Sokoban

Inteligência Artificial 19/20

85044 - Luís Miguel Costa
89285 - Afonso Boto

Estrutura

4 ficheiros:

- Student.py faz a comunicação com o servidor;
- Agente.py serve de camada de abstração entre o student.py e o tree_search.py, fazendo um pré-processamento do estado inicial;
- Tree_search.py onde a pesquisa é feita efetivamente
- Utils.py contém algumas funções de suporte ao funcionamento do agente

Algoritmo

- Optámos por calcular a posição do keeper e das boxes em conjunto para cada novo estado calculado;
- Usámos o algoritmo A*, recorrendo a um Priority Queue de modo a poupar tempo na ordenação dos nós;
- Como heurística utilizámos uma combinação da distância de manhattan e distância euclidiana em que retornamos como heurística a menor de entre as duas;
- A nível de cálculo de custos, atribuímos custos diferentes a diferentes ações, como por exemplo, mover uma box para um goal ou o keeper dar um passo sem empurrar nenhuma box;
- A nível de deadlocks o agente detecta paredes, cantos e boxes encostadas a paredes que não possuam uma goal no seu alinhamento.

Conclusões

- O uso de um Priority Queue permitiu melhorar bastante a velocidade do A*, poupando tempo na ordenação dos nós abertos;
- O uso das duas heurísticas em conjunto permite obter melhores caminhos;
- Para além disso, se a heurística tomar em conta o custo dos movimentos os resultados obtidos são ainda melhores;