Inteligência Artificial

Trabalho 1

Resolução de problemas de pesquisa em espaço de estados

Discentes:

João Santos nº 29634

André Gouveia nº 26918

Évora 2015

Universidade de Évora
Curso de Engenharia Informática
Unidade curricular de Inteligência Artificial
Discentes: André Gouveia nº 26918
João Santos nº 29634

Índice:

Universidade de Évora
Curso de Engenharia Informática
Unidade curricular de Inteligência Artificial
Discentes: André Gouveia nº 26918
João Santos nº 29634

Introdução.

Neste trabalho foi-nos pedido que encarasse-mos o problema da travessia condicionada de passageiros de um margem para a margem oposta de um rio como um problema de pesquisa em espaços de estados. A condição imposta depende do peso que a barca suporta. É dado o peso de cada passageiro e indicado um peso máximo para a barca de 140Kg.

1 – Respostas.

1.(a)

Para a encararmos como espaço de estados vai ser necessário definir os estados e os operadores de transição.

Os estados são definidos como um tuplo com duas listas, cada uma referente a uma das margens (podia ser efectuado apenas com umas lista, no entanto preferimos fazê-lo desta forma de forma a melhor poder acompanhar as transições de estado).

O estado inicial é definido com a margem de partida cheia e a margem de chegada vazia.

```
estado_inicial([28, 65, 45, 57, 98, 120],[]).
```

O estado final é definido por ter a margem de partida vazia, ou seja, já todos os passageiros foram encaminhados.

```
estado_final([],_).
```

Para os operadores das transições definimos a remoção de todas as combinações possíveis , mas não repetidas, de passageiros na margem de partida cujo peso combinado não passa os 140Kg. Essas combinações passam da lista de margem de partida para a lista de margem de chegada.

Universidade de Évora
Curso de Engenharia Informática
Unidade curricular de Inteligência Artificial
Discentes: André Gouveia nº 26918
João Santos nº 29634

1.(b)

O código de pesquisa que resolve este problema devolvendo sempre a melhor solução é um código de pesquisa em largura. Para a resolução criada utilizamos o código fornecido pela docente, adicionando um contador para o numero de nós visitados.

Código para pesquisa não informada:

```
:-dynamic(conta/1).

conta(0).

incC(_):-
 conta(X),
 retractall(conta(_)),
 Y is X+1,
 NovoC =.. [conta,Y],
 asserta(NovoC).

pesquisa(Problema,Alg):-
 consult(Problema),
 estado_inicial(S0),
 pesquisa(Alg,[no(S0,[],[],0,0)],Solucao),
 escreve_seq_solucao(Solucao).

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), incC(_).

pesquisa_largura([E|R],Sol):- expande(E,Lseg), esc(E),

insere_fim(Lseg,R,Resto),

incC(_),

pesquisa_largura(Resto,Sol).

expande(no(E,Pai,Op,C,P),L):- findall(no(En,no(E,Pai,Op,C,P),Opn,Cnn,P1),

 (op(E,Opn,En,Cn),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,

 conta(X),

 write(nos(X)),

 retractall(conta(_)),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.
```

1.(c)

Uma vez que se trata de um problema com uma expansão explosiva, ou seja, a memória e análise necessária para a sua resolução são demasiado elevadas, indicaremos a resposta para apenas um problema com 6 passageiros.

Numero de nós visitados: 4804

Profundidade: 4

Para o algoritmo de pesquisa em largura o numero máximo de nós em memória é dado por

$$n = b^{(d+1)}$$

Sendo b o máximo factor de ramificação (todas as combinações possiveis e não repetidas de passageiros na margem inicial e cujo o peso combinado não exceda os 140Kg) e d a profundidade da solução de menor custo.

Solução:

e([], ([28,65,45,57,98,120],[]))

e(transfere, ([65,57,98,120],[[28,45]]))

e(transfere, ([98,120],[[57,65],[28,45]]))

e(transfere, ([120],[[98],[57,65],[28,45]]))

e(transfere, ([],[[120],[98],[57,65],[28,45]]))

1.(d)

A 1ª heurística que utilizamos é baseada no numero de viagens necessárias para efectuar a viagem. Ao removermos um determinado de grupo, é verificado o peso combinado que sobra na margem de partida e dividido esse valor pela tara da barca. Desta forma obtemos o numero de viagens que falta efectuar removendo esse grupo de passageiros. A opção recai sempre pelo caminho em que menos viagens sobre.

