UNIVERSIDADE
DE ÉVORA

# IA - 1º Trabalho

**Trabalho realizado por:**

- Rui Roque nº42720
- Tomás Dias nº42784

# Respostas às perguntas do 1º problema

| | | | | | | |
|---|---|---|---|---|---|---|
| **0** | **0** | **X** | **0** | **F** | **0** | **0** |
| **X** | **0** | **X** | **0** | **0** | **0** | **X** |
| **0** | **0** | **0** | **0** | **0** | **0** | **0** |
| **0** | **0** | **0** | **X** | **0** | **0** | **0** |
| **0** | **0** | **0** | **X** | **0** | **0** | **0** |
| **0** | **0** | **0** | **X** | **0** | **0** | **0** |
| **0** | **I** | **0** | **0** | **0** | **0** | **0** |

I - Estado inicial do agente
F - Estado final do agente
X - Casas que o agente não pode ocupar.
0 - Casas que o agente pode ocupar.

**a)** Represente em Prolog o espaço de estados e os operadores de transição de estados para este problema.

Representação do estado: *estado(posição do agente)*

Representação dos estados inicial e final do problema:
estado_inicial((7,2))
estado_final((1,5))

Representação do operador: *operador(estado_atual, op, estado_seguinte, custo)*

O custo é incrementado +1 por cada movimento feito.

Operações transição de estados/ações:

- Movimento para cima: op é um tuplo em que $x \in [-1,-6]$ e $y = 0$
- Movimento para baixo: op é um tuplo em que $x \in [1,6]$ e $y = 0$
- Movimento para a esquerda: op é um tuplo em que $x = 0$ e $y \in [-1,-6]$
- Movimento para a direita: op é um tuplo em que $x = 0$ e $y \in [1,6]$

**Agente pode mover-se até 6 casas para qualquer direção, desde que não saia do tabuleiro ou que não ocupe uma casa ocupada.**

**b)** Apresente o código em Prolog do algoritmo de pesquisa não informada mais eficiente a resolver este problema.

O algoritmo de pesquisa não informada mais eficiente dos testados foi a **pesquisa em largura**.

O código em Prolog encontra-se no ficheiro **problema1.pl**.

**c)** Depois de resolver este problema com o algoritmo da alínea anterior indique:

> **I.** Qual o número total (exacto) de estados visitados? Foram visitados **2** estados (o número de nós visitados foi **41**).
> **II.** Qual o número máximo (exacto) de estados que têm que estar simultaneamente em memória? **145** estados.

Para o outro algoritmo testado, a **pesquisa em profundidade**, foram visitados **35** estados (o número de nós visitados foi de **39**) e o número máximo de estados simultaneamente em memória foi **126** estados.

**Nota**: no teste da pesquisa iterativa, o número de nós visitados (207) foi superior ao número de estados em memória (22).

**d)** Proponha duas heurísticas admissíveis para estimar o custo de um estado até à solução para este problema.

As heurísticas utilizadas foram:

- heurística(estado_inicial, resultado)
  resultado = distância(estado_inicial, estado_final) / 3.

- heurística(estado_inicial, 0), que no caso da utilização da pesquisa a* será igual à pesquisa em largura.

O código das heurísticas em Prolog encontra-se no ficheiro **problema1.pl**.

**Nota**: Não foi feita a divisão da distância por um porque de acordo com a nossa interpretação do problema o agente não anda apenas uma casa, pode andar entre uma a seis casas desde que não ultrapasse os limites do tabuleiro ou que passe por uma casa ocupada.

**e)** Apresente o código em Prolog do algoritmo de pesquisa informada mais eficiente para resolver este problema usando as heurísticas definidas na alínea anterior.

O algoritmo de pesquisa informada mais eficiente dos testados foi a **pesquisa a***.

O código em Prolog encontra-se no ficheiro **problema1.pl**.

Para executar as pesquisas informadas deve usar o predicado **pesquisaI**.
Exemplo: pesquisaI(problema1, a) para a pesquisa a*.

**f)** Depois de resolver este problema com o algoritmo da alínea anterior indique para cada função heurística:

- **I.** Qual o número total (exacto) de estados visitados? Foram visitados **2** estados (o número de nós visitados foi **12**)
- **II.** Qual o máximo número (exacto) de estados que têm que estar simultaneamente em memória? **78** estados.

Para o outro algoritmo testado, a **pesquisa greedy**, foram visitados **3** estados (o número de nós visitados foi **4**) e o número máximo de estados simultaneamente em memória foi **38** estados.

