Cellular Automaton on Hanoi Graphs

Geethan Pfeifer

January 10, 2025

1 Introduction

This document informally describes cellular automaton on Hanoi Graphs, which resemble Sierpinski Triangles, and also a basic implementation in Golly.

2 Hanoi Graphs

Read `https://mathworld.wolfram.com/HanoiGraph.html` for an introduction to Hanoi Graphs.

3 Structure of Hanoi Graphs

Every Hanoi graph has 3 vertices of degree 2 (denote these vertices as *corner* vertices), with every other vertex having degree 3 (denote these vertices as *inner* vertices.)[1]
Each inner vertex is part of a 3-clique and a 2-clique.

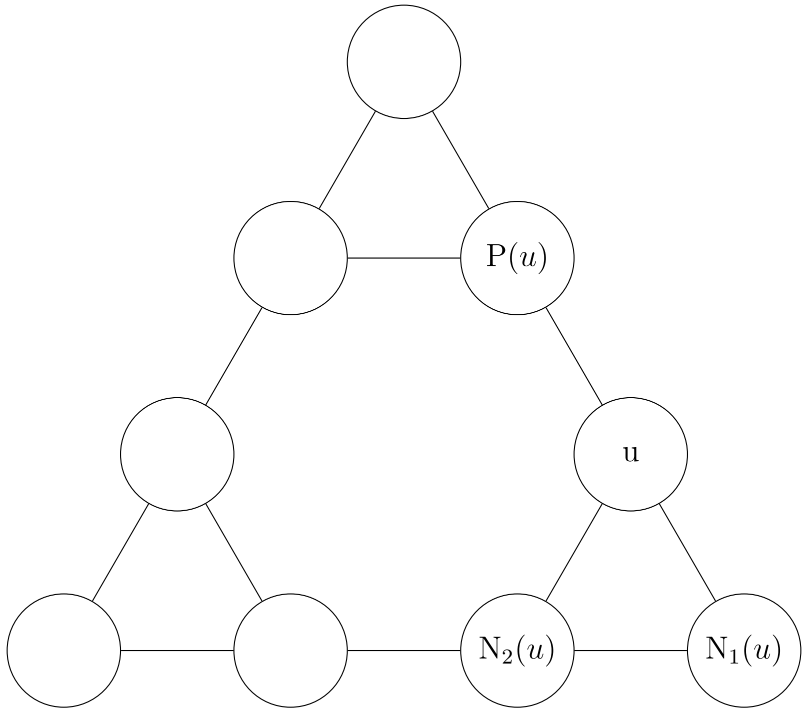
4 Neighbourhood

Consider a Hanoi Graph H_n as you would typically on a plane.
For each inner vertex u, consider the vertex it forms a 2-clique with as $\mathrm{P}(u)$. Consider the vertices it forms a 3-clique with as $\mathrm{N}_1(u)$, $\mathrm{N}_2(u)$, with $\mathrm{N}_1(u)$ being clockwise to u and $\mathrm{N}_2(u)$ being counterclockwise to u.
Define the neighbourhood of u as the tuple $(u, \mathrm{P}(u), \mathrm{N}_1(u), \mathrm{N}_2(u))$.

[1] These are not standard definitions, I'm just defining them as such for convenience.

4.1 Example

For example, consider H_2 below, with labelled inner vertex u, and labelled $\mathrm{P}(u)$, $\mathrm{N}_1(u)$, and $\mathrm{N}_2(u)$.



4.2 Note

On a Hanoi graph, for a inner vertex u with $\mathrm{P}(u)$, $\mathrm{N}_1(u)$, and $\mathrm{N}_2(u)$ also inner vertices, $\mathrm{P}(\mathrm{P}(u)) = u$, $\mathrm{N}_1(\mathrm{N}_1(\mathrm{N}_1(u))) = u$, and $\mathrm{N}_2(\mathrm{N}_2(\mathrm{N}_2(u))) = u$.

5 Cellular Automaton

Take a Hanoi Graph $H_n = (V, E)$, with inner vertices I.
Let Q be the set of states.
Define a transition function $\delta : Q^4 \to Q$.

Define $\mathrm{CA}_0(u) : V \to Q$.
For $t > 0$, define:

$$\mathrm{CA}_t : V \to Q,$$

$$\mathrm{CA}_t(u) = \begin{cases} \delta(\mathrm{CA}_{t-1}(u), \mathrm{CA}_{t-1}(\mathrm{P}(u)), \mathrm{CA}_{t-1}(\mathrm{N}_1(u)), \mathrm{CA}_{t-1}(\mathrm{N}_2(u))), & u \in I \\ \text{undefined}, & u \notin I \end{cases}$$

For convenience, we can define $CA_t(u) = CA_0(u), u \notin I$, but other options are possible.

Concisely, the state of an inner vertex u depends on the states of its neighbours and itself in the previous generation, and the state of corner vertices is undefined.

