Universidade de Évora

Disciplina de Inteligência Artificial

Problema Locomotiva

Autores:
Marcus Santos, 29764
Ricardo Fusco, 29263

Professor:
Irene Pimenta Rodrigues

11/03/2014

Índice

Lista de Figuras

1 Respostas às perguntas do enunciado

1.1 1ª pergunta - Definição do problema (Espaço de estados e operadores)

Para representar o espaço de estados decidimos fazê-lo utilizando listas. Todo o espaço do problema está representado por uma lista, cujo conteúdo será 4 listas, sendo cada uma destas listas representativa de cada área onde o comboio pode estar. Dentro de cada uma destas 4 sub-listas estarão 2 listas. A primeira destas duas listas estará vazia (se o comboio não estiver nesta área) ou irá conter a locomotiva e as carruagens do comboio (se este estiver nesta área). A segunda lista irá representar as carruagens que se encontram num determinado troço/área que não estejam engatadas ao comboio (CarruagensA, CarruagensB, CarruagensC e CarruagensD). Um estado irá ser representado da seguinte maneira (exemplo do estado inicial):

```
estado_inicial([ [Comboio, CarruagensA], [[], CarruagensB, Agulha], [[],CarruagensC], [[],CarruagensD] ]):-
        Comboio = [l, car,car],
        Agulha = baixo,
        CarruagensA = [],
        CarruagensB = [],
        CarruagensC = [],
        CarruagensD = [].
```

Figure 1: Estado inicial

As áreas A,B,C e D encontram-se na representação do espaço do problema seguinte:

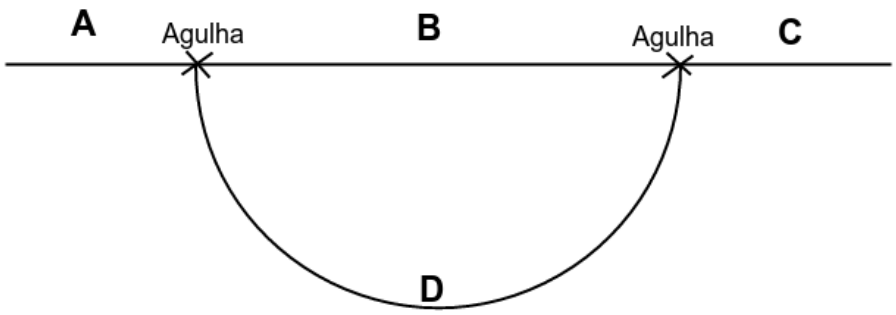

Figure 2: Esquema do espaço do problema

Dentro da lista da área B irá estar um terceiro elemento, a agulha (cima/baixo), que indicará se o comboio irá pela área B ou pela área D.

As operações que é possível fazer são andar para a frente, andar para trás, engatar à frente, engatar atrás, desengatar à frente, desengatar atrás e comutar agulha. Desengatar desengata carruagem a carruagem, e o engatar engata todas as carruagens da área.

```
op(EI, frente, EF, 1):-
        frente(EI, EF).

op(EI, tras, EF, 1):-
        tras(EI, EF).

op(EI, engatar_frente, EF, 1):-
        engatar(EI, frente, EF).

op(EI, engatar_tras, EF, 1):-
        engatar(EI, tras, EF).

op(EI, desengatar_frente, EF, 1):-
        desengatar(EI, EF, frente).

op(EI, desengatar_tras, EF, 1):-
        desengatar(EI, EF, tras).

op(EI, comutar_agulha, EF, 1):-
        comutar(EI, EF).
```

Figure 3: Operações do problema

1.2 2ª pergunta - Algoritmo de pesquisa não informada

O algoritmo de pesquisa não informada que pensamos que resolve este problema é o de pesquisa em largura, ou seja, dá a solução óptima na maior parte dos casos. O de pesquisa em profundidade iterativa também dá a solução a única diferença é que este, em relação ao da pesquisa em largura, visita muitos mais nós (cerca de 9 vezes mais) mas por outro lado tem muito menos nós em memória (cerca de 10 vezes menos) com base nos testes que fizemos. Quanto ao algoritmo da pesquisa em profundidade observámos que este, para a maior parte dos casos estudados, entra em loop infinito não conseguindo achar solução para o problema.

1.3 3ª pergunta - Número de estados visitados e máximo de estados em memória

Para o caso em que no estado inicial o comboio com 3 carruagens se encontra na área B e a agulha está para cima e em que no estado final o comboio com as 3 carruagens se encontra na área D e a agulha está para baixo (figura 4), usando a pesquisa em largura o número de nós visitados é 31 e o número máximo de nós em memória é 93. Usando a pesquisa em profundidade iterativa o número de nós visitados foi 281 e o número máximo de nós em memória foi 9.

```
estado_inicial([ [[], CarruagensA], [Comboio, CarruagensB, Agulha], [[],CarruagensC], [[],CarruagensD] ]):-
        Comboio = [l, car,car],
        Agulha = cima,
        CarruagensA = [],
        CarruagensB = [],
        CarruagensC = [],
        CarruagensD = [].

estado_final([ [[], CarruagensA], [[], CarruagensB, Agulha], [[],CarruagensC], [Comboio,CarruagensD]]):-
        Comboio= [l, car, car],
        Agulha= baixo,
        CarruagensA = [],
        CarruagensB = [],
        CarruagensC = [],
        CarruagensD = [].
```

