Cellular Automata and Computational Universality

Thomas Archbold

T.Archbold@warwick.ac.uk, Department of Computer Science, University of Warwick

Motivation

The motivation behind this project was to provide a whistle-stop tour on some introductory topics in cellular automata and computational universality. It is aimed at people with little-to-no experience in this area of computer science, and aims to provide a solid foundation on the topic on which further reading can be undertaken.

Objectives

- Learn more about a field of Computer Science to which I had so far had little exposure
- Produce a beginner's guide of sorts for anyone interested in cellular automata and computational unversality
- Develop software capable of simulating the automata covered in the paper, as well as a few more
- Undertake a research project and present it in a formal setting

Introduction

A cellular automaton is a discrete model of computation which consists of a finite collection of "cells", each in one of a finite number of states. The state of each of these cells may evolve over the progression of time in discrete steps, and may do so according to a deterministic set of rules based on the states of neighbouring cells. In most cases, updates to the cells of an automaton will occur everywhere and all at once; there also exist asynchronous cellular automata, which are able to update cells independently, allowing the new state of a cell to affect the calculation of states in neighbouring cells.

Automata Covered

While there are many types of cellular automata, the two covered in this project are elementary cellular automata and two-dimensional automata (including totalistic and Life-like automata).

In particular, the paper and the simulation software developed alongside the project is capable of handling:

- Elementary Automata
  - Rules 30, 54, 90, 110, 150, 182, 232, 250
- Life-like automata
  - Game of Life, Brian's Brain, Wireworld, Langton's Ant, Replicator, Life Without Death, Diamoeba, Two By Two, Move, Anneal

Elementary Cellular Automata

Elementary cellular automata are the simplest class of one-dimensional cellular automata: they consist of a one-dimensional array of cells, and each cell may take one of only two values, 0 or 1. Further, rulesets only consider the value of the cell itself and those of its immediate left and right neighbours when deciding on the cell's new state. The project covers Wolfram codes and how to interpret them, using Karnaugh Maps to implement them in a concise manner, particularly useful in programming. It discusses the transformations of elementary automata, mirroring and complementation, and how rules may be equivalent under one or both of these rules, or fundamentally unique under any transformation.

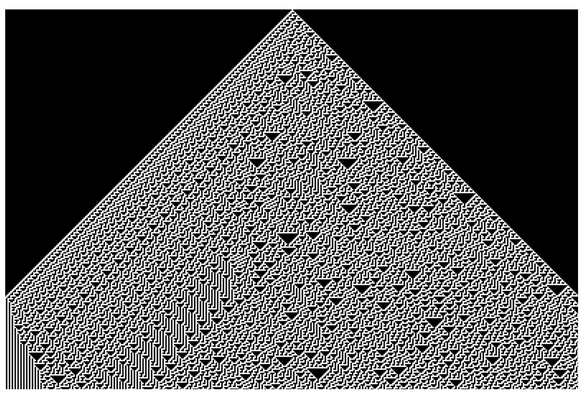

Figure 1: Rule 30, an elementary automaton that can be used for cryptography

Finally for elementary cellular automata, Wolfram classes are briefly discussed, outlining the behaviour exhibited by automata of each of the classes 1, 2, 3, and 4.

Two-dimensional Cellular Automata

As a natural progression from mere one-dimensional automata, two-dimensional automata can be capable of much more, both in terms of computation and in the intricacy of pattern which they give rise to. The history of cellular automata is touched on here, from John von Neumann's pursuit of a self-replicating machine, to automata gaining more exposure with the release of John Conway's Game of Life in 1970. The rules of Life are discussed in greater depth, from the fascinating structures which arise within it, to the family of automata it helped to inspire, to its capacity for universal computation.

Life-like Automata

The requirements for an automaton to be Life-like (similar to Game of Life) are covered, as well as understanding the useful notation behind such automata (e.g. B368/S245).

