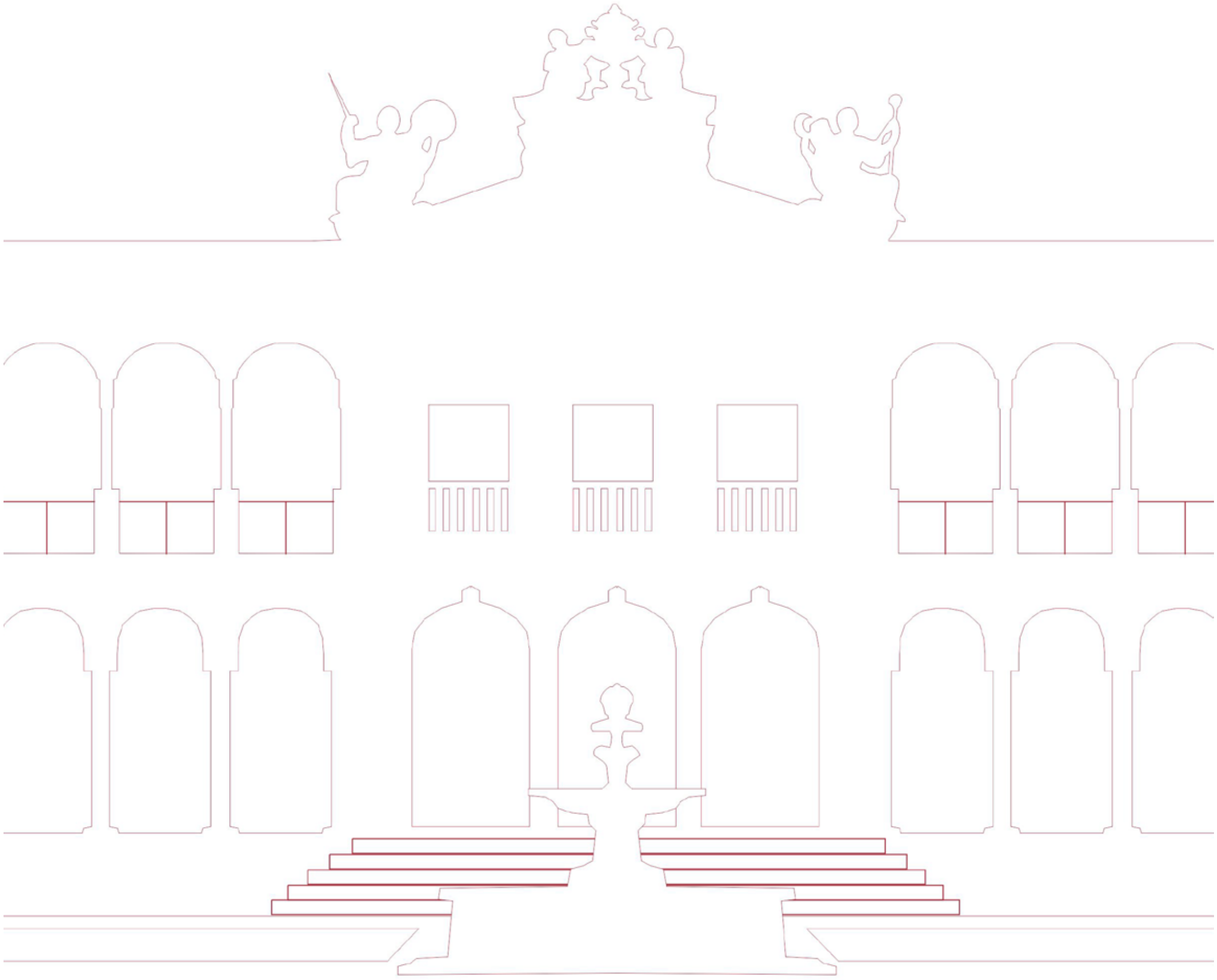
1º Trabalho Prático
Resolução de Problemas

