

JOGO DA VELHA COM IA

Trabalho A3 - Inteligência Artificial

DESENVOLVIMENTO DE AGENTE INTELIGENTE PARA JOGO DA VELHA UTILIZANDO PROGRAMAÇÃO ORIENTADA OBJETO

Arthur Santos Araújo- 12725166488;
Gabriel Santos Rocha– 1272328504;
Matheus Barbosa Costa - 12722118928;
Rubem Matheus Vieira Martins - 1272216869.

Universidade Salvador – Salvador – BA – Brasil

INTRODUÇÃO

O Jogo da Velha é um problema clássico na área de Inteligência Artificial, caracterizado como um ambiente determinístico, de soma-zero e com informação perfeita. Embora aparente simplicidade, serve como excelente campo de provas para a modelagem de Agentes Racionais. Este projeto visa desenvolver uma aplicação onde um agente autônomo utiliza algoritmos de tomada de decisão para competir contra jogadores humanos. Diferente de implementações procedurais simples, este trabalho adota uma arquitetura robusta baseada em Programação Orientada a Objetos (POO), simulando a interação entre um Agente e seu Ambiente.

OBJETIVO

OBJETIVO GERAL:

Desenvolver um agente de software capaz de jogar Jogo da Velha de forma competente, aplicando conceitos de IA simbólica e heurísticas de decisão.

OBJETIVOS ESPECÍFICOS:

- Modelar o sistema utilizando o paradigma de Orientação a Objetos para separar a lógica do jogo da inteligência do agente.
- Implementar um algoritmo de decisão baseado em regras de prioridade (Ataque, Defesa e Estratégia).
- Garantir que o agente seja capaz de identificar vitórias, bloqueios e jogadas ótimas em tempo real.

MÉTODOS

A solução foi desenvolvida em linguagem Python, estruturada em três classes principais que definem a arquitetura do sistema:

1. **Classe Ambiente:** Representa o tabuleiro e as regras do jogo. É responsável por manter o estado (matriz 3x3), validar movimentos e verificar condições de vitória ou empate (o "Juiz").
2. **Classe Agente:** Classe base que define as propriedades genéricas de um jogador (símbolo 'X' ou 'O').
3. **Classe AgenteIA (Herança):** Especialização do agente que implementa o método cognitivo pensar().

Algoritmo de Decisão (Heurística): O método pensar() do agente opera como um Sistema Baseado em Regras Hierárquicas. A cada turno, o agente analisa o Ambiente (percepção) e escolhe a ação seguindo esta ordem de prioridade estrita:

1. **Vitória Imediata:** Verifica se há alguma jogada que encerra o jogo a seu favor.
2. **Bloqueio:** Verifica se o oponente vencerá na próxima rodada e bloqueia a posição.
3. **Ocupação Estratégica:** Prioriza o centro do tabuleiro (1,1), estatisticamente a posição mais valiosa.
4. **Ocupação de Cantos:** Prioriza os vértices do tabuleiro.
5. **Movimento Aleatório:** Caso nenhuma regra anterior se aplique, escolhe uma casa vazia disponível.

REFERÊNCIAS BIBLIOGRÁFICAS

1. RUSSELL, Stuart; NORVIG, Peter. Artificial Intelligence: A Modern Approach. 3. ed. Prentice Hall, 2009.
2. DOCUMENTAÇÃO PYTHON. Classes e POO. Disponível em: python.org. Acesso em: nov. 2025.

