Escola de Ciências e Tecnologia

Departamento de Informática

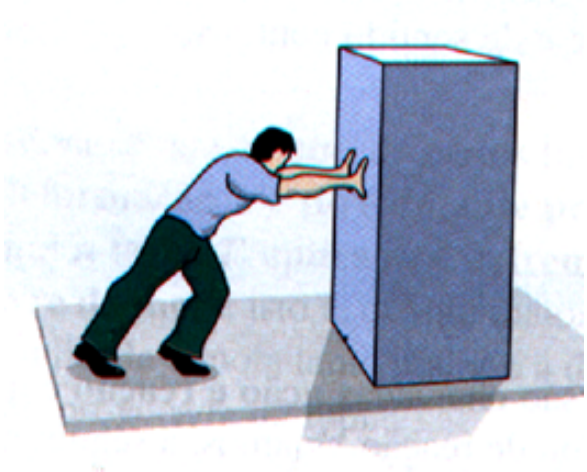
Licenciatura em Engenharia Informática

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)

```
listaX([(1,2),(3,1),(3,2),(4,4),(4,5),(4,6),(7,2)]).

% estado inicial
estado_inicial((2,7)).

% estado final
estado_final((5,1)).

% Verificar se está dentro dos limites do tabuleiro

limites(N,M) :- N>0, M>0, N<8, M<8, listaX(Lx), \+member((N,M),Lx).

% (X,Y) - posicao do A
% Lx    - lista de X's
% Ef    - estado final
% (Nx,Ny) - New X, NewY

op((X,Y), (0,M), (X,Ny), 1) :- member(M, [-1,1]),
                               Ny is Y+M,
                               limites(X,Ny).

op((X,Y), (N,0), (Nx,Y), 1) :- member(N, [-1,1]),
                               Nx is X+N,
                               limites(Nx,Y).
```

Alínea b)

- Pesquisa iterativa (Profundidade = 9)

```
:- dynamic(fechado/1).
:- dynamic(expandido/1).
:- dynamic(maxNL/1).
:- dynamic(nos/1).

maxNL(0).
nos(0).

inc:- retract(nos(N)), N1 is N+1, asserta(nos(N1)).

actmax(N):- maxNL(N1), N1 >= N,!.
actmax(N):- retract(maxNL(_N1)), asserta(maxNL(N)).

pesquisa(Problema,Alg):-
  consult(Problema),
  estado_inicial(S0),
  pesquisa(Alg,[no(S0,[],[],0,0)],Solucao),
  escreve_seq_solucao(Solucao),
  retract(nos(Ns)),retract(maxNL(NL)),retractall(fechado(_)), retractall(expandido(_)),
  asserta(nos(0)),asserta(maxNL(0)),
  write(nos(visitados(Ns),lista(NL))).

pesquisa_it(Ln,Sol,P):- retractall(fechado(_)) ,pesquisa_pLim(Ln,Sol,P).
pesquisa_it(Ln,Sol,P):- P1 is P+1, pesquisa_it(Ln,Sol,P1).

pesquisa(it,Ln,Sol):- pesquisa_it(Ln,Sol,1).

expandePl(no(_,_,_,_,P),[],Pl):- Pl =< P, ! .
expandePl(no(E,Pai,Op,C,P),L,_):-   findall(no(En,no(E,Pai,Op,C,P),Opn,Cnn,P1),
                                    (op(E,Opn,En,Cn), \+ fechado(no(En,_,_,_,_)) ,P1 is P+1, Cnn is Cn+C),
                                    L).
insere_fim([],L,L).
insere_fim(L,[],L).
insere_fim(R,[A|S],[A|L]):- insere_fim(R,S,L).

pesquisa_pLim([no(E,Pai,Op,C,P)|_],no(E,Pai,Op,C,P),_):- estado_final(E), inc. %incrementar depois de estado_final

pesquisa_pLim([E|R],Sol,Pl):- inc, assertz(fechado(E)) ,expandePl(E,Lseg,Pl), %esc(E), %incrementar antes do expandePl
                              insere_fim(R,Lseg,Resto),
                              length(Resto,N), actmax(N), % calcular o tamanho de resto e actualizar maximo
                              pesquisa_pLim(Resto,Sol,Pl).


escreve_seq_solucao(no(E,Pai,Op,Custo,Prof)):- write(custo(Custo)),nl,
                                               write(profundidade(Prof)),nl,
                                               escreve_seq_accoes(no(E,Pai,Op,_,_)).

escreve_seq_accoes([]).
escreve_seq_accoes(no(E,Pai,Op,_,_)):- escreve_seq_accoes(Pai),
                                       write(e(Op,E)),nl.

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)

```
h(E,Val) :- h1(E,Val1), h2(E,Val2), min(Val1,Val2,Val).

max(A, B, A):- A>B,!.
max(_, B, B).

min(A,B,A) :- A<B, !.
min(_,B,B).

