



**Universidade Federal Rural de Pernambuco (UFRPE)**

**Departamento de Estatística e Informática (DEINFO)**

**Cellular Automata**

**Jonas Albuquerque**

**Um Ambiente Interativo para Simulação do Autômato Celular Game of Life utilizando PyQt5**

**Vinícius Ferreira Ferraz**

**26 de Novembro de 2025**

**Resumo:** O estudo dos autômatos celulares tem desempenhado papel fundamental na compreensão de sistemas dinâmicos complexos, capazes de gerar comportamentos emergentes a partir de regras simples. Entre esses modelos, o Game of Life, proposto por John Conway em 1970, destaca-se como um experimento matemático e computacional que demonstra como estruturas auto-organizadas podem surgir a partir de interações locais. Neste trabalho, apresentamos um ambiente computacional interativo, desenvolvido em Python utilizando a biblioteca PyQt5, para simulação visual e análise do Game of Life. A ferramenta permite ao usuário editar padrões diretamente na matriz, avançar e retroceder gerações, além de gerar GIFs animados contendo a evolução de 25 ou 100 gerações. A aplicação demonstrou-se eficiente para fins educacionais, experimentação científica e exploração de padrões emergentes característicos dos autômatos celulares.

## 1. Introdução

O comportamento de sistemas complexos a partir de regras simples tem sido um tema recorrente em diversas áreas do conhecimento, incluindo física, biologia, ciência da computação e teoria da complexidade. Nesse contexto, os **autômatos celulares**, introduzidos originalmente por John von Neumann e Stanislaw Ulam, constituem modelos matemáticos discretos que evoluem no tempo segundo regras locais aplicadas a cada célula de uma grade regular.

Entre os autômatos celulares mais conhecidos está o **Game of Life**, proposto por John Conway (1970). O modelo, composto por células vivas ou mortas organizadas em um plano bidimensional, evolui segundo quatro regras simples que determinam o nascimento, sobrevivência e morte de células. Apesar dessa simplicidade, o sistema apresenta comportamentos extremamente ricos, como osciladores, espaçonaves, padrões estáveis e estruturas em crescimento indefinido. O Game of Life tornou-se uma ferramenta importante para explorar conceitos como emergência, auto-organização e computação natural.

Este trabalho apresenta um ambiente interativo que permite a simulação prática do Game of Life, oferecendo recursos para edição manual da grade, controle da evolução temporal e geração automatizada de GIFs. A abordagem busca unir fins didáticos com uma implementação moderna em Python, utilizando a biblioteca gráfica PyQt5.

## 2. Metodologia

### 2.1 O Modelo do Autômato Celular

O Game of Life baseia-se em uma grade bidimensional onde cada célula assume dois estados possíveis:

- 1 (viva)
- 0 (morta)

A vizinhança utilizada é a vizinhança de Moore, composta pelos oito vizinhos ao redor da célula central. A cada geração, o novo estado é calculado aplicando-se as seguintes regras:

1. Morte por solidão: uma célula viva com menos de 2 vizinhas vivas morre.
2. Superpopulação: uma célula viva com mais de 3 vizinhas vivas morre.
3. Sobrevivência: uma célula viva sobrevive se tiver 2 ou 3 vizinhas vivas.
4. Nascimento: uma célula morta torna-se viva se tiver exatamente 3 vizinhas vivas.

Essas regras produzem comportamentos como:

- padrões oscilatórios (blinker, toad, beacon),
- estruturas estáveis (block, loaf, beehive),
- espaçonaves (glider, lightweight spaceship),
- padrões caóticos e pseudoaleatórios.

## 2.2 Implementação Computacional

A ferramenta foi implementada em Python, empregando a biblioteca PyQt5 para construção da interface e Pillow (PIL) para geração das imagens. A aplicação foi estruturada em três componentes principais

### 2.2.1 Modelo da Matriz (MatrizModel)

A classe `MatrizModel` herda de `QAbstractTableModel`, permitindo representar a grade como uma tabela interativa.

Principais funcionalidades:

- armazenamento da matriz de estados (20×20 por padrão);
- suporte a edição célula a célula;
- cálculo da próxima geração segundo as regras do Game of Life;
- pilha de histórico para retornar gerações anteriores.

O cálculo da geração utiliza uma dupla iteração sobre linhas e colunas, avaliando os oito vizinhos ao redor de cada célula. Para cada posição, uma nova matriz é construída seguindo estritamente as regras de Conway.