RESULTADOS

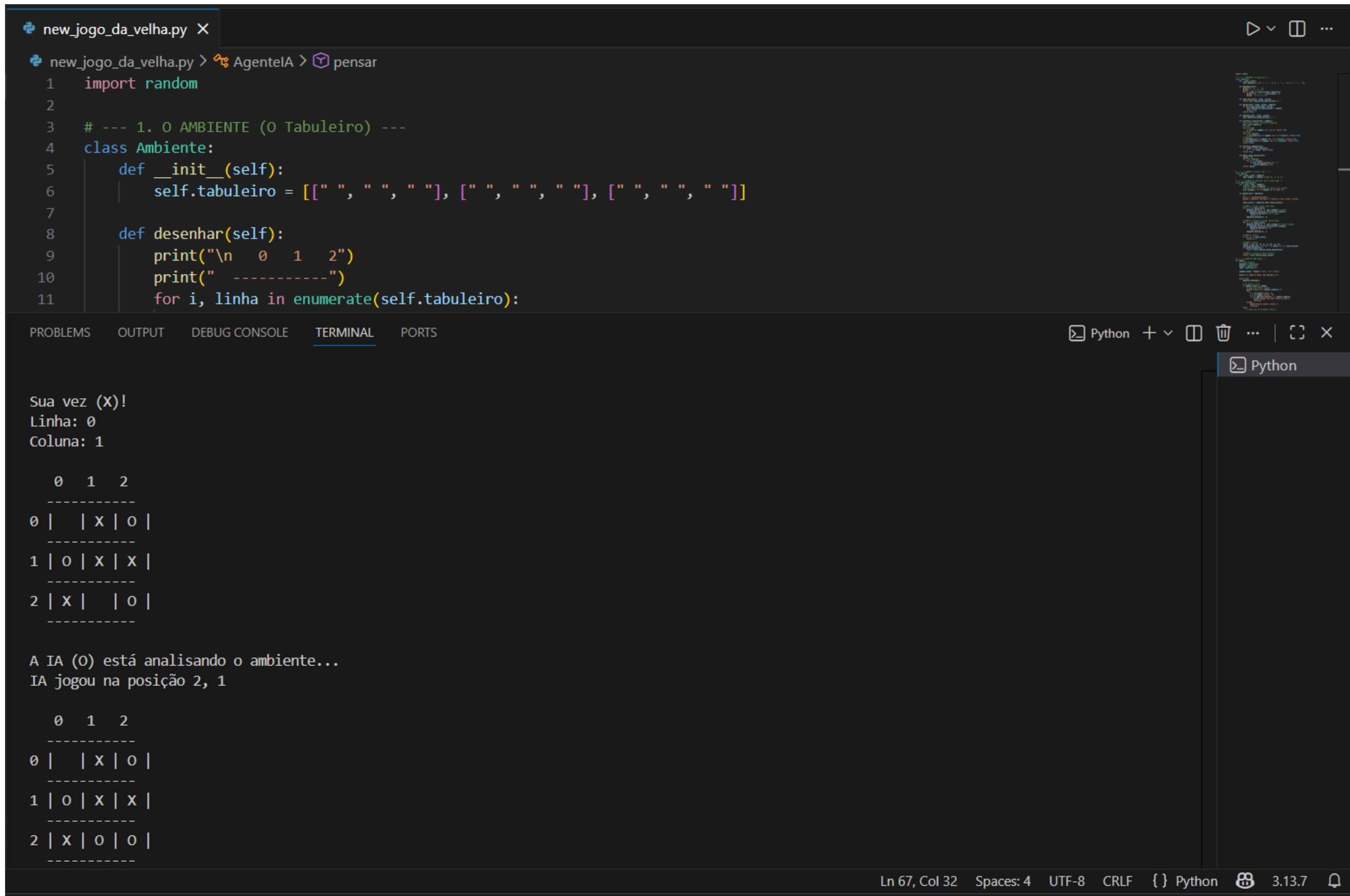
Os testes realizados com o agente demonstraram um comportamento estratégico consistente:

- **Desempenho Defensivo:** O agente obteve 100% de sucesso em bloquear vitórias iminentes do adversário humano, impedindo derrotas por desatenção.
- **Desempenho Ofensivo:** O agente foi capaz de identificar e executar todas as oportunidades de vitória imediata apresentadas.
- **Eficiência:** O tempo de resposta para a tomada de decisão foi imediato (< 0.1s), demonstrando a leveza do algoritmo baseado em regras comparado a buscas profundas em árvore.
- **Limitação:** Observou-se que, contra jogadores humanos perfeitos, o agente tende ao empate, comportamento esperado para o Jogo da Velha.

A análise qualitativa revelou que o Agente IA apresenta pontos fortes decisivos, como a **ausência de erros táticos básicos**, a **identificação imediata de vitórias** e a **execução infalível de bloqueios defensivos** quando ameaçado. O comportamento mostrou-se consistente e determinístico em todas as partidas simuladas.

Como limitações, observou-se um padrão de jogadas repetitivo em aberturas simétricas e uma vulnerabilidade a táticas de 'dupla ameaça' (garfos), uma vez que a heurística baseada em regras prioriza o estado atual sem simular profundidade futura. Contudo, diferente de jogos mais complexos, o tempo de resposta manteve-se **imediato e constante**, independentemente do estágio da partida, validando a eficiência computacional da arquitetura proposta.

Figura 1 – Interface do jogo em execução no terminal, demonstrando a interação entre Humano e Agente IA.



```
new_jogo_da_velha.py X
new_jogo_da_velha.py > AgentelA > pensar
1 import random
2
3 # --- 1. O AMBIENTE (O Tabuleiro) ---
4 class Ambiente:
5     def __init__(self):
6         self.tabuleiro = [
7             [" ", " ", " "], [" ", " ", " "], [" ", " ", " "]
8         ]
9     def desenhar(self):
10        print("\n  0  1  2")
11        print("-----")
12        for i, linha in enumerate(self.tabuleiro):
```

Sua vez (X)!

Linha: 0

Coluna: 1

```
  0  1  2
-----
0 |  | x |  |
1 |  | x | x |
2 | x |  |  |
-----
```

A IA (O) está analisando o ambiente...

IA jogou na posição 2, 1

```
  0  1  2
-----
0 |  | x |  |
1 |  | x | x |
2 | x |  |  |
-----
```

Ln 67, Col 32 Spaces: 4 UTF-8 CRLF Python 3.13.7

CONCLUSÃO

O desenvolvimento deste projeto permitiu a aplicação prática de conceitos fundamentais de Inteligência Artificial e Engenharia de Software. A utilização de Programação Orientada a Objetos facilitou a modelagem do problema, permitindo uma separação clara entre as regras do mundo (Ambiente) e a lógica do indivíduo (Agente). O algoritmo de regras de prioridade provou-se uma solução eficaz para atingir uma complexidade média, criando um oponente desafiador para jogadores casuais e resiliente contra jogadores experientes.