Jelly Field Puzzle

Grupo 42

Alexandre Morais

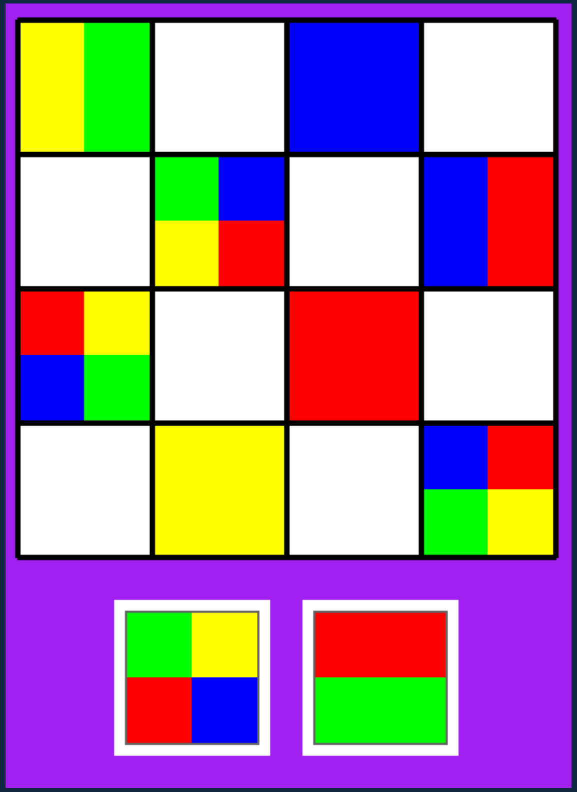
Carlos Costa

Nuno Ramos

Specification of Work

Implementation of a One-Player 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

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:

- Full UI implementation
- 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, A* and greedy search algorithm.
- Hint mechanic where it shows the best current move using a greedy method.
- 3 different levels with increasing difficulty.

Heuristics

1. Heuristic 1
   - Heuristic that calculates the total sum of our goal, resulting in the number of total blocks remaining.

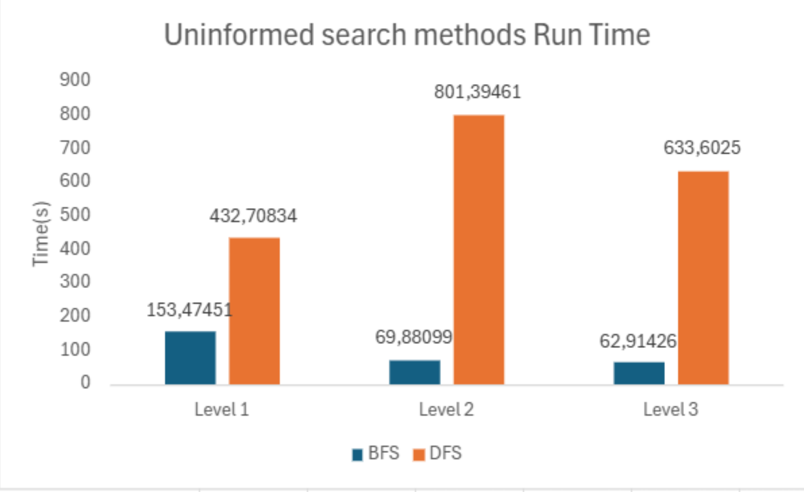
2. Heuristic 2
   - Heuristic that divides the remaining total blocks by 4 and its then rounded up using integer division. It assumes that each piece can clear the max number of blocks and gives the minimum number of moves to reach the goal.

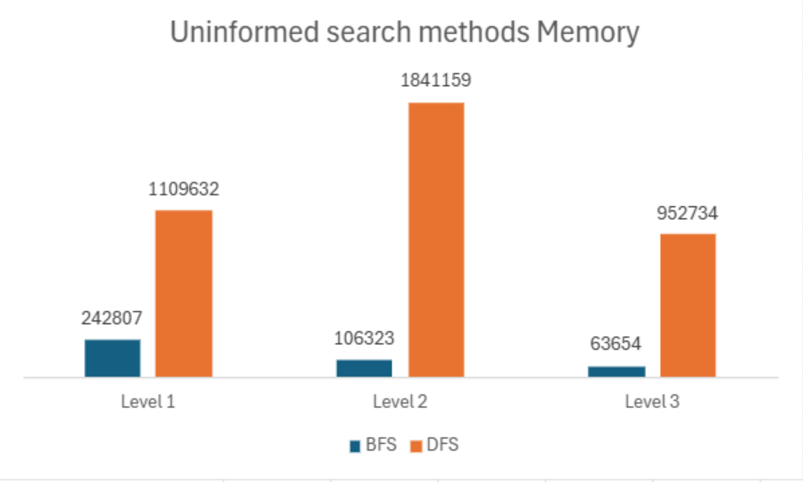
Experimental Results

- Both BFS and DFS guarantees optimal solutions(shortest path) but at the expense of highly computational power as these can take up long amounts of time and explore elevated number of states.

