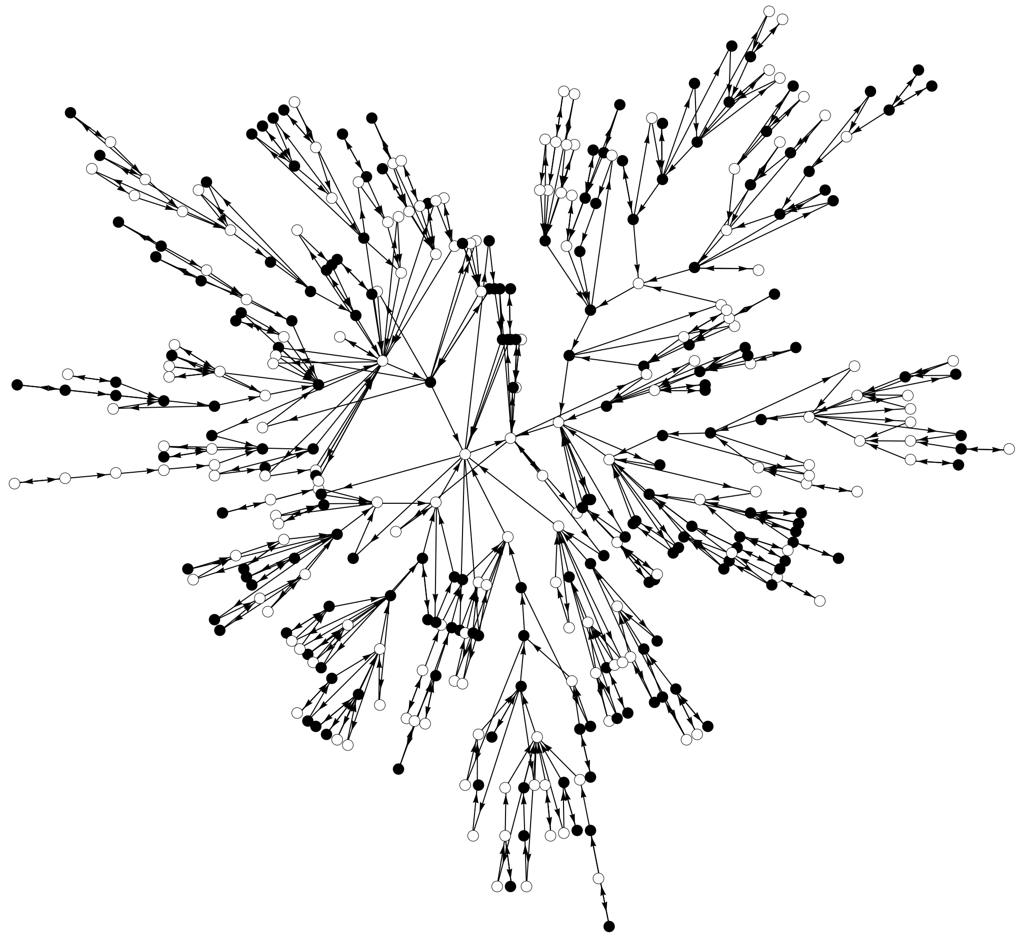
Cellular Automata on Networks

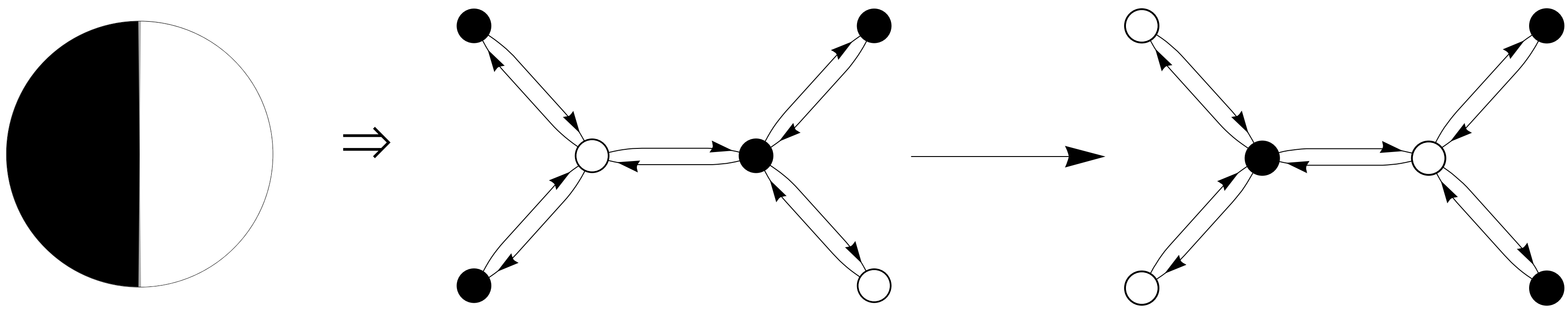
Jesse Nochella

A cellular automaton is loosely defined as a collection of cells with states that change their state depending on at least the states of neighboring cells.

Some cellular automata make further distinctions between their neighbors to some order of placement. This allows a cell that is a certain state to be differentiated from another cell at the same distance to the center cell that is identical in form. In this way, ordered data constructs such as sequences of numbers can be assigned to each individual cell seperate and uniquely, to set conditionals for neighboring cells to meet and cause the center cell to set it's state a certain way.

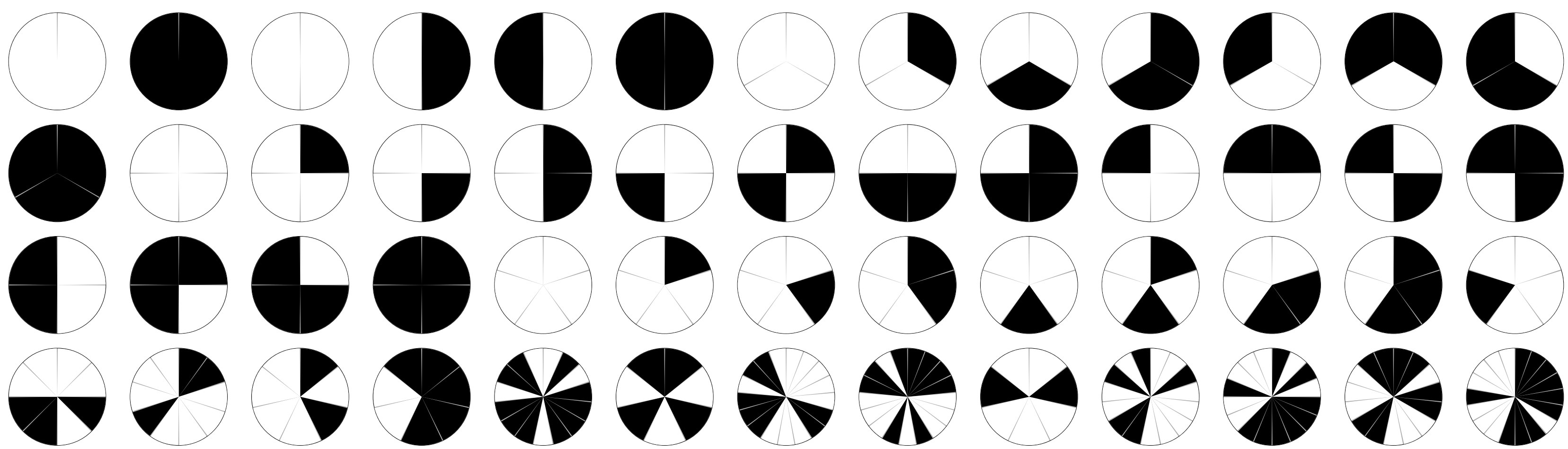
In the systems demonstrated here, a more generalized cellular automaton rule allows a cell to have an arbirtrary number of neighbors, and thus can be set to work on any given network topology. Each node/cell of the network is given a state. Then to run a single step, cells look at the states of other cells with arrows pointing to them to determine based on what percentages of each state the collective neighborhood expresses. A node then simultaneously gives out it's previously set state to all nodes which it points to.





While systems of this kind can have any number of possible states, the ones demonstrated here have only two, represented as black and white. The rules are set up to represent regions of a percentage space going from 0% to 100%. When the percentage of black cells in the neighborhood falls within a given region, the color of that region is given as the next state for the center cell to have. The rule icon represents the clockwise fill of the percentage of black cells. This rule setup allows one to specify a rule that can take an arbitrary number of neighbors and always convert it into a precise value.

