# Resolução da Segunda Atividade de Inteligência Artificial em Java

1st Rui Guedes
*Universidade do Porto*
*FEUP*
Porto, Portugal
up201603854@fe.up.pt

***Resumo*—No âmbito da disciplina de Inteligência Artificial, pretende-se desenvolver um programa em Java capaz de, através da utilização de diferentes métodos de pesquisa e heurísticas/conhecimento, resolver um puzzle, comparando posteriormente o seu desempenho. *N-Puzzle* é o puzzle escolhido. Este puzzle consiste num puzzle 2D cuja representação é feita através de uma matriz bidimensional de tamanho variável.**

**A formulação deste jogo encontra-se descrita no presente artigo com o objetivo de explicitar o problema em questão. Para a resolução deste problema são utilizados três algoritmos de pesquisa: pesquisa em largura, pesquisa gulosa e pesquisa usando o algoritmo A*. Para estes estes dois últimos são utilizadas duas heurísticas distintas. Implementados estes três algoritmos é possível testar e comparar o seu desempenho neste problema. Analisando os resultados obtidos no conjunto de problemas testados, concluiu-se que os algoritmos de pesquisa informada obtiveram melhor desempenho no conjunto de testes realizados.**

***Keywords*—Inteligência Artificial, *N-Puzzle*, Pesquisa em Largura, Pesquisa Gulosa, Algoritmo A*, Pesquisa Heurística.**

## I. INTRODUÇÃO

O presente artigo surge face à necessidade de resolver a segunda atividade proposta na disciplina de inteligência artificial cujo principal objetivo é o desenvolvimento de um programa para a resolução de um problema recorrendo a diferentes métodos de pesquisa e heurísticas/conhecimento e, posteriormente, a comparação do desempenho dos algoritmos utilizados. O problema a abordar é o N-Puzzle, que consiste num puzzle bidimensional cujo objetivo consiste por sua vez na ordenação da matriz por ordem crescente.

Quanto às estratégias de pesquisa, irão ser utilizados algoritmos de pesquisa não informada como a pesquisa pesquisa em largura, bem como algoritmos de pesquisa informada, isto é, que utilizam heurísticas e conhecimento para alcançar a solução, sendo a pesquisa gulosa e o algoritmo A* algoritmos representativos deste tipo de pesquisa e que serão utilizados nesta atividade.

O presente artigo serve para estruturar e apresentar a resolução aos exercícios referentes à atividade proposta. Deste modo o artigo encontra-se estruturado da de forma a responder às diferentes alíneas que constituem o problema.

## II. ALÍNEA A - FORMULAÇÃO DO PROBLEMA

A resolução deste problema requer que seja encontrado um estado da matriz, atingível a partir do estado inicial, que se encontre ordenada por ordem crescente para que o problema possa ser considerado resolvido. Assim sendo, estamos perante um problema de pesquisa.

### A. Representação do Estado

Para a representação dum estado do problema, é necessário ter conhecimento do seu ambiente nesse momento. Esta representação pode ser efetuada através de uma representação matricial de N células, em que N-1 células encontram-se preenchidas e apenas uma se encontra livre.

### B. Estado Inicial

O estado inicial do problema varia de problema para problema e corresponde a uma matriz bidimensional cujos valores presentes na matriz se encontram desordenados.

### C. Teste Objetivo

O teste objetivo consiste em verificar se a matriz corresponde á configuração objetivo, isto é, se corresponde a própria matriz ordenada por ordem crescente.

### D. Operadores

Os operadores correspondem às ações possíveis que possibilitam a transição entre estados. Existem no máximo 4 ações possíveis, são elas mover espaço vazio para esquerda, direita, cima ou abaixo. Na sua totalidade estas quatro ações permitem o ordenamento da matriz por ordem crescente e consequentemente alcançar a resolução do problema.

## III. ALÍNEA B E C - IMPLEMENTAÇÃO DE CÓDIGO

O código implementado encontra-se anexado no mesmo ficheiro zip em que o presente artigo se encontra.

## IV. ALÍNEA D - EXPERIÊNCIAS E RESULTADOS

Usando um conjunto de problemas definidos, testaram-se os três algoritmos para cada problema, recolhendo métricas sobre o seu desempenho, entre estas, o tempo (ms) e memória gastos (MB). Os resultados obtidos podem ser visualizados nas seguintes tabelas:

| Algoritmo | Tempo | Memória |
|---|---|---|
| BFS | 13 | 4,2 |
| Greedy $^{H1}$ | 5 | 5,7 |
| Greedy $^{H2}$ | 1 | 5,7 |
| A* $^{H1}$ | 1 | 6,3 |
| A* $^{H2}$ | 2 | 6,3 |

Tabela 1: Resultados obtidos para o Problema 1

| Algoritmo | Tempo | Memória |
|---|---|---|
| BFS | 21 | 4,7 |
| Greedy $^{H1}$ | 5 | 6,3 |
| Greedy $^{H2}$ | 1 | 6,8 |
| A* $^{H1}$ | 2 | 6,8 |
| A* $^{H2}$ | 1 | 6,8 |

Tabela 1: Resultados obtidos para o Problema 2

| Algoritmo | Tempo | Memória |
|---|---|---|
| BFS | 7 | 24,6 |
| Greedy $^{H1}$ | 27 | 14,1 |
| Greedy $^{H2}$ | 4 | 25,1 |
| A* $^{H1}$ | 2 | 14,6 |
| A* $^{H2}$ | 1 | 14,6 |

Tabela 1: Resultados obtidos para o Problema 3

| Algoritmo | Tempo | Memória |
|---|---|---|
| BFS | 77 | 19,4 |
| Greedy $^{H1}$ | 9 | 21,5 |
| Greedy $^{H2}$ | 27 | 24,6 |
| A* $^{H1}$ | 2 | 25,1 |
| A* $^{H2}$ | 1 | 25,2 |

Tabela 1: Resultados obtidos para o Problema 4

Recorrendo às quatro tabelas anteriores é possível observar que o algoritmo não informado obteve resultados maioritariamente em conformidade com o expectável, dum ponto de vista teórico. Relativamente aos algoritmos de pesquisa informada estes têm um desempenho substancialmente melhor do que os de pesquisa não informada. As heurísticas utilizadas provaram ser eficazes em obter boas soluções em diferentes problemas.

## REFERÊNCIAS BIBLIOGRÁFICAS

[1] GitHub. (2019). aimacode/aima-java. [online] Available at: https://github.com/aimacode/aima-java?fbclid=IwAR0rlZcEBg9rBzeuoziqQCLOFvrhP1vOKHWc4D2j7zRnwbw-Za566lxB110 [Accessed 17 Mar. 2019].