### 2.2.2 Visualização Interativa (MatrizView)

A classe `MatrizView`, derivada de `QTableView`, fornece:

- visualização das células como blocos pretos (vivas) ou brancos (mortas);
- ajuste automático do tamanho das células de acordo com a resolução da tela;
- interação por clique e arraste para desenhar padrões livremente;
- bloqueio de barras de rolagem e cabeçalhos para layout limpo.

O usuário pode criar padrões conhecidos manualmente, como gliders ou oscillators, com interação fluida.

### 2.2.3 Geração de GIFs

O sistema inclui duas funções dedicadas:

- `handle_25_gens_gif()`
- `handle_100_gens_gif()`

Cada uma percorre a matriz, renderizando sua evolução e salvando-a no diretório `gifs/`. Cada frame é produzido ampliando as células por um fator de escala e pintando pixels correspondentes.

O método garante:

- preservação do estado inicial (backup e restauração),
- GIFs suavemente encadeados,
- registro automático das miniaturas no painel lateral.

### 2.2.4 Interface Gráfica e Controle de Execução

A classe `MainWindow` organiza todos os componentes:

- painel lateral com botões grandes e estilizados;
- controle de próxima geração e geração anterior;
- reset total da matriz;
- botão para abrir a pasta dos GIFs;
- galeria com miniaturas das quatro animações mais recentes.

A aplicação abre em tela cheia sem bordas, otimizando o espaço para edição e observação dos padrões.

## 3. Resultados

A aplicação demonstrou desempenho consistente durante a simulação de diferentes padrões do Game of Life. A possibilidade de alternar entre gerações manualmente, retroceder estados e gerar GIFs permitiu observar comportamentos clássicos como:

- **Osciladores simples**, que alternam entre duas ou mais configurações estáveis.
- **Gliders**, que se deslocam diagonalmente pela grade.
- **Padrões pseudoaleatórios**, que eventualmente se estabilizam ou convergem para sistemas oscilatórios.

- **Explosões complexas**, onde regiões densas geram frentes de expansão e subsequente decomposição em estruturas menores.

A galeria lateral exibiu corretamente as quatro últimas animações geradas, oferecendo um mecanismo que facilita a análise comparativa da evolução dos padrões.

## 4. Discussão

A implementação confirma a potência dos autômatos celulares como modelos capazes de produzir fenômenos emergentes complexos a partir de um conjunto mínimo de regras. A ferramenta desenvolvida destaca-se por:

- **Interatividade:** o usuário pode experimentar livremente padrões iniciais, acelerando o aprendizado sobre dinâmica de autômatos.
- **Visualização rica:** a opção de gerar GIFs facilita registro, estudo e compartilhamento dos padrões evolutivos.
- **Modularidade:** o código possui separação clara entre modelo, visualização e controle, o que permite futuras expansões, como:
  - aumento do tamanho da grade,
  - simulação contínua,
  - inserção de padrões pré-definidos,
  - cores dinâmicas,
  - exportação para vídeo MP4 ou PNG sequencial.

Além disso, a utilização de PyQt5 demonstra boa aderência entre interfaces gráficas modernas e modelos matemáticos discretos.

## 5. Conclusão

Este trabalho apresentou um ambiente interativo para simulação do Game of Life, permitindo criação e análise de padrões através de uma interface amigável e responsiva. A ferramenta mostrou ser valiosa tanto em contextos educacionais quanto em pesquisas iniciais sobre dinâmicas emergentes. A possibilidade de gerar GIFs das evoluções amplia seu potencial como instrumento de documentação e exploração visual.

A aplicação demonstra que sistemas complexos podem emergir de regras simples, reafirmando a relevância dos autômatos celulares como ferramentas computacionais e conceituais.

## 6. Disponibilidade do Código

O código-fonte completo encontra-se disponível de forma aberta para análises, modificações e estudos.

**Link de Acesso:** [Vinicius-O-Ferraz/ConwayStudio-Gif-Maker](https://github.com/Vinicius-O-Ferraz/ConwayStudio-Gif-Maker)

## 7. Referências

1. Conway, J. (1970). The Game of Life. *Scientific American*.
2. Wolfram, S. (2002). *A New Kind of Science*. Wolfram Media.
3. Gardner, M. (1970). Mathematical Games: The fantastic combinations of John Conway's new solitaire game "Life". *Scientific American*.
4. Sedgewick, R., & Wayne, K. (2002). *Lecture P4: Cellular Automata*. Princeton University COS 126.