GEOG 178/258
Week 9:

Cellular Automaton / MVC

mike johnson

A cellular automaton (pl. cellular automata, abbrev. CA) is a discrete model studied in automata theory. Cellular automata are also called cellular spaces, tessellation automata, homogeneous structures, cellular structures, tessellation structures, and iterative arrays.[2] Cellular automata have found application in various areas, including physics, theoretical biology and microstructure modeling.

A cellular automaton consists of a regular grid of cells, each in one of a finite number of states, such as on and off (in contrast to a coupled map lattice). The grid can be in any finite number of dimensions.

For each cell, a set of cells called its neighborhood is defined relative to the specified cell.

An initial state (time t = 0) is selected by assigning a state for each cell. A new generation is created (advancing t by 1), according to some fixed rule (generally, a mathematical function) that determines the new state of each cell in terms of the current state of the cell and the states of the cells in its neighborhood.

Typically, the rule for updating the state of cells is the same for each cell and does not change over time, and is applied to the whole grid simultaneously though exceptions are known, such as the stochastic cellular automaton and asynchronous cellular automaton.

Wikipedia

1. Setup

   What does the model look like at t = 0

2. Tick

   What happens in each time step?

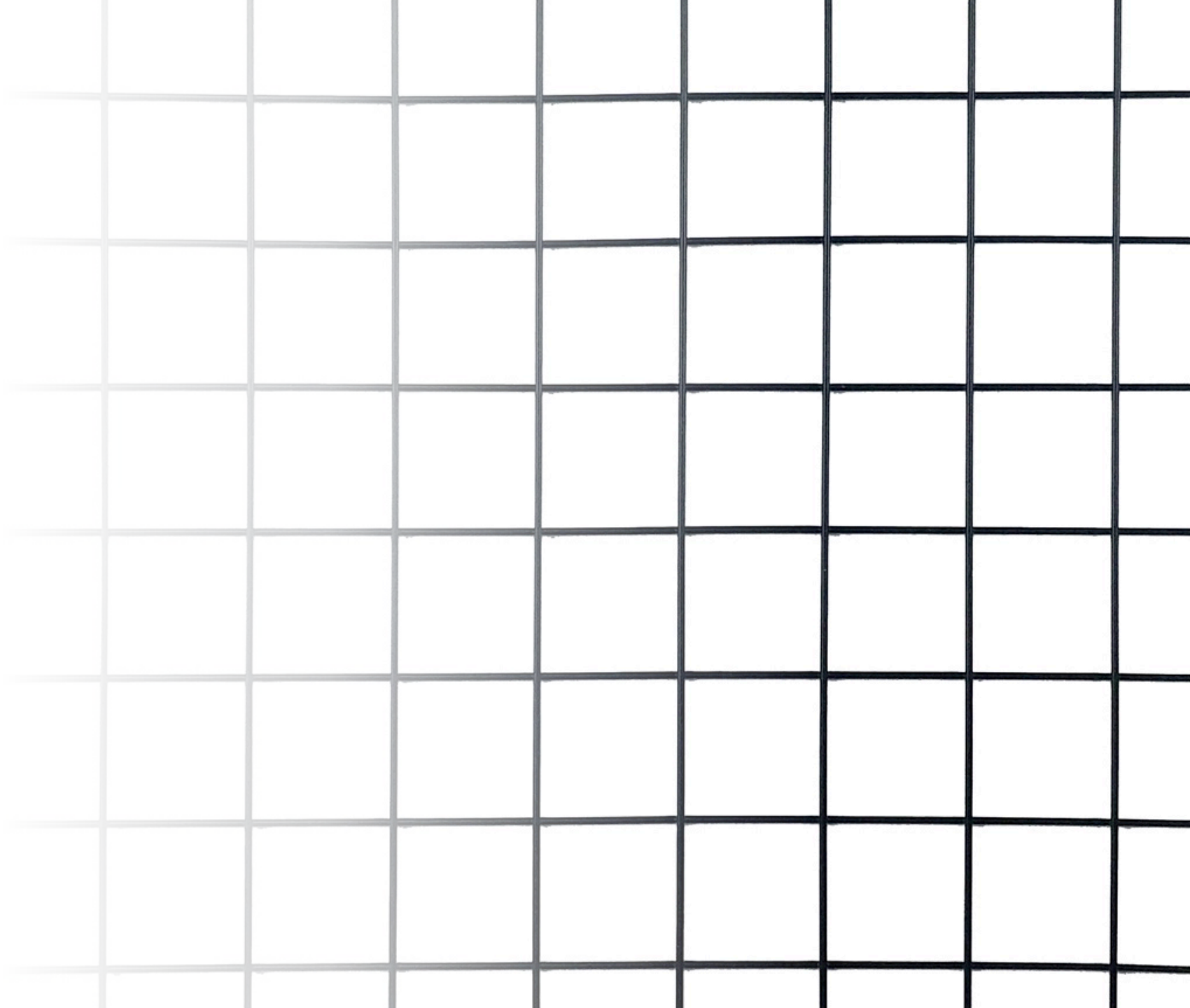
3. Move

   How do agent move, and what rules guide those moves

Setup

- The “model” has an initial state
  - How many cells are in the neighborhood?
  - How many people are in the neighborhood (density or raw count?)?
  - What is the initial state of the people?
  - How are people visualized?

- The SETUP is one of the most difficult parts because in requires a concrete idea of the **model** and the **viewer**.



Tick - timestep

- What happens in each time step?
  - People interact with there *neighbors*.
