Heuristic Search Methods

AMADO

Alberto Serra - up202103627
Eduardo Sousa - up202103342
João Alves - up202108670

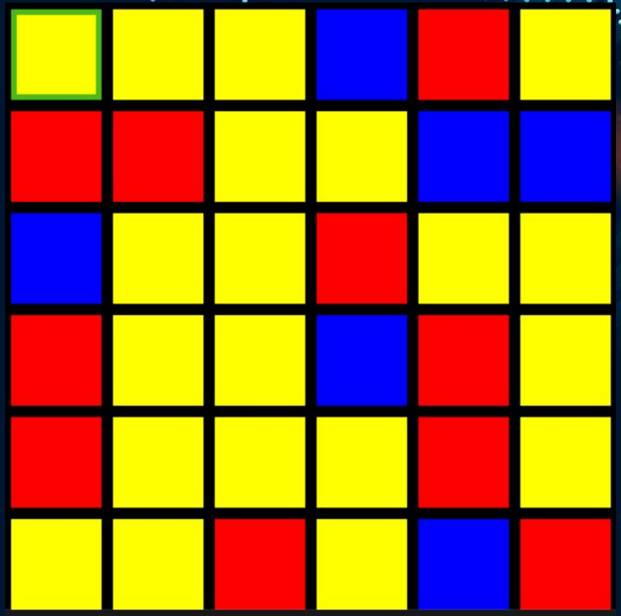
GAME DEFINITION

- Played on one of the four tile patterns, each with tiles of three colors (red, blue, and yellow), **Amado** is a puzzle for one player only.

- The object of the game is to change the color of the **game board** such that it is **equivalent** to the **goal board**.

- In this game, **colored squares are arranged** in a pattern. Moving from one color to another will cause the moved tile to convert to the third color (for example, moving from blue to red will cause the red tile to turn yellow).

- When the player makes the **game board match the goal board**, or when the timer goes off, the **game is over**.



PROBLEM FORMULATION

STATE REPRESENTATION

The symbols **B, R, Y, and ","** which equate to a two-dimensional array, stand in for the **blue, red, yellow, and empty** squares on the board. The most recent play made by the player or machine determines the current condition of the game.

INITIAL STATE

The player's cursor **starts at the upper left square** at the start of the game, and the player's board and goal board are generated at random.

OPERATORS

Movement operators (left, right, up, and down) that are defined with **uniform costs** are essential for assessing how well each action advances the objective state.

OBJECTIVE TEST

Make sure the goal board and the player's board are **equal**.

SEARCH ALGORITHMS

Uninformed Search

Breadth-first search - Guarantees the shortest path by exploring all neighbors before moving on.

Depth-first search - Explores as far as possible along each branch before backtracking.

Iterative deepening - Combines the depth-first search's space-efficiency and breadth-first search's completeness.

Uniform cost - Expands the least cost node, ensuring the cheapest path in a weighted graph.

Monte Carlo - Uses randomness to find solutions, beneficial for complex and unpredictable problems.

Informed Search

Greedy search - Prioritizes moves that seem closest to the goal, focusing on immediate benefits.

A* Algorithm - Balances between the path cost and estimated cost, providing an efficient and optimal path.

Weighted A* - Modifies A* to speed up search at the expense of optimality, by prioritizing estimated cost.

HEURISTICS

01

Color Mismatch

Tally of tiles that don't match the goal state's color, indicating the quantity out of place.

02

Color Distance

Calculates the least number of transitions to reach the goal colors, with intermediary steps if needed, weighting direct changes less than indirect ones.

03

Min Color Transition

Considers the direct and indirect color changes needed for the goal state, emphasizing the efficiency of one-step transitions.

04

Combined

A balanced measure of misplaced tiles and transition costs, combining mismatch count and transition efficiency.

APPROACH

01 Its possible select the level of precision of the algorithm used to solve the game, and the result will give you a **playable pattern**. There are **four patterns available** when choosing low precision, and three patterns when choosing high precision. It is possible to win every game.

02 **Monte Carlo algorithm** is employed to provide **hints**, while **Weighted A*** (which produces results with **higher accuracy** but is **much slower**) and **Quick Solver** (which produces results with **low precision** but is **extremely quick**) are used to complete the game from the starting state of the game.

