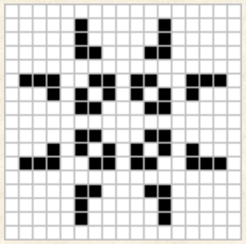
UNIVERSITY of WISCONSIN-MADISON
Computer Sciences Department

CS 202: Introduction to Computation

Professor Andrea Arpaci-Dusseau

How can computation… simulate the future?



Simulations

One of the most popular uses of computation in science and engineering

People invest huge amounts of resources into simulating systems they care about

What types of resources?

- Human time to develop simulation codes
- Many machines to run multiple simulations in parallel
- Many disks to store initial data and final results

Questions you might like to answer

What will the weather be tomorrow? next century?

Which direction will a forest fire move?

How much will stock portfolio be worth next year? at retirement?

How fast can you drive around curve and stay on road?

How quickly will disease spread thru population?

How do you create a simulation?

1. Model capturing characteristics of system
   - Equations or algorithm describing how system behaves
   - Need domain knowledge to construct
   - More accurate, detailed model → more accurate results
2. Initial state of the system
   - What values for variables describe current conditions?
3. Next-step function
   - Given current state, how do you calculate next state at next interval of time? Repeat for many intervals…

Today's Lecture: Two Simulation Examples

1. Game of Life

2. Infectious Disease

Simulation Case Study 1: Cellular Automata

1970s: John Conway introduced Game of Life

Are there simple rules describing a system that lead to complex behavior?

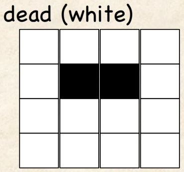
- Self-replicate? Self-organize? Evolve?
- Interesting to biologists, economists, mathematicians, physicists, philosophers

Can "design" and "organization" occur spontaneously without planning?

Rules for Game of Life

Infinite grid of cells, each is live (black) or dead (white)

- Wrap 2-grid around at edges



Simulate time steps to create generations

- Every cell interacts with its 8 neighbors

Apply next-step function to each cell:

- Live cell with < 2 live neighbors dies (loneliness?)
- Live cell with > 3 live neighbors dies (overcrowding?)
- Live cell with 2 or 3 live neighbors lives
- Dead cell with 3 live neighbors becomes live (birth)

Initial pattern constitutes seed of system

Next generation:

- Apply rules simultaneously to all cells in previous
- Births and deaths happen simultaneously

Example Seed 1

If (cell is alive)

If < 2 live neighbors, then dies

If > 3 live neighbors, then dies

If 2 or 3 live neighbors, then ok

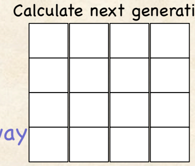
If (cell is dead)

if 3 live neighbors, then live

Count live neighbors

1	2	2	1
1	1	1	1
1	2	2	1
0	0	0	0

Calculate next generation

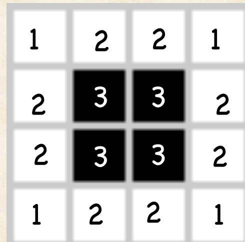


Lots of different initial seeds die away

Example Seed 2

If (cell is alive)
- If < 2 live neighbors, then dies
- If > 3 live neighbors, then dies
- If 2 or 3 live neighbors, then ok

If (cell is dead)
- if 3 live neighbors, then live

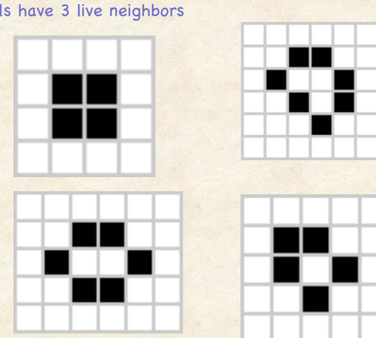


Can find many initial seeds that stay the same!
How would you describe seeds that stay same?

All live cells have 2 or 3 live neighbors
No dead cells have 3 live neighbors

Game of Life:
Stable, Constant Populations

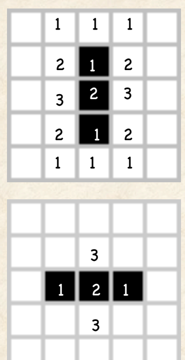
All live cells have 2 or 3 live neighbors
No dead cells have 3 live neighbors



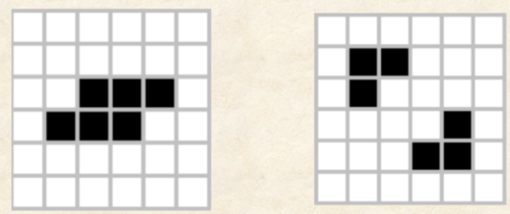
Example Seed 3

If (cell is alive)
- If < 2 live neighbors, then dies
- If > 3 live neighbors, then dies
- If 2 or 3 live neighbors, then ok

