U.PORTO

FEUP FACULDADE DE ENGENHARIA
UNIVERSIDADE DO PORTO

Folding Blocks

Inteligência Artificial 2019/2020

Turma 2 / 1B

Ana Teresa Dias Silva - up201606703
Margarida Alves Pinho - up201704599
Maria Gonçalves Caldeira - up201704507

Especificação do jogo

O *Folding Blocks* é um jogo do tipo solitário cujo objetivo é “desdobrar” os blocos coloridos de forma a que, em cada nível, todos os espaços fiquem preenchidos.

O tabuleiro de jogo pode ter um formato regular e, consequentemente, dimensões NxM ou NxN, ou um formato irregular.

Quando no tabuleiro existem peças da mesma cor, adjuntas ou não, considera-se que as mesmas constituem um bloco.

Qualquer jogada terá um efeito de “espelho”, sobre o bloco, na direção da jogada efetuada, por exemplo, se na jogada efetuada a direção for a direita, o bloco sobre o qual está a ser efetuada a jogada sofrerá uma reflexão para a direita, desde que não ultrapasse as dimensões do tabuleiro de jogo.

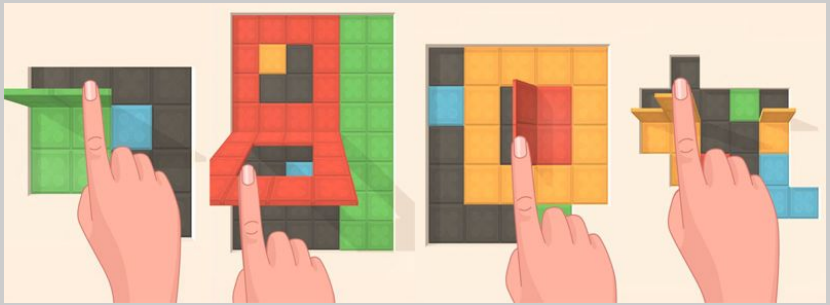

Fig.1: Exemplo de possíveis jogadas, tabuleiros e blocos

Formulação problema

- **Representação de estados:** Especifica a posição de cada uma das peças coloridas e dos espaços vazios (diversas representações)
- **Estado inicial:** Tabuleiro de tamanho e formato aleatórios, com alguns blocos dispostos aleatoriamente
- **Teste objetivo:** Todos os espaços do tabuleiro ficam preenchidos com blocos coloridos
- **Operadores**

Nome	Pré-condições	Efeitos	Custo
Direita	A reflexão do bloco à direita não ultrapassa as margens do tabuleiro e não sobrepõe outros blocos	Reflexão do bloco, segundo eixo vertical, à direita	1
Esquerda	A reflexão do bloco à esquerda não ultrapassa as margens do tabuleiro e não sobrepõe outros blocos	Reflexão do bloco, segundo eixo vertical, à esquerda	1
Cima	A reflexão do bloco para cima não ultrapassa as margens do tabuleiro e não sobrepõe outros blocos	Reflexão do bloco, segundo eixo horizontal, para cima	1
Baixo	A reflexão do bloco para baixo não ultrapassa as margens do tabuleiro e não sobrepõe outros blocos	Reflexão do bloco, segundo eixo horizontal, para baixo	1

- **Custo da solução:** Número de jogadas efetuadas

Implementação

- **Linguagem de programação:** C++
- **Ambiente de desenvolvimento:** Visual Studio Code
- **Estruturas de dados:**
    - Classes:
        i. Board
        ii. Game
- **Estrutura de ficheiros:**
    i. Board.c/Board.h: informação sobre o tabuleiro de jogo e respetivas peças
    ii. Game.c/Game.h: informação sobre as regras do jogo e possíveis jogadas
    iii. Algorithms.c/Algorithms.h: algoritmos de pesquisa utilizados na resolução do problema
    iv. Node.h: armazena informação sobre um nó de um grafo(árvore)
    v. Graph.h/Graph.cpp: informação sobre uma árvore de pesquisa
    vi. Algorithms2.cpp: opções alternativas dos algoritmos BFS e DFS

Abordagem

Inicialmente desenvolvemos o jogo na versão player que identifica todos os blocos, implementa corretamente todas as regras tendo em conta os operadores previamente definidos e verifica e efetua jogadas.

