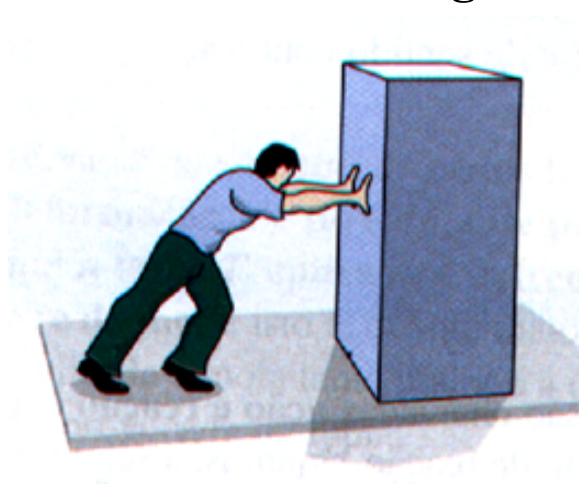


Escola de Ciências e Tecnologia
Departamento de Informática
Licenciatura em Engenharia Informática
Unidade curricular Inteligência Artificial
Ano letivo 2020/2021

Relatório

1º Trabalho Prático de Inteligência Artificial



Docentes

Professora Irene Pimenta Rodrigues

Discentes

José Santos, nº43017

Ludgero Teixeira, nº41348

Pedro Claudino nº39870

Évora, abril de 2021

Exercício 1

Alínea a)

```
1  listaX([(1,2),(3,1),(3,2),(4,4),(4,5),(4,6),(7,2)]).
2
3  % estado inicial
4  estado_inicial((2,7)).
5
6  % estado final
7  estado_final((5,1)).
8
9  % Verificar se está dentro dos limites do tabuleiro
10
11 limites(N,M) :- N>0, M>0, N<8, M<8, listaX(Lx), \+member((N,M),Lx).
12
13 % (X,Y) - posicao do A
14 % Lx     - lista de X's
15 % Ef     - estado final
16 % (Nx,Ny) - New X, NewY
17
18 op((X,Y), (0,M), (X,Ny), 1) :- member(M, [-1,1]),
19 |                               Ny is Y+M,
20 |                               limites(X,Ny).
21
22 op((X,Y), (N,0), (Nx,Y), 1) :- member(N, [-1,1]),
23 |                               Nx is X+N,
24 |                               limites(Nx,Y).
```

Alínea b)

➤ Pesquisa iterativa (Profundidade = 9)

```

1  :- dynamic(fechado/1).
2  :- dynamic(expandido/1).
3  :- dynamic(maxNL/1).
4  :- dynamic(nos/1).
5
6  maxNL(0).
7  nos(0).
8
9  inc:- retract(nos(N)), N1 is N+1, asserta(nos(N1)).
10
11 actmax(N):- maxNL(N1), N1 >= N,!.
12 actmax(N):- retract(maxNL(_N1)), asserta(maxNL(N)).
13
14 pesquisa(Problema,Alg):-
15     consult(Problema),
16     estado_inicial(S0),
17     pesquisa(Alg,[no(S0,[],[],0,0)],Solucão),
18     escreve_seq_solucão(Solucão),
19     retract(nos(Ns)),retract(maxNL(NL)),retractall(fechado(_)), retractall(expandido(_)),
20     asserta(nos(0)),asserta(maxNL(0)),
21     write(nos(visitados(Ns),lista(NL))).
22
23 pesquisa_it(Ln,Sol,P):- retractall(fechado(_)) ,pesquisa_pLim(Ln,Sol,P).
24 pesquisa_it(Ln,Sol,P):- P1 is P+1, pesquisa_it(Ln,Sol,P1).
25
26 pesquisa(it,Ln,Sol):- pesquisa_it(Ln,Sol,1).
27
28 expandePl(no(_,_,_,_),P,[],Pl):- Pl <= P, !.
29 expandePl(no(E,Pai,Op,C,P),L,_):- findall(no(En,no(E,Pai,Op,C,P),Opn,Cnn,P1),
30                                     (op(E,Opn,En,Cn), \+ fechado(no(En,_,_,_)) ,P1 is P+1, Cnn is Cn+C),
31                                     L).
32 insere_fim([],L,L).
33 insere_fim(L,[],L).
34 insere_fim(R,[A|S],[A|L]):- insere_fim(R,S,L).
35
36 pesquisa_pLim([no(E,Pai,Op,C,P)|_],no(E,Pai,Op,C,P),_):- estado_final(E), inc. %incrementar depois de estado_final
37
38 pesquisa_pLim([E|R],Sol,Pl):- inc, assertz(fechado(E)) ,expandePl(E,Lseg,Pl), %esc(E), %incrementar antes do expandePl
39                                     insere_fim(R,Lseg,Resto),
40                                     length(Resto,N), actmax(N), % calcular o tamanho de resto e actualizar maximo
41                                     pesquisa_pLim(Resto,Sol,Pl).
42
43
44 escreve_seq_solucão(no(E,Pai,Op,Custo,Prof)):- write(custo(Custo)),nl,
45                                     write(profundidade(Prof)),nl,
46                                     escreve_seq_accões(no(E,Pai,Op,_,_)).
47
48 escreve_seq_accões([]).
49 escreve_seq_accões(no(E,Pai,Op,_,_)):- escreve_seq_accões(Pai),
50                                     write(e(Op,E)),nl.
51
52 esc(A):- write(A), nl.

