Wire Brains: an extension to Wireworld allowing for evolvable digital brains

Clifford Bohm[1,4], Chris Reeves[3], and Jory Schossau[2,4]

[1]Michigan State University, Department of Integrative Biology
[2]Michigan State University, Department of Computer Science and Engineering
[3]Michigan State University, Department of Psychology
[4]BEACON Center for the Study of Evolution in Action
cliff@msu.edu

Abstract

The Wireworld cellular automaton effectively models the behavior of electrical currents flowing through wires, enabling the creation of circuits capable of performing logic operations. Importantly, Wireworld has been proven Turing complete, further underscoring its computational potential. In this work, we expand Wireworld from a two-dimensional grid to a three-dimensional cuboid and harness the power of digital evolution to construct operational digital brains. These brains are evolved to control simulated robots that must perform simple navigation and harvesting tasks. To the best of our knowledge, this work represents the first time evolutionary computation has been employed in conjunction with the Wireworld cellular automaton.

Introduction

Wireworld is a cellular automaton invented by Brian Silverman and first presented in his program Phantom Fish Tank in 1987 (Dewdney, 1990). In Wireworld, every cell can exist in one of four states: **`empty`**, **`wire`** (a.k.a., conductor), **`charge`** (a.k.a., electron head), and **`reset`** (a.k.a., electron tail). There are also four rules that determine the state of each cell on each update.

1. **`empty`** $\rightarrow$ **`empty`**
2. **`wire`** $\rightarrow$ **`charge`**,
   if exactly 1 or 2 neighbors are **`charge`**
3. **`charge`** $\rightarrow$ **`reset`**
4. **`reset`** $\rightarrow$ **`wire`**

Wireworld's rules result in behavior similar to electrical currents in copper wires. These rules can give rise to interesting dynamics. For example, we can use the neighbor limit in rule (2) to make a diode, a simple circuit that allows charge to flow in one direction but not the other. As Figure 1(a) shows, charge can flow from the top wire cell to the bottom one. Conversely, in Figure 1(b), charge originating at the bottom is not able to flow through the structure.

Different arrangements of wire can implement logical operators and conditionals, and Wireworld has been shown to be Turing complete (Tringham, 2014; Cook et al., 2004).

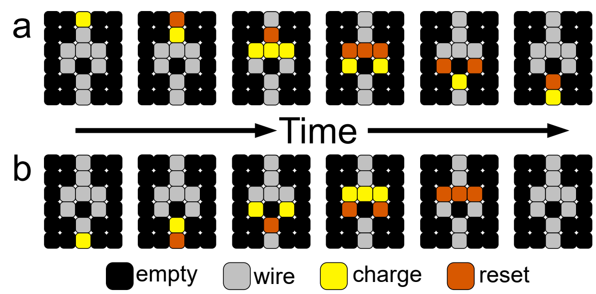

Figure 1: Example of wireworld arrangement implementing a diode.

However, since wires cannot cross, laying out complex programs can require huge grids. One solution to this problem is the introduction of strong and weak charge (Gladkikh and Nigay, 2018), though this is not explored further in this study.

We utilize digital evolution to construct three-dimensional "wire brains" guided by the Wireworld rules. A wire brain's functionality stems from the positions of **`wire`** and **`empty`** cells plus designated input and output cells.

We show that wire brains can achieve high performance on a maze task and a harvesting task, particularly when given access to external memory. However, brains without external memory were also able to implement memory in wire constructs.

Methods

In this work, we employ the Modular Agent-Based Evolver (MABE) Bohm and Hintze (2017), a tool specifically designed for digital evolution research. We chose to utilize two navigation-based tasks and other components, such as the selection process, already present in MABE, with the only new code being our novel "wire brain."

Wire Brains A "wire brain" consists of two components: a cuboid array of cells, each either **`wire`** or **`empty`**, and two lists determining which cells connect to inputs and outputs. Our implementation uses a direct encoding: mutations act directly on the cells and connection lists. In our experiments, inputs and outputs were restricted to the bottom and top layers of the brain, respectively.

In order to process inputs and generate outputs, cells associated with active inputs are set to **`charge`**. The brain then runs for a certain number of time steps using standard Wireworld rules, after which the state of the cell associated with each output determines the brain's output.

We incorporated several parameters to allow for customization, three of which were tested and are described here. First, we added support for external memory, implemented as additional outputs collected each time step and supplied as inputs on the next. Second, we introduced an option to clear the **`charge`** and **`reset`** states of the brain before new task inputs were introduced (Clear Between Updates, or CBU). Finally, we allowed customization of the brain's width, height, and depth.

