Departamento de Engenharia Informática

Sokoban Problem

Introdução à Inteligência Artificial
2016-2017

Gabriel Cardoso, 2014195477, gbc@student.dei.uc.pt, PL5
João Lopes, 2014205453, jllopes@student.dei.uc.pt, PL5
Luís Almeida, 2014214377, lmsimoes@student.dei.uc.pt, PL1

02 de Abril de 2017

Conteúdo

1 Introdução **3**

2 Problema **3**

3 Problema de procura **3**
3.1 O que é um estado? . . . 3
3.2 Qual é o estado inicial? . . . 3
3.3 Como se define um estado final? . . . 4
3.4 Quais são os operadores de mudança de estado? . . . 4
3.5 Qual a natureza da solução pretendida? . . . 4
3.6 Qual o custo associado a cada movimento? . . . 4
3.7 Que tipo de Heurísticas são aplicáveis? . . . 4

4 Algoritmos **4**
4.1 Pesquisa em Profundidade Limitada . . . 4
4.2 Aprofundamento Progressivo . . . 4
4.3 Pesquisa Sôfrega . . . 5
4.4 A* . . . 5

5 Heurísticas **5**
5.1 Bola de neve . . . 5
5.2 Heurística Manhattan . . . 6
5.3 Heurística Euclides . . . 6
5.4 Evitar deadlocks . . . 6

6 Resultados Obtidos **7**
6.1 Algoritmos Base . . . 7
6.2 Tratamento de Deadlocks sobre Bola de Neve . . . 9
6.3 Heurística Manhattan 1- Distância entre caixa e objetivo . . . 10
6.3.1 Aplicada sobre Procura Sôfrega . . . 10
6.3.2 Aplicada sobre A* . . . 10
6.4 Heurística Manhattan 2- Distância entre player e caixa . . . 11
6.4.1 Aplicada sobre Procura Sôfrega . . . 11
6.4.2 Aplicada sobre A* . . . 11
6.5 Heurística Euclides 1- Distância entre caixa e objetivo . . . 12
6.5.1 Aplicada sobre Procura Sôfrega . . . 12
6.5.2 Aplicada sobre A* . . . 12
6.6 Heurística Euclides 2- Distância entre player e caixa . . . 13
6.6.1 Aplicada sobre Procura Sôfrega . . . 13
6.6.2 Aplicada sobre A* . . . 13

7 Discussão **13**

8 Bibliografia **14**

1 Introdução

Neste projeto de Introdução à Inteligência Artificial tinhamos como objetivo a implementação de vários algoritmos de procura de forma a encontrar soluções para múltiplos níveis de Sokoban, bem como a criação de heurísticas, de forma a tentar melhorar a eficiência desses algoritmos. De forma a testar estes algoritmos e heurísticas foram disponibilizados pelos docentes doze mapas, todos com complexidades diferentes.
Era também pedido que implementássemos uma determinada heurística para os algoritmos de Procura Sôfrega e A*, de forma a melhorar a eficiência dos mesmos.

2 Problema

O Sokoban é um puzzle cujo objetivo é mover uma série de caixas para posições determinadas (objetivos) obedecendo a um conjunto de regras:

- O jogador só se pode movimentar uma célula de cada vez na horizontal ou vertical (não são permitidos movimentos na diagonal);
- O jogador pode empurrar, mas não pode puxar as caixas;
- O jogador só pode mover uma caixa de cada vez;
- As caixas não podem ser sobrepostas, isto é, não podem haver duas caixas na mesma célula;
- O jogador não se pode sobrepor a caixas;
- As paredes são obstáculos intransponíveis tanto pelo jogador como por caixas;
- O jogo termina quando todas as caixas estiverem nas posições objetivo;

3 Problema de procura

3.1 O que é um estado?

Um estado é composto pelo personagem, paredes, objetos e número de objetivos. Distinguem-se entre si pelas posições dos objetos e personagem.

3.2 Qual é o estado inicial?

O estado inicial caracteriza-se pelas posições em que o jogo é inicializado, isto é, nas quais, o ficheiro do mapa, coloca o conteúdo do jogo e em que o personagem não efetuou nenhum movimento. O estado inicial não difere entre algoritmos de procura.

3.3 Como se define um estado final?

Um estado final trata-se de um estado em que todas as células objetivo contém objetos. Ao contrário do estado inicial, o estado final pode variar, conseiderando algoritmos completos, pois a posição do personagem pode também variar conforme a solução apresentada.