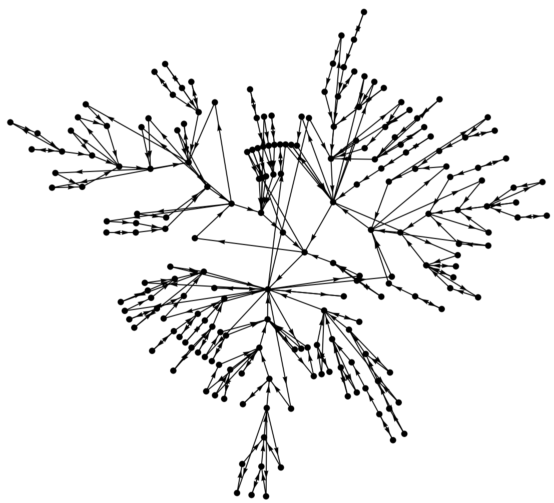
Here are some examples of rules of this kind. Note that there are some redundancies because of the enumeration scheme used to generate the first 3 rows.

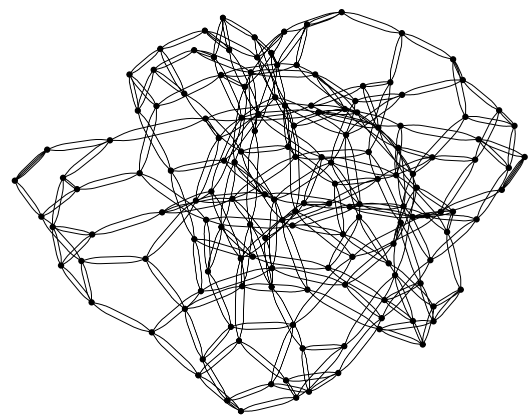


Ideas on variations

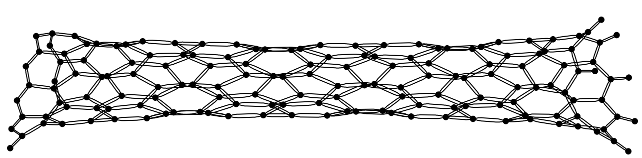
- •Different fundamental rule setups. Different than percentage based? What kinds?
- •More colors added to the space.
- •Different network topologies. What kinds?
- •Including nodes at a distance other than one from the center cell. Subvariations of this include:
  - —Just counting the nodes regardless of how far they are from the center cell. This would include the typical variation in totalistic cellular automaton rules that count the center cell as well and equal to it's neighbors.
  - —Differentiating 'levels' of nodes. So that separate collections of sums are made for each level of nodes and can be used for more precise cellular automaton rules.
  - —Deciding whether or not to 'spread' to the next greater distance depending on the color of the nodes? On the number of connections?
  - —Implementing a 'memory' of the states of cells throughout their time history and using that data to expand the rule set.

