Hexagonal Game of Life

DEFNE DEMİR, Akdeniz University, Türkiye
HASAN SAYGILI, Akdeniz University, Türkiye
SEMİH AVCI, Akdeniz University, Türkiye
ZAYİMHAN KORKMAZ, Akdeniz University, Türkiye
ZEYNEP TANRIVERMİŞ, Akdeniz University, Türkiye

This research aims to discover hexagonal Game Of Life's unique approach to cellular automata, by different rules and outcomes. Based on our hexagonal GOF simulation code it is experimented that how different environments would react in varying conditions and how would they benefit us.

ACM Reference Format:
DEFNE DEMİR, HASAN SAYGILI, SEMİH AVCI, ZAYİMHAN KORKMAZ, and ZEYNEP TANRIVERMİŞ. 2024. Hexagonal Game of Life. 1, 1 (December 2024), 14 pages. https://doi.org/XXXXXXX.XXXXXXX

1 Introduction

Conway's Game of Life [2], or simply Life, is a well-known example of a cellular automaton— . Traditionally it is played on a 2D square grid with set of rules. It is considered a zero-player game since it only needs an initial state to be given. After the initial state, game will determine the next step, and steps after according to the predetermined rules.

Hexagonal Game of Life is one of the different versions of the Conway's game. Even though rules differ only slightly, there's important points we need to look at. In this research, we compared these two games, ideas behind them and inspected hexagonal Game of Life.

2 Background

2.1 How It Started?

John Conway was inspired by a philosophical idea initially proposed by John von Neumann, who tried to integrate the biological reproduction idea to a hypothetical machine. With this idea humanity could simulate how life would occur like in different settings. Working on this, Conway actually created a model for a such machine on a rectangular grid.

The first prototypes of the game focused on finding the perfect rule set that would allow growing without limits Adding onto that, he tried to find "simple initial patterns" before settling into repeating loops. Conway late wrote the basic motivation of Life was to create a "universal" cellular automaton. Could reproduction be simulated in computer environment? Could we make an ecosystem by giving the initial position ourselves? John Conway researched these questions. He successfully created the project, too.

Authors' Contact Information: DEFNE DEMİR, Akdeniz University, Antalya, Türkiye; HASAN SAYGILI, Akdeniz University, Antalya, Türkiye; SEMİH AVCI, Akdeniz University, Antalya, Türkiye; ZAYİMHAN KORKMAZ, Akdeniz University, Antalya, Türkiye; ZEYNEP TANRIVERMİŞ, Akdeniz University, Antalya, Türkiye.

ACM XXXX-XXXX/2024/12-ART
https://doi.org/XXXXXXX.XXXXXXX

After the publication of the game in October 1970 in issue of *Scientific American*, it gained a huge interest because of the infinitely different things the game could evolve into. People found different oscillators—patterns that repeats itself after a certain amount of generations— and still lives—patterns that do not change.

2.2 Ideas Behind

Cellular automata [1] are used to model systems where both space and time are discrete, and where certain states evolve in a stepwise manner. Discrete mathematical methods help us define the rules governing the state transitions of these cells and understand the evolution of the entire system [7]. In this context, hexagons' similarity to nature and symmetry lets us simulate the system more realistically.

3 Methodology

To inspect Hexagonal Game of Life, we need to understand Original Game of Life. The rules are given as follows:

- A cell could have one of the two possible states: dead or alive.
- Cells have "neighbors" which are the other cells that are horizontally, vertically, or diagonally adjacent.
- If a cell has fewer than two alive neighbors, the cell dies, using the idea of underpopulation.
- If a cell has two or three alive neighbors, the cell lives onto the next generation.
- If a cell has more than three alive neighbors, the cell dies, using the idea of overpopulation.
- If a dead cell has exactly three neighbors alive, the cell becomes alive, as if by reproduction.

Hexagonal Game of Life's rules differ slightly, since each hexagonal cell has 6 neighbors while squares have 8. The rules for Hexagonal version are given as follows:

- A cell could have one of the two possible states: dead or alive.
- Cells have "neighbors" which are other cells who shares a side.
- If a cell has fewer than two alive neighbors, the cell dies, using the idea of underpopulation.
- If an alive cell has two or three alive neighbors, the cell lives onto the next generation.
- If a cell has more than two alive neighbors, the cell dies, using the idea of overpopulation.
- If a dead cell has exactly two neighbors alive, the cell becomes alive, as if by reproduction.