3.4 Quais são os operadores de mudança de estado?

O personagem verifica se é válida a movimentação para as diversas posições, pela seguinte ordem, Norte, Este, Sul e Oeste. Caso o movimento seja inválido devido à existência de uma parede ou de múltiplas caixas agrupadas, o personagem irá verificar o sentido seguinte. Caso encontre uma caixa, e seja possível movimentá-la (não exista outra caixa seguida a essa no mesmo sentido a ser verificado) o personagem irá empurrar a caixa, aumentando desta forma o custo.

3.5 Qual a natureza da solução pretendida?

Qualquer estado obtido que contenha os objetos nos objetivos é aceite.

3.6 Qual o custo associado a cada movimento?

Cada movimento tem sempre o custo associado da deslocação, e, caso seja deslocado um objeto, soma-se também o custo de deslocar esse mesmo objeto.

3.7 Que tipo de Heurísticas são aplicáveis?

Heurísticas que estimem encontrar a solução com igual ou menor custo que os métodos de pesquisa cega são admissíveis.

4 Algoritmos

4.1 Pesquisa em Profundidade Limitada

O algoritmo de pesquisa em profundidade limitada procura evitar o problema de lidar com caminhos infinitos fixando o nível máximo de procura. No entanto, nem sempre é possível determinar o valor máximo para o nível que deve ser usado, o que cria um problema para este algoritmo.
Assumindo que é possível definir um valor máximo, este algoritmo torna-se num algoritmo completo, tem complexidade temporal $O(r^l)$ e complexidade espacial $O(r * l)$. Neste algoritmo, a indicação do nível faz parte do estado.

4.2 Aprofundamento Progressivo

Este algoritmo combina aspetos de procura em profundidade com aspetos de procura por níveis. Isto tudo é atingido pela alteração do principio de procura

limitada, fazendo para isso variar o limite entre 0 e infinito, isto é, este algoritmo consiste na chamada repetida do algoritmo de procura limitada, aumentando a cada chamada o valor do limite máximo até encontrar uma solução.
Por estas razões, podemos dizer que este algoritmo é completo. É também, caso o custo das transições seja o mesmo, discriminador. Em caso contrário, a solução encontrada, pode não ser a solução mais próxima da raiz. A sua complexidade espacial é $O(r * l)$ e a sua complexidade temporal é $O(r^n)$, sendo n o limite máximo usado na solução.
Este algoritmo deve ser utilizado quando o método tem de ser cego, o espaço de procura é grande ou o nível da solução é desconhecido.

4.3 Pesquisa Sôfrega

Este algoritmo consiste em escolher, conforme o valor estimado em $h(n)$, o nó na fronteira da árvore de procura que aparenta ser o mais promissor. Para isto ser possível, o algoritmo mantém a fronteira da árvore ordenada a partir dos valores de $h(n)$, sendo sempre escolhido o nó mais abaixo.
Este algoritmo não é completo, também não é discriminador, tem complexidade espacial de $O(r^n)$ e complexidade temporal de $O(r^n)$.

4.4 A*

Este algoritmo utiliza como informação heurística o custo do nó atual n ao nó solução, determinado através da função $h(n)$, de forma a fazer uma escolha adequada, e ainda o valor determinado pela função $g(n)$, isto é, o custo do caminho que nos leva ao nó n. Desta forma, o algoritmo A* utiliza a função $f(n) = g(n) + h(n)$ de forma a descobrir, a cada instante, o melhor caminho passando pelo nó n, até ao nó solução.
Este algoritmo é, no caso da função h(n) ter boas propriedades, não só completo, como também encontra a melhor solução. Para isto acontecer $h(n)$ tem de ser admissível, isto é, não deve ser maior que o custo real. $h(n) \leq h_{real}(n)$.
Este algoritmo é completo e óptimo. A sua complexidade depende da qualidade da função heurística.

5 Heurísticas

5.1 Bola de neve

Esta heurística criada pelo grupo, verifica qual a caixa mas próxima do personagem e calcula a sua distância ao objetivo livre mais próximo. Assim sendo, tende a prioritizar os estados em que deslocamos objetos na direção dos objetivos. Mau quando existe uma parede a separar o objetivo do objeto.É admissivel pois no mínimo custa o número de movimentos para mover os objetos para a posição correta 1 passo. No pior caso, custa o nº de passos de empurrar os objetos para a posição correta.