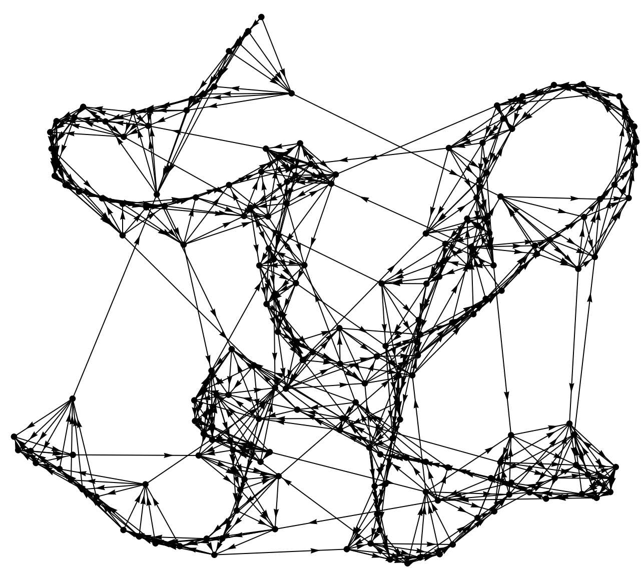
Experiments

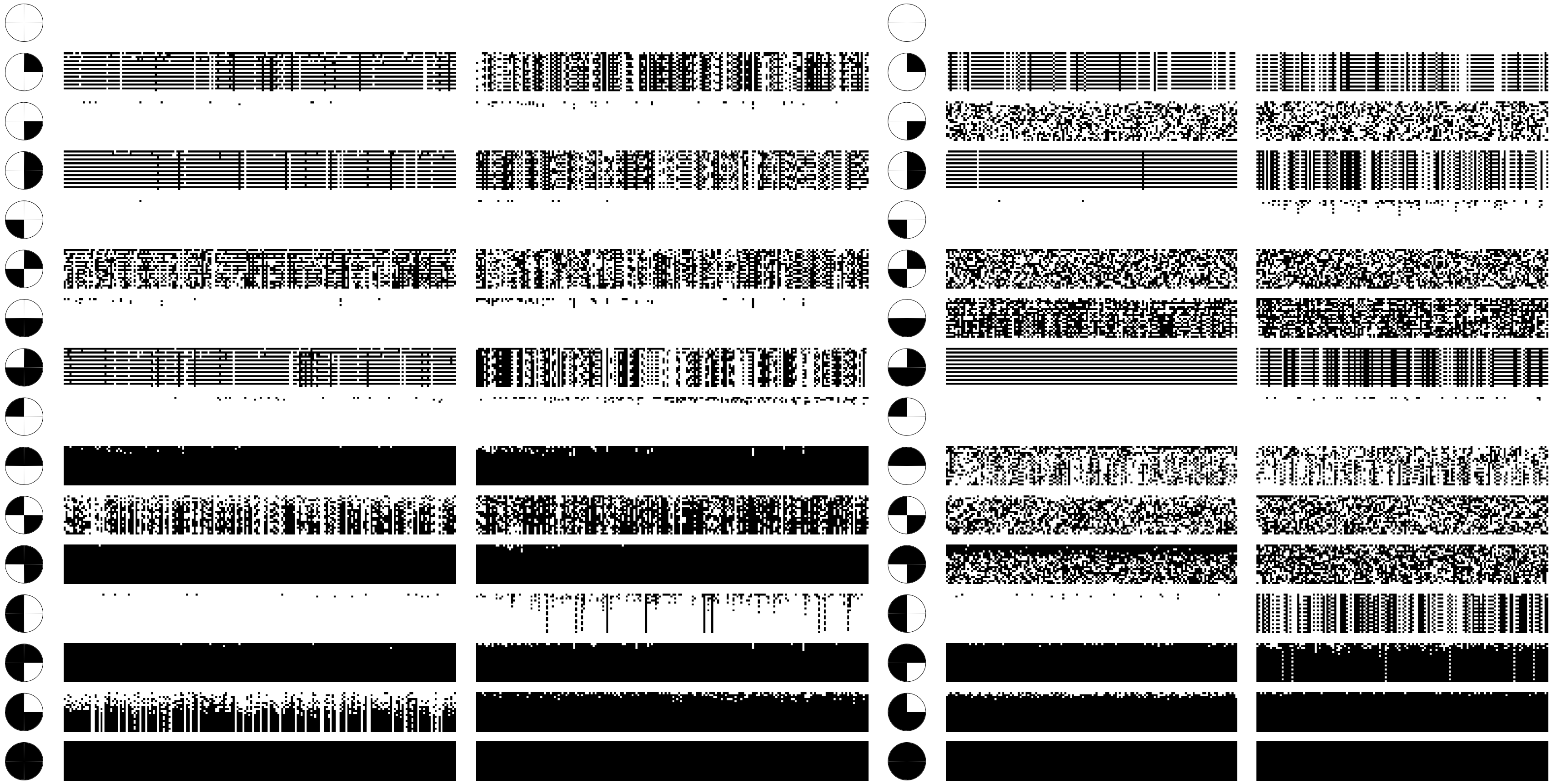




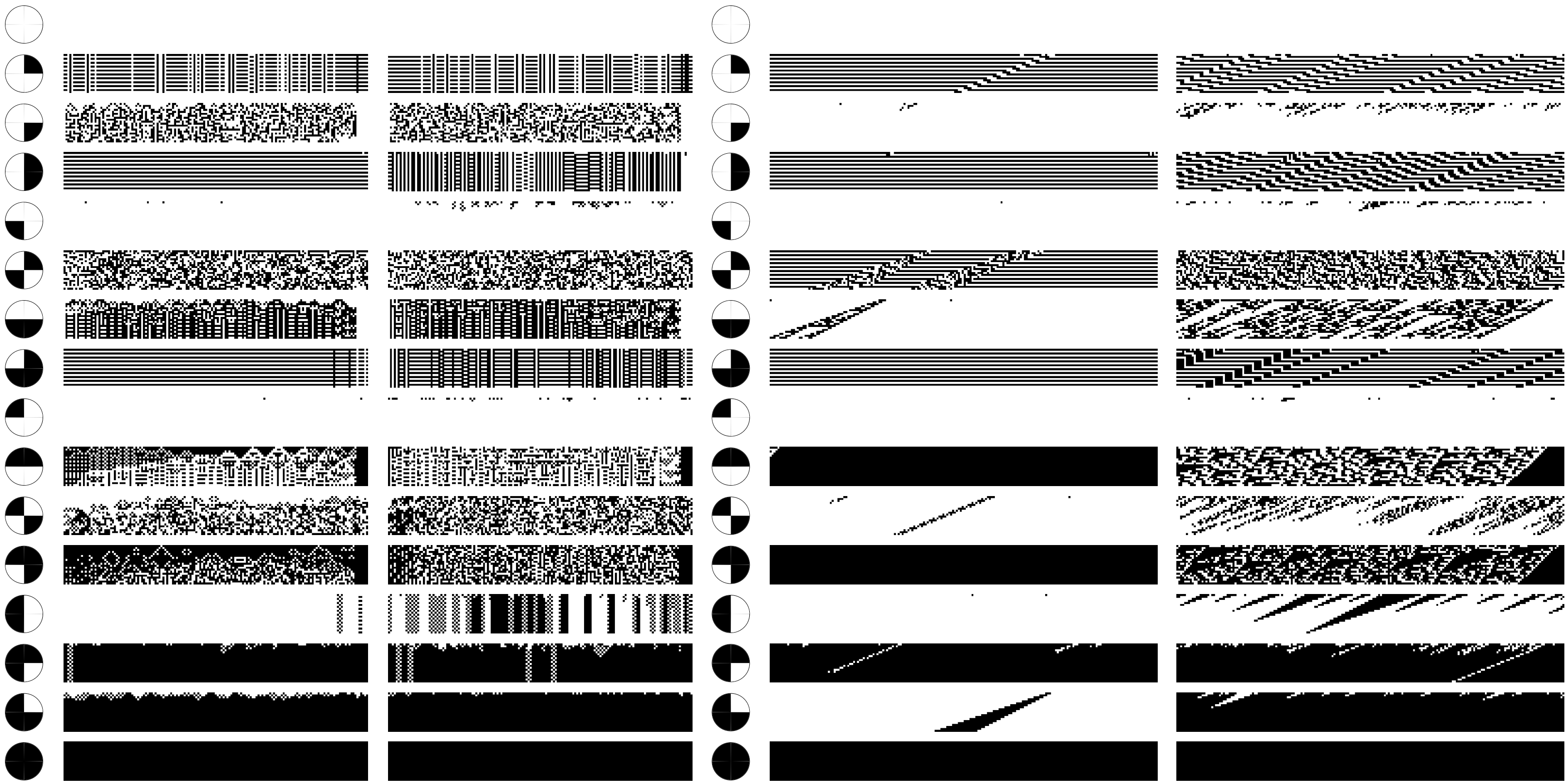
Four sample networks to demonstrate network cellular automata on. The first is an inward—flowing tree with the tips of the roots being fed by nodes a few jumps up. The second is an undirected random trivalent network of 150 nodes. The third is a roughly made hexagonal lattice tube, similar to chicken wire and microtubules. The fourth network was composed from a list of numbers 1 to 200 with a rule designating that any numbermust become a rule no itself plus 1, plus 2, plus 3, plus 4, and for every fifth node a random number between 1 and 200; then finally all nodes with no inbound connections are deleted.







Runs from simple and random initial conditions. The outputs in the left column of each array starts with a black/white cell ratio of 1/10, the right hand ratio is 50/50. Multiple samples help one discern between features of the cellular automaton and artifacts implicit by the network. What better ways to visualizing network state time series exist?



Additive behavior can be seen in netowrk 3 via rules and . Because of network 4's general topology, it seems to emulate one-dimensional cellular automata, including rule 110 via , and what looks like rule 30 via .

What could cellular automata on networks do?