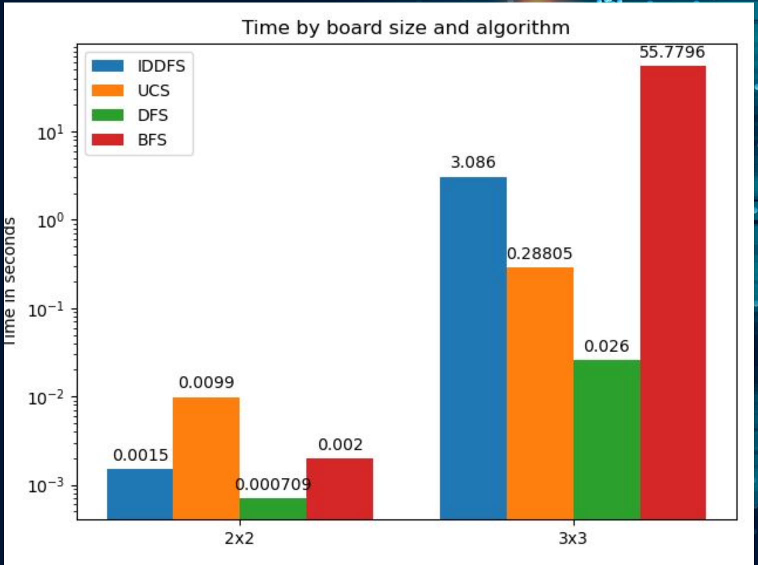
03 Our best algorithm, **Quick Solver**, applies weighted A* to smaller patterns before combining all the solutions to solve a larger pattern. Although we don't always get the greatest solving with that, this strategy nearly always yields decent outcomes.

04 Since we **nearly always obtained the same results using every heuristic**, the informed algorithms typically display the best result regardless of the heuristic that was applied.

EXPERIMENTAL RESULTS
Uninformed Search

Breadth-first search - The BFS method always finds a solution, despite the fact that it is regarded to be **extremely space-consuming**. This is because it visits many alternatives and requires exponential time complexity as the board gets larger, making it an **impractical algorithm for larger boards**.

Depth-first search - As expected, given that each game has a large number of potential solutions, the DFS algorithm is still our best uninformed approach in terms of time and space. However, in certain games, the **algorithm failed to discover a solution** because it became trapped in an infinite search that did not result in a winning scenario.

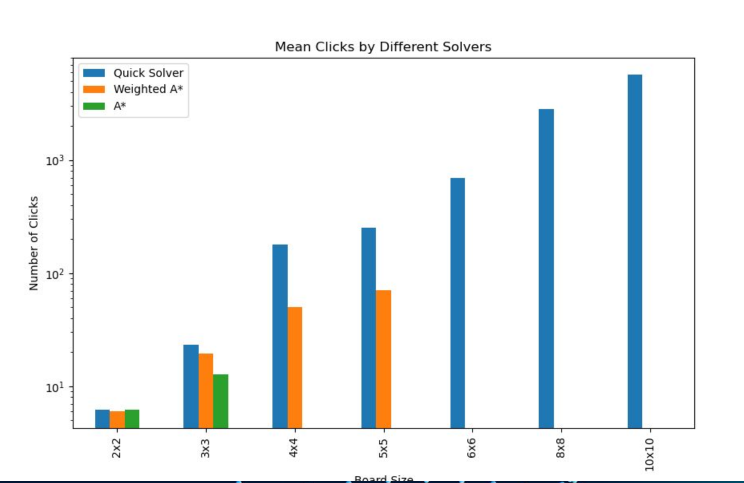


EXPERIMENTAL RESULTS

Uninformed Search

Iterative deepening - The IDDFS's methodology alone meant that its overall results were subpar. It took up more space and quickly became impractical for small boards since it required looking for particularly deep boards that required around 15 degrees of deepening, which **made the times less convenient**. For bigger boards it would **take days** to get a solution.

Uniform cost - Even though the UCS was still slower than DFS, it still provided us some decent times. For mid-sized boards, it might work, but **not for larger ones**. Up to mid-size boards, this is a good technique because it requires a lot less time and space than iterative deepening and the breadth-first search while still **guaranteeing a solution path**.



