Dots and Boxes: Manual Utilizador

Inteligência Artificial - Escola Superior de Tecnologia de Setúbal

André Dias - 201901690

João Caetano - 201901961

2022/2023

Prof. Joaquim Filipe Eng. Filipe Mariano

1. Acrónimos e convenções usadas

- Nó - Uma estrutura de dados que contem informação de um ponto numa árvore de procura.
- BFS - Breadth-first-search (Pesquisa em largura primeiro). Um algoritmo de procura onde os nós de um nível de uma árvore de procura são todos gerados primeiro, tipicamente da esquerda para a direita.
- DFS - Depth-first-search (Pesquisa em profundidade primeiro). Um algoritmo de procura onde o nó mais à esquerda na árvore é expandido primeiro, até uma profundidade máxima.
- A* - A star (A asterisco). Um algoritmo de procura informado, onde a procura é guiada por uma função *h* que avalia o melhor nó a explorar a seguir.
- h - No contexto deste projeto, H quer simplesmente dizer Heurística ou Função heurística.

2. Introdução

Este documento servirá como guia de instalação, utilização e troubleshooting do utilizador comum. Visa explicar de forma sucinta o objetivo do projeto **Dots and Boxes**, feito para quem quer estudar algumas soluções para certas configurações de um tabuleiro do jogo **Dots and Boxes**.

3. Instalação e utilização

Para que o programa corra, sugere-se a instalação de um ambiente de programação lisp

- Lispworks
- GNU CLISP

Após a sua instalação, abra a linha de comandos (listener no lispworks / escrever clisp na linha de comandos de escolha caso opte por GNU CLISP) e escreva o seguinte comando:

```
(load (compile-file "projeto.lisp"))
```

Ao carregar em enter, se não houve qualquer tipo de erro, o programa fica pronto a executar! Se ocorrer algum erro durante a compilação do ficheiro "projeto.lisp" verifique se os ficheiros que vieram distribuídos com este programa estão todos na mesma pasta.

Para executar o programa, basta escrever:

```
(start)
```

E seguir as instruções que aparecem na linha de comandos.

4. Input/Output

Após execução do comando anterior, deverá estar perante o seguinte output:

```
Welcome to the Dots and Boxes problem solving project!
Developed by: Andre Dias and Joao Caetano

Choose the problem to solve:
1. Board:
... ; truncado por extensibilidade do output
```

O programa lê os tabuleiros a partir de um ficheiro na mesma pasta chamado "problemas.dat". Aí estarão por linha a representação do tabuleiro inicial e o número de caixas que se quer fechar para esse problema particular. Se não forem apresentados tabuleiros nenhuns, ou encontre um erro, certifique-se que o ficheiro "problemas.dat" existe e tem conteúdo devidamente formatado:

```
(((linhas)(colunas))numero de caixas) ; 0 - casa não preenchida, 1 - casa
preenchida
```

Se os tabuleiros forem imprimidos no seu ecrã, siga os próximos passos para escolher o algoritmo a utilizar e processar o problema. Os resultados serão mostrados no final da execução do algoritmo e serão escritos para um ficheiro "results.txt".

5 Exemplo de aplicação

O funcionamento expectável segue um caminho parecido ou idêntico a este:

```
Welcome to the Dots and Boxes problem solving project!
Developed by: Andre Dias and Joao Caetano

Choose the problem to solve:
1. Board:
.    .    .    .

.    .    .----.
     |         |
.    .----.----.
          |    |
.    .    .----.
```

```
2. Board:
.    .    .----.    .
          |    |
.----.----.----.----.
          |         |
.    .    .----.----.
|    |    |    |    |
.    .    .----.----.
|    |    |    |    |
.    .    .----.----.

1 (input)
Choose the searching algorithm:

1.BFS
2.DFS
3.A*
1 (input)

Calculating...
Board:
.    .    .    .

.    .----.----.
     |    |    |
.    .----.----.
          |    |
.    .    .----.


Number of generated nodes: 91
Number of expanded nodes: 6
Algorithm used: BFS
Run time: 0.00996ms
Penetrance: 0.021978023
Effective branching factor: 2.043457
Path to solution: ((((0 0 0) (0 1 1) (0 1 1) (0 0 1)) ((0 0 0) (0 1 0) (0 1 1) (0
1 1))) (((0 0 0) (0 1 1) (0 1 1) (0 0 1)) ((0 0 0) (0 1 0) (0 0 1) (0 1 1))) (((0
0 0) (0 0 1) (0 1 1) (0 0 1)) ((0 0 0) (0 1 0) (0 0 1) (0 1 1))))
```

O caminho mais comum que gera a erro, é na utilização dos algoritmos BFS ou DFS para problemas que tenham uma grande quantidade de jogadas possíveis, pois demora demasiado tempo ou acaba por acontecer uma exceção de overflow. Nesses casos, o output "Calculating..." acaba por ficar preso na consola. Se acontecer isto, basta reiniciar o ambiente de programação que escolheu no passo de instalação e recomeçar o processo de instalação a partir do comando para compilar o ficheiro "projeto.lisp".