Rules seems reasonable and understandable for hexagonal grid. Yet it wasn't that easy to actually make these rules. Since every time we change our polygon , we have to initiate new rules for it. Furthermore, we need to follow another set of rules for a valid game of life rule. These rules are:

- When counting the neighbors of a cell, all touching neighbors are considered the same.
- At least one glider(oscillator that does movement across the grid.) exists.
- Start with a finite wrapped universe that is completely filled with a random pattern. Then after a finite number of generations, all such patterns eventually must either disappear, or decompose into one or more oscillators. Rules exhibiting this property are said to be stable

Hexagonal Game Of Life is checked whether it's accurate or not by these standarts. Hexagons have their neighbors from their every sides, securing the first rule.

According to these rules, game was experimented on Python. Libraries used are: numpy for managing the grid data, math for performing necessary calculations and pygame for the front-end application and visualisation. Main idea is by starting with an initial pattern and letting the pattern evolve with the predetermined rules.

First of all, we initialize front-end parameters like size, colors, button settings. After that we draw the hexagon using pygame and compute corners of a hexagon using math library. Simulation also has a next step button and start button. These are necessary and very basic methods we will frequently use but we would like to talk more about the Game Of Life algorithm itself, so we will not discuss the basic aspects in detail.

Listing 1. Main Loop Of The Algorithm

```
while running:
screen.fill(BACKGROUND_COLOR)
draw_grid()
draw_buttons(running_simulation)
pygame.display.flip()

for event in pygame.event.get():
    if event.type == pygame.QUIT:
        running = False
    elif event.type == pygame.MOUSEBUTTONDOWN:
        mx, my = pygame.mouse.get_pos()
        if start_button_rect.collidepoint(mx, my):
            running_simulation = not running_simulation
        elif next_button_rect.collidepoint(mx, my):
            show_next_step = True
        elif not running_simulation:
            select_hex(mx, my)
if running_simulation or show_next_step:
    grid = update_grid()
    show_next_step = False
```

This loop effectively manages both continuous simulation updates and user-controlled actions, allowing for real-time visualization and interaction with the hexagonal grid structure. Even if there's no user interference during the simulation, cells will continue to evolve with game rules. Following method aims to update the grid and achieves it.

Listing 2. Method that updates the grid every step

```
1  def update_grid():
2  new_grid = np.copy(grid)
3  for row in range(rows):
4      for col in range(cols):
5          live_neighbors = sum(grid[nr][nc] for nr, nc in get_neighbors(row, col))
6          if grid[row][col] == 1:  # alive cell
7              if live_neighbors !=2 :  # over/nder population
8                  new_grid[row][col] = 0
9              else:  # survival
10                 new_grid[row][col] = 1
11         elif grid[row][col] == 0:  # dead cell
12             if live_neighbors == 2:  # reproduction
13                 new_grid[row][col] = 1
14 return new_grid
15
```

This method is straightforward with checking the rules, then updating the cell. Important thing to note here is how we get neighbors in the fifth line.

Listing 3. Method that gets neighbors of a cell

```
def get_neighbors(r, c):
offsets = [(-1, 0), (0, -1), (0, +1), (+1, 0)]
if c % 2 == 0:
    offsets += [(-1, +1), (-1, -1)]
else:
    offsets += [(+1, +1), (+1, -1)]
neighbors = [(r + dr, c + dc) for dr, dc in offsets
             if 0 <= r + dr < rows and 0 <= c + dc < cols]
return neighbors
```

Note that offset matrix represents the hexagonal grid, since normal matrix represents square grid. The difference is crucial because of the neighboring relations, where normal matrix also counts the corner cells as neighbor. Thanks to how offset matrix adjusts hexagonal grid's symmetry accurately, we are able to calculate a cell's neighbors.