Licenciatura em Eng. Informática

Inteligência Artificial

UNIVERSIDADE DE ÉVORA
ESCOLA DE CIÊNCIAS E TECNOLOGIA
DEPARTAMENTO DE INFORMÁTICA



André Baião 48092
Gonçalo Barradas 48402
Guilherme Grilo 48921

Docente: Irene Pimenta

1 Problema 1

1. (a)

```
estado_inicial(e(6, 1)).
estado_final(e(0, 4)).
```

A lista abaixo contém as casas bloqueadas com um "X".

```
bloqueada([(3,1),(3,2),(3,3),(0,5),(2,5),(6,5),(2,6)]).
```

O seguinte troço de código verifica se as casas em questão são válidas.

```
dentro_tabuleiro((X,Y)):- X@>=0, X@=<6, Y@>=0, Y@=<6.
```

Seguem-se os operadores.

```
op(e(X,Y), dir, e(W,Y), 1):- W is X+1, dentro_tabuleiro((W,Y)), bloqueada(L),
    \+member((W,Y),L).

op(e(X,Y), esq, e(W,Y), 1):- W is X-1, dentro_tabuleiro((W,Y)), bloqueada(L),
    \+member((W,Y),L).

op(e(X,Y), cima, e(X,W), 1):- W is Y+1, dentro_tabuleiro((X,W)), bloqueada(L),
    \+member((X,W),L).

op(e(X,Y), baixo, e(X,W), 1):- W is Y-1, dentro_tabuleiro((X,W)), bloqueada(L),
    \+member((X,W),L).
```

(b) Segundo os testes realizados podemos concluir que o algoritmo mais eficiente para a resolução deste exercício é a pesquisa em profundidade.

```
pesquisa_profundidade([no(E, Pai, Op, C, P)|_],no(E, Pai, Op, C, P)):-
    estado_final(E), inc.

pesquisa_profundidade([E|R], Sol):- inc,
                                    asserta(fechado(E)),
                                    expande(E, Lseg),
                                    insere_fim(R, Lseg, Resto),
                                    length(Resto, N),
                                    actmax(N),
                                    pesquisa_profundidade(Resto, Sol).
```

(c) i. O número de estados visitados é 23.

ii. O número máximo de estados que se encontram simultaneamente em memória é 17.

(d)

```
%Manhattan
h_manhattan(e(X1,Y1),e(X2,Y2),D) :-
    DeltaX is abs(X1-X2),
    DeltaY is abs(Y1-Y2),
    D is DeltaX + DeltaY.

%Euclidiana
h_euclidiana(e(X1,Y1),e(X2,Y2),D):-
    DeltaX is X1-X2,
    DeltaY is Y1-Y2,
    D is round(sqrt((DeltaX*DeltaX - DeltaY*DeltaY))).
```

(e)

```
pesquisa_g([no(E, Pai, Op, C, HC, P)|_], no(E, Pai, Op, C, HC, P)):-
    estado_final(E), inc.

pesquisa_g([E|R], Sol):- inc,
    asserta(fechado(E)), expande_g(E, Lseg),
    insere_ord(Lseg, R, Resto), length(Resto, N),
    actmax(N), pesquisa_g(Resto, Sol).
```

(f)

Algoritmo	Heurística	Nós visitados	Nós em memória	Profundidade	Custo
Greedy	Manhattan	10	12	9	9
A*	Euclidiana	132	59	9	9
A*	Manhattan	60	43	9	9
Greedy	Euclidiana	112	59	9	9

Tabela 1: Resposta às alineas f) i) e f) ii).

