



FACULDADE DE  
CIÊNCIAS E TECNOLOGIA  
UNIVERSIDADE DE  
**COIMBRA**

*Departamento de Engenharia Informática*

Introdução à Inteligência Artificial  
2019/2020 - 2º Semestre

Trabalho Prático Nº2:  
*D31*  
*The Last Search*

**Nota:** A fraude denota uma grave falta de ética e constitui um comportamento inadmissível num estudante do ensino superior e futuro profissional licenciado. Qualquer tentativa de fraude levará à anulação da componente prática tanto do facilitador como do prevaricador, independentemente de acções disciplinares adicionais a que haja lugar nos termos da legislação em vigor. Caso haja recurso a material não original, as **fontes** devem estar explicitamente indicadas.

# 1 Introdução

O nosso D31, depois de bater tantas vezes contra a parede no TP1, ficou com alguns dos seus módulos avariados e teve de ser sujeito a manutenção. Apesar deste infortúnio, o D31 sofreu algumas actualizações, e foram-lhe adicionados módulos mais poderosos, que lhe vão permitir coleccionar os recursos de forma mais eficaz. Em concreto, foi-lhe adicionado um módulo de procura básico que lhe permite calcular o caminho mais curto até aos recursos o que lhe permite mover-se de um recurso para outro de forma eficiente. No entanto isso não chega, e para apanhar todos os recursos de forma eficiente o D31 tem que planear as suas acções descobrindo a melhor ordem de recolha dos recursos. Infelizmente, o módulo de planeamento foi adquirido numa lixeira, e está com alguns problemas.

Assim o seu objectivo passa por ajudar a desenvolver um novo módulo de planeamento através do desenvolvimento de métodos de procura que permitam ajudar o D31 a encontrar a ordem pela qual deve apanhar os recursos presentes no ambiente percorrendo a menor distância possível.

A Fig. 1 mostra um exemplo de um ambiente que o D31 terá de percorrer. Como podemos observar, o nosso agente é colocado no canto inferior direito de um ambiente cheio de recursos numerados (neste caso de 1 a 12). O objectivo passará por encontrar a sequência pelo qual cada recurso deve ser apanhado de forma a minimizar a distância total percorrida. Uma possível solução é a seguinte: [12, 8, 10, 1, 9, 2, 6, 11, 4, 3, 5, 7]. Isto significa que o D31 ira recolher primeiro o recurso com o número 12, depois o recurso com o número 8, seguido do recurso com o número 10 e assim sucessivamente até chegar ao recurso 7. Uma solução alternativa, com um custo mais baixo seria [12, 10, 1, 2, 6, 9, 5, 7, 8, 3, 11, 4].

## 2 Objectivos Genéricos

O presente trabalho prático tem como objectivos genéricos:

1. Aquisição de competências no que diz respeito à modelação de algoritmos de procura heurística;
2. Aquisição de competências relacionadas com a análise, desenvolvimento, implementação de algoritmos de procura estocástica;
3. Compreensão das forças, fraquezas e dificuldades inerentes às diferentes abordagens;

Estes objectivos genéricos serão alcançados através do trabalho em grupo e da experimentação, promovendo-se, assim, estas capacidades.