4 Experiments

4.1 Objective of the Experiment

The main goal of this experiment is to analyze the evolutionary behavior of various initial patterns in the Hexagonal Game of Life on a hexagonal grid and to identify the observed differences compared to the traditional square grid structure. Focusing on particularly interesting initial patterns (e.g. static life, spaceships, oscillation, methuselah or infinite growth patterns), we will examine how these patterns evolve under certain rules and the patterns they produce. We will also examine how the same patterns lead to different outcomes when the rule is changed.

4.2 Scope of the Experiment

This experiment focuses on analyzing the evolutionary dynamics of the Game of Life on a hexagonal grid. The study encompasses different initial configurations, long-term simulations with random distributions, and the effects of various rules on the grid. The main scope of the experiment is detailed below:

4.2.1 Examined Initial Configurations.

- The evolutionary behaviors of classic and intriguing initial patterns on the hexagonal grid will be analyzed.[3] [4]
  - Still Life: Patterns exhibiting stationary characteristics will be examined.
  - Oscillators: The periods of periodically repeating patterns will be measured, and their behaviors under environmental influences will be observed.
  - Spaceships: Patterns that move in a specific direction will be analyzed for their direction, speed, and stability.
  - Methuselahs: Patterns that display complex evolution over an extended period from a sparse initial configuration will be studied.
  - Infinite Growth Patterns: The behaviors of theoretically infinite-growing structures.

4.2.2 Effects of Rule Modifications.

- The results on the grid will be studied by modifying Conway's classic rules:
  - Survival Condition: The impact of having fewer or more neighboring cells on survival
  - Reproduction Condition: The outcomes of varying neighbor counts for reproduction (e.g., requiring 2-4 neighbors instead of 3).

- Changes in Active/Inactive Ratios: Observations of how different rules affect the ratio of active to inactive cells.

Through this scope, both classic patterns and new features unique to the hexagonal grid will be explored, and the resulting differences will be thoroughly analyzed.

4.3 Experimental Setup

4.3.1 *Simulation Environment.*

- The simulation is implemented using Python. The rules of Conway's Game of Life are adapted for use with a hexagonal grid.
- Libraries:
  - NumPy: Used for processing grid data and performing mathematical operations.
  - Pygame: Used for dynamic visualization, user interaction, and simulation control.

4.3.2 *Hexagonal Grid Structure.*

- The hexagonal grid is represented as a 2D matrix in Python. Each cell stores its state (0 or 1).
  - The neighborhood is coded according to the hexagonal structure. Each cell has six neighbors, and the neighborhoods are computed with column-parity checking. Different neighborhood calculation methods are used for even and odd columns.
- Initial Configurations
  - Classic Game of Life patterns, such as Glider and Blinker, were adapted for the hexagonal grid and placed on the grid.
  - The grid was also initialized with randomly populated cells based on specific random density levels.

4.3.3 *Rule Implementation.*

- Original Conway Rules (Hexagonal Adaptation):
  - Survival: A cell remains alive (1) if it has exactly 3 to 4 neighbors.
  - Reproduction: A dead cell (0) becomes alive if it has exactly 2 neighbors.
  - Death: A cell dies if it has fewer than 3 or more than 4 living neighbors.
- Rule Modifications:
  - Experiments included testing variations of survival and reproduction rules.

4.3.4 *Simulation Control and Visualization.*

- User Interaction:
  - Users can alter the state of cells by clicking on the grid. The simulation can be controlled using "Start/Stop," "Next Step," and "Random" buttons.
- Visualization:
  - Cells are visualized in yellow when alive (1) and black when dead (0).
  - The hexagonal grid structure is dynamically drawn on the screen.

4.4 Data Collection

During simulations conducted on the hexagonal grid, data was systematically collected and analyzed throughout the experiments. The data collection aimed to understand the evolutionary characteristics of initial configurations and the long-term effects on random distributions. The following metrics and methods were utilized:

4.4.1 *Collected Data.*

- Number of Live Cells (Burning Cells):

  - The total number of active cells on the grid was recorded at the start of the program and each time the simulation was paused. This metric was particularly used to determine whether patterns reached a stable state, exhibited periodic behavior, or continued evolving.
- Pattern Characteristics:
  - Information about the nature of initial configurations, such as whether they were static, oscillating, moving, or explosive, was recorded. Additional notes were taken to identify new and intriguing patterns specific to the hexagonal grid.
- Stabilization Time for Patterns:
  - The number of iterations required for initial configurations to reach a stable state, a periodic loop, or a completely empty grid was recorded.

