

**CENTRO UNIVERSITÁRIO FEI**

**DIOGO F. DE M. SANTIAGO**

**TRAB. 02**

**Implementação Algoritmos de Busca**

**Documentação**

São Bernardo do Campo

2023

## Overview

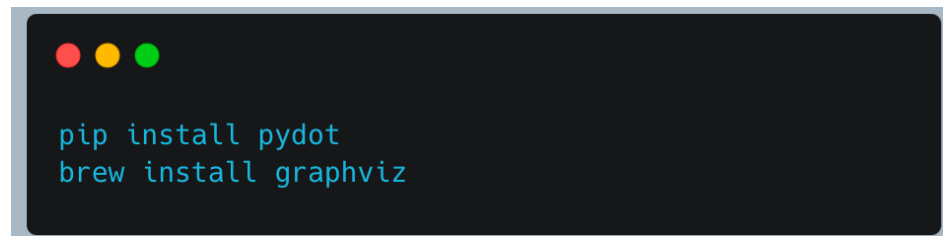
Este trabalho visa implementar os seguintes algoritmos: BFS, DFS, Greedy Best-First Search e A Star. Ele será entregue via Moodle e via um *github* para na intenção de visualizar os resultados.

## Instalação

Há a necessidade da instalação de 2 bibliotecas: pydot e graphviz.

O pydot é um framework que se utiliza do graphviz para manipulação facilitada de seus elementos, Nodes, Edges, etc.

O graphviz deve ter uma instalação local pois é uma aplicação independente do Python para visualização de grafos. Maiores detalhes de como da instalação no link: <https://graphviz.org/download/>

A terminal window with a dark background and three colored window control buttons (red, yellow, green) in the top-left corner. It contains two lines of text in a light blue monospace font: 'pip install pydot' and 'brew install graphviz' on separate lines.

```
pip install pydot
brew install graphviz
```

## Abordagem

A aplicação é feita da seguinte maneira, os estados têm o seguinte estilo **(M, C, L)** onde **M** é o número de missionários, **C** o número de canibais e **L** o lado em que o barco se encontra onde sendo 1 o lado inicial e 0 o lado oposto. A contagem de missionários e canibais é dada por sua quantidade no lado inicial, com o estado inicial sendo **(3, 3, 1)**. Para análise dos resultados foi usado o link do jogo web <https://www.novelgames.com/en/missionaries/> e neste caso, o **L** é o lado direito(origem). O estado final é o **(0, 0, 0)** onde não há nem missionários nem canibais do lado de origem. A intuição desta abordagem vem de um *assignment* de outra Universidade sob o título "AI Assignment II.pdf" que consta junto com os documentos enviados.

As ações são caracterizadas pelas 5 possibilidades:

- (1, 0): Move-se 1 missionário.
- (2, 0): Movem-se 2 missionários.
- (0, 1): Move-se 1 canibal.
- (0, 2): Movem-se 2 canibais.
- (1, 1): Movem-se 1 missionário e 1 canibal.

## Heurísticas

Foi utilizada 2 heurísticas, uma para o GBFS e para o AStar, pelo fato de escolhermos sempre o estado com o menor custo a soma de **M** + **C** do lado de origem precisa ser a menor possível ou o estado precisa ser o menor possível comparado ao estado final **(0, 0, 0)**, ambas abordagens funcionam.

## Resultados

Os resultados são visualizados pelos gráficos gerados no *graphviz*. Sua interpretação é muito simples onde estados impossíveis nem aparecem estados válidos, porém sem filhos são pintados de **vermelho** e o estado final pintado de **verde**.

Nas arestas há a informação da ação feita e no caso do GBFS e AStar que possuem custos, este também é adicionado p/ visualização. Lembrando que o GBFS tem custo  $f(x) = h(x)$  e o AStar  $f(x) = g(x) + h(x)$ , ou seja, o AStar leva em consideração a profundidade também e adiciona ao custo total de se movimentar para aquele estado.

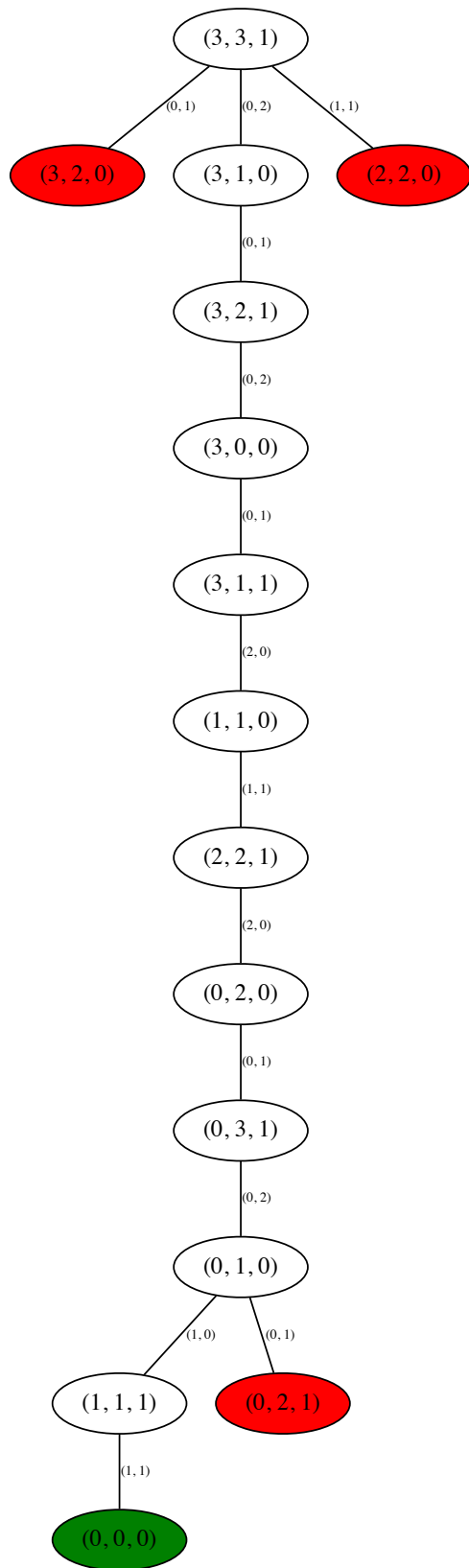


Gráfico gerado do BFS

## Curiosidades

Tanto o GBFS e o AStar, que se utilizam de custos, foi adicionado um campo ao estado irrelevante que é a posição em que foi visitado em caso válido. Isto foi utilizado pois em caso de empate na prioridade(custo), essa biblioteca (*PriorityQueue*) ordena usando-se do *hash* do valor de entrada e a ordem dos estados não era priorizada na ordem em que foi visitada.