Development of an autonomous agent for the game Rush Hour

Group Assignment for Artificial Intelligence

João Sousa, 103415
Vânia Morais, 102383

2022/2023

univers
de aveir

Architecture

Our code is split into two files: **student.py** and **RushHourSolver.py**. All the classes and functions used are in the RushHourSolver.py file:

- Within the **Solver** class we have the **recalcula()** function and the **run()** function. The **run()** function calls the **recalcula()** function under some specific conditions and puts the solution in place, meanwhile the **recalcula()** function invokes the functions for searching the best result.
- Inside the **SearchTree** and **SearchNode** classes the algorithm explained in the next slide is applied.

The Algorithm

To develop the autonomous agent for the game Rush Hour, the first step was to find an efficient way to solve the problem.

We then decided to apply what we were taught in class and used an informed search strategy (A*). This search consists of choosing the node where the total cost is lowest.

This total cost is given by the sum of the cost from the initial node to node n and the heuristics (described on the next slide).

Heuristics

The quality of a heuristic can be measured through the effective branching factor and the better the heuristic, the lower this factor.

In this work we use two heuristics:

- **The distance from the red car to the wall**
- **Number of cars that prevent the red car from advancing**

The combination of these two heuristics gave origin to the function **heuristicas()**

Conclusion

Regarding the first delivery, we made one more heuristic and improved some points, which improved the response time, but not enough to complete all levels.
We believe that the problem is the algorithm that is not the most appropriate for the desired result.