4.4.2 *Data Collection Methods.*

- Recording Each Iteration:
  - During the simulation, real-time data about the grid's state at each iteration was collected. This data, reflecting the burned/extinguished states of cells over time, was saved to a file for further analysis.

Listing 4. Printing The Grid Data

```
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            running = False
        elif event.type == pygame.MOUSEBUTTONDOWN:
            mx, my = pygame.mouse.get_pos()
            if start_button_rect.collidepoint(mx, my):
                running_simulation = not running_simulation
                if running_simulation:
                    print(f"Beginning of simulation  - living grid count: {count_alive_cells()}")

```

- Visualization of Patterns:
  - The state of the grid at specific iterations during the evolution of patterns was recorded and visual outputs were generated.

4.4.3 *Data Collected Under Different Rules.*

- The effects of survival and reproduction rule changes on the ratio of live to dead cells.
  - The effects of survival and reproduction rule changes on the ratio of live to dead cells.
  - The stability or growth dynamics of patterns under modified rules.

This data collection process enabled a detailed and comprehensive analysis of Game of Life simulations on a hexagonal grid.

4.5 Result of the Experiment

The evolutionary behaviors of initial configurations and random distributions were thoroughly analyzed in the Game of Life simulations conducted on the hexagonal grid. The results revealed distinctive features of the hexagonal structure compared to the traditional square grid and introduced new observations. The main findings are summarized below:

4.5.1 *Behavior of Initial Configurations.*

- Still Life Patterns
  - No stationary patterns were observed on the hexagonal grid.
  - Adaptations of classical square grid patterns, such as "Block" and "Beehive," failed to remain static on the hexagonal grid.

– Due to the rule requiring a cell to have 3 or 4 live neighbors to survive, and the nature of hexagonal structures where cells often have fewer live neighbors, patterns that are stable on the square grid became unstable. For instance, the "Block" pattern on the square grid transformed into an oscillator on the hexagonal grid.

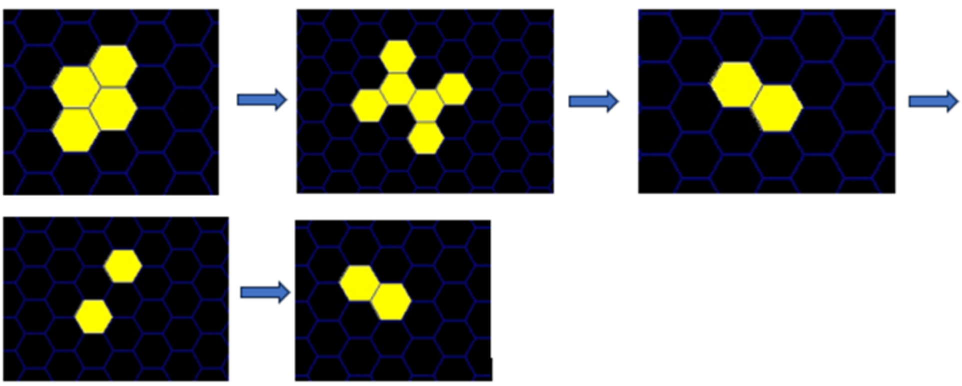


Fig. 1. Behavior of "Block" pattern on Hexagonal Grid

– However oscillators can support alive cell groups and keeping them alive. This pattern was named semi-still life because of the need for oscillators.

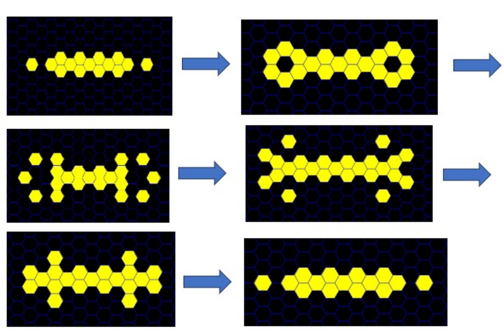


Fig. 2. The pattern which has oscillators on two sides has static pattern on the middle