```

Alínea c)

	Largura	Profundidade	Profundidade Iterativa
Nós visitados	102	45	313
Número máximo de estados simultaneamente em memória	19	31	11

(todos os testes foram feitos com os exemplos do enunciado)

Alínea d)

```

1  h(E,Val) :- h1(E,Val1), h2(E,Val2), min(Val1,Val2,Val).
2
3  max(A, B, A) :- A>B, !.
4  max(_, B, B).
5
6  min(A,B,A) :- A<B, !.
7  min(_,B,B).
8
9  % módulo da subtração
10 modSub(A, B, Res) :- max(A, B, Max), min(A, B, Min), Res is Max - Min.
11
12 % Heurística 1 - Distância à casa final
13 h1(E,Val) :- estado_final((Fx,Fy)),
14     E=(Cx,Cy),
15     modSub(Fx, Cx, DistX),
16     modSub(Fy, Cy, DistY),
17     Val is (DistX + DistY).
18
19 % Heurística 2 - Distância no eixo do X à casa final
20 h2(E,Val) :- estado_final((Fx,_)),
21     E = (Cx,_),
22     modSub(Fx, Cx, Val).

```

Alínea e)

➤ Greedy Search (Profundidade = 9)

```

1  :- dynamic(fechado/1).
2  :- dynamic(maxNL/1).
3  :- dynamic(nos/1).
4
5      maxNL(0).
6      nos(0).
7
8      inc:- retract(nos(N)), N1 is N+1, asserta(nos(N1)).
9
10     actmax(N):- maxNL(N1), N1 >= N,!.
11     actmax(N):- retract(maxNL(_N1)), asserta(maxNL(N)).
12
13 pesquisa(Problema,Alg):-
14     consult(Problema),
15     estado_inicial(S0),
16     pesquisa(Alg,[no(S0,[],[],0,1,0)],Solucao),
17     escreve_seq_solucao(Solucao),
18     retract(nos(Ns)),retract(maxNL(NL)),retractall(fechado(_)),
19     asserta(nos(0)),asserta(maxNL(0)),
20     write(nos(visitados(Ns),lista(NL))).
21
22 pesquisa(g,E,S):- pesquisa_g(E,S).
23
24 pesquisa_g([no(E,Pai,Op,C,HC,P)|_],no(E,Pai,Op,C,HC,P)):- estado_final(E), inc.
25
26 pesquisa_g([E|R],Sol):- inc, asserta(fechado(E)), expande_g(E,Lseg),
27     insere_ord(Lseg,R,Resto),length(Resto,N), actmax(N),
28     pesquisa_g(Resto,Sol).
29
30 expande_g(no(E,Pai,Op,C,HC,P),L):- findall(no(En,no(E,Pai,Op,C,HC,P),Opn,Cnn,H,P1),
31     (op(E,Opn,En,Cn),
32     \+ fechado(no(En,_,_,_,_))),
33     P1 is P+1, Cnn is Cn+C, h(En,H)), L).
34
35 insere_ord([],L,L).
36 insere_ord([A|L],L1,L2):- insereE_ord(A,L1,L3), insere_ord(L,L3,L2).
37
38 insereE_ord(A,[],[A]).
39 insereE_ord(A,[A1|L],[A,A1|L]):- menor_no(A,A1),!.
40 insereE_ord(A,[A1|L],[A1|R]):- insereE_ord(A,L,R).
41
42 menor_no(no(_,_,_,_,N,_), no(_,_,_,_,N1,_)):- N < N1.
43
44 escreve_seq_solucao(no(E,Pai,Op,Custo,_HC,Prof)):- write(custo(Custo)),nl,
45     write(profundidade(Prof)),nl,
46     escreve_seq_accos(no(E,Pai,Op,_,_,_)).
47
48
49 escreve_seq_accos([]).
50 escreve_seq_accos(no(E,Pai,Op,_,_,_)):- escreve_seq_accos(Pai),
51     write(e(Op,E)),nl.
52
53 esc(A):- write(A), nl.

