









Sokoban

 REPOSITÓRIO: 93305_93301

 ANA LUÍSA FERREIRA, 93301

 JOÃO DIOGO FERREIRA, 93305

Arquitetura

-  **Representação do mapa:** Cada posição do mapa é representada por um número. A posição no canto superior esquerdo é a zero e aumenta da esquerda para a direita de cima para baixo.
-  **Representação do estado:** A mudança de estado dá-se quando uma caixa muda de sítio.
-  **Deadlocks:** Antes de começar a pesquisa são calculados as posições que são deadlocks, ou seja, as posições em que as caixas não podem estar porque não conseguem ser empurradas para um objetivo.
-  **Frozen Deadlocks:** Em cada estado é calculado se alguma caixa fica presa por estar ao pé de outra caixa. Se isto acontecer o estado não é expandido.
-  **Hash (Zobrist keys):** Consiste em atribuir dois números aleatórios, um para o Keeper e outro para a caixa em cada posição do mapa, e calcular em cada estado o resultado da operação xor em função das posições do Keeper e da caixa.
-  **Evitar a duplicação de estados:** É usado um dicionário que utiliza a nossa função de hash. Estados que já estejam no dicionário são filtrados. Ou seja, estados repetidos são todos os estados em que as posições das caixas são iguais e ,ao mesmo tempo, posições do Keeper em que ele consegue chegar aos mesmos sítios.

Métodos de pesquisa

HEURÍSTICA:

A heurística utilizada retorna o primeiro valor mínimo para a soma das distâncias das caixas aos objetivos, não significa que seja a solução ótima. Não repete caixas nem objetivos. Ex:((caixa, objetivo), distância)

```
[((39, 38), 10), ((39, 40), 10), ((23, 38), 50), ((23, 40), 50)]  
Soma: 60
```

CUSTO:

De acordo com a definição de estado que nós temos, o custo de estado em estado aumenta um, representando a quantidade de pushes efetuadas.

A*:

A pesquisa principal é a A*. A pesquisa usa o custo mais a heurística para ordenar os estados.

GREEDY:

A pesquisa secundária é a Greedy. Este método só usa a heurística para ordenar os estados e só começa a ser utilizada se a pesquisa principal não tiver conseguido arranjar uma solução antes dos 1500 steps contados no nível atual.

Resultados

APLICANDO A*:




Aplicando A*, o agente consegue resolver até ao nível 131, iniciando a sua resolução tarde de mais e assim acabando por perder o nível.

APLICANDO A* + GREEDY:





Usando os dois métodos em conjunto o agente no nível 131, a partir dos 1500 steps troca de estratégia para a Greedy o que lhe permite achar a solução mais cedo, porém o número de pushes não é otimizado. Com a combinação das duas estratégias o agente consegue chegar ao nível 146 sem o completar.

Conclusões

NÃO IMPLEMENTADO:

-  **Túneis:** Tentamos implementar túneis, ou seja, quando uma caixa se encontrava no início de um túnel, como depois mais cedo ou mais tarde o ia percorrer, para evitar a criação desses estados a ideia era fazer com que a caixa fosse logo para o fim do túnel.
-  **Corrais:** Pensamos em implementar, não chegamos a tentar. Ocorre quando há uma zona delimitada por 3 lados com paredes e do outro lado por caixas, tornando o lado oposto ao que o Keeper está inacessível a não ser que se mova uma caixa.
-  Não tentamos otimizar o número de moves.

O QUE DE BOM FOI FEITO:

-  Filtragem de estados duplicados utilizando o hash.
-  É calculado o caminho do Keeper até à caixa e não step a step.
-  A utilização do método Greedy ajuda a completar mais níveis, que não iriam ser feitos, sem nos termos de preocupar com a otimização de pushes que é dada pelo método A*.
-  As distâncias de cada posição a cada objetivo são calculadas no início do nível o que facilita a utilização destes valores na heurística.