  - The “life” of each COVID instance decreases

- ”Move” could be a part of “tick” but we will separate for ease

- Remember that CA models require that all actions happen independently of one another and **not** sequentially! This requires the generation of a new “model state” that is computed and then plugged into the viewer.

Move

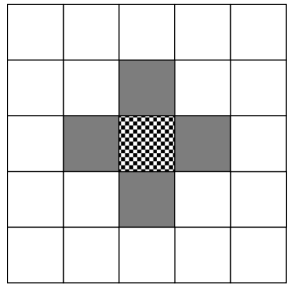
- How should people move? Can they move to any where in the board? Only their neighbors? All 8 neighbors?
- Moving happens either before or after the “tick”
  - E.g. People move and then infect or People infect and then move

Neighbors

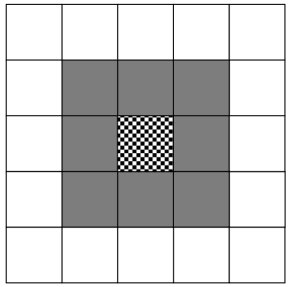
- Neighborhood functions appear in both tick and move

- Neighborhood functions are very tricky on the edge of the board!

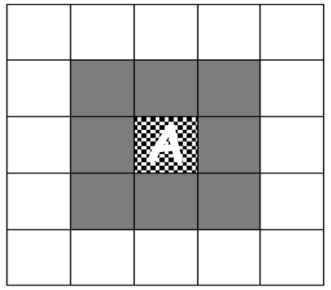
Possible Neighborhoods for 2D CA



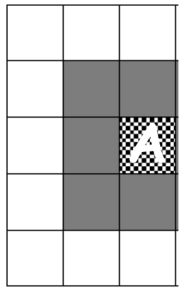
Von Neumann Neighborhood



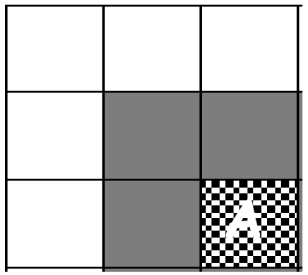
Moore Neighborhood



Interior agent A
up to 8 neighbors



Border agent A
up to 5 neighbors



Corner agent A
up to 3 neighbors

Assignment

- **Finish the SIR cellular automaton** by implementing movement, different rates/chances of infection, and stay-at-home orders (a fraction of the agents will not move).
- Test your code by showing different settings.
- Implement a way to store the state of the model and load it again (from the filesystem) **[GEOG 258 only]**
- Read chapters 10 (**Exception Handling**) and 21 (**Java IO**).
- Upload a zip file [LN1W5.zip] with the *.java files to GauchoSpace.
- Assignments and **executable** programs are due the day before lecture at **5pm PST** of each week.

Homework:

- Use the set up we build today
- Create a move and tick method
- Choose one additional action to build into your model
  - Could be a stay at home order
  - Could be a text box that show number of cases
  - Could be a changed visualization (circles instead of squares)
  - Could be a slider to define transfer rate