Cellular Automata

An introduction to Conway's Game of Life

Aniruddha Deb

IIT Delhi

April 16, 2021

Determinism and Non-Determinism

A **Deterministic Algorithm** is an algorithm that repeatedly gives the same set of outputs for a given set of outputs. An example is a simple function $f(x) = x^2$

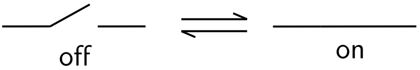
$$5 \longrightarrow \boxed{f(x) = x^2} \longrightarrow 25$$

A **Nondeterministic Algorithm** is an algorithm that does not give repeatable outputs for a given input. An example is a cup in which n dice are rolled. $(g : \mathbb{N} \rightarrow \mathbb{N}^n)$

$$1 \longrightarrow \boxed{\quad g \quad} \longrightarrow \{1, 2, 3, 4, 5, 6\}$$

State Machines

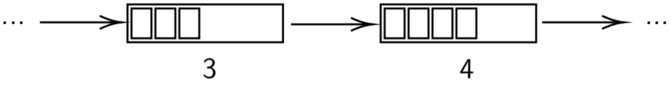
A **Finite State Machine** or **Finite State Automata** is a machine that can exist in exactly one of a finite number of states at a given point in time, and can *transition* between these states as a result of inputs. A simple example is a switch.



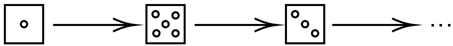
Can we make an infinite state machine by combining infinite finite state machines? Theoretically yes, but practically no. We'll cover this in more detail when we go to Cellular Automata.

Deterministic and Nondeterministic Finite State Machines

Combining both these concepts, a **Deterministic Finite State Automata (DFA)** is one whose next state is uniquely determined by the current state. As an example, a loading bar at n% can only progress to a state at (n+1)%



In contrast, a **Nondeterministic Finite State Automata (NFA)** is one whose next state is not uniquely determined by it's current state. An example would be the output of a dice roll: it's a NFA, because the number of states are finite, and the next state is independent of the current state



Cellular Automata

A **Cellular Automata** is a collection of cells (FSM with on/off state), each of which changes it's state based on the states of it's neighbouring cells.

Elementary cellular automata simply involve a 1D grid (a continuous array of cells). I'll not be covering those now. Instead, let's focus on 2D grids.

Each cell has two types of neighbourhoods: **Moore** and **Von Neumann**. Rules (either Deterministic or Nondeterministic) based on these neighbourhoods determine the next state of the cell.

Figure: Moore(L) and Von Neumann(R) neighbourhoods

Bounds for Cellular Automata

We'll cover Life in a while, but a question:

Are Cellular Automata Finite state or Infinite state?

Theoretically, nothing prevents cellular automata from being infinite-state automata. However, they would have a countably infinite number of states, which makes things (theoretically) simpler. For all practical purposes, however, cellular automata are finite and are bounded by the memory space of the computer simulating them.

If we do have to bound an automata, we generally impose a different set of conditions on the boundary: something like reflection or stasis or vanishing conditions.

Conway's Game of Life (finally!)

Conway's Game of Life is a Cellular Automata that follows the following rules on the Moore neighbourhood N of a cell C:

- If C is dead, and exactly 3 cells in N are alive, then C becomes alive in the next iteration i.e. C **is born**
- If C is alive, and exactly 2 or 3 cells in N are alive, then C remains alive in the next iteration i.e. C **survives**

These rules can be summarized as **B3/S23**. Variations of this rule set give several other automata. What's interesting about Life is that it's stable: it doesn't explode or decay very fast.

Few other things: Life is Turing complete (proof left as an exercise), and Life is Undecidable: you cannot predict the initial state from the final state, or directly compute n states into the future (as a consequence of the Halting problem).

Interesting patterns in Life

- Still Life
- Oscillators
- Gliders
- Spaceships
- Methuselahs
- Glider Gun

Further reading/watching

- Numberphile video on Life (With Conway himself): `https://www.youtube.com/watch?v=R9Plq-D1gEk`
- LifeWiki, the reference on all Game of Life Patterns: `https://www.conwaylife.com/wiki/`
- Golly, an open-source program for simulating life: `http://golly.sourceforge.net`
- PyGameOfLife, a Game of Life program implemented in Python: `https://github.com/Aniruddha-Deb/PyGameOfLife`