Elementary cellular automata as conditional Boolean formulæ

Trace Fleeman y Garcia

October 2015

Abstract

I show that any elementary cellular automata – a class of 1-dimensional, 2-state cellular automata – can be deconstructed into a set of two Boolean operators; I also present a conjecture concerning the computational completeness of a rule and its relationship to complete Boolean operators.

1 Introduction

Cellular automata are defined as 4-tuples in the form $(\mathcal{L}, \Sigma, \mathcal{N}, \phi)$, where $\mathcal{L}$ is some finite or infinite lattice of interconnected finite state automata (the cells), Σ is the set of states, $\mathcal{N}$ is the neighborhood, b is a boundary condition, and ϕ is the local transition rule. (1)
Cellular automata were first introduced in the 1950s by John von Neumann and Stanislaw Ulam, in order to develop an abstract model for biological self-reproduction. (2; 3) A computationally universal cellular automaton was presented by Albert and Culick in 1987. (4) It was shown be Stephen Wolfram that two-state, 1-dimensional, nearest-neighbour rules are sufficient constructing universal rules. (5)

1.1 Elementary cellular automata

Elementary cellular automata are those cellular automata where $\mathcal{L} = \mathbb{Z}$, $|\Sigma| = 2$, with a neighborhood consisting of the two closest neighhboring cells. (6)

2 Conditional Boolean forms of cellular automata

We may first separate configurations by the state of their center cell, and then evaluate truth tables for these rules, where c represents the state of the center cell in the next generation, while p and q represent the state of the right and left cells respectively.

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.2553v1 | CC BY 4.0 Open Access | rec: 24 Oct 2016, publ:

p	q	c
0	0	
0	1	
1	0	
1	1	

p	q	c
0	0	
0	1	
1	0	
1	1	

Rule 110, when broken into constituent Boolean operators, can be represented as the truth tables

p	q	c
0	0	1
0	1	1
1	0	1
1	1	0

p	q	c
0	0	0
0	1	1
1	0	0
1	1	1

One can see that the left truth table ($c = 1$) is equivalent to the truth table of the NAND Boolean function. On the contrary, the right truth table ($c = 0$) is equivalent to the Boolean function q (in which the center cell inherits the state of the right cell).
Therefore, we may represent Rule 110 as the tuple (NAND, q), or alternatively *if NAND, else q*. When construing elementary cellular automata as conditionals, we may implement them more exhaustively in pseudocode:

```
If c == true:
    print(p NAND q)
Else:
    print(q)
End if
```

Here, c (the center cell) acts as what I label a *control value*. When c is true (or 1), the NAND operation is performed on p and q and the center cell takes the state of the solution of the Boolean function. In all other cases, the center cell inherits the state of q.

3 Relation to computational universality

It should be noted that Rule 110, a proven computationally-complete rule, (8) contains the NAND function specifically. It is common knowledge that the NAND function is functionally complete, and Aaronson *et al* state that it is computationally universal.[1;2] (9)
Following from this, I conjecture there is some form of relationship between so-called "computationally universal" logic gates and computationally universal cellular automata. I have devised two separate versions of this conjecture.

[1] "The Toffoli gate is computationally universal, because... [computes] the NAND function." Page 4.
[2] "...it contains a NAND gate, which is computationally universal." Page 8.

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.2553v1 | CC BY 4.0 Open Access | rec: 24 Oct 2016, publ:

3.1 Strong conjecture

The *strong* form of the conjecture states that

> If an elementary cellular automaton is composed of at least one computationally universal logic gate, then the cellular automaton is computationally universal.

Take rule 54, which, as of 2016, has not been shown to be computationally universal. (10)

p	q	c
0	0	1
0	1	0
1	0	0
1	1	0

p	q	c
0	0	0
0	1	1
1	0	1
1	1	1

The table for $c = 1$ is the NOR gate, a universal gate. The table for $c = 0$ is the OR gate. If the strong form of the conjecture holds true, then rule 54 is a computationally universal.

3.2 Weak form of conjecture

The *weak* form of the conjecture rather states that

> All computationally universal elementary cellular automata are composed of at least one computationally universal logic gate.

This differs from the strong form of the conjecture insomuch that it allows for the existence of rules that contain complete Boolean operations yet are not computationally complete themselves.

References

[1] Kroc, Jiri, Peter Sloot, and Alfons Hoekstra. "'Classical Cellular Automata." *Simulating Complex Systems by Cellular Automata*. Heidelberg: Springer, 2010. 6-10. Print.

[2] Wolfram, Stephen. "Why These Discoveries Were Not Made Before." *A New Kind of Science*. Champaign, IL: Wolfram Media, 2002. 876. Print.

[3] Sarkar, Palash. "A Brief History of Cellular Automata." *CSUR ACM Comput. Surv. ACM Computing Surveys* 32.1 (2000): 80-107. Web.

[4] J. Albert, K. Culik II, "A simple universal cellular automaton and its one-way and totalistic version", *Complex Systems 1* (1987) 1–16.

[5] Wolfram, S. A New Kind of Science. Champaign, IL: Wolfram Media, pp. 646-647, 2002.

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.2553v1 | CC BY 4.0 Open Access | rec: 24 Oct 2016, publ:

[6] Weisstein, Eric W. "Elementary cellular automaton." (2002).

[7] *Computation*, 1967, p. 255-258

[8] Cook, Matthew (2004). "Universality in Elementary Cellular Automata". *Complex Systems*. 15 (1). ISSN 0891-2513.

[9] Aaronson, Scott, Daniel Grier, and Luke Schaeffer. "The classification of reversible bit operations." *arXiv preprint arXiv:1504.05155* (2015).

[10] Wolfram, Stephen. *A New Kind of Science*. p. 697. Wolfram Media, Inc. ISBN 1-57955-008-8. (2002)