6 Elementary Automaton

Take $Q = \{0, 1\}$. Clearly, there are $2^{2^4} = 65536$ possible transition functions.
We can number them with 16-bit integers, with the m-th bit of n being the value of $\delta_n(m_0, m_1, m_2, m_3)$, where $m_3 m_2 m_1 m_0$ is the binary representation of m.
Call δ_n `Sierpinski n`[2]

6.1 Example

For example, take `Sierpinski 49980`.
The binary representation of 49980 is 1100001100111100_2.
This is the transition table:

m	$\mathrm{CA}_{t-1}(u)$	$\mathrm{CA}_{t-1}(\mathrm{P}(u))$	$\mathrm{CA}_{t-1}(\mathrm{N}_1(u))$	$\mathrm{CA}_{t-1}(\mathrm{N}_2(u))$	$\mathrm{CA}_t(u)$
0	0	0	0	0	0
1	1	0	0	0	0
2	0	1	0	0	1
3	1	1	0	0	1
4	0	0	1	0	1
5	1	0	1	0	1
6	0	1	1	0	0
7	1	1	1	0	0
8	0	0	0	1	1
9	1	0	0	1	1
10	0	1	0	1	0
11	1	1	0	1	0
12	0	0	1	1	0
13	1	0	1	1	0
14	0	1	1	1	1
15	1	1	1	1	1

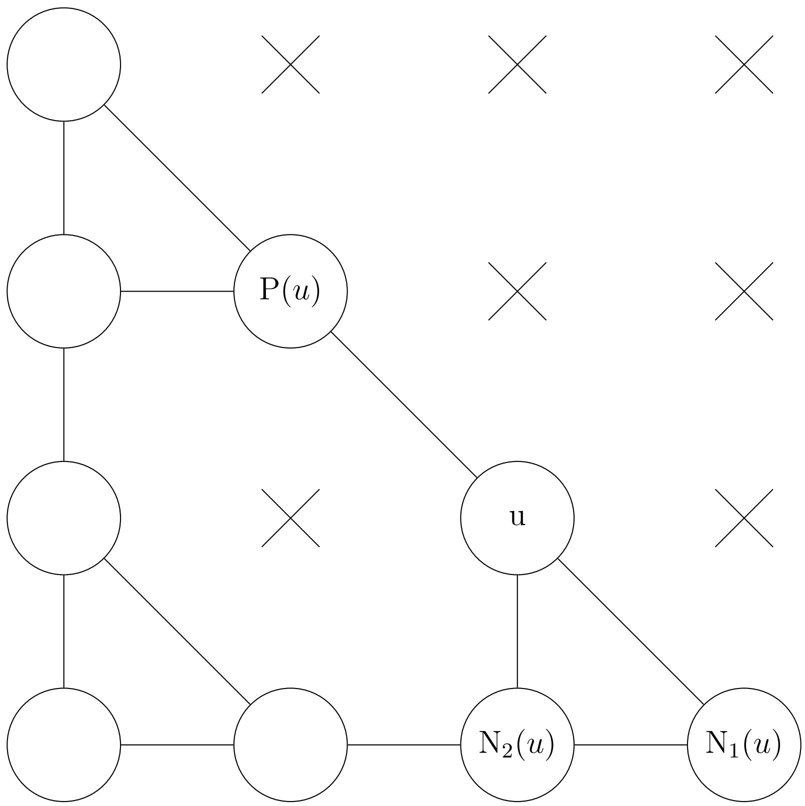
(This rule updates to the XOR sum of the neighbours of a vertex.)

[2]Perhaps `Hanoi n` is a better name, but I already named it `Sierpinski n` in my Golly implementation.

7 Golly

7.1 Representation on a 2-D grid

If you "skew" the planar representation of a Hanoi Graph slightly it fits nicely on a 2D grid. For example, we can "skew" the graph in 4.1 as such:



Each cross represents an empty square.

7.2 Advantages of this representation

The neighbourhood of an inner vertex u as defined in 4 (and their states) can be determined solely by the Moore neighbourhood of u on the 2D grid, meaning that we can create Golly rules for it.[3]
Also, this representation of the Hanoi Graph can be easily generated by `Wolfram 60`.

7.3 Generating golly rules

sierpinskirulegen generates a Golly rule corresponding to a specified Sierpinski rule. In the rule, state `0` corresponds to an empty square, and states `1` and `2` correspond to states 0 and 1 as per 6 respectively.

[3] I was shocked to discover that there are very few neighbourhoods supported by Golly! I initially planned to use a different representation, but that representation wouldn't be possible with a neighbourhood supported by Golly.

8 Future work

- Develop an "infinite Hanoi graph" such that all vertices are inner.
- Develop analogues of tori such that all vertices are inner.
- Create a custom program for cellular automaton on fractal patterns–Golly is quite limited.
- Develop the concept of a "direction". (How would spaceships work with these cellular automaton? Are they even possible?)

References

[1] Weisstein, Eric W. "Hanoi Graph." From *MathWorld*–A Wolfram Web Resource.
https://mathworld.wolfram.com/HanoiGraph.html

[2] Weisstein, Eric W. "Elementary Cellular Automaton." From *MathWorld*–A Wolfram Web Resource.
https://mathworld.wolfram.com/ElementaryCellularAutomaton.html

[3] LifeWiki. "Moore neighbourhood." Retrieved on January 10, 2025.
https://conwaylife.com/wiki/Moore_neighbourhood

[4] LifeWiki. "Tutorials/Creating custom rules". Retrieved on January 10, 2025.
https://conwaylife.com/wiki/Tutorials/Creating_custom_rules