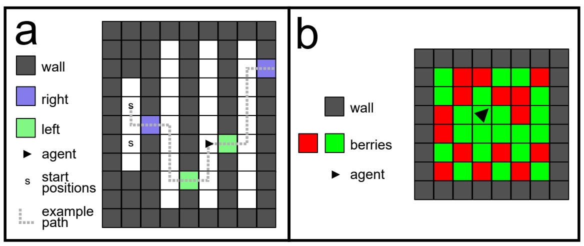

Figure 2: Illustrations of Maze (a) and Berry (b) tasks.

Tasks In the Maze task (Edlund et al., 2011), agents must navigate mazes made up of long hallways separated by walls (Figure 2(a)). In each wall, there is an opening with a marker that informs the agent if the next opening is to their left or right. Every time step, agents can move left, right, or forward. Agents have a limited lifetime in which they must respond to and remember the markers in order to solve each maze before time runs out.

In the Berry task (Figure 2(b)), agents are provided a limited amount of time to collect red and green berries in a walled grid arena. On each time step, agents can move forward, turn left, turn right, or collect. If an agent collects, the empty location will be filled with a new random berry when they move to a new location. They receive 1 point for every berry they collect if that berry is the same color as the last berry collected; otherwise, they lose 0.4 points. Thus, agents must minimize switching to maximize their score.

Experiments For each task, we tested eight conditions, consisting of all combinations of: Clear Between Updates (CBU): on / off, brain size: small $(5 \times 10 \times 5)$ / large $(10 \times 10 \times 10)$, and external memory: 0 / 4.

For each condition, we ran 62 replicates. We used tournament selection with discrete generations and population size 100.

Results and Discussion

We found that wire brains can perform well on both tasks under most conditions. Wire brains were capable of evolving connections between inputs and outputs, employing memory, and developing circuits for reactionary behavior.

The maze task requires agents to use memory, so we were not surprised that we found no successful solutions in conditions where CBU was on and external memory was unavailable. While we did see perfect performance develop in all but one condition with CBU on and external memory, we did see perfect agents develop in a few conditions with CBU off and no external memory (1 "small brain" and 3 "large brain"). This confirms that evolved wire brains are capable of implementing memory algorithms directly and need not rely on external memory.

Scoring in the berry task is somewhat dependent on the randomness of berry replacements. Previous work suggests that a trivial agent can achieve a score of 60, whereas scores exceeding 100 require sophisticated strategies. Of the conditions with CBU off, we observed no high-performing agents. We found that all conditions with CBU on were able to generate high-preforming agents (at least 12 out of 62), but, contrary to our expectations, the best performance was seen in conditions with CBU on and without external memory. We had believed that memory of the color of the last berry collected was essential for high-performance; however, evolution found a simple reactionary strategy using environmental cues that worked effectively without memory.

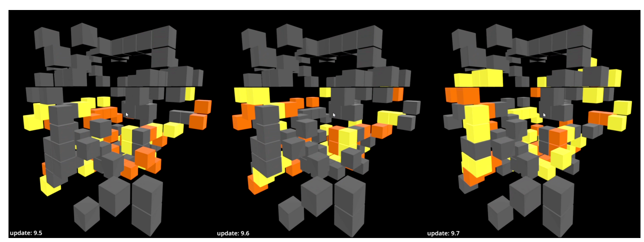

Figure 3: Three consecutive frames from an active wire brain trained on the berry task. Yellow is **`charge`**, red is **`reset`**, and gray is **`wire`**.

Figure 3 shows an example of a wire brain evolved on the berry task. Unlike human-designed instances of Wireworld, this brain appears chaotic and unorganized. A video of the brain in Fig. 3 can be found at `https://youtu.be/MB3BaS5sT2M`.

Conclusion We have shown that Wireworld is evolvable. Considering that we have only shown a single implementation and that there are several alternatives to the update rules, genetic encoding, and mutational operators which could be considered, we feel that this is a promising substrate and should be a target for future research.

References

Bohm, C. and Hintze, A. (2017). Mabe (modular agent based evolver): A framework for digital evolution research. In *Artificial Life Conference Proceedings*, pages 76–83. MIT Press One Rogers Street, Cambridge, MA 02142-1209, USA journals-info . . . .

Cook, M. et al. (2004). Universality in elementary cellular automata. *Complex systems*, 15(1):1–40.

Dewdney, A. K. (1990). Computer recreations. *Scientific American*, 262(1):146–149.

Edlund, J. A., Chaumont, N., Hintze, A., Koch, C., Tononi, G., and Adami, C. (2011). Integrated information increases with fitness in the evolution of animats. *PLoS computational biology*, 7(10):e1002236.

Gladkikh, V. and Nigay, A. (2018). Wireworld++: A cellular automaton for simulation of nonplanar digital electronic circuits. *Complex Systems*, 27(2018).

Tringham, J. (2014). Properties of cellular automata.