5.2 Heurística Manhattan

A distância de Manhattan refere-se à distância entre dois pontos numa grelha baseada em caminhos horizontalmente e verticalmente restritos, resumindo o cálculo simplesmente à soma das componentes horizontais e verticais.
Foram implementadas duas funções que dão uso a este cálculo de forma a calcular o menor caminho entre o objeto A e B.
Estas duas funções encontram-se no script *SokobanProblem.cs* e têm como nome *manhaGetGoal()* que trata de calcular a menor distância entre uma caixa e o objetivo; e *manhaGetPlayer()* que trata de encontrar a menor distância agora entre o jogador e a caixa, isto sobre aplicação direta da fórmula.

5.3 Heurística Euclides

A distância Euclidiana é a distância entre dois pontos, dada pelo comprimento do segmento de reta que une esses pontos.
Tal como na distância de Manhattan foram implementadas duas funções que dão uso a este cálculo de distância, com o mesmo objetivo. Estas duas funções encontram-se também no script *SokobanProblem.cs* e têm como nome *euclGetGoal()* que trata de calcular a menor distância entre uma caixa e o objetivo; e *euclGetPlayer()* que trata de encontrar a menor distância agora entre o jogador e a caixa, isto sobre aplicação direta da fórmula.

5.4 Evitar deadlocks

Quando um objeto tem um obstáculo ou outro objeto em cima, à direita e no canto superior direito, este é impossivel de ser deslocado. O mesmo se aplica para os outros 3 cantos.
Sabendo isso, estados que contenham objetos presos fora de objetivos, não deverão ser expandidos pois é impossível de encontrar a solução a partir deles. Utíl em situações sensíveis a colocar o bloco preso, como o map07 fornecido. Apesar de não ser uma heurística em si, esta melhora a complexidade temporal dos algoritmos de pesquisa.

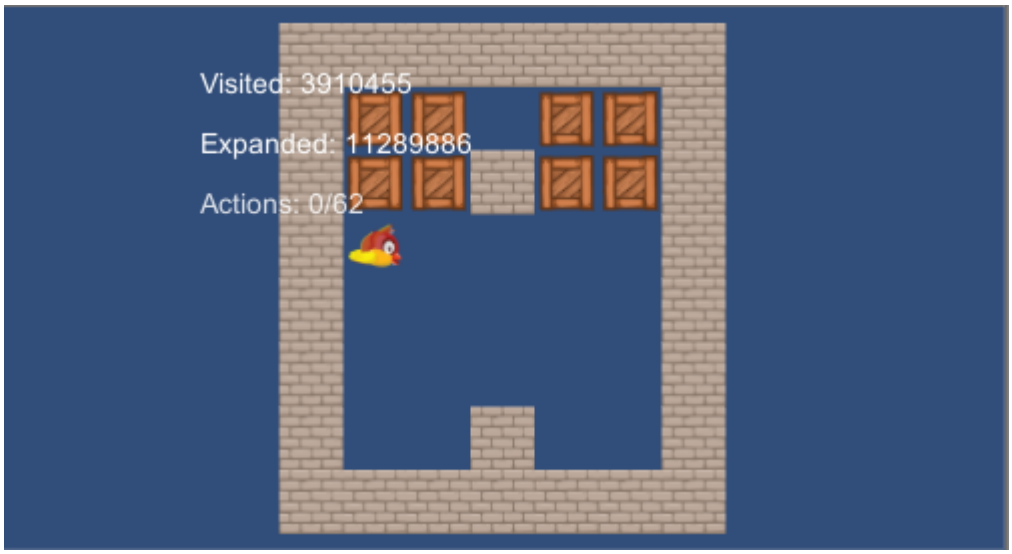

Figura 1: map07 com a Heurística Bola de Neve e a evitar deadlocks

6 Resultados Obtidos

6.1 Algoritmos Base

Algoritmo	Mapa	Visitados	Expandidos	Comprimento
Pesquisa em Largura	1	22	58	5
	2	386	1123	8
	3	134251	382032	29
	4	13812	33496	50
	5	708161	1964594	35
	6	71617	158845	89
	7	6988364	20000000	-
	8	1073204	2802189	93
	9	7539660	20000001	-
	10	6440355	20000000	-
	11	7759758	20000000	-
	12	7691837	20000000	-