```

Alínea f)

	A*	Greedy
Nós visitados	81	10
Número máximo de estados simultaneamente em memória	30	12

(todos os testes foram feitos com os exemplos do enunciado)

Exercício 2

Alínea a)

```

1  listaX([(1,2),(3,1),(3,2),(4,4),(4,5),(4,6),(7,2)]).
2
3  % estado_inicial
4  estado_inicial(((2,7), (2,6))).
5
6  % estado_final
7  estado_final( (_, (5,1))).
8
9  % Verificar se está dentro dos limites do tabuleiro
10
11 limites(N,M) :- N>0, M>0, N<8, M<8, listaX(Lx), \+member((N,M),Lx).
12
13 op(((Xa, Ya),(Xc, Yc)), (N,0), ((Nxa, Ya),(Nxc, Yc)), 1) :- member(N, [-1,1]),
14                                     Nxa is Xa+N,
15                                     ((Nxa,Ya) = (Xc,Yc) -> Nxc is Xc+N ; Nxc is Xc),
16                                     limites(Nxa, Ya), limites(Nxc, Yc),
17                                     verificarX((Nxa, Ya)), verificarX((Nxc, Yc)).
18
19 op(((Xa, Ya),(Xc, Yc)), (0,M), ((Xa, Nya),(Xc, Nyc)), 1) :- member(M, [-1,1]),
20                                     Nya is Ya+M,
21                                     ((Xa,Nya) = (Xc,Yc) -> Nyc is Yc+M ; Nyc is Yc ),
22                                     limites(Xa, Nya), limites(Xc, Nyc),
23                                     verificarX((Xa, Nya)), verificarX((Xc, Nyc)).

```

Alínea b)

➤ Pesquisa em largura (Profundidade = 16)

```

1  :- dynamic(fechado/1).
2  :- dynamic(expandido/1).
3  :- dynamic(maxNL/1).
4  :- dynamic(nos/1).
5
6  maxNL(0).
7  nos(0).
8
9  inc:- retract(nos(N)), N1 is N+1, asserta(nos(N1)).
10
11 actmax(N):- maxNL(N1), N1 >= N,!.
12 actmax(N):- retract(maxNL(_N1)), asserta(maxNL(N)).
13
14
15 pesquisa(Problema,Alg):-
16     consult(Problema),
17     estado_inicial(S0),
18     pesquisa(Alg,[no(S0,[],[],0,0)],Solucao),
19     escreve_seq_solucao(Solucao),
20     retract(nos(Ns)),retract(maxNL(NL)),retractall(fechado(_)), retractall(expandido(_)),
21     asserta(nos(0)),asserta(maxNL(0)),
22     write(nos(visitados(Ns)),lista(NL)).
23
24 pesquisa(largura,Ln,Sol):- pesquisa_largura(Ln,Sol).
25
26 pesquisa_largura([no(E,Pai,Op,C,P)|_],no(E,Pai,Op,C,P)):- estado_final(E), inc.
27
28 pesquisa_largura([E|R],Sol):- inc, expande(E,Lseg), %esc(E),
29                               insere_fim(Lseg,R,Resto),
30                               length(Resto,N), actmax(N),
31                               pesquisa_largura(Resto,Sol).
32
33
34
35 % Para evitar ciclos infinitos, usamos o assert e o fechado
36 expande(no(E,_,_,_),[]):- fechado(E), !.
37
38 expande(no(E,Pai,Op,C,P),L):- assertz(fechado(E)),
39                               findall(no(En,no(E,Pai,Op,C,P),Opn,Cnn,P1),
40                               (op(E,Opn,En,Cn), \+ fechado(no(En,_,_,_)) ,P1 is P+1, Cnn is Cn+C),
41                               L).
42 insere_fim([],L,L).
43 insere_fim(L,[],L).
44 insere_fim(R,[A|S],[A|L]):- insere_fim(R,S,L).
45
46 escreve_seq_solucao(no(E,Pai,Op,Custo,Prof)):- write(custo(Custo)),nl,
47                                                  write(profundidade(Prof)),nl,
48                                                  escreve_seq_accos(no(E,Pai,Op,_,_)).
49
50 escreve_seq_accos([]).
51 escreve_seq_accos(no(E,Pai,Op,_,_)):- escreve_seq_accos(Pai),
52                                       write(e(Op,E)),nl.
53
54 esc(A):- write(A), nl.