- BFS ends up having better results as it only requires to explore states until it finds the first solution

- DFS is tied to its max depth of exploration which results in higher values.





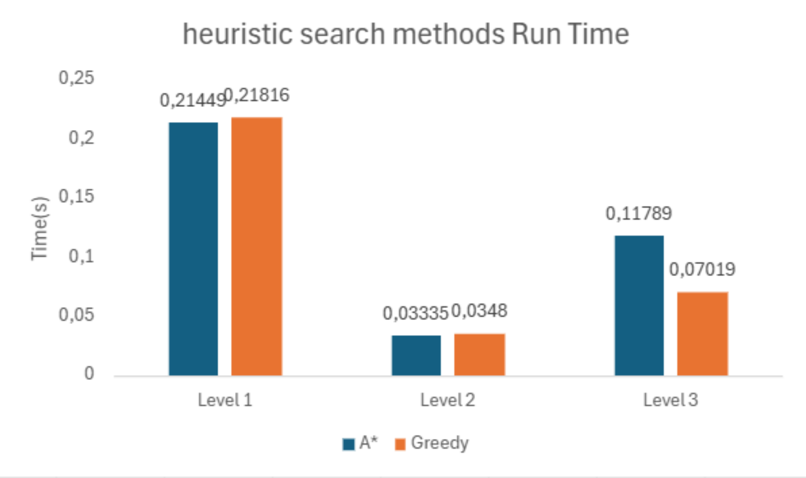
Experimental Results

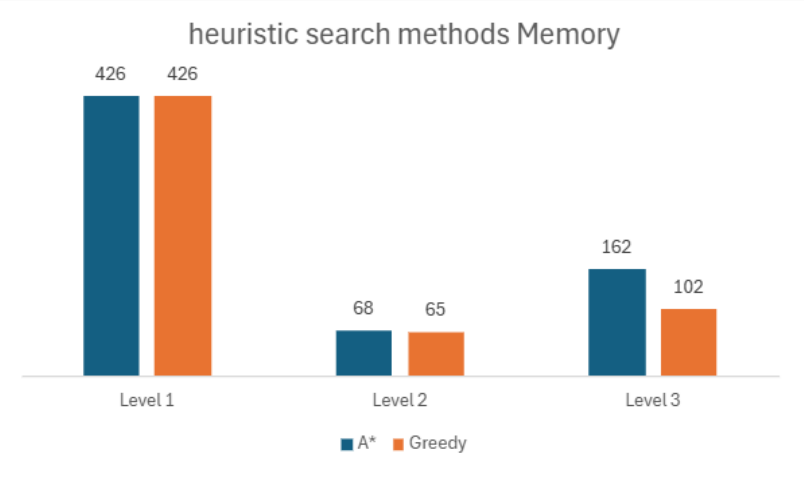
Both A* and Greedy showed similar results when compared to BFS and DFS:

- Non-optimal solutions
- Lower run times and number of states explored

But when comparing A* to Greedy:

- Having Greedy use heuristic 1 results in the algorithm always encountering a solution, while A* using heuristic 2 highly depends on the disposition of colors on our board which can lead to infinite loops.
- A* ends up finding close to optimal solutions while showing higher number of states explored.





Conclusions

This project highlights the trade-offs and distinctions between search algorithms in solving the Jelly Field Puzzle. While uninformed search methods guarantee optimal solutions, their exponential time and memory demand make them unsuitable for larger and more complex boards. In contrast, informed search strategies leverage heuristics to significantly reduce computation time, though they may sacrifice solution optimality in favor of efficiency.

References and Materials used

1. Course Resources
    - Artificial Intelligence Modern Approach 4thEd 1stPart

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

3. Python programming language
    - Pygame library for UI implementation