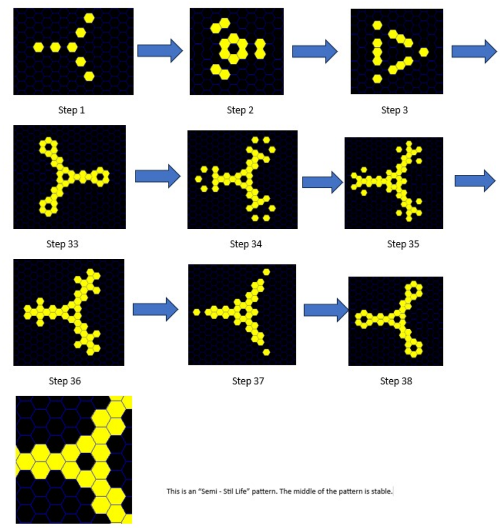


Fig. 3. The pattern which has oscillators on three sides has static pattern on the middle after couples steps

- Oscillator Patterns
  - Oscillators like "Blinker" displayed similar periodic behavior on the hexagonal grid. However, a significant difference in the period length was observed between the square and hexagonal grids.

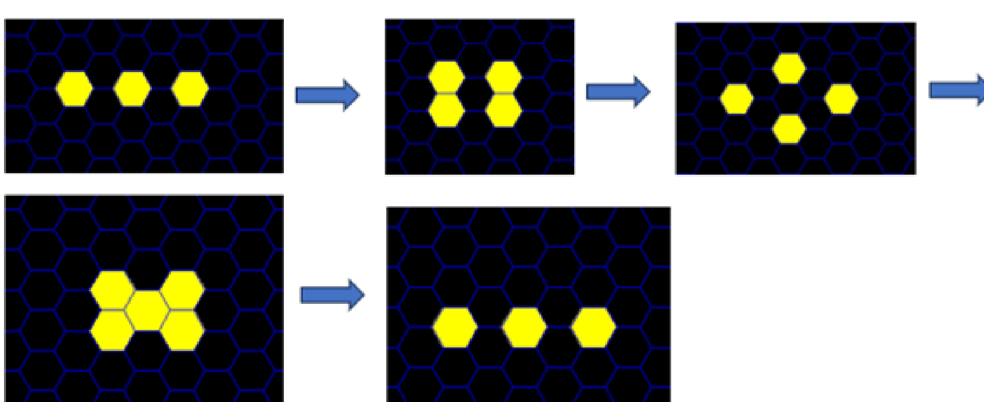


Fig. 4. Behavior of "Blinker" pattern on Hexagonal Grid

  - Unique oscillators specific to the hexagonal grid were identified, characterized by their extended periods.

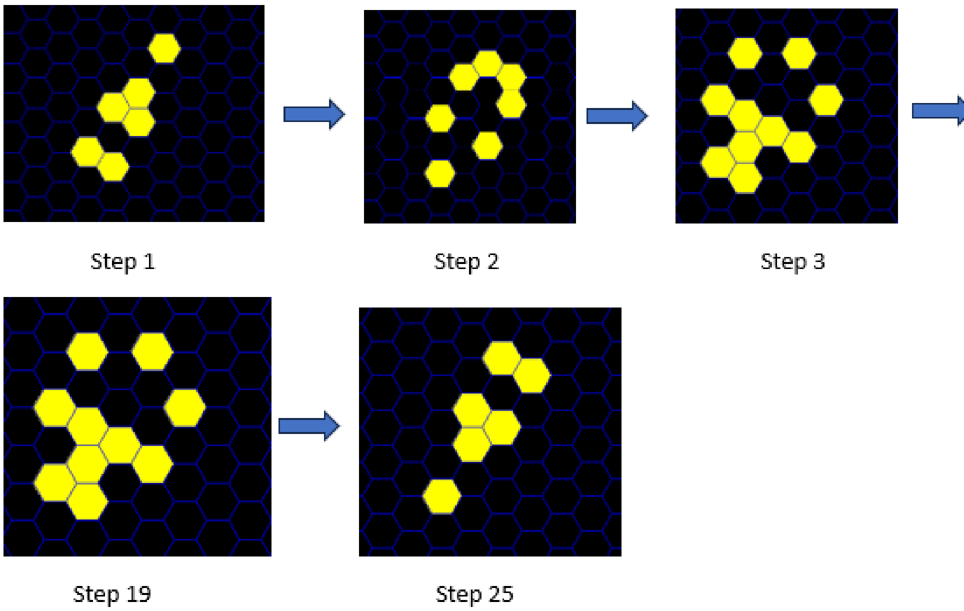
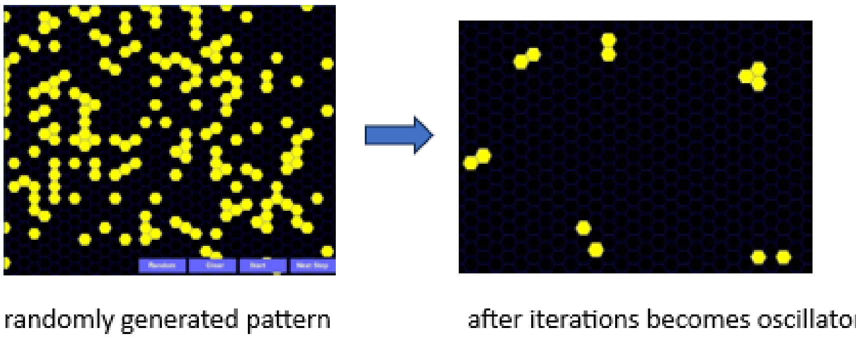