% modulo da subtração
modSub(A, B, Res):- max(A, B, Max), min(A, B, Min), Res is Max - Min.

% Heuristica 1 - Distância à casa final
h1(E,Val) :- estado_final((Fx,Fy)),
             E=(Cx,Cy),
             modSub(Fx, Cx, DistX),
             modSub(Fy, Cy, DistY),
             Val is (DistX + DistY).

% Heuristica 2 - Distância no eixo do X à casa final
h2(E,Val) :- estado_final((Fx,_)),
             E = (Cx,_),
             modSub(Fx, Cx, Val).
```

Alínea e)

- Greedy Search (Profundidade = 9)

```
:- dynamic(fechado/1).
:- dynamic(maxNL/1).
:- dynamic(nos/1).

  maxNL(0).
  nos(0).

  inc:- retract(nos(N)), N1 is N+1, asserta(nos(N1)).

  actmax(N):- maxNL(N1), N1 >= N,!.
  actmax(N):- retract(maxNL(_N1)), asserta(maxNL(N)).

pesquisa(Problema,Alg):-
  consult(Problema),
  estado_inicial(S0),
  pesquisa(Alg,[no(S0,[],[],0,1,0)],Solucao),
  escreve_seq_solucao(Solucao),
  retract(nos(Ns)),retract(maxNL(NL)),retractall(fechado(_)),
  asserta(nos(0)),asserta(maxNL(0)),
  write(nos(visitados(Ns),lista(NL))).

pesquisa(g,E,S):- pesquisa_g(E,S).

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).

expande_g(no(E,Pai,Op,C,HC,P),L):- findall(no(En,no(E,Pai,Op,C,HC,P),Opn,Cnn,H,P1),
                          (op(E,Opn,En,Cn),
                          \+ fechado(no(En,_,_,_,_,_)),
                          P1 is P+1, Cnn is Cn+C, h(En,H)), L).

insere_ord([],L,L).
insere_ord([A|L],L1,L2):- insereE_ord(A,L1,L3), insere_ord(L,L3,L2).

insereE_ord(A,[],[A]).
insereE_ord(A,[A1|L],[A,A1|L]):- menor_no(A,A1),!.
insereE_ord(A,[A1|L], [A1|R]):- insereE_ord(A,L,R).

menor_no(no(_,_,_,_,N,_), no(_,_,_,_,N1,_)):- N < N1.

escreve_seq_solucao(no(E,Pai,Op,Custo,_HC,Prof)):- write(custo(Custo)),nl,
                                write(profundidade(Prof)),nl,
                                escreve_seq_accoes(no(E,Pai,Op,_,_,_)).


escreve_seq_accoes([]).
escreve_seq_accoes(no(E,Pai,Op,_,_,_)):- escreve_seq_accoes(Pai),
                                    write(e(Op,E)),nl.

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)

```
listaX([(1,2),(3,1),(3,2),(4,4),(4,5),(4,6),(7,2)]).

% estado_inicial
estado_inicial(((2,7), (2,6))).

% estado_final
estado_final((_, (5,1))).

% Verificar se está dentro dos limites do tabuleiro

limites(N,M) :- N>0, M>0, N<8, M<8, listaX(Lx), \+member((N,M),Lx).

op(((Xa, Ya),(Xc, Yc)), (N,0), ((Nxa, Ya),(Nxc, Yc)), 1) :- member(N, [-1,1]),
                                                            Nxa is Xa+N,
                                                            ((Nxa,Ya) = (Xc,Yc) -> Nxc is Xc+N ; Nxc is Xc),
                                                            limites(Nxa, Ya), limites(Nxc, Yc),
                                                            verificarX((Nxa, Ya)), verificarX((Nxc, Yc)).

op(((Xa, Ya),(Xc, Yc)), (0,M), ((Xa, Nya),(Xc, Nyc)), 1) :- member(M, [-1,1]),
                                                            Nya is Ya+M,
                                                            ((Xa,Nya) = (Xc,Yc) -> Nyc is Yc+M ; Nyc is Yc ),
                                                            limites(Xa, Nya), limites(Xc, Nyc),
                                                            verificarX((Xa, Nya)), verificarX((Xc, Nyc)).
```

Alínea b)

