Simulations of the lattice

Jyoti - 277156

January 2024

1 Introduction

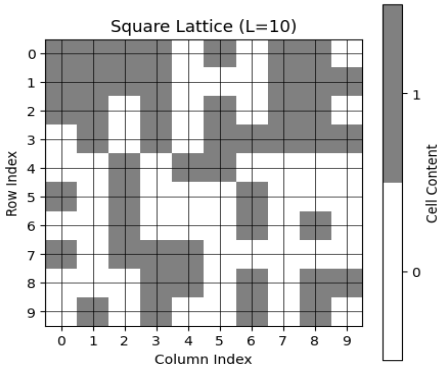
This report details the implementation and exploration of simulations involving a square lattice with dogs and fleas, along with the implementation of a cellular automaton. The simulations are conducted using Python programming language, primarily utilizing array manipulations and random number generation.

2 Task 1: Square Lattice Simulation

Implementation: A square lattice of size $L \times L$ is represented as an array $A[i, j]$, where each cell can either be occupied by a dog ('1') with probability p or empty ('0') with probability $1 - p$.

Procedure:

1. Load parameters p and L from the initialization file `ini.txt`.
2. Generate the lattice array based on the specified probabilities.
3. Save the lattice configuration to a file.
4. Visualize the lattice with dogs represented by gray color and empty cells by white color.



3 Task 2: Flea Movement Simulation

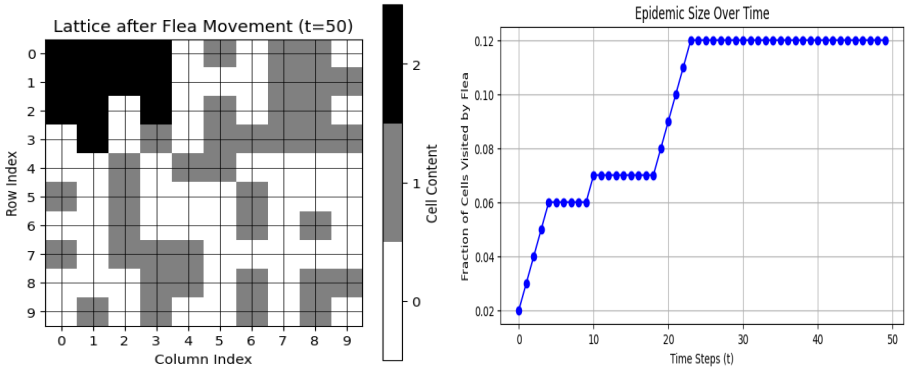
Implementation: A flea is placed on the very left dog in the first row of the lattice and can randomly jump to one of the four adjacent cells occupied by dogs. The flea's path is marked with '2', indicating visited cells.

Chosen Implementation:

- The flea moves randomly to one of the adjacent cells with equal probability.
- If the chosen cell is occupied by a dog, the flea moves to that cell and marks it with '2'. Otherwise, it stays in its current position.

Procedure:

1. Initialize the lattice with dogs and place the flea.
2. Perform flea movements for a specified time t.
3. Save the updated lattice configuration to a file.
4. Visualize the lattice with dogs represented by gray color, empty cells by white color, and the flea by black color.



Exploration:

- Varying parameters p and L to observe their effects on flea movement.
- Implementing simple animation to visualize flea movements over time.

4 Task 3: Cellular Automaton Implementation

Choice of Cellular Automaton: The Game of Life, a classic cellular automaton devised by John Conway.

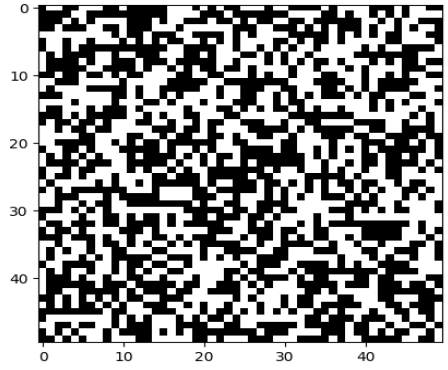
Implementation:

- Represent the grid as a 2D array where each cell is either alive ('1') or dead ('0').

- Apply the rules of the Game of Life to update the grid for each generation.

Procedure:

1. Initialize the grid with random initial configuration.
2. Apply the rules iteratively to generate subsequent generations.
3. Save each generation's configuration to visualize the evolution of the system.



5 Conclusion

Through these simulations, we gain insights into the behavior of complex systems governed by simple rules. By exploring different parameter values and visualizing the dynamics, we can observe emergent patterns and behaviors, providing a deeper understanding of the underlying processes. Additionally, the implementation of cellular automata like the Game of Life showcases the versatility and applicability of such models in studying various phenomena in diverse fields.