

**Inteligência Artificial**

**Projeto Wumpus Agent**

Nome: André Garcia Dobermann

RA: 201610586

Curso: Engenharia da Computação

# Introdução

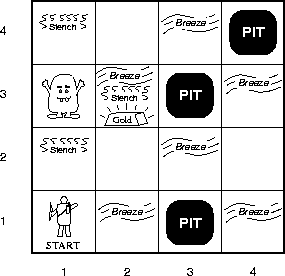
O projeto “Wumpus agent” foi proposto pelo professor Rodrigo Almeida Gonçalves na Facamp relacionada a matéria de inteligência artificial. O projeto consiste na criação de um agente inteligente que consiga resolver o problema do Wumpus, proposto inicialmente pelo livro Artificial Inteligence – A Modern Approach, Stuart Russell e Peter Norvig.

# O problema

O problema do Wumpus consiste basicamente em um jogo onde o jogador controla um explorador com o objetivo de achar ouro em um complexo de cavernas, que podem conter um abismo ou um monstro chamado Wumpus.

O complexo de cavernas é representado por uma grade de 10x10, em que cada célula representa uma caverna, cada caverna pode conter um abismo ou um Wumpus ou o ouro. O jogador pode andar somente em quadro direções, “NORTE”, “SUL”, “OESTE”, “LESTE”, ele também só consegue enxergar a caverna em que ele está, ou seja, o jogador não consegue olhar o que tem nas cavernas adjacentes, a não ser que ele se mova até elas. Porém o jogador pode deduzir o que existe nas cavernas adjacentes baseado no que ele percebe na caverna atual, pois as ameaças existentes como o Wumpus e o abismo possuem algumas características como: todas as cavernas adjacentes a uma caverna que existe um abismo, o jogador pode perceber uma brisa, assim como quando existe um Wumpus em uma caverna, todas as cavernas adjacentes o jogador pode sentir um cheiro ruim como pode ser observado na imagem 1.

Imagem 1: Exemplo de mapa 4x4



Fonte: voidexception.weebly.com/wumpus-world-agent.html

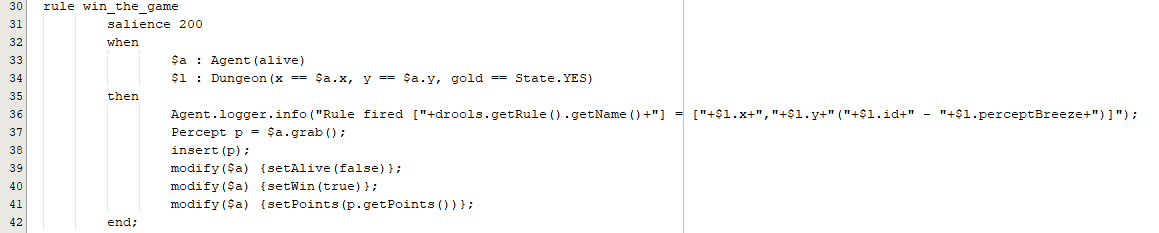
Com essas percepções o jogador pode deduzir o que existe nas cavernas adjacentes e assim ele pode tomar ações, como se mover para uma dessas cavernas, atirar uma flecha em uma dessas cavernas para matar o Wumpus ou arremessar uma pedra para saber se existe um abismo na caverna adjacente ou não.

# Ferramentas

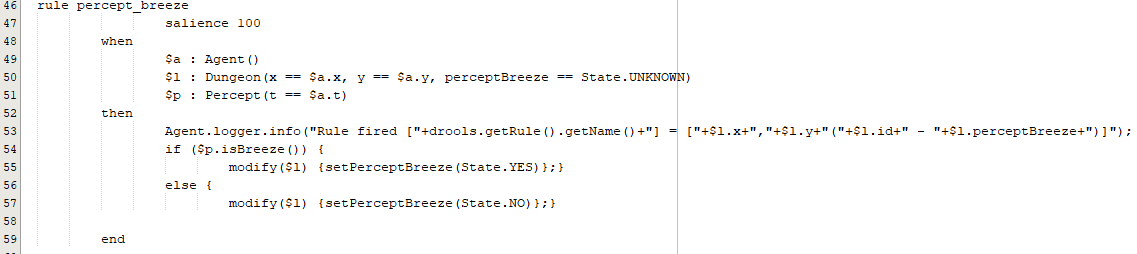
O jogo foi alocado em um servidor particular do professor Rodrigo, assim os alunos poderiam acessa-lo através do endereço “http://66.228.62.78:8080/server-0.5-SNAPSHOT/”. Para acessar inicialmente o servidor foi usado uma plataforma chamada Postman, que possui uma interface amigável e intuitiva para realizar operações com json, assim os alunos poderiam jogar o jogo e entender suas mecânicas e dificuldades, podendo tomar pensar em estratégias que ajudassem no desenvolvimento do agente.

# Regras

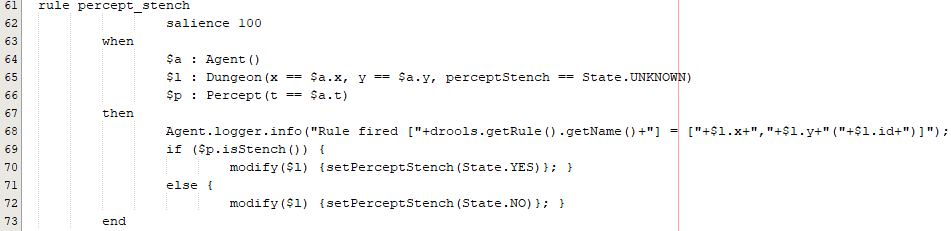
A regra “win\_the\_game”, mostrada na imagem 2, é acionada quando o agente está vivo e a caverna em que ele está possui gold, assim a regra faz com que o agente realize a ação grab, para pegar o gold, e modifica alguns atributos do agente, mudando o estado dele para “não vivo”, estado de que venceu e modifica os pontos somando a pontuação atual com a pontuação anterior acumulada.