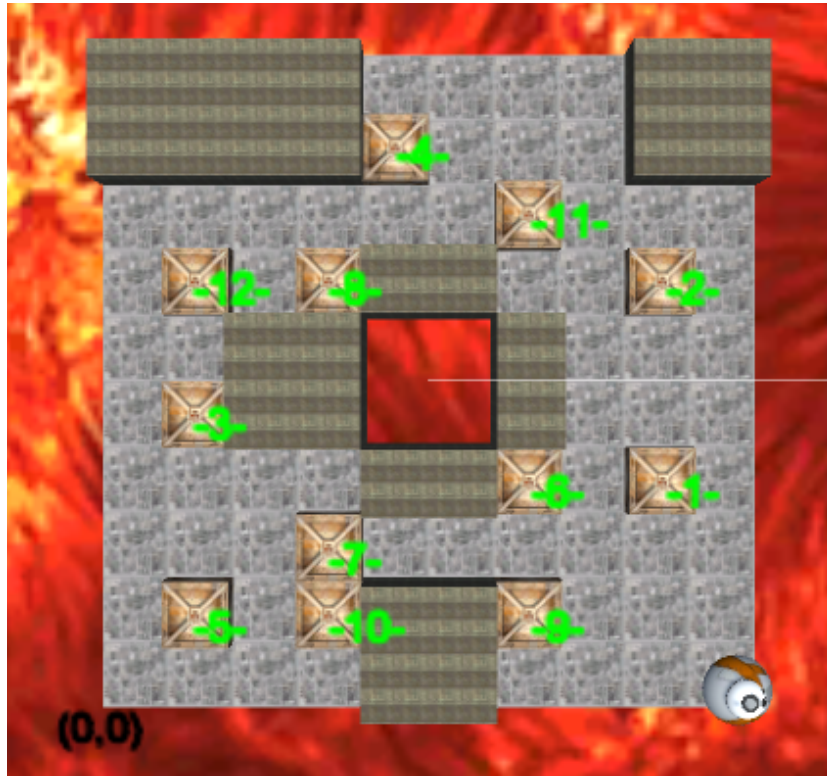


Figura 1: Exemplo do ambiente do agente D31.

### 3 Enunciado

De forma genérica, pretende-se utilizar métodos de procura heurística e estocástica para ajudar o D31 a coleccionar todos os recursos que existam num ambiente de forma eficiente. Tal implica 3 passos fundamentais:

**Modelação** – Há problemas de procura (p. ex., procurar o melhor caminho entre duas cidades, missionários e canibais, etc.) em que conhecemos o estado inicial e o final, e em que pretendemos encontrar uma sequência estados (operadores de mudança de estado) que nos leve do estado inicial ao final. Temos outro tipo de problemas, por exemplo o das N Rainhas, onde apenas se pretende saber qual a melhor configuração possível. Nestes problemas o estado inicial é arbitrário, o estado final é desconhecido, e a solução é apenas um estado (ex. no caso das N rainhas, a configuração do tabuleiro de xadrez). Optou-se por modelar o problema aqui apresentado seguindo esta segunda abordagem, ou seja um estado é uma sequência ordenada de recursos (ex. [12, 10, 1, 2, 6, 9, 5, 7, 8, 3, 11, 4]) que indica uma ordem de recolha. O estado inicial, p.

ex. [12, 8, 10, 1, 9, 2, 6, 11, 4, 3, 5, 7]) é arbitrário, e a forma como se transitou do estado inicial para o final é irrelevante, não fazendo parte da solução.

Tendo em conta esta opção de modelação do problema, importa encontrar soluções adequadas a uma série de questões: Como criar um estado inicial? Quais são os operadores de mudança de estado? Quando deve terminar a procura? Que tipo de funções de escalonamento de temperatura são aplicáveis? Como parametrizar os algoritmos?

**Implementação** – Deve implementar os algoritmos de pesquisa, **Trepa Colinas** e **Recristalização Simulada**;

**Experimentação** – Deve conduzir um conjunto de testes empíricos comparando os algoritmos e as soluções que estes permitem obter.;

O presente trabalho prático encontra-se dividido em 2 metas distintas:

1. Meta 1 – Pela Encosta Abaixo
2. Meta Final – Recristalizar e Simular

### 3.1 Meta 1 – Pela Encosta Abaixo

O “Unity Package” disponibilizado já inclui a implementação do método de pesquisa **Aleatória**. Deve começar por analisar a implementação fornecida, e testar a mesma em diferentes ambientes, de modo a compreender as opções de modelação e implementação tomadas. Note que deverá criar um novo projecto em **3D**, visto que o código fornecido está criado nesse modo. Ao executar o projecto fornecido, em modo interactivo, deverá carregar na **tecla Espaço** para iniciar a procura.

