U.PORTO
FEUP FACULDADE DE ENGENHARIA
UNIVERSIDADE DO PORTO

# Folding Blocks

## Métodos de Pesquisa Heurística para Resolução de Jogo do Tipo Solitário

João Araújo, 201705577
Jorge Pacheco, 201705754
Leonor Sousa, 201705377

**Inteligência Artificial**
Mestrado Integrado em Engenharia Informática e Computação

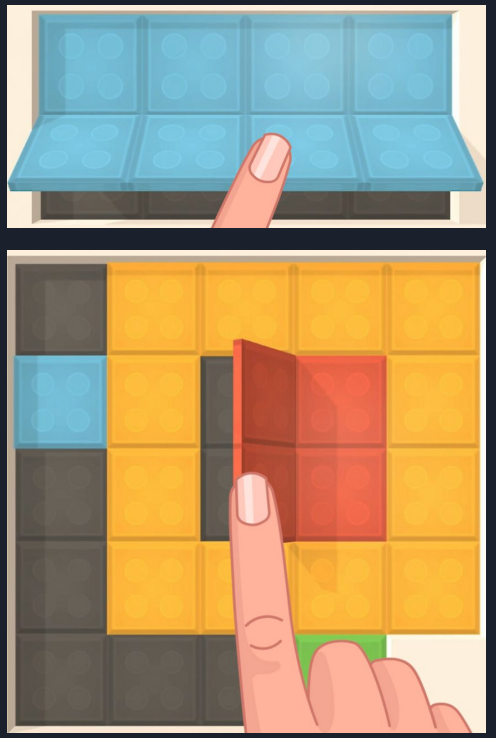
# Especificação do Trabalho a Efetuar

O jogo Folding Blocks é um jogo do tipo solitário.

Cada nível do jogo é representado por um tabuleiro que, inicialmente tem algumas células coloridas (de uma ou mais cores) e outras cinzentas. Para a especificação do trabalho, convém considerar grupos de células, em que cada grupo corresponde ao conjunto de todas as células da mesma cor.

O jogo permite movimentos que correspondem a duplicar um grupo de células coloridas, aplicando-lhes uma simetria no tabuleiro, sendo que todas as células duplicadas têm que ficar em locais anteriormente ocupados por células cinzentas.

O objetivo de cada nível consiste em preencher o tabuleiro com células coloridas.



# Pesquisa Efetuada

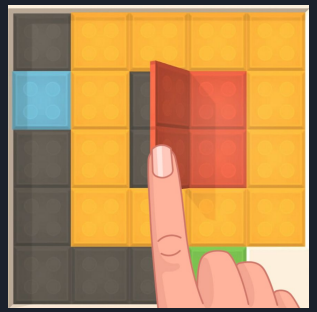
- Encontrámos uma solução/implementação possível para o problema, em python. LINK
- Encontrámos algumas tips que podem ajudar a encontrar o melhor método de resolução/a melhor heurística a ser usada. LINK LINK
    - Dividir o tabuleiro em secções e tentar resolver uma secção de cada vez
    - Começar por “duplicar” grupos de células maiores, visto que são, por norma, os mais difíceis de encaixar no tabuleiro.
    - Detetar a simetria nos níveis, adquirindo uma estratégia diferente consoante haja simetria (ou falta dela)

# Formulação do Problema como um Problema de Pesquisa

**Representação do Estado**: matriz retangular de células M, em que:

- 0 -> célula cinzenta
- A-Z ->célula colorida, em que cada letra representa uma cor diferente
- _ ->célula “vazia” (buraco no tabuleiro)

**Estado Inicial**: matriz com células de vários tipos, problema com resolução
**Teste Objetivo**: matriz sem células cinzentas/sem símbolos 0.
**Exemplo**:



```
M=[[0, A, A, A, A],
   [B, A, 0, C, A],
   [0, A, 0, C, A],
   [0, A, A, A, A],
   [0, 0, 0, D, _]]
```

**Estado Inicial**

```
M=[[B, A, A, A, A],
   [B, A, C, C, A],
   [B, A, C, C, A],
   [B, A, A, A, A],
   [D, D, D, D, _]]
```

**Teste Objetivo**

| Operador | Pré-condição | Efeito | Custo |
|---|---|---|---|
| $B_k$ | Para cada $M_{ij}$=k, a posição $M_{(2*imax-i+1)j}$=0 | Para cada $M_{ij}$=k, a posição $M_{(2*imax-i+1)j}$=k | 1 |
| $C_k$ | Para cada $M_{ij}$=k, a posição $M_{(2*imin-i-1)j}$=0 | Para cada $M_{ij}$=k, a posição $M_{(2*imin-i-1)j}$=k | 1 |
| $E_k$ | Para cada $M_{ij}$=k, a posição $M_{i(2*jmin-j-1)}$=0 | Para cada $M_{ij}$=k, a posição $M_{i(2*jmin-j-1)}$=k | 1 |
| $D_k$ | Para cada $M_{ij}$=k, a posição $M_{i(2*jmax-j+1)}$=0 | Para cada $M_{ij}$=k, a posição $M_{i(2*jmax-j+1)}$=k | 1 |

## Heurística

Comparar os tamanhos e as possibilidades de duplicação dos grupos com as células ainda livres, de forma a utilizar os grupos maiores que podem ser duplicados permitindo o preenchimento do tabuleiro:

$$T=PA*2^{NA}+PB*2^{NB}+...$$

em que:

T= número total de células não “vazias”(buracos)

PA = número de células do grupo A

NA = número de vezes que as células do grupo A são duplicadas

# Trabalho de Implementação já Realizado

- Classe que define um nível do jogo (Level)
  - Função que calcula se uma jogada é possível (analise_move)
  - Função que realiza uma jogada (make_move)
  - Função que verifica se o nível já está resolvido (solved)
  - Display do “tabuleiro” de um nível (display()
- Classe que define uma jogada (Move)
- Classe que define um nó  (Node)
- Classe que define uma árvore de pesquisa  (SearchTree)
  - Função para pesquisa com Custo Uniforme quase concluída
- Classe que define o jogo (FoldingBlocks)
  - Função que faz o processamento do jogo em modo bot (play_bot)
  - Função que permite a um jogador humano jogar (play_human)

**A Implementar:**

- Funções para pesquisa cega: Em Profundidade, Em Profundidade Iterativa
- Funções para pesquisa informada: Gulosa (tentar encher o máximo de células vazias em cada jogada), usando a heurística feita.