Fig. 5. Unique oscillators specific to the hexagonal grid

- Spaceship Patterns
  - Because of the rule structure of Hexagonal Game of Life, there has no spaceships, in other words gliders, found yet. Even with our experiments using different rules, we unfortunately could not find any.
- Methuselah Patterns
  - Martin Gardner defined methuselahs as patterns of fewer than ten cells that take longer than 50 generations to stabilize.[5] That is greatly opposed to hexagonal grid's nature, which barely has stable patterns. Therefore a methuselah pattern could not be found yet.
- General Observations
  - Most random patterns evolved into oscillators after a certain number of iterations under the applied rules.
  - Experiments with various randomly distributed patterns confirmed that the number of live cells decreased over time, with most patterns eventually stabilizing as oscillators.




```
Beginning of simulation  - living grid count: 164
Beginning of simulation  - living grid count: 43
Beginning of simulation  - living grid count: 40
Beginning of simulation  - living grid count: 40
Beginning of simulation  - living grid count: 41
```

```
Beginning of simulation  - living grid count: 19
Beginning of simulation  - living grid count: 17
Beginning of simulation  - living grid count: 15
Beginning of simulation  - living grid count: 16
Beginning of simulation  - living grid count: 16
Beginning of simulation  - living grid count: 15
```

Fig. 6. Living grids on a random pattern and oscillator result

4.5.2 *Effects of Rule Modifications.*

- Rule modifications were implemented as follows:
  - o A dead cell becomes alive if it has exactly 2 live neighbors.
  - A live cell survives if it has exactly 2 live neighbors.
  - Cells with fewer than 2 or more than 2 live neighbors die.
- Observations with these modified rules included:
  - No stationary patterns emerged under the new rules.
  - The "Blinker" pattern, previously behaving as a simple oscillator, evolved into a more complex structure.
  - The "Spaceship" pattern, which originally displayed a 5-iteration oscillatory behavior, also developed into a more complex formation.

4.6 General Findings

- The hexagonal grid demonstrated a structure that grows more rapidly and produces more chaotic patterns compared to the traditional square grid.
- It was observed that stationary patterns exhibit less diversity, but oscillating and moving patterns acquire unique characteristics specific to the hexagonal grid.
- Different initial configurations and rules significantly diversified the evolutionary outcomes of the hexagonal grid compared to the classical Conway rules.

