



Centro Universitário de Belo Horizonte - UNIBH
Instituto de Engenharia e Tecnologia - IET

Agente reativo simples - Jogo da Velha

**Marcos Magno de Carvalho
Jonas Oliveira da Silva Filho
José Mateus de Lima Xisto
Mateus Ruas Duarte**

Belo Horizonte - Minas Gerais

2018

Lista de ilustrações

Figura 1 – Agentes interagem com ambientes.	a
Figura 2 – Jogo da velha	a
Figura 3 – Tabuleiro	c
Figura 4 – Humano	c
Figura 5 – Agente	c
Figura 6 – Ganho	c

Sumário

1	INTRODUÇÃO	a
2	METODOLOGIA	c
2.1	Implementação	c
2.1.1	Componentes do jogo	c
2.1.2	Humano	d
2.1.3	Agente	d
2.1.4	Jogo da Velha	e
3	CONCLUSÃO	f
	REFERÊNCIAS	g

1 Introdução

A inteligência artificial (IA) é um ramo da ciência da computação que se propõe a elaborar dispositivos que simule a capacidade humana de raciocinar, resolver problemas e tomar decisões. Dentro de IA, temos o conceito de agente, sendo esse definido em sua simplicidade apenas como algo que age, porém, espera-se que um agente faça mais que apenas agir. A partir disso surge os agentes racionais, capaz de operar sob controle autônomo, perceber seu ambiente, persistir por um período de tempo e adaptar a mudanças, com a finalidade de alcançar o melhor resultado possível ou, quando há incerteza, o melhor resultado esperado (RUSSEL S.;NORVIG, 2013). Para isso, o agente deve levar em consideração a percepção do seu ambiente, utilizando sensores e de agir, por meio de atuadores (Figura 1). Tomando o ser humano como exemplo de um agente, os olhos e ouvidos serverm como sensores, o que é utilizado para perceber o ambiente em que se está. As pernas e mãos servem como atuadores, sendo utilizado para atuar e modificar o ambiente.

Dentre os programas de agentes, existe os reativos, sendo representado por agentes reativos baseados em modelo e reativos simples. Esse é caracterizado pelo fato de selecionar ações com base na percepção atual, ignorando o restante do histórico, podendo ser utilizados em vários ambientes, até mesmo nos mais complexos, como por exemplo, em carros automatizados.

Entre os vários ambientes, a área de jogos de tabuleiro tem se mostrado favorável para estudo e aplicação de agentes (COSTA, 2016). Um exemplo é o jogo da velha. O mesmo é composto por uma matrix de tamanho 3X3 e tem como objetivo fazer uma sequência de três símbolos iguais (Figura 2), normalmente é utilizado o X e O para representar cada um dos dois jogadores.

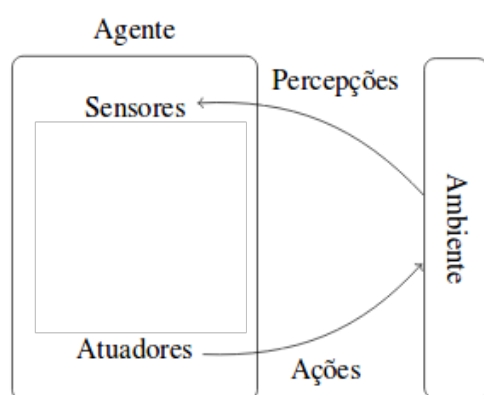


Figura 1 – Agentes interagem com ambientes.

Fonte – Adaptada (STANGE R. L.; CEREDA, 2017)

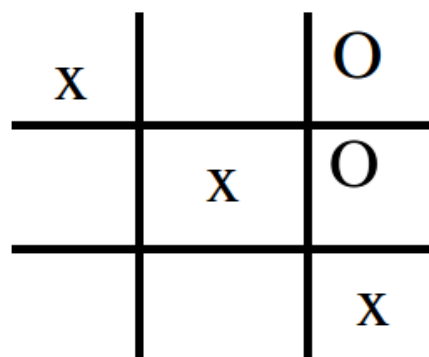


Figura 2 – Jogo da velha

Fonte – Elaborado pelos autores

Diante de tudo, o presente trabalho tem como objetivo estudar e desenvolver um programa de agente reativo simples, bem como implementá-lo em um software que simule o jogo da velha. Desta forma, um dos jogadores será o humano e o outro o agente. Além disso, o agente em questão deverá fazer, no mínimo, 20 percepções. Por fim, este próprio documento será composto como parte do objetivo, fazendo dele um relatório final.

² <https://www.linuxmint.com/>

2.1.2 Humano

Aqui desenvolveu-se a classe Humano com os métodos `set_jogada_humano` e `get_jogada_humano`. Esses métodos são responsáveis por registrar a posição das jogadas feitas pelo jogador humano em um vetor.

2.1.3 Agente

Para o agente, implementou-se a classe `Agente()` com os métodos abaixo:

1. agente()

O comportamento do agente é dado abstratamente pela função do agente

$$f : P^* \rightarrow A$$

onde P^* é uma sequência de percepções e A é uma ação (ZADROZNY, 2010). Para isso, o seguinte procedimento é chamado:

$$self.atuador(self.percepcao())$$

Ou seja, para cada ação (atuador), passa-se uma sequência de percepções.

2. atuador (acao)

O método atuador recebe como parâmetro uma função de percepções e retorna uma ação possível. Ele é responsável por enviar uma ação para o jogo da velha.