2 Problema 2

1. (a)

```
estado_inicial(p(2, 7, 2, 6)).
estado_final(p(_, _, 5, 1)).
```

A lista seguinte contém as casas bloqueadas com um "X".

```
casa_bloq(p(1,2)).
casa_bloq(p(3,1)).
casa_bloq(p(3,2)).
casa_bloq(p(4,4)).
casa_bloq(p(4,5)).
casa_bloq(p(4,6)).
casa_bloq(p(7,2)).
```

Seguem-se os operadores.

```
op(p(X,Y,P,Q),cima,p(X,Y1,P,Q1),1):-
    Y1 is Y - 1,
    (iguais(p(X, Y1), p(P, Q)) ->
        ( Q1 is Q - 1,
            lim(P, Y1),
            \+ casa_bloq(p(P, Q1))
        );
        ( Q1 is Q,
            lim(X, Y1),
            \+ casa_bloq(p(X, Y1))
        )
    ).
op(p(X,Y,P,Q),direita,p(X1,Y,P1,Q),1) :-
    X1 is X+1,
    (iguais(p(X1,Y),p(P,Q)) ->
        ( P1 is P+1,
            lim(P1,Q),
            lim(X1,Y),
            \+ casa_bloq(p(P1,Q))
        );
        ( P1 is P,
            lim(X1,Y),
            \+ casa_bloq(p(X1,Y))
        )
    ).
op(p(X,Y,P,Q),baixo,p(X,Y1,P,Q1),1) :-
    Y1 is Y+1,
    (iguais(p(X,Y1),p(P,Q)) ->
        (
            Q1 is Q+1,
            lim(P,Q1),
            lim(X,Y1),
            \+ casa_bloq(p(P,Q1))
        );
        ( Q1 is Q,
            lim(X,Y1),
            \+ casa_bloq(p(X,Y1))
        )
    ).
```

```
    op(p(X,Y,P,Q),esquerda,p(X1,Y,P1,Q),1) :-
X1 is X-1,         (iguais(p(X1,Y),p(P,Q)) ->
            ( P1 is P-1,
                lim(P1,Q),
                lim(X1,Y),
                \+ casa_bloq(p(P1,Q))
            );
            ( P1 is P,
                lim(X1,Y),
                \+ casa_bloq(p(X1,Y))
            )
        ).
```

(b) Segundo os testes realizados podemos concluir que o algoritmo mais eficiente para a resolução deste exercício é a pesquisa em profundidade.

```
    pesquisa_profundidade([no(E, Pai, Op, C, P)|_],no(E, Pai, Op, C, P)):-
                estado_final(E), inc.

    pesquisa_profundidade([E|R], Sol):- inc,
                asserta(fechado(E)),
                expande(E, Lseg),
                insere_fim(R, Lseg, Resto),
                length(Resto, N),
                actmax(N),
                pesquisa_profundidade(Resto, Sol).
```

(c) i. O número total de estados visitados é 250.

ii. O número máximo de estados que se encontra simultaneamente em memória é 50.

(d) Abaixo encontram-se as heurísticas utilizadas para este problema. A heuristica `h1` corresponde ao cálculo da distância de Manhattan e a heuristica `h2` corresponde ao cálculo da diferença entre os y's.

```
    h1(p(_,_,Ix,Iy),SOMA):-
                    estado_final(p(_,_,Fx,Fy)),
                    Dx is abs(Ix - Fx),
                    Dy is abs(Iy - Fy),
                    SOMA is Dx + Dy.

    h2(p(_,_, _,Iy),SOMA):-
                    estado_final(p(_,_,_,Fy)),
                    Dy is abs(Iy - Fy),
                    SOMA is Dy.
```

(e) Após alguns testes, concluímos que a melhor pesquisa para este problema, com as duas heurísticas é o algoritmo de Greedy. Estes testes podem ser verificados na tabela da alínea abaixo.

```
    pesquisa_g([no(E, Pai, Op, C, HC, P)|_], no(E, Pai, Op, C, HC, P)):-
                estado_final(E), inc.

    pesquisa_g([E|R], Sol):- inc,
                asserta(fechado(E)),
                expande_g(E, Lseg),
                insere_ord(Lseg, R, Resto),
                length(Resto, N),
                actmax(N),
                pesquisa_g(Resto, Sol).
```

Após os testes podemos concluir também que a pesquisa iterativa demora imenso tempo a encontrar uma solução para este problema.

