Jogo da Velha em C++

Implementação utilizando programação orientada a objetos e árvore de decisão

Jesuino Vieira Filho
Engenharia Mecatrônica
Universidade Federal de Santa Catarina
Joinville, Brasil
jesuinovieira_@hotmail.com

***Resumo*—Este relatório é um documento referente ao trabalho da disciplina EMB5631 - Programalção III, ministrada pelo professor Gian Ricardo Berkenbrock.**

Palavras-chave—jogo da velha, programação; àrvore de decisão;

I. Introdução

Esse relatório tem como objetivo apresentar o desenvolvimento do jogo da velha, um jogo de raciocínio que foi implementado na linguagem de programação C++, com foco na aplicação dos conceitos da programação orientada a objetos. O documento é dividido em três seções primárias. A primeira apresenta a descrição do jogo, tal como suas regras, ilustrações do seu funcionamento e demais aspectos relacionados. Essa também é composta pela explicação da implementação do programa, demonstrando aonde foi aplicado a orientação a objetos e a codificação da árvore de decisão. A outra etapa apresenta uma breve discussão sobre o projeto, relatando algumas dificuldades e similares. Finalmente, uma sessão com as conclusões encerra o documento falando sobre as perspectivas do desenvolvimento e o progresso obtido com a realização do trabalho.

II. Desenvolvimento

A. Regras do jogo

O Jogo da Velha é um jogo de fácil entendimento, realizado por dois jogadores que revezam suas jogadas por turnos, marcando uma posição de um tabuleiro 3 x 3, desde que esta esteja vazia.

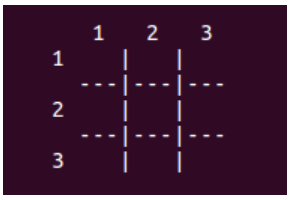

Fig. 1. Representação do tabuleiro 3 x 3.

Dessa forma, a jogada passa para o outro jogador, e o processo se repete até que um deles consiga colocar três de seus marcadores – representador por X ou O – em uma linha vertical, horizontal, diagonal ou até o tabuleiro ser completamente preenchido sem que ocorra os eventos anteriores, situação na qual ocorre empate.

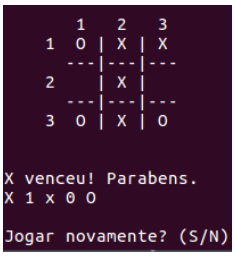

Fig. 2. Representação de uma situação em que o jogador que possui o marcador X vence o jogo.

Uma curiosidade é que, se os dois jogadores fizerem suas melhores jogadas em todos os turnos, o jogo sempre terminara em empate. Isso porque apesar do número total de possíveis jogadas ser grande, a maioria delas são simétricas e possuem o mesmo efeito.

Devido a sua simplicidade, o jogo é muitas vezes utilizado como ferramenta pedagógica para estimular o raciocínio de crianças e para ensinar o ramo da inteligencia artificial que trata das árvores de busca – utilizada na implementação desse trabalho.

B. A implementação

Antes de começar a implementação do código no computador, foi extremamente importante elaborar o funcionamento do jogo no papel, tal como as classes que seriam implementadas e a ordem de execução dos comandos.

Desse modo, a decisão final foi a criação de quatro classes: TicTacToe, Tabuleiro, Jogador e ArvoreDeDecisao.

1. Classe TicTacToe: esta é a classe gerente de jogo, tem como objetivo cuidar de toda a execução, marcações no tabuleiro, interação com o usuário, se comunicar com a árvore de decisão e imprimir na tela as informações do jogo em andamento. Desse modo, foi utilizado o conceito de template, com objetivo de possibilitar dois tipos de jogo: Usuário 01 x Usuário 02 (representado por: TicTacToe<Jogador>) ou

Usuário x Máquina (representado por: TicTacToe<ArvoreDeDecisao>). Apesar da duplicação de código em alguns métodos, o template foi necessário para evitar a criação de atributos e métodos desnecessários – principalmente a criação da árvore no modo de jogo Usuário 01 x Usuário 02, a qual exige um pouco mais de processamento.

A classe também faz o uso de composição, tendo em vista que possui dois objetos do tipo Jogador, um objeto do tipo Tabuleiro, e no template TicTacToe<ArvoreDeDecisao> possui um objeto do tipo ArvoreDeDecisao.

Um objeto do tipo TicTacToe é instanciado na main para dar início ao jogo.

2. Classe Tabuleiro: o tabuleiro é bem simples, uma das primeira classes a se pensar quando precisamos desenvolver um jogo da velha. Possui seus métodos setters e getters, sempre respeitando o encapsulamento e permitindo a possibilidade de suas instâncias manipularem-o indiretamente, possuindo também, uma matriz de caractere 3 x 3 para representar o tabuleiro em si.

Um objeto do tipo Tabuleiro compõe a classe TicTacToe e a classe ArvoreDeDecisão.

3. Classe Jogador: outra classe simples, tem como objetivo modularizar o programa e dar mais clareza ao código. Possui o nome do jogador, seu marcador e o numero de vitorias.

Um objeto do tipo Jogador compõe a classe TicTacToe.

4. Classe ArvoreDeDecisão: aqui se encontra o grande desafio do programa. As árvores de decisões são comumente usadas na análise de decisões, para ajudar a identificar uma estratégia com maior probabilidade de atingir um objetivo.