3. percepcao()

Esse método é responsável por perceber cenários possíveis de jogada, levando em consideração posições livres no tabuleiro e as jogadas do humano. Como resultado, o método retorna uma ação possível.

No algoritmo Percepções (Algoritmo 1), o procedimento JOGADAS HUMANO (Linha 7) é responsável por comparar se, para cada posição i do vetor de lista ganho (Figura 9), existe uma lista que contenha o valor da posição jogada pelo humano. Caso tenha, armazena o valor no vetor `jogHumano`. Uma verificação de tamanho (se for maior que 1) no mesmo vetor é feita a cada interação, com o intuito de perceber se há alguma intenção de ganho nas jogadas do humano. Caso seja verdadeiro a verificação, chama-se o método POSIÇÃO DE GANHO (Linha 9). Esse procedimento é responsável por verificar qual posição da lista de ganho resta para ser marcada, por exemplo: Caso o humano jogue na posição 1, o método JOGADAS HUMANO registra-o no vetor `jogHumano` e verifica quais são as listas que contém a posição 1 no vetor ganho. Neste caso, temos a lista [1,2,3] e [1,5,9] 6.

Ná próxima jogada do humano, caso ele jogue na posição 2 ou 5 o vetor `jogHumano` será maior que 1, portanto, chamará o procedimento POSIÇÃO DE GANHO. Suponha-se que ele jogue na posição 5, então, o procedimento retornará a posição 9 para o atuador.

Já o procedimento POSIÇÃO LIVRE (Linha 12) é responsável por verificar e montar possíveis jogadas de ganho, de forma a verificar quais às posições livres no tabuleiro e comparar com às combinações de lista de ganho.

Por fim, caso durante o jogo nenhum procedimento supracitado seja chamado, a ação retornada é aleatória, por meio do procedimento RANDOM CHOICE (Linha 15), levando em consideração as posições livres no tabuleiro.

Algorithm 1 Percepções

```

1: function PERCEPCAO
2:   jogAgent  $\leftarrow$  []
3:   jogHumano  $\leftarrow$  []
4:   jogAcao  $\leftarrow$  []
5:   jogAgent  $\leftarrow$  []
6:   for i in range(0, 8) do                                     ▷ Vetor ganho
7:     jogHumano  $\leftarrow$  JOGADAS HUMANO
8:     if jogHumano > 1 then
9:       jogAcao  $\leftarrow$  POSIÇÃO DE GANHO
10:      DESEMPILHA (jogHumano)
11:      return jogAcao                                             ▷ Ação baseada nas jogadas do Humano
12:   jogAgent  $\leftarrow$  POSIÇÃO LIVRE
13:   if jogAgent > 1 then
14:     return jogAgent                                             ▷ Ação baseada nas posições livres
15:   jogAleatorio  $\leftarrow$  RANDOM CHOICE(posicaoLivre)             ▷ Ação aleatória
16:   return jogAleatorio

```

2.1.4 Jogo da Velha

Para o funcionamento do jogo, implementou-se o método `main`, responsável por criar os objetos necessários (Agente, Humano e JogoDaVelha).

O primeiro jogador é iniciado de forma aleatória a cada inicio de jogo, por meio da função `random.randint(0, 1)`, sendo 0 a opção de se começar com o humano e 1 agente. A condição `While True` coordena todo o jogo, sendo a condição de parada (`false`) definida quando algum jogador ganha ou dá velha.

3 Conclusão

Para o presente trabalho apresentou-se o estudo de agentes reativos, especificamente, o simples, em um contexto de jogo da velha. Para conseguir cumprir com objetivo de ter no mínimo 20 percepções, desenvolveu-se um algoritmo que leva em consideração as posições livres no tabuleiro, visando favorecer o agente. Para dificultar que o humano ganhe, mapeou-se todas as jogadas do mesmo, na qual, por meio de várias percepções, é possível escolher uma posição no tabuleiro que era favorável para o seu ganho.

Em vários testes e jogadas, percebeu-se que o algoritmo, de fato, dificulta o vencimento do humano, contudo, por se tratar de um agente reativo simples, algumas estratégias mais elaborada não impede que o humano ganhe.

Referências

COSTA, J. *Inteligência Artificial em jogos de tabuleiro: proposição de uma Heurística para o Jogo de Dominós*. [s.n.], 2016. Disponível em: <<http://dspace.bc.uepb.edu.br/jspui/bitstream/123456789/10078/4/PDF%20-%20Jos%c3%a9%20Aldo%20Silva%20da%20Costa.pdf>>.

Acesso em: 27 mai. 2018. Citado na página [a](#).

RUSSEL S.;NORVIG, P. *Inteligência Artificial*. Brooklin – São Paulo – SP: Campus, 2013. Citado na página [a](#).

STANGE R. L.; CEREDA, P. R. M. J. N. J. Agentes adaptativos reativos: formalização e estudo de caso. Memórias do XI Workshop de Tecnologia Adaptativa–WTA, p. 63–71, 2017. Citado na página [a](#).

ZADROZNY, B. Inteligência artificial ementa agentes inteligentes agentes - ic/uff. Agosto 2010. Disponível em: <<http://www2.ic.uff.br/~bianca/ia/aulas/IA-Aula2.pdf>>. Acesso em: 20 mai. 2018. Citado na página [d](#).