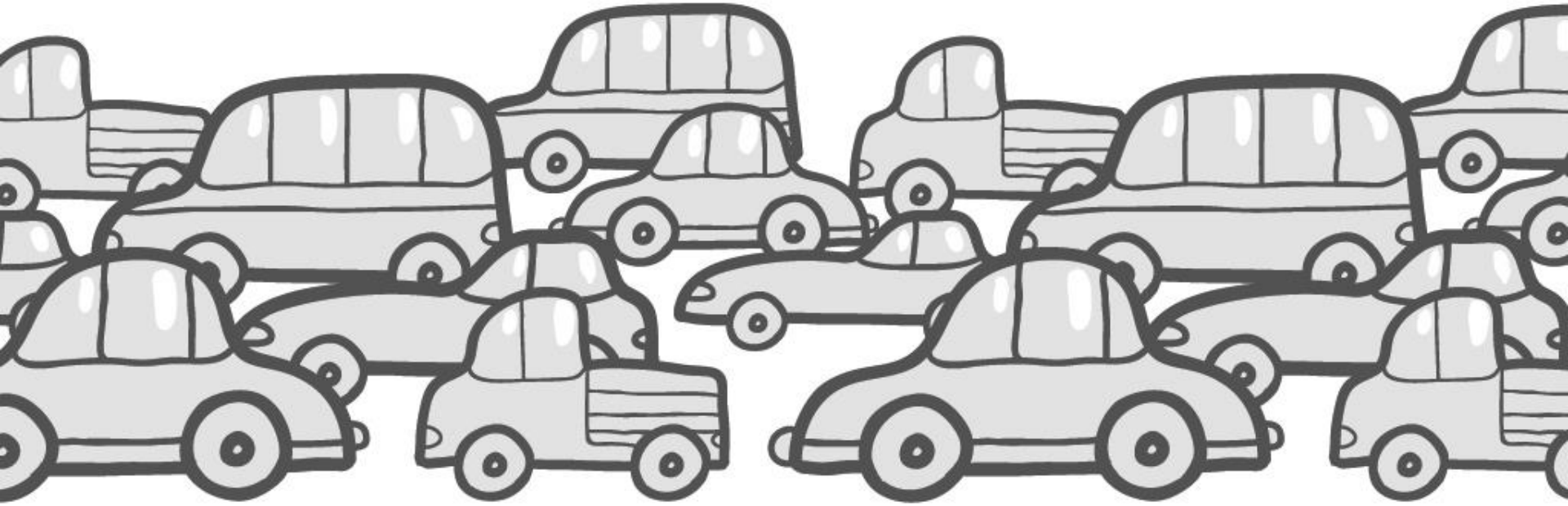

TPG - Rush Hour

João Figueiredo, 98506

DETI – Universidade de Aveiro

Inteligência Artifical

Prof. Diogo Gomes

Prof. Luís Lopes

Introdução – Criação do agente

- Para a criação do nosso agente, foram usados os módulos da treeSearch do guião prático e adaptados para o contexto do jogo, **RushHour**.

- Foram criadas funções adicionais para auxiliar a expansão de nós na árvore e mais facilmente manipular a grelha do jogo.

Funções adicionais

- Criou-se uma cópia do Common.py, com.py e criou-se a função:
  - **Type_actions()** → Para um dado carro, calcula o seu tamanho e direção. Com esta informação verifica se pode realizar um ou os dois movimentos possíveis. Se está na vertical, poderá deslocar-se para baixo e/ou cima. Se está na horizontal, poderá deslocar-se para a esquerda e/ou direita. Todos os movimentos são de 1 unidade.
- Criou-se o ficheiro Domain.py, onde possuímos as funções de auxílio à searchTree:
  - **Actions()** → Para um dada grid, devolve todas as ações possíveis, ou seja, todos os movimentos que todos os carros podem realizar.
  - **Result()** → Para um dada grid e um movimento (action) de um carro, calcula como ficará a grelha após esse movimento e devolve-a com a nova posição do carro.
  - **Cost()** -> Função a qual atribui um custo ao nó.
  - **Heuristic()** -> Função heurística, que atribui o custo heurístico ao nó. Este custo é igual ao número de carros que estão a bloquear o caminho.
- No Student.py, foram usadas threads e foi criada uma função para lidar com o plano/sequência de movimentos após a pesquisa ter sido realizada:
  - **Get_Next_Move()**

Pesquisa/TreeSearch

- A função search(), pertencente à class searchTree, a partir do nó inicial (grid inicial), faz as expansões dos vários nós.
- É usado um algoritmo A* para realizar a pesquisa.
- À medida que a função ia sendo testada, alterações foram feitas, tais como a implementação de uma priorityQueue para ordenar os elementos em base do custo.
- Foi também usada a estrutura de dados, set(), para evitar expandir um nó que já foi expandido.

Estado do agente

- Numa fase inicial, criou-se um levels.txt novo, um subconjunto de levels.txt original (80 níveis), no qual estavam os níveis mais difíceis para testar o agente.
- Após cada jogo realizado, várias otimizações foram feitas, assim chegando ao estado em que o agente se encontra atualmente.
- Uma vez que o agente é capaz de resolver todos os níveis, posso concluir que a solução desenvolvida é eficiente.