Uma vez ultrapassada esta fase de aprendizagem inicial deve implementar o método **Trepa Colinas** (Alg. 1) no ficheiro correspondente. Deve testar e validar as respectivas implementações.

Deve testar a implementação nos diferentes ambientes fornecidos.

```

1 CurrentSolution  $\leftarrow$  GenerateRandomSolution(ProblemSize);
2 CurrentSolutionCost  $\leftarrow$  Evaluate(CurrentSolution) ;
3 while CurrentNumberOfIterations < MaxNumberOfIterations do
4   /* Não se gera a vizinhança toda, apenas um vizinho
   aleatoriamente */;
5   newSolution  $\leftarrow$  GenerateNeighbourSolution(CurrentSolution);
6   newSolutionCost  $\leftarrow$  Evaluate(newSolution) ;
7   /* Minimização */;
8   if newSolutionCost  $\leq$  SolutionCost then
9     CurrentSolution  $\leftarrow$  newSolution;
10    CurrentSolutionCost  $\leftarrow$  newSolutionCost ;
11  end
12 end
13 return CurrentSolution

```

**Algoritmo 1:** Pseudocódigo do Algoritmo Trepas Colinas. Note que assume que estamos a lidar com um problema de minimização, i.e., quanto menor for o custo da solução, melhor.

## 3.2 Meta Final – Recrystalizar e Simular

### 3.2.1 Recrystalizar

Nesta última meta deve implementar o algoritmo de Recrystalização Simulada (Alg. 2). Deverá ter em especial atenção aos parâmetros a utilizar (e.g, temperatura inicial, função de escalonamento) e comparar vários valores dos mesmos.

```
1  /* Tipicamente um valor alto */;
2  Temperature ← InitialTemperatureValue ;
3  CurrentSolution ← GenerateRandomSolution(ProblemSize);
4  CurrentSolutionCost ← Evaluate(CurrentSolution) ;
5  while CurrentNumberOfIterations < MaxNumberOfIterations and
    Temperature > 0 do
6    newSolution ← GenerateNeighbourSolution(CurrentSolution);
7    newSolutionCost ← Evaluate(newSolution) ;
8    /* notem que (currentSolutionCost - newSolutionCost) é negativo
       sempre que a nova solução é pior que a actual */;
9    probability ←  $e^{((currentSolutionCost - newSolutionCost)/Temperature)}$  ;
10   /* Minimização */;
11   /* Muda de estado sempre que o custo é inferior */ ;
12   /* Se o custo for superior pode mudar probabilisticamente */;
13   if newSolutionCost ≤ CurrentSolutionCost or
     probability > Random([0, 1]) then
14     CurrentSolution ← newSolution ;
15     CurrentSolutionCost ← newSolutionCost ;
16   end
17   /* Escalonamento da Temperatura */;
18   Temperature ← TemperatureSchedule(Temperature) ;
19 end
20 return CurrentSolution
```

**Algoritmo 2:** Pseudocódigo do Algoritmo de Recrystalização Simulada. Note que assume que estamos a lidar com um problema de minimização, i.e., quanto menor for o custo da solução, melhor.

Deverá analisar a performance deste algoritmo, e fazer um comparação com os restantes, explicando devidamente no relatório quais os seus pontos fortes e quais as suas fraquezas ilustrando o seu funcionamento com exemplos.

### 3.2.2 Simular

A etapa de experimentação e análise empírica dos resultados constitui uma componente **fundamental** deste trabalho prático que não pode ser descuidada. Nesta análise deve considerar todos os algoritmos disponibilizados e implementados (i.e. deve incluir a pesquisa aleatória) e todas as variantes dos mesmos (i.e. deve conduzir experiências utilizando todas os métodos desenvolvidos).

