Search Algorithms in Java language (Unblock Me/ Group Nº38)

Dinis Moreira (up201503092)  
MIEIC  
FEUPPorto, Portugal  
up201503092@fe.up.pt

Diogo Dores (up201504614)  
MIEIC  
FEUPPorto, Portugal  
up201504614@fe.up.pt

Luís Oliveira (up201604343)  
MIEIC  
FEUPPorto, Portugal  
up201604343@fe.up.pt

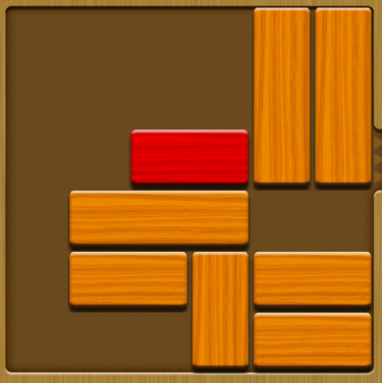
Abstract—In this document we describe a simple game, formulate it as a search problem, solve it with different algorithms, then we analyse the results and how we can benefit from them.

Keywords—Artificial Intelligence, Search, A\* Algorithm, Uniform Cost Algorithm, Greedy Algorithm, Depth First Algorithm and Breadth First Algorithm

# Introduction

In this project we will implement an application capable of resolving instances of the game “Unblock Me”, without any external interaction, using search algorithms of Artificial Intelligence, namely A\* Algorithm, Uniform Cost Algorithm, Greedy Algorithm, Depth First Algorithm and Breadth First Algorithm.

# Problem Description

* The game is played on a 6x6 board, with one special piece (typically 1x2), and several other that can vary in size and orientation, but always 1xY (pieces that move vertically) or Xx1 (pieces that move horizontally). The objective of the game is to move the special piece to the right side of the board, moving the other pieces out of the way to make room for it. The challenge lays in the fact that the small size of the board does not allow the pieces to move much without being obstructed by others, sometimes several pieces need to be moved before making room for a certain one to move just one position.
* Released in 2009, the game is the most popular release of the Thai company Kiragames [2]. Although hard to find evidence, the game resembles an early 20th century game called Klotski, which was included in Windows3.1’s Windows Entertainment Pack [3].

# Formulation of the problem

State Representation: A board can be represented with a 6x6 matrix whose positions range from (0, 0) on the topmost left to (5, 5) on the bottommost right. Every piece can be represented with letters ranging from “A” to “Z” where every piece that occupies multiple positions is represented with the same letter. Two or more pieces cannot occupy the same position and blank space is represented with a dot (“.”).

Initial State: A board will always start with a key piece facing a target side and there might be pieces obstructing its movement but, by moving the surrounding pieces, it’s always possible to achieve a target state. In the earlier levels played, the key piece is always located on the third row.

Target State: The game ends when a key piece reaches a target side. In the earlier levels played, this target side is always the rightmost of the board and can be achieved if the key piece reaches the matrix position (2, 5) (third row, sixth column). As with most puzzles of this nature, the goal should be to finish a level within the least amount of moves possible.

Operators:

P – Letter of the current piece

xPos – Position of the current piece within the horizontal axis.

yPos – Position of the current piece within the vertical axis.

|  |  |  |  |
| --- | --- | --- | --- |
| Name: | Pre-conditions: | Effects: | Costs: |
| MoveUp | yPos > 0;  (xPos, yPos-1)  = “.” | (xPos, yPos-1)  = P;  (xPos, yPos) = “.” | 1 |
| MoveDown | yPos < 5;  (xPos, yPos+1)  = “.” | (xPos, yPos+1)  = P;  (xPos, yPos) = “.” | 1 |
| MoveLeft | xPos > 0;  (xPos-1, yPos)  = “.” | (xPos-1, yPos)  = P;  (xPos, yPos) = “.” | 1 |
| MoveRight | yPos < 5;  (xPos+1, yPos)  = “.” | (xPos+1, yPos)  = P;  (xPos, yPos) = “.” | 1 |

# Related Work

During our research there were several projects that we found which will likely assist us throughout the course of the development of the project. The first example details the development of a solver of the Unblock Me game [4]. It uses a Breadth First algorithm and it is implemented in C++ [5]. The second project is also an implementation of this game, but with a couple differences. It is developed using the Python language and the algorithm implemented is the A\* [6].

# game Implementation

This project was developed in the Java programming language

Descrevendo o projeto e implementação, na linguagem selecionada, do jogo incluindo a forma de representação do estado do tabuleiro, operadores (verificação do cumprimento das regras do jogo) aplicáveis com determinadas pré-condições e que têm efeitos sobre o estado do jogo e um dado custo, teste objetivo (determinação do final do jogo). Entre outras devem ser implementadas funções: ler nível de ficheiro (lendo um dado nível/estado de um ficheiro de texto), visualizar em modo de texto/gráfico um dado estado, validar uma dada jogada/operador (tendo em conta as suas pré-condições), executar uma dada jogada/operador, num dado tabuleiro, tendo em conta os seus efeitos e gerando o respetivo estado sucessor, listar todas as jogadas/operadores disponíveis num dado tabuleiro, avaliar um dado estado (tendo em conta a sua “proximidade” à solução final), testar se um dado estado é solução (teste objetivo). Os métodos de pesquisa para cálculo das jogadas a realizar que permitam ao computador jogar sozinho e resolver os puzzles devem ser descritos na secção seguinte assim como o método geral para os chamar e resolver o puzzle (utilizando um dado método selecionado de entre os disponíveis).

# Search algorithms

Descrevendo os vários algoritmos de pesquisa utilizados e a sua implementação de modo a calcular a próxima jogada do PC ou retornar a solução final (conjunto de operações para transformar o estado inicial no estado objetivo). Devem ser implementados algoritmos para cálculo da solução utilizando pesquisa em largura, pesquisa em profundidade (se aplicável), aprofundamento progressivo, custo uniforme (se aplicável), pesquisa gulosa e Algoritmo A\* (estes último método utilizando várias heurísticas).

# Experiences and results

Descrevendo as experiências realizadas com os vários algoritmos para resolver diversos puzzles e os resultados obtidos a nível de tempo e custo da solução obtida em cada nível, por cada um dos métodos experimentados. Devem ser incluídas tabelas comparativas dos resultados obtidos na aplicação dos vários métodos aos vários puzzles (níveis do jogo) e discutidos os resultados.

# Conclusions and Development Prespectives

Sumário do trabalho realizado e conclusões que retira deste projeto. Análise crítica dos resultados obtidos em comparação com os resultados teóricos que seriam esperados. Trabalho futuro, ou seja, formas de melhorar o trabalho desenvolvido.

##### References

1. Stuart Russel and Peter Norvig, “Artificial Intelligence: A Modern Approach”, Third Edtition, Pearson Education Inc., 2010, ISBN: 978-0-13-604259-4.
2. Stuart Russel and Peter Norvig, “AimaCode - Code for the Book Artificial Intelligence: A Modern Approach", 2019, [online], available at: <https://github.com/aimacode> , consulted on March 2019.

Livros, artigos e páginas Web utilizados para desenvolver o trabalho. Todos os elementos bibliográficos devem ser citados no texto do trabalho, incluindo qualquer código fonte adaptado de uma dada fonte para a realização do trabalho.