- Pesquisa em largura (Profundidade =16)

```
:- dynamic(fechado/1).
:- dynamic(expandido/1).
:- dynamic(maxNL/1).
:- dynamic(nos/1).

maxNL(0).
nos(0).

inc:- retract(nos(N)), N1 is N+1, asserta(nos(N1)).

actmax(N):- maxNL(N1), N1 >= N,!.
actmax(N):- retract(maxNL(_N1)), asserta(maxNL(N)).


pesquisa(Problema,Alg):-
    consult(Problema),
    estado_inicial(S0),
    pesquisa(Alg,[no(S0,[],[],0,0)],Solucao),
    escreve_seq_solucao(Solucao),
    retract(nos(Ns)),retract(maxNL(NL)),retractall(fechado(_)), retractall(expandido(_)),
    asserta(nos(0)),asserta(maxNL(0)),
    write(nos(visitados(Ns),lista(NL))).

pesquisa(largura,Ln,Sol):- pesquisa_largura(Ln,Sol).

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

pesquisa_largura([E|R],Sol):- inc, expande(E,Lseg), %esc(E),
                              insere_fim(Lseg,R,Resto),
                              length(Resto,N), actmax(N),
                              pesquisa_largura(Resto,Sol).



% Para evitar ciclos infinitos, usamos o assert e o fechado
expande(no(E,_,_,_,_),[]):- fechado(E), !.

expande(no(E,Pai,Op,C,P),L):- assertz(fechado(E)),
                              findall(no(En,no(E,Pai,Op,C,P),Opn,Cnn,P1),
                              (op(E,Opn,En,Cn), \+ fechado(no(En,_,_,_,_)) ,P1 is P+1, Cnn is Cn+C),
                              L).
insere_fim([],L,L).
insere_fim(L,[],L).
insere_fim(R,[A|S],[A|L]):- insere_fim(R,S,L).

escreve_seq_solucao(no(E,Pai,Op,Custo,Prof)):- write(custo(Custo)),nl,
                                               write(profundidade(Prof)),nl,
                                               escreve_seq_accoes(no(E,Pai,Op,_,_)).

escreve_seq_accoes([]).
escreve_seq_accoes(no(E,Pai,Op,_,_)):- escreve_seq_accoes(Pai),
                                       write(e(Op,E)),nl.

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)

```
h(E, Val) :- h(E, Val).
max(A, B, A):- A>B,!.
max(_, B, B).

min(A,B,A) :- A<B, !.
min(_,B,B).

% modulo da subtração
modSub(A, B, Res):- max(A, B, Max), min(A, B, Min), Res is Max - Min.

% heuristica 1 - distância da caixa ao estado final
h1(E,Val) :- estado_final((_,(Fx,Fy))),
             E= (_,(Cx,Cy)),
             modSub(Fx, Cx, DistX),
             modSub(Fy, Cy, DistY),
             Val is (DistX + DistY).

% Heuristica 2 - Distância no eixo do X à casa final
h2(E, Val) :- estado_final((_,(Fx,_))),
             E = (_,(Cx,_)),
             modSub(Fx,Cx,Val).
```

Alínea e)

- Greedy Search (Profundidade =16)

```
:- dynamic(fechado/1).
:- dynamic(maxNL/1).
:- dynamic(nos/1).

  maxNL(0).
  nos(0).

  inc:- retract(nos(N)), N1 is N+1, asserta(nos(N1)).

  actmax(N):- maxNL(N1), N1 >= N,!.
  actmax(N):- retract(maxNL(_N1)), asserta(maxNL(N)).

pesquisa(Problema,Alg):-
  consult(Problema),
  estado_inicial(S0),
  pesquisa(Alg,[no(S0,[],[],0,1,0)],Solucao),
  escreve_seq_solucao(Solucao),
  retract(nos(Ns)),retract(maxNL(NL)),retractall(fechado(_)),
  asserta(nos(0)),asserta(maxNL(0)),
  write(nos(visitados(Ns),lista(NL))).

pesquisa(g,E,S):- pesquisa_g(E,S).

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).

expande_g(no(E,Pai,Op,C,HC,P),L):- findall(no(En,no(E,Pai,Op,C,HC,P),Opn,Cnn,H,P1),
                                (op(E,Opn,En,Cn),
                                \+ fechado(no(En,_,_,_,_,_)),
                                P1 is P+1, Cnn is Cn+C, h(En,H)), L).

insere_ord([],L,L).
insere_ord([A|L],L1,L2):- insereE_ord(A,L1,L3), insere_ord(L,L3,L2).

insereE_ord(A,[],[A]).
insereE_ord(A,[A1|L],[A,A1|L]):- menor_no(A,A1),!.
insereE_ord(A,[A1|L], [A1|R]):- insereE_ord(A,L,R).

menor_no(no(_,_,_,_,N,_), no(_,_,_,_,N1,_)):- N < N1.

escreve_seq_solucao(no(E,Pai,Op,Custo,_HC,Prof)):- write(custo(Custo)),nl,
                                    write(profundidade(Prof)),nl,
                                    escreve_seq_accoes(no(E,Pai,Op,_,_,_)).


escreve_seq_accoes([]).
escreve_seq_accoes(no(E,Pai,Op,_,_,_)):- escreve_seq_accoes(Pai),
                                      write(e(Op,E)),nl.

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)