(f)

Algoritmo	Heurística	Nós visitados	Nós em memória	Profundidade	Custo
Greedy	Manhattan	468	54	16	16
A*	distância dos y	2305	513	16	16
A*	Manhattan	1203	459	16	16
Greedy	distância dos y	2098	198	16	16

Tabela 2: Resposta às alineas f) i) e f) ii).

3 Comandos

- De forma a executar o algoritmo de pesquisa em largura para o exercício 1, executar o comando `[pni].` no gprolog e de seguida executar `pesquisa(ex1, largura).` .
- De forma a executar o algoritmo de pesquisa em profundidade para o exercício 1, executar o comando `[pni].` no gprolog e de seguida executar `pesquisa(ex1, profundidade).` .
- De forma a executar o algoritmo de Greedy com a heurística de Manhattan para o exercício 1, colocar o predicado do ficheiro `ex1.pl` como `h(E, V) :- estado_final(Ef), h_manhattan(E, Ef, V).`, de seguida executar o comando `[pi].` no gprolog e finalmente executar `pesquisa(ex1, g).` .
- De forma a executar o algoritmo de Greedy com a heurística Euclidiana para o exercício 1, colocar o predicado do ficheiro `ex1.pl` como `h(E, V) :- estado_final(Ef), h_euclidiana(E,Ef,V).`, de seguida executar o comando `[pi].` no gprolog e finalmente executar `pesquisa(ex1, g).` .
- De forma a executar o algoritmo A* com a heurística de Manhattan para o exercício 1, colocar o predicado do ficheiro `ex1.pl` como `h(E, V) :- estado_final(Ef), h_manhattan(E,Ef,V).`, de seguida executar o comando `[pi].` no gprolog e finalmente executar `pesquisa(ex1, a).` .
- De forma a executar o algoritmo A* com a heurística Euclidiana para o exercício 1, colocar o predicado do ficheiro `ex1.pl` como `h(E, V) :- estado_final(Ef), h_euclidiana(E,Ef,V).`, de seguida executar o comando `[pi].` no gprolog e finalmente executar `pesquisa(ex1, a).` .
- De forma a executar o algoritmo de pesquisa em largura para o exercício 2, executar o comando `[pni].` no gprolog e de seguida executar `pesquisa(ex2, largura).` .
- De forma a executar o algoritmo de pesquisa em profundidade para o exercício 2, executar o comando `[pni].` no gprolog e de seguida executar `pesquisa(ex2, profundidade).` .
- De forma a executar o algoritmo de Greedy com a heurística de Manhattan para o exercício 2, colocar o predicado do ficheiro `ex2.pl` como `h(A, B) :- h1(A, B).`, de seguida executar o comando `[pi].` no gprolog e finalmente executar `pesquisa(ex2, g).` .
- De forma a executar o algoritmo de Greedy com a heurística distância dos y para o exercício 2, colocar o predicado do ficheiro `ex2.pl` como `h(A, B) :- h2(A, B).`, de seguida executar o comando `[pi].` no gprolog e finalmente executar `pesquisa(ex2, g).` .
- De forma a executar o algoritmo A* com a heurística de Manhattan para o exercício 2, colocar o predicado do ficheiro `ex2.pl` como `h(A, B) :- h1(A, B).`, de seguida executar o comando `[pi].` no gprolog e finalmente executar `pesquisa(ex2, a).` .
- De forma a executar o algoritmo A* com a heurística distância dos y para o exercício 2, colocar o predicado do ficheiro `ex2.pl` como `h(A, B) :- h2(A, B).`, de seguida executar o comando `[pi].` no gprolog e finalmente executar `pesquisa(ex2, a).` .