EXPERIMENTAL RESULTS

Informed Search

Greedy search - The only factor guiding board expansion was proximity to the final solution, which is a move that prioritizes changing a tile's color to match the goal color. Because there are circumstances in which the combination of local optimal solutions **does not ensure a global optimal solution**, the algorithm did not always produce the best answer.

A* Algorithm - The nodes were expanded in a way that minimized both the cost of the path to each step and the distance to the **ultimate solution**. Compared to the DFS, the A* algorithm consistently produced the best result taking less time and space.

Weighted A* - The weighted version **produced better outcomes** than the normal A*, although the same ideology was used. It took a lot less time to complete and maintained results that were extremely near to the typical A* by taking up a little more space.

All of the final version of these algorithms use the **color distance heuristic**.

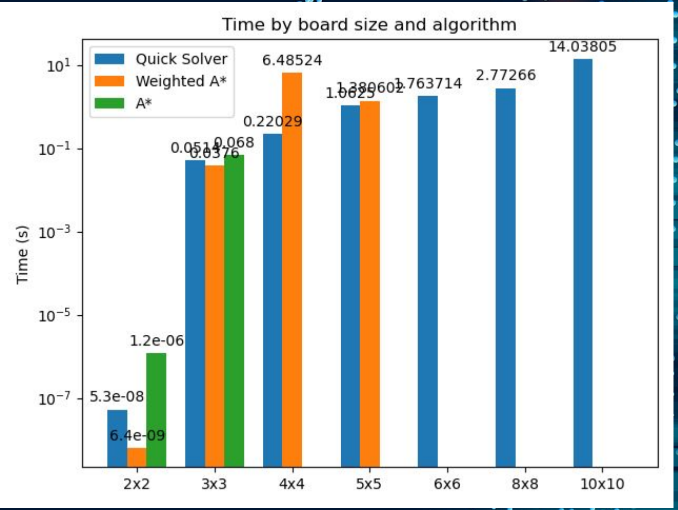
EXPERIMENTAL RESULTS

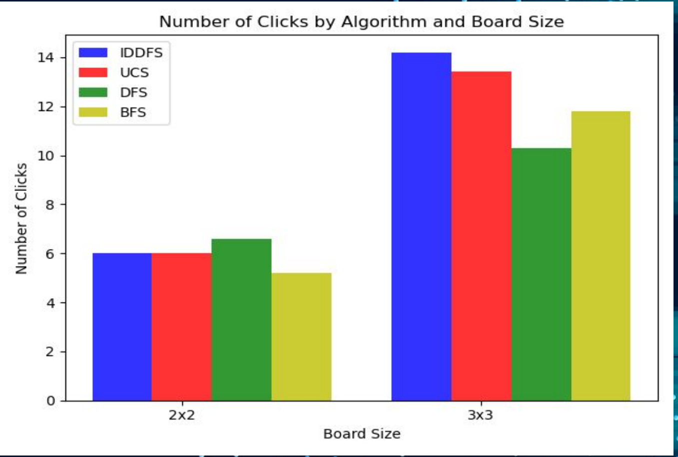
Informed & Uninformed Search

Monte Carlo - used for the hints, so in order to guante the best play at the moment a Monte Carlo with 15 simulations is played.

Quick Solver - by separating the board in smaller boards and then using the weighted A* algorithm to solve the smaller boards and then combining solutions gave us a good way to solve big boards in a really good time. Still there are boards that the algorithms can give a lot more moves than the optimal solution and takes a lot more space to solve.

All of the final version of these algorithms use the color distance heuristic.





Conclusion

Amado highlights the importance of understanding **algorithms**, showcasing their applicability not just within the game but also in **real-life problem-solving**. Through engaging gameplay, it demystifies complex concepts, making them accessible and enjoyable to learn. This project underlines the significant role that interactive learning can play in education, bridging the gap between **theoretical knowledge** and **practical application**. It's a reminder of how games can be a potent tool for learning, inspiring curiosity and insight into the algorithms that shape our digital world.

References

- Course Bibliography ("Artificial Intelligence: A Modern Approach" by Stuart Russell and Peter Norvig, and "Reinforcement Learning: An Introduction" by Richard S. Sutton)
- AI in Problem-Solving (ex: ChatGPT, Gemini, and Copilot)
- Open Source Implementations (ex: Github)
- Technical Blogs and Tutorials (GeeksforGeeks, and Stack Overflow)