A 2ª heurística é semelhante mas em vez do numero de viagens para levar o peso total, utiliza o a relação entre o peso médio e a tara da barca.

Código para as heurísticas:

```
pesototal([],0):-!.

pesototal([X|Ls],V1):-
        pesototal(Ls,V),
        V1 is V+X.

numerodepessoas([],V,V):-!.

numerodepessoas([_|L],X,V):-
        X1 is X+1,
        numerodepessoas(L,X1,V).

h1((L,_),V):- pesototal(L,T), V is T/140.

h2((L,_),V):- numerodepessoas(L,0,N),pesototal(L,T), T\=0,!, V is T/(N*140).

h2((L,_),V):- V is 0.
```

Universidade de Évora
Curso de Engenharia Informática
Unidade curricular de Inteligência Artificial
Discentes: André Gouveia nº 26918
João Santos nº 29634

1.(e)

Tal como na alinea 1.b, utilizamos o algoritmo de pesquisa informada dado pela docente, incluindo apenas o contador para o numero de nós visitados.

Código para a pesquisa informada:

```
:-dynamic(conta/1).

conta(0).

incC(_):-
  conta(X),
  retractall(conta(_)),
  Y is X+1,
  NovoC =.. [conta,Y],
  asserta(NovoC).

pesquisai(Problema,Alg):-
  consult(Problema),
  estado_inicial(S0),
  pesquisa(Alg,[no(S0,[],[],0,1,0)],Solucao),
  escreve_seq_solucaoi(Solucao).

pesquisa(a,E,S):- pesquisa_a(E,S).

pesquisa_a([no(E,Pai,Op,C,HC,P)|_],no(E,Pai,Op,C,HC,P)):- estado_final(E), incC(_).

pesquisa_a([E|R],Sol):- expande_a(E,Lseg), incC(_), esc(E),
  insere_ord(Lseg,R,Resto),
  pesquisa_a(Resto,Sol).

expande_a(no(E,Pai,Op,C,HC,P),L):- findall(no(En,
  no(E,Pai,Op,C,HC,P),Opn,Cnn,HCnn,P1),
```

```
  (op(E,Opn,En,Cn),P1 is P+1, Cnn is Cn+C, h1(En,H),

  HCnn is Cnn+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_solucaoi(no(E,Pai,Op,Custo,_HC,Prof)):- write(custo(Custo)),nl,

  write(profundidade(Prof)),nl,

  conta(X), write(nos(X)),nl,retractall(conta(_)),

  escreve_seq_accoesi(no(E,Pai,Op,_,_,_)).

escreve_seq_accoesi([]).

escreve_seq_accoesi(no(E,Pai,Op,_,_,_)):- escreve_seq_accoesi(Pai),

  write(e(Op,E)),nl.

esc(A):- write(A), nl.
```

Universidade de Évora
Curso de Engenharia Informática
Unidade curricular de Inteligência Artificial
Discentes: André Gouveia nº 26918
João Santos nº 29634

1.(f)

Heuristica 1 :

Numero de nós visitados: 943

Profundidade: 4

Solução:

e([], ([28,65,45,57,98,120],[]))

e(transfere, ([65,45,57,120],[[28,98]]))

e(transfere, ([45,120],[[57,65],[28,98]]))

e(transfere, ([45],[[120],[57,65],[28,98]]))

e(transfere, ([],[[45],[120],[57,65],[28,98]]))

Heurística 2 :

Numero de nós visitados: 2373

Profundidade: 4

Solução:

e([], ([28,65,45,57,98,120],[]))

e(transfere, ([28,65,45,57,98],[[120]]))

e(transfere, ([28,65,45,57],[[98],[120]]))

e(transfere, ([28,45],[[57,65],[98],[120]]))

e(transfere, ([],[[28,45],[57,65],[98],[120]]))

Para a pesquisa informada, o numero máximo de nós em memória é inferior ao da pesquisa em largura uma vez que a pesquisa é direcionada e não guardamos os nós associados a expansão de caminhos que não percorremos.