Imagem 2: Regra “win\_the\_game”

A regra “percept\_brezze”, mostrado na imagem 3, é acionada quando o agente não sabe se na caverna em que ele está existe brisa ou não, assim a regra busca saber se na resposta do servidor no t atual existe brisa ou não, caso exista ela insere na base de conhecimento que na posição atual existe brisa, caso contrário ela insere que na posição atual não existe brisa.

Imagem 3: Regra “percept brezze”

A regra “percept\_stench”, mostrada na imagem 4, é acionada do mesmo modo que a regra “percept\_brezze”, porem só é válida quando o agente não sabe se na posição atual existe stench, caso exista a regra insere na base de conhecimento que naquela posição existe brisa, caso contrário ela insere que não existe brisa na posição atual.

Imagem 4: Regra “percept\_stench”

A regra “percept\_glitter”, mostrada na imagem 5, é acionada quando o agente não sabe se existe glitter na posição atual, caso a retorne que o gritter for igual a “true” a regra insere na base de conhecimento que existe glitter portanto existe gold, isserindo também na base de conhecimento que existe gold na posição atual, caso o servidor retorne “false” para glitter a regra insere na base de conhecimento que não existe glitter nem gold na posição atual.

A regra “mark\_visited\_and\_register\_I\_survived” é acionada quando a posição atual do agente está marcada como não visitada, então ela verifica se o estado do agente está ativo, e marca a posição x,y como uma posição sem pit nem wumpus.

A regra “realize\_that\_I\_moved\_without\_bumping” é acionada quando o agente realizou um movimento e o servidor retorna bump igual a “false”, alegando que o agente não colidiu com uma parede após o movimento. Após ser acionada a regra insere a nova posição do agente na base de conhecimento e retira o movimento realizado.

A regra “lost” é acionada quando o percept no instante atual retornar que o estado é “LOST”

A regra ” realize\_that\_I\_moved\_bumping” é acionada quando o agente realizou um movimento e o percept retornou bump como “true”, ou seja, quando o agente se move e bate em uma parede. Após ser acionada a regra verifica qual direção que o agente se moveu e delimita as coordenadas máximas e mínimas baseada no movimento. Por exemplo, se o agente se moveu para cima e bateu na parede, ele está no valor máximo da coordenada y, assim ele sabe que não pode existir nenhuma caverna acima desse valor.

A regra “removing\_invalid\_dungeons” é acionada quando a coordenada de uma caverna não condizer com o tamanho do mapa, ou coordenadas máximas e mínimas, caso uma caverna possua valor do eixo x menos do que coordenada mínima, ela é dada como inválida, pois está fora do mapa.

A regra “expand\_horizons\_north” é acionada quando a caverna a norte do agente possui a coordenada y menor ou igual a coordenada máxima no eixo y, ou seja, se é possível ir para o norte, além disso a regra só é acionada se essa caverna ainda não foi explorada, o contrário acontece com a regra “expand\_horizons\_south”, porem nela é verificada se existe uma caverna ao sul da posição atual.

Assim como nas regras “expand\_horizons\_north” e “expand\_horizons\_south” as regras “expand\_horizons\_east” e “expand\_horizons\_west” verificam a existência de caverna em uma determinada direção, onde a regra “expand\_horizons\_east” verifica se existe uma caverna não explorada a leste da posição atual, e o contrário acontece com a regra “expand\_horizons\_west”, que verifica se existe uma caverna não explorada a oeste da posição atual. Assim essas regras adicionam as cavernas existentes na base de conhecimento.

A regra “wumpus\_is\_dead” é acionada quando a o percept diz que existe stench e se em alguma caverna adjacente o wumpus está como estado “dead”, assim a regra modifica a caverna atual como sem stench.

A regra “free\_of\_wumpus” é acionada quando o agente nao perceber stench e se as cavernas adjacentes estão com estado de “unknown” ou “maybe”, assim a regra pode mudar o estado de cada caverna adjacente para sem wumpus. A mesma ideia é aplicada para a regra “free\_of\_pit” que modifica as cavernas adjacentes como sem pit, se na caverna atual não houver brisa e as adjacentes estiverem com estados de “unknow” ou “maybe”.

A regra “maybe\_pit” é acionada quando o agente perceber brisa e as cavernas adjacentes estiverem com estado de “unknown”, ou seja, se não existe nenhuma informação das cavernas adjacentes na base de conhecimento. O mesmo conceito é aplicado para a regra “maybe\_wumpus”, que só é acionada quando o agente perceber stench e não souber o que existe nas células adjacentes.

A regra “explore\_neighborhood\_safe\_walk” é acionada quando o agente estiver em “modo” de explorar a vizinhança e se ele não tiver realizado um movimento no instante anterior, além de as cavernas adjacentes não possuírem pit ou wumpus e não ter sido visitada. Assim a regra cria um movimento para as cavernas adjacentes e as insere na base de conhecimento.