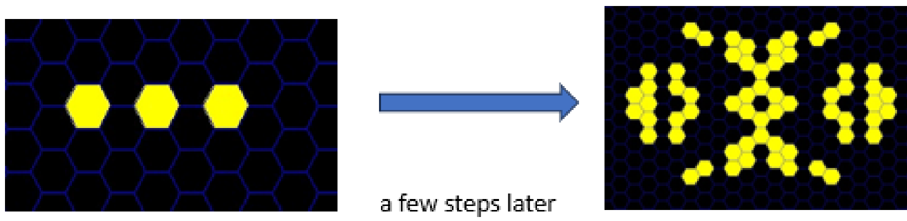


Fig. 7. Behavior of "Blinker" pattern on Hexagonal Grid when rules changed

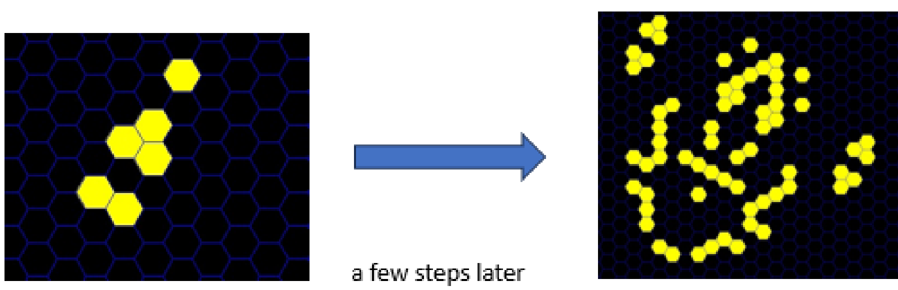


Fig. 8. Behavior of "Spaceship" pattern on Hexagonal Grid when rules changed

These findings provide critical insights into how the Game of Life diverges on a hexagonal grid and serve as a valuable guide for future research.

5 Discussions

5.1 Mathematical Implications of Neighbor Relationships

The neighbor relationships in hexagonal grids enhance the precision of system evolution. Each cell is equidistant from its six neighbors, ensuring a more uniform application of neighbor-based rules compared to square grids. This characteristic enables a more balanced progression of cellular interactions, particularly in multi-generational simulations, as the influence of each neighbor is evenly distributed.

5.2 Why Hexagons?

Hexagons can be found in many places in nature from beehives to snowflakes. These structures stand out in terms of durability and energy efficiency. They provide a more realistic representation in complex modeling such as chemical reactions, where square grids are inadequate [6]. Since each cell's neighbors are equidistant, hexagonal systems provide a more balanced and symmetrical modeling.

One of the most striking features of hexagonal grids is that they do not produce "still life" (static patterns). This feature makes the system dynamic and open to constant change. However, hexagonal grids can be more difficult and complex to program than square grids. Despite all its difficulties, the unique advantages it offers makes his model ideal for implementation.

5.3 Comparative Analysis

In this section, we compared the performance of hexagonal and classical square grid systems, focusing specifically on two main questions:

5.3.1 Which system stabilizes faster ?

Experiments have shown that the classical square grid stabilizes faster than the hexagonal grid. This is because the fixed patterns (still life) are more common in the square grid. In the hexagonal grid, the fixed patterns do not occur, causing the system to remain dynamic for a longer period of time.

5.3.2 Which system produces more moving patterns ?

It has been observed that the hexagonal grid produces more dynamic patterns (gliders, oscillators) than the classical square grid. The symmetry of the neighborhood relations in the hexagonal grid supports the formation of such dynamic patterns.

5.4 Future Directions

As said before, this simulation is incomplete in terms of having finalized rules, so it still has a lot to improve. Our goal with this research was to understand how Hexagonal Game of Life works and how it could be more beneficial to our usage. After experimenting here is some aspects for where simulation could develop.

(1) Machine Learning-Enhanced Algorithms: The integration of machine learning techniques could improvise the detection and analysis of gliders, oscillators, and other patterns. By training models on existing data, machine learning algorithms could predict pattern evolution, identify rare occurrences, and even optimize rules for specific applications. This approach would significantly enhance computational efficiency since hexagonal computation is an expensive method.
(2) Focus on Large-Scale Simulations: To incresing the scope of the usability in fields like biology and ecology, research should prioritize large-scale simulations. Hexagonal grids are particularly well-suited for modeling complex natural systems due to their symmetrical structure and realistic representation of spatial interactions yet too complex for basic algorithms to follow. So if the large-scale simulations could be developed, that would allow researchers to observe emergent behaviors over extended periods, offering deeper insights into the system's potential for real-world applications.
(3) Exploration of More Complex Rules: With the development of large-scale simulations, complexity The Game already has [8] increases exponentially. Adding to that, incorporating non-linear rules or weighted neighbor relationships into Hexagonal Game of Life could be possible with newly discovered techniques. These modifications would enable the emergence of new dynamic behaviors and patterns, overcoming the limitations observed in simpler rule sets. For example, assigning different weights to neighbors based on their position relative to the cell could simulate interactions in more nuanced systems, such as ecosystems or chemical reactions