Figure 4: Teste - estados inicial e final

1.4 4ª pergunta - Heurísticas admissíveis

1.4.1 Heurística 1

A heurística 1, dado um estado actual, calcula a distância que o comboio está da área final e o custo para lá chegar. Se a área tiver a agulha e essa estiver para baixo, então o custo de andar para essa área é 2 (comutar a agulha e andar para frente ou para trás), caso contrário é 1.

h1(E, V)

Onde (E) é estado actual e (V) o custo da mudança de estado

1.4.2 Heurística 2

A segunda heurística calcula o número de carruagens que estão fora do lugar e esse calculo é feito da seguinte forma:

(O número de carruagens nos estados seguintes é apenas um exemplo para facilitar a compreensão do algoritmo.)

1º*passo* :
Cria uma lista com o número de carruagens no estado actual:

Estado actual: **[0|2|1|0]**

Cria uma lista com o número de carruagens no estado final:

Estado final: **[1|1|0|1]**
2º*passo* :
Subtrai uma lista da outra e faz o módulo de cada posição da lista, tendo como resultado a seguinte lista que diz em cada área quantas carruagens estão fora do lugar:

Resultado: **[1|1|1|1]**

A soma de todos os valores da lista diz que 4 carruagens estão fora do lugar.

3º*passo* : Seja X o número de carruagens na locomotica no estado actual e Y o número de carruagens na locomotica no estado final, então o módulo de X - Y diz quantas carruagens estão fora do lugar na locomotiva (ou seja quantas carruagens faltam para perfazer o total de carruagens que pertencem ao comboio no estado final) .

4º*passo* : A soma do nº de carruagens fora do lugar na locomotiva com o nº de carruagens fora do lugar nas área diz o nº total de carruagens fora do lugar.

Por fim a heurística final é a soma destas duas heurísticas
h(E, V):-
h1(E,V1),
h2(E,V2),
V is V1+V2.

1.5 5ª pergunta - Algoritmo de pesquisa informada

O algoritmo de pesquisa informada que resolve este problema é o algoritmo A*. Foi testado este algoritmo para alguns casos e observou-se de facto uma diminuição vísivel tanto do número de nós visitados como do número máximo de nós em memória. Houve até alguns casos em que não é encontrado qualquer tipo de solução para o problema usando a pesquisa não informada e com a pesquisa A* usando as 2 heurísticas é encontrada uma solução (figura 5) em que foram visitados 11 nós e o número máximo de nós em memória simultânemente foi 23. No caso considerado na figura 4 foram visitados 9 nós e número máximo de nós em memória em simultâneo foi 19 verificando-se uma diminuição considerável tanto do número de nós visitados como do número máximo de nós em memória em simultâneo.

```
estado_inicial([ [[], CarruagensA], [Comboio, CarruagensB, Agulha], [[],CarruagensC], [[],CarruagensD] ]):-
        Comboio = [l, car,car],
        Agulha = cima,
        CarruagensA = [],
        CarruagensB = [],
        CarruagensC = [],
        CarruagensD = [].

estado_final([ [[], CarruagensA], [[], CarruagensB, Agulha], [[],CarruagensC], [Comboio,CarruagensD]]):-
        Comboio= [l],
        Agulha= baixo,
        CarruagensA = [],
        CarruagensB = [car,car],
        CarruagensC = [],
        CarruagensD = [].
```

Figure 5: Teste - estados inicial e final

1.6 6ª pergunta - Para cada heurística indique: o nº de estados visitados e o máximo de estados em memória

Considerando novamente o caso da figura 4, usando apenas a heurística 1 (h1) o número de nós visitados foi 9 e o número máximo de nós em memória foi 19. Usando apenas a heurística 2 (h2) o número de nós visitados foi 18 e o número máximo de nós em memória foi 47. Tendo em conta os valores do número de nós visitados e número máximo de nós em memória para a pesquisa em largura, 31 e 93, podemos concluir que ambas as heurísticas são aceitáveis reduzindo bastante o número de nós.

Considerando o caso da figura 5, usando apenas a heurística 1 (h1) o número de nós visitados foi 11 e o número máximo de nós em memória foi 23. Usando apenas a heurística 2 (h2) o número de nós visitados foi 44 e o número máximo de nós em memória foi 123.

Tendo em conta que para este caso não foi possível encontrar solução usando qualquer tipo de pesquisa não informada devido ao facto de serem demasiados nós a serem expandidos para encontrar a solução, podemos concluir que estas heuristicas são aceitáveis para este problema.

Há apenas um pequeno problema relativamente à maneira como o trabalho foi implementado, no caso demonstrado nas imagens do enunciado do trabalho, o comboio anda para a frente (da área A para a área B, com a agulha para cima) de seguida anda para a frente (para a área D) e desengata atrás e engata à frente as carruagens todas na mesma área. Tirando os casos semelhantes a este encontra soluções para muitos dos casos do problema da locomotiva.