Usando a heurística com valor 0, o número de estados visitados e o número de estados em memória foi igual aos obtidos na pesquisa em largura.

# Respostas às perguntas do 2º problema

| | | | | | | |
|---|---|---|---|---|---|---|
| **0** | **0** | **X** | **0** | **F** | **0** | **0** |
| **X** | **0** | **X** | **0** | **0** | **0** | **X** |
| **0** | **0** | **0** | **0** | **0** | **0** | **0** |
| **0** | **0** | **0** | **X** | **0** | **0** | **0** |
| **0** | **0** | **0** | **X** | **0** | **0** | **0** |
| **0** | **0** | **0** | **X** | **0** | **0** | **0** |
| **0** | **I** | **0** | **0** | **0** | **0** | **0** |

I - Estado inicial do agente
F - Estado final do agente
X - Casas que o agente não pode ocupar.
0 - Casas que o agente pode ocupar.

**a)** Represente em Prolog o espaço de estados e os operadores de transição de estados para este problema.

Representação do estado: *estado([posição do agente, posição da caixa])*

Representação dos estados inicial e final do problema:
estado_inicial([(7,2), (6,2)])
estado_final([(_,_), (1,5)])

Representação do operador: *operador(estado_atual, op, estado_seguinte, custo)*

O custo é incrementado +1 por cada movimento feito pelo agente.

Operações transição de estados/ações:

- Movimento para cima: op é um tuplo em que $x \in [-1,-6]$ e $y = 0$
- Movimento para baixo: op é um tuplo em que $x \in [1,6]$ e $y = 0$
- Movimento para a esquerda: op é um tuplo em que $x = 0$ e $y \in [-1,-6]$
- Movimento para a direita: op é um tuplo em que $x = 0$ e $y \in [1,6]$

**O Agente pode mover-se até 6 casas para qualquer direção, desde que não saia do tabuleiro ou que não ocupe uma casa ocupada, equanto que a caixa só anda um e para a direção onde o agente vai, não podendo esta ocupar também uma casa ocupada ou sair do tabuleiro.**

## **b)** Apresente o código em Prolog do algoritmo de pesquisa não informada mais eficiente a resolver este problema.

O algoritmo de pesquisa não informada mais eficiente dos testados foi a **pesquisa em largura**.

O código em Prolog encontra-se no ficheiro **problema2.pl**.

## **c)** Depois de resolver este problema com o algoritmo da alínea anterior indique:

> **I.** Qual o número total (exacto) de estados visitados? Foram visitados **8** estados (o número de nós visitados foi **9616**).
> **II.** Qual o número máximo (exacto) de estados que têm que estar simultaneamente em memória? **20031** estados.

Para o outro algoritmo testado, a **pesquisa em profundidade**, foram visitados **35** estados (o número de nós visitados foi de **39**) e o número máximo de estados simultaneamente em memória foi **126** estados.

**Nota**: para testar as pesquisas em largura e profundidade deve alterar o valor de GLOBALSZ para 652000 e 1752000 respectivamente.

## **d)** Proponha duas heurísticas admissíveis para estimar o custo de um estado até à solução para este problema.

As heurísticas utilizadas foram:

- heurística(estado inicial da caixa, resultado)
  resultado = distância(estado inicial da caixa, estado final da caixa) / 1.

O código das heurísticas em Prolog encontra-se no ficheiro **problema2.pl**.

**Nota**: Como o que interessa é a distância do estado inicial da caixa ao estado final da caixa, e a caixa anda sempre de uma em uma casa, a distância foi dividida por um.

**e)** Apresente o código em Prolog do algoritmo de pesquisa informada mais eficiente para resolver este problema usando as heurísticas definidas na alínea anterior.

O algoritmo de pesquisa informada mais eficiente dos testados foi a **pesquisa g**.

O código em Prolog encontra-se no ficheiro **problema2.pl**.

Para executar as pesquisas informadas deve usar o predicado **pesquisaI**.
Exemplo: pesquisaI(problema2, g) para a pesquisa greedy.

**f)** Depois de resolver este problema com o algoritmo da alínea anterior indique para cada função heurística:

- **I.** Qual o número total (exacto) de estados visitados? Foram visitados **8** estados (o número de nós visitados foi **131**)
- **II.** Qual o número máximo (exacto) de estados que têm que estar simultaneamente em memória? **481** estados.

Para o outro algoritmo testado, a **pesquisa a***, foram visitados **8** estados (o número de nós visitados foi **464**) e o número máximo de estados simultaneamente em memória foi **1425** estados.