If (cell is dead)
- if 3 live neighbors, then live

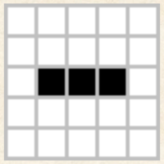


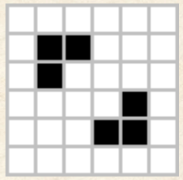
Oscillating Seeds:
Pattern Repeats in Cycles



Oscillating Seeds: Pattern Repeats in Cycles

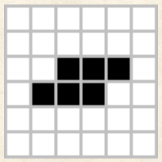
Period = Two

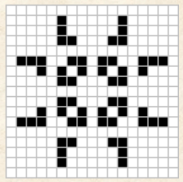




Period = Two

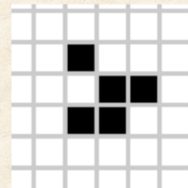
Period = Two

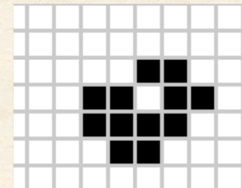




Period = Three

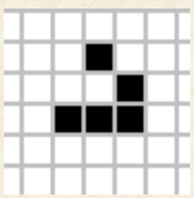
Spaceship Seeds

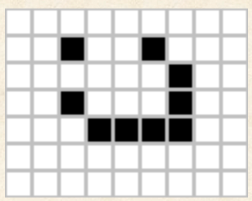




Spaceships continuously move across grid
while changing pattern in cycles

Spaceship Seeds





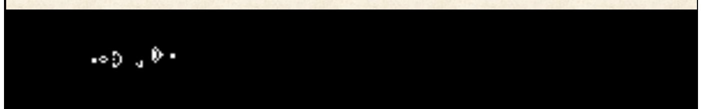
Spaceships continuously move across grid
while changing pattern in cycles

Glider Seeds

Generates "gliders" forever...
Can grow infinitely...

Glider Seeds

Generates "gliders" forever…
Can grow infinitely…



Summary of Game of Life

Surprisingly simple rules can lead to arbitrarily complex systems

- Simple computation + interesting data (state) leads to interesting results
- Very hard to predict how system will behave
- Easiest to run "game" and see how it turns out

Fun for mathematicians

- Create smallest glider seed or longest repeating cycle and prove it!

Simulation Case Study 2: Avoiding Disease

Questions to answer with simulation

- Will the whole population eventually become ill?
- Will disease die out or continuously cycle?
- How do parameters affect answers?

Model of how some disease spreads:

- Population of people who are sick or well or immune
- When person becomes sick
  - Remain sick for "infectious" number of days
  - While sick, probability "prob" infect each of 4 neighbors
  - After recover, immune for "immune" days
  - Example: Infectious = 3, immune = 2, probability = 0.25

How to Represent Data?

How to represent Population of N people?

- List of N items: Each element corresponds to 1 person

What should values of List be?

- Value at Index j corresponds to person j
  - Represents how many days sick or immune
    - Positive integers: sick
    - Negative integers: immune
- List: 0 0 0 0 0 0 0 0 0 0
  - 10 people : Everyone is healthy (but not immune either)
- List: 0 0 0 0 1 0 0 0 0 0
  - Person 5 is sick for the first day
- List: 0 1 2 2 3 2 0 2 2 1
  - P2 sick for 1^{st} day, P3 and P4 sick for 2^{nd} day in a row
- List: 0 -1 -2 2 4 3 -1 2 1 0
  - Person 2 immune 1 day, person 3 for 2 days…

How to Model Time-Step (1 Day?)

Each day, create new list based on current list

Repeat for each item of current list:

- If sick:
  - Infect others for next day
    - Random chance of infecting each of 4 neighbors
      - Can't infect them if already sick or immune
      - Set next state of neighbor to sick for 1 day
  - Increment number of days sick
    - If last infectious day, become healthy (and immune…)
- If immune:
  - Increment number of days immune
  - If last immune day, set back to 0

Pencil-Paper Simulation

Infectious=3, Immune=2

	1	2	3	4	5	6	7	8	9	10
# 5 is infected										
5 infects 4										
4->2, 5 -> 7										
2->1, 4-5, 5-4, 7->6										
1->3, … 6->8, 7-5										
7->9, 8->10										

Each column represents 1 of 10 people
Each row represents a different day

Observations from Simulation

All infections must occur simultaneously and instantaneously

- Example: If person 3 infects person 4 on day 2, person 4 can't infect someone else on day 2

Must separate current day from next day

Implication: Must have multiple lists

- List for every day?
  - Wasteful, painful to code
- Two lists:
  - Current day: Use this for seeing who is sick or immune today
  - Next day: Set this list set state for tomorrow
  - Copy Next Day list to Current day when done

Code for Simulation

Code for Simulation

Today's Summary

Simulation

- Important in many domains
- Simple rules can lead to complex behavior
- Initial state of system (input data) has strong impact on results!

Announcements

- Homework 7 due by 5pm Today
- Homework 8 available today
  - Design and Implement Trivia Game in Scratch
- Next week
  - Keeping data secret, very difficult problems, exam 2 review