Note que os algoritmos têm uma componente estocástica, i.e. a sua performance depende de alguns factores aleatórios (p.ex. geração da solução inicial). Para evitar que a sua análise seja enviesada por questões de “sorte deve efectuar – para cada algoritmo, combinação de parâmetros e ambiente – várias execuções usando diferentes “random seeds”<sup>1</sup>. Idealmente deveriam fazer pelo menos 30 execuções, mas tal pode ser demasiado moroso, como tal sugere-se que façam 5 execuções para cada para cada algoritmo, combinação de parâmetros e ambiente. Devem comparar algoritmos e combinações de parâmetros em função da média das 5 repetições, bem como do melhor e pior valor obtido.

De forma sucinta deve:

- Considerar vários ambientes com diferentes graus de complexidade, i.e., que demonstrem os pontos fortes e fracos de cada um dos algoritmos;
- Comparar os diferentes algoritmos entre si, analisando a qualidade das soluções encontradas bem como o número de iterações necessário para chegar a essas soluções;
- Analisar e comparar a escalabilidade dos algoritmos, ou seja analisar a forma como a sua performance se degrada à medida que a complexidade dos ambientes aumenta;
- Analisar a robustez dos algoritmos em relação à parametrização e as condições iniciais;

---

<sup>1</sup>Para o efeito devem alterar o valor do parâmetro `randomSeed`. Se não sabe o que é uma random seed, veja, por exemplo: <https://www.statisticshowto.com/random-seed-definition/>

## 4 Datas e Modo de Entrega

### 4.1 Meta 1 – Pela Encosta Abaixo

#### Material a entregar:

- O código desenvolvido até ao momento, devidamente comentado;
- Um breve documento (max. 3 páginas), em formato pdf, com a seguinte informação:
  - Identificação dos elementos do grupo (Nomes, Números de Estudante, e-mails, Turma(s) Prática(s))
  - Informação pertinente relativamente a esta meta, objectivos alcançados e dificuldades.

#### Modo de Entrega:

Entrega electrónica através do Inforestudante. **Data Limite: 12 de Abril de 2020**

### 4.2 Meta Final – Recrystalizar e Simular

#### Material a entregar:

- O código desenvolvido, devidamente comentado, para cada uma das metas;
- Um relatório (max. 15 páginas), em formato pdf, com a seguinte informação:
  - Identificação dos elementos do grupo (Nomes, Números de Estudante, e-mails, Turma(s) Prática(s))
  - Informação pertinente relativamente à globalidade do trabalho realizado

Num trabalho desta natureza o relatório assume um papel importante. Deve ter o cuidado de descrever detalhadamente todas as funcionalidades implementadas, dando particular destaque aos problemas e soluções encontradas. Deve ser fácil ao leitor compreender o que foi feito e ter por isso capacidade de adaptar/modificar o código.

Para cada agente desenvolvido, deve descrever o comportamento esperado e a forma como esse comportamento foi alcançado.



A **experimentação** é uma parte essencial do desenvolvimento de aplicações de IA. Assim, deve descrever detalhadamente as experiências realizadas, analisar os resultados, extrair conclusões e efectuar alterações (caso se justifique) em função dos resultados experimentais, por forma a optimizar o desempenho dos seus agentes.

O relatório deve conter informação relevante tanto da perspectiva do utilizador como do programador. Não deve ultrapassar as 15 páginas, formato A4. Todas as opções tomadas deverão ser devidamente justificadas e explicadas. Poderá incluir em anexo as experiências realizadas.

### **Modo de Entrega:**

Entrega electrónica através do Inforestudante. **Data Limite: 26 de Abril de 2020.**

## **5 Bibliografia**

- **Inteligência Artificial: Fundamentos e Aplicações**  
*Ernesto Costa, Anabela Simões*