1. The “neighborhood” in a cellular automata model defines the geometry of the space. In class, we discussed 4 different kinds of neighborhoods for 2-dimensional cellular automata -- the 8-neighbor Moore, the 6-neighbor triangular, the 4-neighbor von Neumann, and the 3-neighbor hexagon.

(a) Let \( N_k(x, y) \) be the set of neighbors of site \((x, y)\) where \(x\) and \(y\) are integers. If \(k = 3\), then
\[
N_3(x, y) = \begin{cases} 
\{(x + 1, y), (x, y + 1), (x, y - 1)\} & \text{if } x + y \text{ is even}, \\
\{(x - 1, y), (x, y + 1), (x, y - 1)\} & \text{if } x + y \text{ is odd}.
\end{cases}
\]

Find similar formulas for \(N_4(x, y), N_6(x, y)\), and \(N_8(x, y)\).

(b) In a regular lattice, the atomic loop length is the smallest number of neighboring nodes in loop from \((0,0)\) back to itself where the same edge between two neighbors is never traversed more than once. Each such loop is called an “atomic loop”. For each \(k \in \{3, 4, 6, 8\}\), find the atomic loop length and the number of atomic loops containing \((0,0)\) for lattices with neighborhoods \(N_k(x, y)\).

(c) Given a neighborhood \(N_k(x, y)\), we can define a matrix \(d_k((x, y), (u, v))\) to measure the distance between points \((x, y)\) and \((u, v)\) recursively as follows:
\[
d_k((x, y), (u, v)) = \begin{cases} 
0 & \text{if } (x, y) = (u, v), \\
1 + \min\{d_k((w, z), (u, v)) : (w, z) \in N_k((x, y))\} & \text{otherwise}.
\end{cases}
\]

For each \(k \in \{3, 4, 6, 8\}\), draw \(\{(u, v) : d_k((0,0), (u, v)) \leq 2\}\).

(d) For each \(k \in \{3, 4, 6, 8\}\), find \(d_k((0,0), (3,3))\).

2. In class, we were able to visually calculate percolation depth because of our awesome pattern analysis wet-ware. Calculating percolation depth algorithmically is a little more complicated, but can be done recursively. See [http://www.math.psu.edu/treluga/450/percolation_example.py](http://www.math.psu.edu/treluga/450/percolation_example.py)

(a) Let \(A(p, N)\) be a random 0/1 matrix with shape \(N \times N\) where entries are 1 with probability \(p\). Recall that in class, we generated example matrices like this with the python code
\[
A = \text{floor}(\text{rand}(N,N) + p)
\]
For each value of \(p \in \{0.3, 0.4, 0.45, 0.5, 0.6\}\), simulate 1000 \(40 \times 40\) matrices. Let \(g(x, p)\) be the fraction of these matrices with percolation depth less than or equal to \(x\). Plot \(g(x, p)\) for each value of \(p\), all in one plot. Remember to label your plot axes and include a legend.

(b) What particular feature of your plot change when \(p\) is between 0.45 and 0.5? What does this change mean?

(c) Suppose we define \(h(p, N)\) as the fraction of \(N \times N\) \(A(p, N)\) matrices which percolate all the way through. Plot \(h(p, N)\) as a function of \(p \in [0.3, 0.6]\) for \(N \in \{5, 10, 20, 50, 100\}\). Use atleast 1000 matrices for each. (Warning: This calculation may take you a long time.)

(d) Extrapolating from your plot, what do you think will happen to \(\lim_{N \to \infty} h(p, N)\) ?

(e) Reality, of course, often three-dimensional, rather than two-dimensional. Describe how you think the percolation threshold will change when we switch from two dimensions to three dimensions and why it will change.

3. Search the web for a new cellular-automata model that we have not already discussed in class.

(a) Specify the rules of the cellular automata in enough detail that we can program it.

(b) Describe the emergent phenomena exhibited by your automata. (waves, particles, solitons, oscillators, interfaces, spirals, freezing, clustering, bursting, ...)

---
due Friday, April 18