Algoritmo	Mapa	Visitados	Expandidos	Comprimento
Aprofundamento Progressivo	1	204	558	25
	2	153633	468538	24
	3	8159140	20000002	-
	4	7542594	20000001	-
	5	6662640	20000001	-
	6	9496466	20000000	-
	7	7247524	20000001	-
	8	8348646	20000000	-
	9	6923858	20000000	-
	10	6315415	20000001	-
	11	8330667	20000000	-
	12	8701062	20000002	-

Algoritmo	Mapa	Visitados	Expandidos	Comprimento
Pesquisa em Profundidade Limitada	1	185	528	5
	2	6578	20522	8
	3	7697372	20000002	-
	4	7358275	20000001	-
	5	6866193	20000000	-
	6	8747210	20000000	-
	7	7758174	20000000	-
	8	7569600	20000002	-
	9	6844886	20000002	-
	10	6274295	20000000	-
	11	6915244	20000000	-
	12	6915866	20000003	-

Algoritmo	Mapa	Visitados	Expandidos	Comprimento
Procura Sôfrega	1	19	50	5
	2	30	96	8
	3	3506	9568	37
	4	9099	21967	52
	5	1767	4631	39
	6	9542	21360	89
	7	7068491	20000000	-
	8	28793	73864	97
	9	7652545	20000001	-
	10	91555	280561	48
	11	181196	470834	155
	12	4183703	10299021	246

Algoritmo	Mapa	Visitados	Expandidos	Comprimento
A*	1	21	55	5
	2	187	564	8
	3	null	null	null
	4	13126	31822	50
	5	null	null	null
	6	71123	157717	89
	7	6947626	20000000	-
	8	1063579	2775791	93
	9	7541360	20000000	-
	10	6440459	20000000	-
	11	7761388	20000001	-
	12	7691185	20000001	-

6.2 Tratamento de Deadlocks sobre Bola de Neve

Mapa	Visitados	Expandidos	Comprimento
1	19	50	5
2	211	621	8
3	7697372	20000002	-
4	5784	14100	50
5	6866193	20000000	-
6	4067	9142	91
7	3910455	11289886	62
8	433734	1137266	93
9	1527117	4021789	64
10	6416270	20000001	-
11	7762549	20000001	-
12	7879903	20000000	-

6.3 Heurística Manhattan 1- Distância entre caixa e objetivo

6.3.1 Aplicada sobre Procura Sôfrega

Mapa	Visitados	Expandidos	Comprimento
1	19	50	5
2	202	603	8
3	132351	369874	29
4	6375	15504	50
5	295507	816755	35
6	26914	60783	89
7	6981110	20000001	-
8	173327	445536	95
9	7590293	20000000	-
10	6545416	20000001	-
11	7773973	20000001	-
12	7885092	20000001	-

6.3.2 Aplicada sobre A*

Mapa	Visitados	Expandidos	Comprimento
1	21	55	5
2	347	1014	8
3	134251	382032	29
4	11746	28590	50
5	640160	1776737	35
6	59551	131735	89
7	6980832	20000001	-
8	1027130	2681059	93
9	7537497	20000000	-
10	6374054	20000002	-
11	7762727	20000000	-
12	7743134	20000001	-

6.4 Heurística Manhattan 2- Distância entre player e caixa

6.4.1 Aplicada sobre Procura Sôfrega

Mapa	Visitados	Expandidos	Comprimento
1	6	18	5
2	153	462	8
3	17481	47896	29
4	5482	13675	52
5	40815	116298	40
6	6713	14681	95
7	7153465	20000000	-
8	90885	243596	99
9	1115256	2994531	68
10	6328040	20000002	-
11	7389197	20000001	-
12	7326328	20000001	-

6.4.2 Aplicada sobre A*

Mapa	Visitados	Expandidos	Comprimento
1	9	27	5
2	312	926	8
3	133845	380234	29
4	11057	26922	50
5	591763	1657344	35
6	34380	75873	89
7	6994701	20000001	-
8	971557	2538194	93
9	7548801	20000001	-
10	6308530	20000002	-
11	7783034	20000001	-
12	7696620	20000002	-

