Introdução à Inteligência Artificial

Trabalho Prático II - 2024/2

Igor Lacerda Faria da Silva

1. Introdução

O Trabalho Prático II de Introdução à Inteligência Artificial consistiu na implementação (em Python) de 3 variações do algoritmo *Q-Learning* para *Path-Finding*, em mapas de um jogo simples (o mesmo jogo do TP I). As variações foram: o *Q-Learning* padrão (`standard`); com matriz de recompensa positiva (`positive`) e com movimento estocástico (`stochastic`). O repositório deste trabalho pode ser encontrado neste link.

A documentação está divida da seguinte maneira: esta breve introdução descreve o que foi realizado, a Seção 2 descreve a modelagem do programa, entrando nos detalhes das estruturas de dados; a Seção 3, por sua vez, foca no algoritmo (e variações) implementadas. Por fim, a Seção 4 é uma análise experimental dos diversos algoritmos implementados.

2. Estruturas de Dados e Modelagem

Similarmente ao TP I, a estrutura de dados mais prevalente foi a matriz. As matrizes foram usadas em diversos contextos: para armazenar o mapa (caracteres), as recompensas das posições e, claro, gerenciar a matriz (tensor) Q. A matriz Q é inicializada com valores aleatórios entre -1 e 1 (somente posições válidas são preenchidas). Além disso, outra estrutura auxiliar utilizada ao longo do programa foi uma tupla, para representar as coordenadas no mapa.

Também foi criado um tipo (`enum`) `Action` para auxiliar no mapeamento das direções em que o agente pode se mover. Ele é particularmente útil no que diz respeito à escolha de uma direção perpendicular, na variação estocástica do algoritmo. Assim como no TP I, foi tomado certo cuidado para lidar com o sistema de coordenadas. Após a leitura do mapa, os dados são transpostos e, no final, após o processamento do algoritmo, os dados são novamente transpostos.

No mais, o restante da implementação é bem similar ao pseudocódigo visto nas aulas. Começando do estado inicial, um *loop* escolhe uma ação. Geralmente esta é a melhor ação (para aquele estado), mas devido ao mecanismo de exploração ε-*greedy*, há a chance (10%) de ser alguma outra. A próxima posição é calculada, considerando que é possível não sair do lugar ao tentar realizar um movimento para uma posição inválida.

Caso o agente tenha atingido o estado objetivo (`0`) ou um estado com fogo (`x`), a simulação é reiniciada e a matriz Q é atualizada com base na recompensa do estado atingido. Caso contrário, é buscada a recompensa mais promissora do estado atingido, e é ela que é utilizada para atualizar a recompensa da transição realizada (considerando, claro, a taxa de aprendizado de 0.1 e a taxa de desconto de 0.9). A variação `positive` consiste apenas na troca da matriz de recompensas, enquanto a variação `stochastic` traz nuances que são detalhadas na Seção 3.

Em resumo, o estado contém todas as posições atingíveis; as ações são os 4 movimentos permitidos; o agente sempre se encontra em uma dada posição e a recompensa é calculada pela estimativa inicial (aleatória) e as sucessivas iterações, conforme descrito nos parágrafos anteriores.

3. *Q-Learning*

4. Análise Comparativa