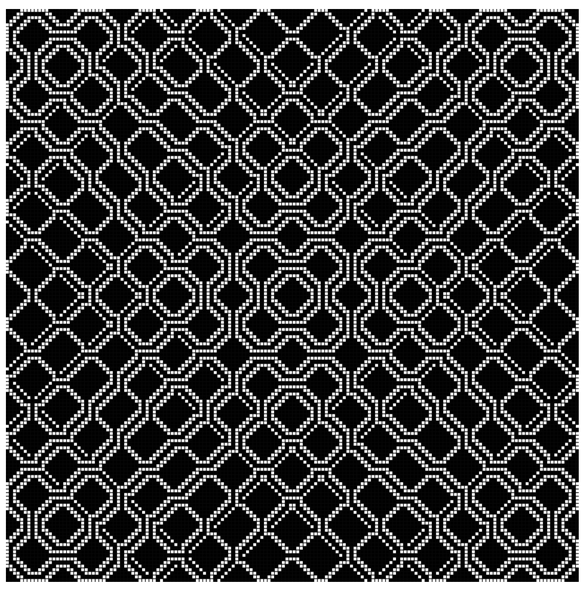

Figure 2: Replicator, a Life-like automaton, showing some of the wonderful patterns which may arise from simple rules

Computational Universality

Turing Machines are introduced, with both formal and high-level descriptions, as well as the concepts of computable functions and universal computation, with respect to this class of automata. The process of building a universal machine in cellular automata is covered, in particular the approaches taken by Game of Life [1], Wireworld [2], and Rule 110 [3].

Firing Squad Synchronisation Problem

The Firing Squad Synchronisation Problem is one which asks for the construction of a one-dimensional cellular automaton which, starting with a single "general" giving the fire command, ends with all "soldiers" firing simultaneously, wherre each soldier may only change his state based on his current one and those of his two immediate neighbours. The overview for two minimal-time solutions are discussed, namely Waksman's [4] solution and Mazoyer's least-states solution [5].

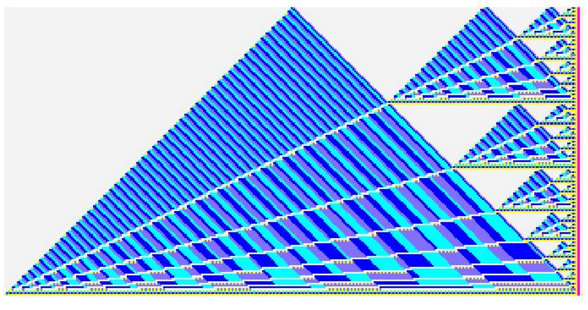

Figure 3: Mazoyer's 6-state solution to FSSP

Majority Problem

The Majority Problem is a problem asking that, for a one-dimensional cellular automaton of $i + j$ cells, each cell of which is in one of two states. i are in state 0, and the remaining j are in state 1. If $i > j$, all cells must eventually be set to i's value, and if $j > i$ all cells must eventually equal j's value. The outcome of the automaton is unspecified when $i = j$.

The paper discusses a Monte Carlo solution to the problem from Gács, Kurdyumov, and Levin, and covers how the definition of recognising a majority may be relaxed to allow for a deterministic solution, in the form of Rule 184.

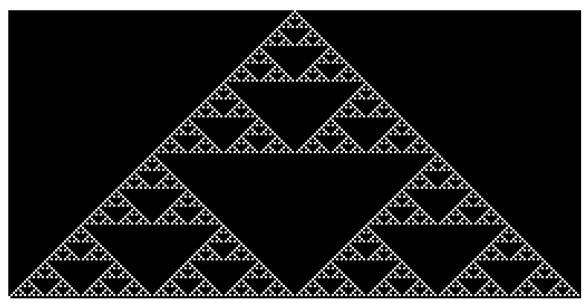

Figure 4: Rule 90 elementary cellular automaton, with remarkable similarity to the Sierpinski Sieve

Simulation software

In parallel with writing this primer on cellular automata, a program was developed to simulate all of the automata covered. It was written using only C and SDL2, and while better software exists, such as Golly, it forced a deeper understanding of the intricacies behind each automaton, especially as they became more complex with von Neumann's machine.

References

[1] This is a fully universal turing machine (utm) implemented in conway's game of life.
`http://www.rendell-attic.org/gol/fullutm/index.htm`.
Page accessed: 2018-09-13.

[2] The wireworld computer.
`https://quinapalus.com/wi-index.html`.
Page accessed: 2018-09-17.

[3] Matthew Cook.
Universality in elementary cellular automata.
Complex Systems.

[4] Abraham Waksman.
An optimum solution to the firing-squad synchronization problem.
Information and Control, 9:66–78.

[5] Jacques Mazoyer.
A six-state minimal time solution to the firing squad synchronization problem.
Theoretical Computer Science, pages 183–238.