Desse modo, a àrvore possui uma espécie de “mapa” dos possíveis resultados do jogo da velha. Cada nó representa uma jogada e assim, conforme o andamento do jogo, a árvore acompanha o tabuleiro da classe TicTacToe.

A construção da árvore é feita recursivamente, e quando chegamos em um nó que possui um tabuleiro onde o jogo chegou ao fim, encerramos as chamadas recursivas e retornamos os valores: 10 (se a árvore venceu o jogo), -10 (se o oponente venceu o jogo) e 0 (no caso de empate).

Afim de melhorar o desempenho da árvore, na hora de retornar os valores citados anteriormente, foi somado a altura em que a árvore se encontra vezes 10 (se a árvore venceu o jogo) ou vezes -10 (se o oponente venceu o jogo ou se houve empate).

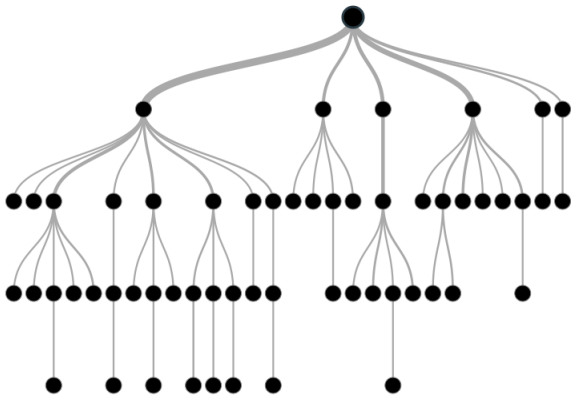

Fig. 3. Ilustração de um ramo da árvore, os nós folhas possuem tabuleiros em que o jogo chegou ao fim.

C. Execução

O tipo de jogo é definido em tempo de chamada do programa, por meio da passagem de parâmetros (ver arquivo “README”).

Iniciando o programa, é solicitado ao(s) usuário inserir o seu(s) nome(s), dependendo do modo de jogo. Em seguida o jogo começa, demandando a jogada do(s) usuário(s) até que ocorra um situação de fim de jogo – empate ou vitória de algum jogador.

Em seguida, é impresso na tela o resultado da partida atual e o placar do jogo. Assim, o(s) usuário(s) pode decidir entre continuar jogando ou sair do programa.

A seguir, uma breve ilustração do funcionamento do jogo.

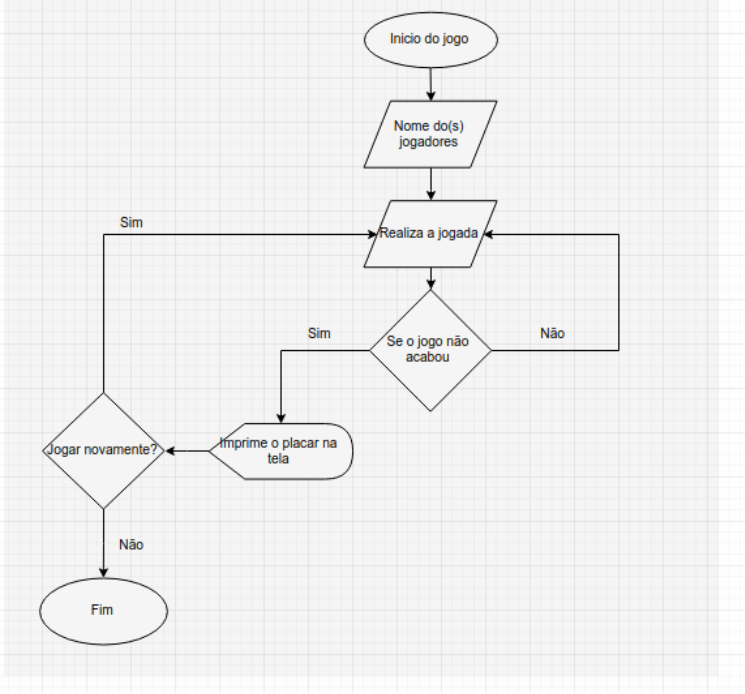

Fig. 4. Fluxograma básico.

III. Discussão

O maior desafio do programa foi a implementação da árvore de decisão. Inicialmente, é relativamente difícil de interpretá-la devido a sua complexidade e abstração, de modo que com o tempo o programador vai se familiarizando com o seu funcionamento.

Sendo assim, esporadicamente foram encontradas umas jogadas que parecem não ser criadas pela árvore, de modo que quando é realizada a comunicação entre o tabuleiro da classe TicTacToe e da classe ArvoreDeDecisão, ocorra alguns bugs.

A organização das classes foi realizada ilustrando o mundo real. Portanto, aplicando as técnicas da orientação a objeto, ficou evidente como a linguagem C++ permite implementar códigos mais robustos do que, por exemplo, a linguagem C. Isso porque suas técnicas de encapsulamento permitem criar classes com comportamentos e estados bem definidos, que depois de implementadas e testadas, não são corrompidas por outras implementações.

IV. Conclusão

O projeto apresentado proporciona uma boa noção inicial da programação orientada a objetos, permitindo ao aluno absorver bem a idéia da representação de objetos do mundo real no mundo virtual – feita através do uso de classes. Além disso, certificando-se de encapsular corretamente os dados, fica evidente o poder desse estilo de programação, o qual permite os programadores implementarem códigos mais estáveis e com maior reusabilidade.