Uso de Orientação a Objetos para a Modelagem do Autômato Celular Jogo da Vida

Francisco M. Palermo[1], **Samuel R. Cassanego**[1], **Sergio Luis S. Mergen**[1]

[1]Sistemas de Informação – Universidade Federal de Santa Maria (UFSM)

[2]Centro de Tecnologia da UFSM – (CT-UFSM)
Universidade Federal de Santa Maria (UFSM) – Santa Maria, RS – Brazil

fmpalermo@inf.ufsm.br, srcassanego@inf.ufsm.br, mergen@inf.ufsm.br

Resumo. *O Jogo da vida (Game of life) é um conhecido programa que usa autômatos celulares para simular a evolução de seres biológicos. Este artigo apresenta a transcrição do código estrutural deste programa para um código orientado a objetos. O objetivo da transcrição é permitir que novos comportamentos sejam adicionados de maneira simples, sem que código pre-existente precise ser modificado. Os testes demonstram que a transcrição foi bem sucedida, e abre caminho para que novos comportamentos sejam testados.*

Abstract. *The Game of Life (Game of life) is a well-known program that uses cellular automata to simulate the evolution of biological beings. This article presents the transcription of the structural code of this program to an object-oriented code. The purpose of the transcription is to allow new behaviors to be added in a simple way, without pre-existing code having to be modified. Tests demonstrate that the transcription was successful, and paves the way for new behaviors to be tested.*

1. Introdução

O jogo da vida, ou *Game of life*, é denominado como um autômato celular bidimensional, inventado pelo matemático britânico John Horton Conway. Basicamente, este autômato simula a evolução de seres biológicos. Ele ficou bastante conhecido pela capacidade de construir sistemas complexos a partir de um conjunto de regras simples.

Como apresentado no [Izhikevich et al. 2015], os seres são representados em células de uma matriz bi-dimensional. Cada célula pode conter os valores 1 ou 0, indicando se o ser está vivo ou morto, como podemos observar na Figura 1.

O programa é dividido em gerações. Em cada geração, o estado da célula pode mudar de acordo com um conjunto de regras condicionais absolutas, descritas a seguir:

- Qualquer célula viva com menos de dois vizinhos vivos morre de solidão;
- Qualquer célula viva com mais de três vizinhos vivos morre de superpopulação;
- Qualquer célula morta com exatamente três vizinhos vivos se torna uma célula viva;
- Qualquer célula viva com dois ou três vizinhos vivos continua no mesmo estado para a próxima geração;

Figura 1. Matriz (parte gráfica e numérica)

A ideia, apesar de simples, é capaz de demonstrar como a estabilidade populacional pode ser alcançada através de resultados visualmente atraentes. Essa característica despertou o interesse científico em estudar variações do conjunto de regras para testar outras hipóteses.

Surgiram muitos modelos de implementação do Jogo da Vida desenvolvidos em várias linguagens diferentes. Em repositórios de código, são geralmente encontradas várias soluções usando os conceitos da programação estruturada, onde trechos do código implementam as regras verificando o conteúdo armazenado em cada célula.

No entanto, a programação estruturada gera um obstáculo para que o jogo da vida possa ser adaptado de maneira prática e eficaz, pois a criação de novas regras envolve adaptações em códigos pre-existentes. Nesse sentido, a programação orientada a objetos é capaz de fornecer um ambiente propício para a exploração de regras variadas, uma vez que novos comportamentos são facilmente adicionados através da criação de novas classes em uma hierarquia de herança.

A proposta deste artigo é adaptar o código-fonte do Jogo da Vida para que sejam usados os conceitos de orientação a objetos para modelar os seres, seu comportamento e suas interações. No decorrer do artigo, pretende-se demonstrar a viabilidade da transcrição, e enfatizar as possibilidades que se abrem quando se trabalha com código orientado a objetos.

2. Trabalhos relacionados

O jogo da vida vem sendo amplamente utilizado em pesquisas acadêmicas. Nesta seção, serão apresentados alguns trabalhos que demonstram o interesse em manipular as regras básicas do jogo para alcançar resultados inovadores.

Em [Mathrani et al. 2019], surgiu uma variação muito interessante do *game of life* (GoL), onde os seres contam com um atributo 'gênero'. As regras foram reformuladas de forma que apenas interações de seres com gêneros opostos produzissem novos seres. Essa mudança permitiu analisar como sociedades diferentes podem evoluir ao longo dos anos.

O trabalho de [Vilcarromero et al. 2010] propôs outro autômato celular semelhante ao *Game of Life*, porém com um foco voltado para a análise de como duas espécies diferentes, onde uma ocupa a categoria de presa e a outra de predadora, interagem entre si. O trabalho estudou como essas espécies se comportam vivendo em um mesmo espaço, analisando seus pontos de equilíbrio e os pontos de ápice da presa e do predador.

Já no [Levene and Roussos 2003] se discute a possibilidade de usar em um mesmo cenário espécies que compitam entre si. Nesse sentido os autores apresentam o *P2life* um jogo da vida onde existem dois grupos celulares distintos fazendo com que sua sobrevivência e expansão gerem uma competitividade no seu universo.

Em nenhum desses casos, o paradigma de programação foi um assunto abordado. Assume-se que não houve a preocupação em criar um ambiente propício para a expansão e a incorporação de novos comportamentos.

3. Proposta

A proposta, como mostra o diagrama de classes da Figura 2, é de inicialmente separar as funções do universo (*Universe*) e do ser (*Being*).

