Any live cell with two or three neighbours moves on to the next generation.

Problem description

- The Conway's game of life, as described above, has quite a lot of potential for AI applications.
- We shall treat this as a cellular automata and use apply the genetic algorithm in order to explore how we can maximize life for structures that structures that are not permanent

Motivation

- We feel that we will be tackling a unique and interesting idea as a part of this project
- The existing literature in AI does not contain detailed descriptions of the role that Conway's game of life plays in this field
- Give AI the tools to play God and cultivate civilizations.

Plan of Action

Metrics to be used:

- maximum number of cells at k units of time
- maximum cell life in the grid over k intervals of time

Implementation details

- We initially implemented a simple game of life implementation in C++.
- We then added a genetic algorithm on top of our initial game of life code in such a way so as to maximize the life of long-lasting structures
- We run our genetic algorithm with 10 per cent elitism rate

Variations of the initial structure

For the entire civilisation

- Extinction of colonies
- Addition of cancerous cells

Rules of extinction

- We are accepting as user input the time period for which the extinction will take place
- In that time period, we are consistently killing most of the cells off.

Rules of cancer spread

- Spawn a dormant cancer cell at any random location
- If any cell is in the vicinity of a cancer cell, it becomes a cancer cell itself

Optimisations

- The sorting parameter was initially a function. It was replaced by a comparator.
- Initially, the data structure of our choice was an array. We replaced that with a boolean vector for reasons of data structure flexibility
- Instead of using structs, we decided to use a vector of pointers for speeding up the code.