6 Conclusion

6.1 Innovative Contribituons Of Hexagonal Game Of Life

In this research, we examined in detail how the Hexagonal Game of Life offers differences and innovations compared to the classical Game of Life. We observed that the hexagonal cell structure has a more balanced neighbourhood relationship than the square cell structure. This relationsip both makes system evolution realistic and increases its applicability in fields such as biology and ecology. We noticed that the hexagonal structure prevents the formation of stable patterns. Therefore we understood that this model is dynamic and open to change any time.

6.2 Main Findings And Contributions

6.2.1 Symmetry in Neighbourhood Relationships

. The hexagonal structure offers a more balanced modeling, as each cell is equidistant from its neighbors.

6.2.2 Does Not Allow Fixed Patterns

. Our research shows that hexagonal grids do not allow stable patters to form. This finding proves that hexagonal systems present a structure in which there is constant change and movement.

6.2.3 Resemblance To Nature Life

. Hexagonal grids offer a potential advantage in modeling natural systems, as they are similar to the hexagonal symmetries we often encounter in nature (e.g. beehives or snowflakes).

6.3 Difficulties That May Be Encountered

In addition to the advantages of hexagonal cells, some disadvantages have also emerged. These difficulties include the complexity of defining new rules for hexagonal cells and the resulting high transaction costs. These problems could be solved by these approaches:

Improving Resource Management: Calculating simulations by dividing them into parts that require lower processing power can provide more efficient use.

Using Machine Learning Models: Manual processing load can be reduced by using machine learning models in pattern recognition and optimization processes.

6.4 Future Directions and Suggestions For The Future Works

As we talked before at 5.2 chapter, this simulation is incomplete. So it still has unfinalized rules and is open to improvement.

6.4.1 New Rule Sets

. More complex rules can be developed to take full advantage of the differences offered by hexagonal grids. For example, the behavior of the system can be diversified by adding multidimensional or weighted neighborhood rules. Thus, it may be possible for this model to reach a much wider usage area.

6.4.2 Large-Scale Simulations

. With larger data sets and long-term simulations, the potential of hexagonal structures in natural systems can be examined more comprehensively.

6.5 Importance And Conclusion Of The Study

The hexagonal game of life offers a different perspective compared to classical square cell models. In this study, it has been shown that the hexagonal structure can have a wider use in scientific research and makes a valuable contribution to the literature. Future studies may better demonstrate the theoretical and practical potential of this model.

References

[1] Rahul Awati. 2024. What is a cellular automaton? https://www.techtarget.com/searchenterprisedesktop/definition/cellular-automaton
[2] J. H. Conway. 1970. "The Game of Life". *Scientific American, 223(4),120-123* (1970).
[3] davidsiaw. 2024. HexLife. https://davidsiaw.github.io/blog/2014/11/21/hexlife/
[4] davidsiaw. 2024. HexLife Part 2. https://davidsiaw.github.io/blog/2014/11/21/hexlife/
[5] M. Gardner. 1983. The Game of Life, Part III. *Wheels, Life and Other Mathematical Amusements: 246, W.H. Freeman.* (1983).
[6] R. Hollerbach. 2016. "The Game of Life Revisited: Cellular Automata and Complex Systems.". *Knots, a Mathematical Review, 5(1), 21-34.* (2016).
[7] C. Langton. 1984. Computational Attractor States in Cellular Automata and Their Relation to the Dynamics of Simple Models of Life. *Physica D: Nonlinear Phenomen*, 10(1), 135-144* (1984).
[8] J. E. Perez. 1996. Conway's Game of Life: A Study in Complexity. *Proceedings of the International Workshop on Cellular Automata for Research and Industry (ACRI), 300-313* (1996).

Received 16 October 2024; revised 8 December 2024