No modo AI, as regras e jogadas são efetuadas também de forma correta apesar das dificuldades sentidas na implementação dos algoritmos.

Algoritmos implementados

Quando executado no modo AI, é possível resolver o problema recorrendo aos seguintes algoritmos:

- **Métodos de pesquisa não informada:**
  - Pesquisa em largura (BFS)
  - Pesquisa em profundidade (DFS)
- **Métodos de pesquisa heurística:**
  - Pesquisa gulosa (Greedy)
  - A*

Resultados experimentais

Comparação entre métodos de pesquisa não informada (número de jogadas necessárias para atingir solução)

	Pesquisa em largura(bfs_v1)		**Pesquisa em profundidade(dfs_v1)**	
	Número jogadas	***Nível árvore***	***Número jogadas***	***Nível árvore***
Nível 1 (1 bloco, 16 células)	13	4	4	4
Nível 2 (1 bloco, 8 células)	5	2	2	2
Nível 3 (2 blocos, 16 células)	37	6	6	6
Nível 4 (3 blocos, 22 células)	30	6	6	6
Nível 5 (4 blocos, 32 células)	-	-	9	9
Nível 6 (5 blocos, 56 células)	460	13	-	-
Nível 7 (6 blocos, 31 células)	179	9	-	-
Nível 8 (6 blocos, 60 células)	172	16	-	-

Resultados experimentais

U.PORTO
FEUP FACULDADE DE ENGENHARIA UNIVERSIDADE DO PORTO

Comparação entre métodos de pesquisa heurística (número de jogadas necessárias para atingir solução)

	Algoritmo "guloso"		Algoritmo A*					
			Função 1		Função 2		Função 3	
	Jogadas	*Nível árvore*	*Jogadas*	*Nível árvore*	*Jogadas*	*Nível árvore*	*Jogadas*	*Nível árvore*
Nível 1	4	4	4	4	4	4	4	4
Nível 2	2	2	2	2	2	2	2	2
Nível 3	10	6	7	6	7	6	10	6
Nível 4	10	6	12	6	12	6	9	6
Nível 5	-	-	-	-	-	-	17	9
Nível 6	-	-	-	-	-	-	-	-
Nível 7	-	-	-	-	-	-	-	-
Nível 8	-	-	-	-	-	-	-	-

Conclusões

Na nossa implementação fizemos duas abordagens para o bfs e o dfs. Numa fase inicial as funções que implementamos para cada um funcionavam, mas o tempo de pesquisa era muito elevado. Para tentarmos melhorar esta situação tentamos alterar as funções dos dois algoritmos, contudo, estas novas versões são mais rápidas mas não passam em todos os níveis de jogo presentes na nossa implementação.

Em suma, este trabalho permitiu-nos aplicar os conhecimentos adquiridos nas aulas teóricas e nas teórico-práticas no que toca à implementação dos métodos de pesquisa, e apesar de não ter sido implementado totalmente com sucesso, ajudou-nos a consolidar e a aumentar a nossa compreensão sobre toda matéria lecionada na cadeira.

Referências

U.PORTO
FEUP FACULDADE DE ENGENHARIA UNIVERSIDADE DO PORTO

Durante a elaboração do trabalho, baseamo-nos na aplicação disponível no PlayStore (https://play.google.com/store/apps/details?id=com.popcore.foldingblocks) para a implementação dos diversos níveis.

Referências utilizadas para elaboração de algoritmos de pesquisa:

- Slides da Unidade Curricular disponibilizados no moodle
- https://www.softwaretestinghelp.com/cpp-bfs-program-to-traverse-graph/
- https://www.geeksforgeeks.org/depth-first-search-or-dfs-for-a-graph/
- https://www.geeksforgeeks.org/c-program-for-greedy-algorithm-to-find-minimum-number-of-coins/
- https://www.hackerearth.com/practice/algorithms/greedy/basics-of-greedy-algorithms/tutorial/
- https://www.redblobgames.com/pathfinding/a-star/implementation.html
- https://www.geeksforgeeks.org/a-search-algorithm/