6.5 Heurística Euclides 1- Distância entre caixa e objetivo

6.5.1 Aplicada sobre Procura Sôfrega

Mapa	Visitados	Expandidos	Comprimento
1	19	50	5
2	202	603	8
3	132351	369874	29
4	4686	11358	50
5	295507	816755	35
6	27499	62036	91
7	6981110	20000001	-
8	190571	489231	95
9	7590293	20000000	-
10	6545416	20000001	-
11	7774011	20000002	-
12	7884096	20000001	-

6.5.2 Aplicada sobre A*

Mapa	Visitados	Expandidos	Comprimento
1	21	55	5
2	347	1014	8
3	134251	382032	29
4	12818	31114	50
5	640160	1776737	35
6	60127	133059	89
7	6980832	20000001	-
8	1071150	2796836	93
9	7538088	20000001	-
10	6385852	20000003	-
11	7771544	20000002	-
12	7727376	20000003	-

6.6 Heurística Euclides 2- Distância entre player e caixa

6.6.1 Aplicada sobre Procura Sôfrega

Mapa	Visitados	Expandidos	Comprimento
1	6	18	5
2	234	692	8
3	97089	265824	29
4	6371	15821	74
5	339260	963799	37
6	6887	15151	99
7	7047982	20000003	-
8	172937	457348	97
9	2278001	5991952	100
10	6355152	20000001	-
11	7458869	20000001	-
12	7402569	2000000	-

6.6.2 Aplicada sobre A*

Mapa	Visitados	Expandidos	Comprimento
1	15	43	5
2	354	1040	8
3	134170	381747	29
4	12094	29460	50
5	654244	1823892	35
6	66679	148213	89
7	6980690	20000000	-
8	1044069	2726984	93
9	7537854	20000001	-
10	6312590	20000001	-
11	7761536	20000002	-
12	7632442	20000001	-

7 Discussão

Definimos como limite 20000000 nós visitados para evitar correr um algoritmo até encontrar uma solução, e mostrou-se suficiente pois encontrámos pelo menos uma solução admissível para todos os mapas fornecidos.
Com base neste limite, também definimos o limite para o algoritmo de Pesquisa em Profundidade limitada, para ser avaliado nas mesmas condições que os outros, isto é, percorrer uma árvore com um número de nós superior a 20000000, daí atribuírmos um limite de 25 na tabela acima apresentada, de maneira a cobrir os pontos positivos e negativos do algoritmo. Apesar de conseguir descobrir

uma solução para a maioria dos casos, este poderia descobrir uma solução para outros casos se se alterar o limite.

Tabela 1: Melhores resultados

Algoritmo/Heurística	**Mapa**	**Visitados**	**Comprimento**
Procura Sôfrega/Manhattan 2	1	6	5
Procura Sôfrega	2	30	8
Procura Sôfrega/Euclides 1	3	97089	29
Procura Sôfrega/Manhattan 2	4	5482	50
Procura Sôfrega/Manhattan 2	5	40815	35
Bola de Neve sem Deadlocks	6	4067	89
Bola de Neve sem Deadlocks	7	3910455	62
A*/Manhattan 2	8	433734	93
Procura Sôfrega/Manhattan 2	9	1527117	64
Procura Sôfrega	10	91555	48
Procura Sôfrega	11	181196	155
Procura Sôfrega	12	4183703	246

Note: Euclides/Manhattan 1: Distância entre caixa e objetivo
Euclides/Manhattan 2: Distância entre o player e a caixa

O algoritmo de Pesquisa Sôfrega, com e sem heurísticas aplicadas, revelou-se ser a mais eficaz a descobrir a solução mais curta com menor o menor nº de nós visitados, mostrando assim ter a melhor solução mais rapidamente relativamente às Heurísticas estudadas.
A Heurística Bola de Neve sem a pesquisa sobre Deadlocks também se mostrou eficaz ao encontrar a solução, pois não pesquisa sobre nós inúteis.
Algo relevante de ser mencionado é o facto de para mapas complexos como o 10, 11 ou 12, apenas o algoritmo de Pesquisa Sôfrega ter sido capaz de encontrar uma solução, mesmo para o mapa 12 com 246 passos até à solução.
Outro mapa interessante de se analisar é o mapa 7, onde as possibildades de prender um bloco são elevadas e apenas o algoritmo Bola de Neve sem Deadlocks encontrou uma solução com menos de 20 milhões de nós expandidos.

8 Bibliografia

- Ernesto Costa, Anabela Simões, Inteligência Artificial: Fundamentos e Aplicações (2008)