Manual do Utilizador

IPS Instituto Politécnico de Setúbal
Escola Superior de Tecnologia de Setúbal

Projeto: Resolução do Jogo **O Solitário (Peg Solitaire)**

1. Capa

- **Unidade Curricular:** Inteligência Artificial
- **Projeto:** Projeto 01 – Peg Solitaire Solver
- **Ano Letivo:** 2025/2026
- **Linguagem:** Common Lisp
- **Professor:** Filipe Mariano

Autores:

- Dinis Xavier | 202300148
- Rui Monteiro | 202300265
- Daniel Pais | 202200286

2. Acrónimos e Convenções Usadas

Acrónimos

- **BFS** – *Breadth-First Search* (Procura em Largura)
- **DFS** – *Depth-First Search* (Procura em Profundidade)
- **A*** – Algoritmo A estrela (procura informada)
- **IDA*** – *Iterative Deepening A**
- **IA** – Inteligência Artificial

Convenções

- **Comandos introduzidos pelo utilizador** aparecem em `fonte monoespaçada`.
- **Ficheiros** são apresentados em *itálico* (ex.: *problemas.dat*).
- **Saída no ecrã** é mostrada como texto formatado.
- Valores `1`, `0` e espaço representam respetivamente **pino**, **cavidade vazia** e **posição inválida** do tabuleiro.

3. Introdução

Para quem é este programa?

Este programa destina-se a estudantes e docentes da área de Inteligência Artificial, bem como a qualquer utilizador interessado em compreender e comparar algoritmos de procura aplicados à resolução do jogo **O Solitário (Peg Solitaire)**.

Para que serve?

O programa permite resolver automaticamente diferentes configurações do jogo Peg Solitaire, explorando o espaço de estados através de vários algoritmos de procura:

- Procura não informada (BFS e DFS)
- Procura informada (A* e IDA*)

Além de encontrar soluções, o programa recolhe e apresenta **estatísticas de desempenho**, permitindo a comparação entre algoritmos e heurísticas.

Problemas que resolve

- Encontrar uma sequência válida de jogadas desde um estado inicial até um estado final (um único pino no tabuleiro).
- Determinar soluções ótimas (menor número de jogadas) quando aplicável.
- Avaliar eficiência dos algoritmos de procura.

---

4. Instalação e Utilização

4.1 Requisitos

- Interpretador Common Lisp (ex.: **SBCL**, **CLISP**)
- Sistema operativo compatível com Common Lisp

4.2 Estrutura do Projeto

O projeto deve conter os seguintes ficheiros no mesmo diretório:

- `projeto.lisp` – Front-end e interface com o utilizador
- `procura.lisp` – Implementação genérica dos algoritmos de procura
- `puzzle.lisp` – Definição do domínio, operadores e heurísticas
- `problemas.dat` – Lista de problemas (tabuleiros iniciais)

4.3 Instalação

1. Copiar todos os ficheiros do projeto para um diretório local.
2. Abrir o interpretador Common Lisp.
3. Definir o diretório de trabalho para a pasta do projeto.
4. Carregar o ficheiro principal:

```
(load "projeto.lisp")
```

4.4 Arranque do Programa

Para iniciar o programa, executar o seguinte comando no Listener:

```
(iniciar)
```

O sistema carregará automaticamente os módulos e os problemas definidos no ficheiro *problemas.dat*.

---

5. Input / Output

5.1 Input (Entrada)

Interativo (teclado)

O utilizador deverá fornecer:

1. **Escolha do problema** (número do tabuleiro)
2. **Escolha do algoritmo** (BFS, DFS, A*, IDA*)
3. **Parâmetros adicionais**:
    - Limite de profundidade (no caso do DFS)
    - Heurística (no caso de A* e IDA*)

Ficheiros

- *problemas.dat*: contém os estados iniciais dos tabuleiros, separados por `###`.

5.2 Output (Saída)

Ecrã

- Visualização do tabuleiro inicial
- Indicação do algoritmo em execução
- Apresentação da solução (se encontrada)
- Sequência completa de estados
- Mensagens de erro ou insucesso

Ficheiros

- `resultados.txt`: ficheiro gerado automaticamente com:
    - Problema utilizado
    - Algoritmo utilizado
    - Existência de solução
    - Profundidade da solução
    - Número de nós gerados
    - Número de nós expandidos
    - Fator de ramificação médio
    - Penetrância
    - Tempo de execução
    - Caminho completo da solução

---

6. Exemplo de Aplicação

6.1 Execução com Sucesso (Linha Típica)

1. Iniciar o programa:

```
(iniciar)
```

2. Selecionar um problema:

```
1 - Problema 1
2 - Problema 2
Escolha: 1
```

3. Selecionar o algoritmo:

```
1 - BFS
2 - DFS
3 - A*
4 - IDA*
Escolha: 3
```

4. Selecionar a heurística:

```
1 - h1
2 - h2
Escolha: 1
```

5. O programa executa o algoritmo, mostra a solução no ecrã e grava os resultados em `resultados.txt` que será gerado no mesmo diretório dos ficheiros.

6.2 Situações de Insucesso

- **Solução não encontrada**: o programa informa o utilizador e regista as estatísticas disponíveis.
- **Escolha inválida**: o menu é reapresentado automaticamente.
- **Limite de profundidade insuficiente (DFS)**: pode impedir a descoberta da solução.

---

7. Limitações do Programa

O programa apresenta as seguintes limitações:

- A interface é **exclusivamente textual** (linha de comandos).
- O utilizador deve conhecer conceitos básicos de algoritmos de procura para interpretar os resultados.
- O tempo de execução pode ser elevado para problemas complexos, especialmente com BFS.
- O formato do ficheiro *problemas.dat* deve ser rigorosamente respeitado.

- Não existe validação gráfica dos tabuleiros introduzidos.
- Só estão implementados 4 algoritmos com a ausência de SMA* e RBFS.

---

8. Considerações Finais

Este programa constitui uma ferramenta educativa para o estudo e comparação de algoritmos de procura em espaço de estados, aplicados a um problema clássico da Inteligência Artificial. O utilizador pode experimentar diferentes algoritmos, heurísticas e configurações, analisando tanto as soluções obtidas como o seu desempenho computacional.

Fim do Manual do Utilizador