

1º Trabalho Prático

Resolução de Problemas

Licenciatura em Eng. Informática
Inteligência Artificial



André Baião 48402
Gonçalo Barradas 48402
Guilherme Grilo 48921
Docente: Irene Pimenta



1 Problema 1

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

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

```
casas_bloq([(0, 2), (1, 0), (1, 2), (1, 6), (3, 3), (4, 3), (5, 3)]).
```

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

```
pos_valida((X, Y)) :- casas_bloq(L), dentro_sala((X, Y)), \+member((X, Y), L).  
dentro_sala((X, Y)) :- X>=0, X<=6, Y>=0, Y<=6.
```

Seguem-se os operadores.

```
op((X, Y), sobe, (Z, Y), 1) :- Z is X-1, pos_valida((Z, Y)).  
op((X, Y), desce, (Z, Y), 1) :- Z is X+1, pos_valida((Z, Y)).  
op((X, Y), esq, (X, Z), 1) :- Z is Y-1, pos_valida((X, Z)).  
op((X, Y), dir, (X, Z), 1) :- Z is Y+1, pos_valida((X, Z)).
```

- (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 26.
ii. O número máximo de estados que se encontram simultaneamente em memória é 18.

- (d) **%Manhattan**

```
h1((A,B),C):-  
    estado_final((X,Y)),  
    X1 is abs(A - X),  
    Y1 is abs(B - Y),  
    C is X1 + Y1.
```

%Euclidiana

```
h2((Ix,Iy),SOMA):-  
    estado_final((Fx,Fy)),  
    Dx is abs(Ix - Fx),  
    Dy is abs(Iy - Fy),  
    SOMA is round(sqrt(Dy ** 2 + Dx ** 2)).
```

- (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).
```



	Algoritmo	Heurística	Nós visitados	Nós em memória	Profundidade	Custo
	Greedy	Manhattan	12	14	9	9
(f)	A*	Euclidiana	66	39	9	9
	A*	Manhattan	60	43	9	9
	Greedy	Euclidiana	12	15	9	9

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

2 Problema 2

- (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) teste

(e)

	Algoritmo	Heurística	Nós visitados	Nós em memória	Profundidade	Custo
	Greedy	Manhattan	468	54	16	16
(f)	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 alíneas f) i) e f) ii).

Após os testes podemos concluir também que a pesquisa iterativa não encontra uma solução para este problema.

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(A, B) :- h1(A, B).`, 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(A, B) :- h2(A, B).`, 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(A, B) :- h1(A, B).`, 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(A, B) :- h2(A, B).`, 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).` .