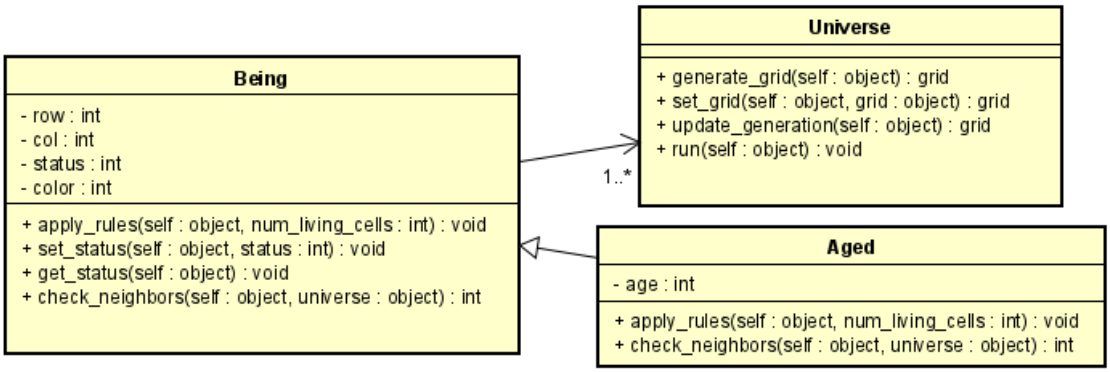

Figura 2. Diagrama de classes

A classe *Universe* é responsável por servir de meio para que os seres possam interagir. Nela são modeladas características como a quantidade máxima de gerações que devem ser geradas e as dimensões da matriz do mundo criado. Uma das principais funções é a *update_generation*, responsável por comunicar-se com os seres do universo.

Por sua vez, a classe *Being* modela as características exclusivas de seres. Para o jogo da vida, a única característica é o *status*, indicando se o ser está vivo ou morto. Uma das principais funções é a *apply_rules*, responsável por decidir se o ser deve ou não permanecer vivo. Essa função se vale do resultado obtido pela função *check_neighbors*, que verifica quantos seres com o *status* vivo estão próximos da instância do ser sendo analisado.

Com o código organizado dessa forma, pode-se criar outros comportamentos para os indivíduos a partir de extensões da classe *Being*. Por exemplo, o diagrama mostra a classe *Aged*, que possui a característica de idade, sendo que essa informação é usada pelo método sobrescrito *apply_rules* para modificar o comportamento principal da classe. A função *check_neighbors* também pode ser sobrescrita caso se deseja analisar um conjunto

diferente de células vizinhas, ou até mesmo verificar o tipo da célula vizinha antes de retornar uma contagem.

Convém destacar que a criação de novos comportamentos é alcançada sem que nenhum código pre-existente precise ser modificado, o que facilita que novos modelos sejam testados e até mesmo combinados. Por exemplo, o modelo de competitividade proposto por [Levene and Roussos 2003] (*P2life*) poderia ser implementado de uma maneira simplificada, através da criação de novas classes.

4. Resultados alcançados

O objetivo desta seção é demostrar que tanto o programa estruturado quanto o orientado a objetos, partindo de um mesmo cenário inicial, conseguem reproduzir os mesmos resultados finais. Para isso, criou-se um *framework* de testes que executa e avalia os dois programas. Neste *framework* pode ser definido quantas vezes deve ser realizado um novo teste, o número máximo de gerações em cada teste e as configurações relacionadas ao tamanho das matrizes.

A execução ocorre a partir de uma geração aleatória de um cenário inicial que é testado nos dois programas. Após isso, a matriz resultante de cada execução é salva em um arquivo de texto, onde é realizada a verificação de igualdade entre os dois resultados. Caso haja alguma incompatibilidade entre as matrizes, isso significa que os dois algoritmos se comportaram de maneira distinta.

Foi realizado um teste com 1000 execuções de cenários diferentes, até a 30ª geração, com uma matriz de 30 linhas e 40 colunas. O resultado demonstrou que os algoritmos geraram as mesmas matrizes em todos os casos, o que atribui um alto grau de confiabilidade de que a transcrição foi realizada com sucesso.

5. Conclusão

Ao fim do trabalho, pode-se concluir que a implementação realizada até o momento mostrou a viabilidade de modificar o paradigma de programação clássico do jogo da vida. O uso da orientação a objetos abre caminho para que novas regras sejam avaliadas.

Como trabalhos futuros, pretende-se transformar o código em um *framework*, para que tanto o universo quanto os seres possam ser expandidos. Esse ajuste abre um leque maior para experimentações. Por exemplo, uma possibilidade a ser explorada envolve simular outros modelos voltados à estabilidade populacional e até mesmo usar autômatos celulares para estudar a disseminação de um vírus durante uma epidemia.

Referências

Izhikevich, E. M., Conway, J. H., and Seth, A. (2015). Game of life. *Scholarpedia*, 10(6):1816.

Levene, M. and Roussos, G. (2003). A two-player game of life. *International Journal of Modern Physics C*, 14(02).

Mathrani, A., Scogings, C., and Mathrani, S. (2019). Gender diversity population simulations in an extended game of life context. *IEEE Access*, 7.

Vilcarromero, Â. C., Jafelice, R. S., and Barros, L. C. (2010). Autômato celular no estudo de um modelo presa-predador.