Jelly Field Puzzle

Grupo 42

Alexandre Morais

Carlos Costa

Nuno Ramos

Specification of Work

Implementation of a One-Plater Solitaire Game for a Human player providing Hints related to game state, algorithms to solve it using search methods, focusing on the comparison between uninformed search methods and heuristic search methods.

The game in question is the Jelly Field Puzzle game. A board with random disposition of blocks of color is presented alongside a hand of blocks to be played to reach a determined number of colors completed.

Related Works

1. AI Player for Matching Games
    - Same style of popping clusters as Jelly Field
    - https://github.com/khangluong2004/COMP10001_AutoBot_Proj1

2. Source code of N-puzzle solver using DFS , BFS, A *
    - Same board style as Jelly field
    - https://github.com/lesquerra/AI-n-puzzle

3. GeeksForGeeks webpage
    - https://www.geeksforgeeks.org/heuristic-search-techniques-in-ai/
    - https://www.geeksforgeeks.org/difference-between-bfs-and-dfs/

Formulation of the Problem

State Representation

- Matrix Board with predetermined pieces placed T[x,y]
- Hand with pieces to place
- Queue with next playable pieces

Initial State

- Board with predefined pieces
- Hand with predefined piece
- Goal with predefined values
  - Blue:B Green:G Red:R Yellow:Y

Objective State

- Blue:0 Green:0 Red:0 Yellow:0

Formulation of the Problem

Operators

- Place Piece
  - Preconditions
    - Valid placement on board
  - Effects
    - Piece is placed and board is updated
  - Cost 1
- Pop Cluster
  - Preconditions
    - Cluster of same color exists
  - Effects
    - Cluster is removed, goal and board is updated
  - Cost 0
- Refill Hand
  - Preconditions
    - Missing piece on hand
  - Effects
    - New playable piece is added to hand
  - Cost 0

Implemented work

We decided on using Python for development. And implemented the following:

- Although with only text-based UI the full game logic is implemented.
- The BFS search method was implemented giving the first solution it finds which will always be the solution with the smaller number of moves needed.
- DFS search method. Printing the first solution it finds with minimal moves.