```

Alínea c)

	Largura	Profundidade	Profundidade Iterativa
Nós visitados	2815	3951	17148
Número máximo de estados simultaneamente em memória	338	363	32

(todos os testes foram feitos com os exemplos do enunciado)

Alínea d)

```

1  h(E, Val) :- h(E, Val).
2  max(A, B, A) :- A>B, !.
3  max(_, B, B).
4
5  min(A,B,A) :- A<B, !.
6  min(_,B,B).
7
8  % modulo da subtração
9  modSub(A, B, Res) :- max(A, B, Max), min(A, B, Min), Res is Max - Min.
10
11 % heuristica 1 - distância da caixa ao estado final
12 h1(E,Val) :- estado_final((_,(Fx,Fy))),
13             E= (_,(Cx,Cy)),
14             modSub(Fx, Cx, DistX),
15             modSub(Fy, Cy, DistY),
16             Val is (DistX + DistY).
17
18 % Heuristica 2 - Distância no eixo do X à casa final
19 h2(E, Val) :- estado_final((_,(Fx,_))),
20             E = (_,(Cx,_)),
21             modSub(Fx,Cx,Val).
```


Alínea e)

➤ Greedy Search (Profundidade =16)

```

1  :- dynamic(fechado/1).
2  :- dynamic(maxNL/1).
3  :- dynamic(nos/1).
4
5  maxNL(0).
6  nos(0).
7
8  inc:- retract(nos(N)), N1 is N+1, asserta(nos(N1)).
9
10 actmax(N):- maxNL(N1), N1 >= N,!.
11 actmax(N):- retract(maxNL(_N1)), asserta(maxNL(N)).
12
13 pesquisa(Problema,Alg):-
14     consult(Problema),
15     estado_inicial(S0),
16     pesquisa(Alg,[no(S0,[],[],0,1,0)],Solucao),
17     escreve_seq_solucao(Solucao),
18     retract(nos(Ns)),retract(maxNL(NL)),retractall(fechado(_)),
19     asserta(nos(0)),asserta(maxNL(0)),
20     write(nos(visitados(Ns),lista(NL))).
21
22 pesquisa(g,E,S):- pesquisa_g(E,S).
23
24 pesquisa_g([no(E,Pai,Op,C,HC,P)|_],no(E,Pai,Op,C,HC,P)):- estado_final(E), inc.
25
26 pesquisa_g([E|R],Sol):- inc, asserta(fechado(E)), expande_g(E,Lseg),
27                             insere_ord(Lseg,R,Resto),length(Resto,N), actmax(N),
28                             pesquisa_g(Resto,Sol).
29
30 expande_g(no(E,Pai,Op,C,HC,P),L):- findall(no(En,no(E,Pai,Op,C,HC,P),Opn,Cnn,H,P1),
31                                     (op(E,Opn,En,Cn),
32                                     \+ fechado(no(En,_,_,_,_))),
33                                     P1 is P+1, Cnn is Cn+C, h(En,H)), L).
34
35 insere_ord([],L,L).
36 insere_ord([A|L],L1,L2):- insereE_ord(A,L1,L3), insere_ord(L,L3,L2).
37
38 insereE_ord(A,[],[A]).
39 insereE_ord(A,[A1|L],[A,A1|L]):- menor_no(A,A1),!.
40 insereE_ord(A,[A1|L],[A1|R]):- insereE_ord(A,L,R).
41
42 menor_no(no(_,_,_,_,N,_), no(_,_,_,_,N1,_)):- N < N1.
43
44 escreve_seq_solucao(no(E,Pai,Op,Custo,_HC,Prof)):- write(custo(Custo)),nl,
45                                                     write(profundidade(Prof)),nl,
46                                                     escreve_seq_accoes(no(E,Pai,Op,_,_,_)).
47
48
49 escreve_seq_accoes([]).
50 escreve_seq_accoes(no(E,Pai,Op,_,_,_)):- escreve_seq_accoes(Pai),
51                                         write(e(Op,E)),nl.
52
53 esc(A):- write(A), nl.

```

Alínea f)

	A*	Greedy
Nós visitados	1166	341
Número máximo de estados simultaneamente em memória	436	46

(todos os testes foram feitos com os exemplos do enunciado)