



FACULDADE DE
CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE DE
COIMBRA

Departamento de Engenharia Informática

Fundamentos de Inteligência Artificial
Introdução à Inteligência Artificial
2021/2022 - 2º Semestre

Trabalho Prático N^o2:
The Slow & The Calm:
Darwin's Edition

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

Fascinado pela forma como a Natureza resolve os problemas que encontra, o Homem usa-a frequentemente como fonte de inspiração para desenvolver soluções computacionais para problemas difíceis. É neste sentido que surgem os Algoritmos Evolucionários (AE), que são modelos computacionais inspirados na Teoria da Selecção Natural de Charles Darwin, e nos princípios de herança genética descobertos por Gregor Mendel. Estes métodos tem sido aplicados com sucesso em diversas áreas, desde optimização até ao desenho automático de antenas para satélites.

O objectivo deste trabalho passa por desenvolver as componentes de um AE para desenhar um veículo motorizado. Para isso, iremos recorrer a um simulador virtual¹, que permite avaliar o desempenho do veículo em vários cenários, nomeadamente deslocar-se em percursos com buracos, e em percursos com subidas.

O principal objectivo é que veículo seja capaz de chegar o mais longe possível no percurso onde é inserido. A Figura 1 mostra um exemplo do ambiente ambiente de simulação.

2 Enunciado

Este trabalho prático tem como objectivo principal a aquisição de competências relacionadas com a análise, desenvolvimento, implementação e teste de agentes adaptativos.

Assim, pretende-se desenvolver uma Algoritmo Evolucionário (AE) que permite o desenho de veículos para completar um conjunto de percursos. O AE irá evoluir um conjunto de parâmetros relacionados com a estrutura do veículos (e.g., número de rodas, posição das rodas, raio, etc.) de forma a melhorar a sua performance ao longo do tempo.

Os nossos veículos são compostos por:

1. 8 vectores com um tamanho máximo de 64m, com um ângulo que é calculado a partir do centro do carro;
2. De 0 e 15 rodas:
 - (a) Com um raio entre 0 e 3;
 - (b) Posição da roda no carro;

¹O simulador é baseado na Framework GeneticSharp desenvolvida por Diego Giacomelli

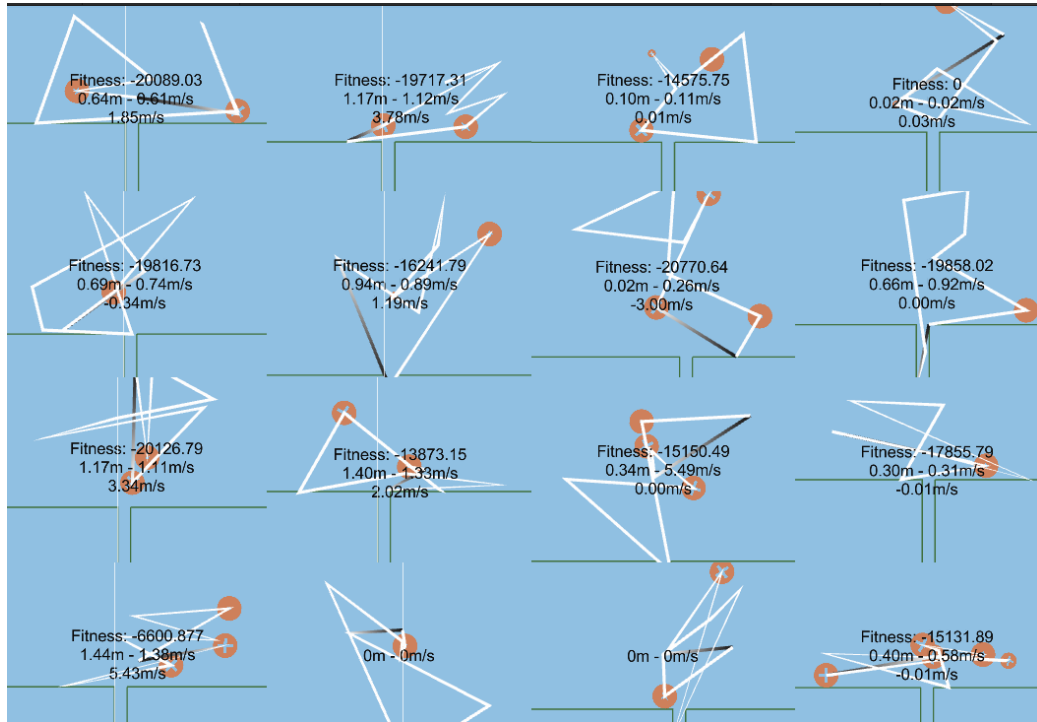


Figura 1: Exemplo do ambiente de simulação.

Tanto as rodas como os vectores tem uma massa associada, pelo que o seu tamanho tem influencia no peso do carro.

Para representar o veículo acima descrito no nosso AE, vamos utilizar um cromossoma binário com a seguinte estrutura:

1. Um número real que corresponde ao tamanho de cada vector representado utilizando 7 bits;
2. Um número real que corresponde ao ângulo de cada vector representado por 9 bits;
3. Um número inteiro que corresponde ao índice do vector onde deverá ser colocada a roda representado por 7 bits;
4. Um número real que corresponde ao raio de cada roda representado por 4 bits;

No total, necessitamos de 28 bits para descrever a estrutura acima descrita. Como o nosso veículo é composto por 8 vectores, temos de a repetir 8 vezes, o que resulta num cromossoma com tamanho 216. Um exemplo do

cromossoma é apresentado abaixo, bem como o fenótipo associado, i.e., o veículo (Fig. 2).

Listing 1: Genótipo

```
0,1,1,0,0,0,0,0,0,0,0,0,0,0,1,1,0,1,1,0,0,1,0,1,1,1,0,0,
0,0,0,0,1,1,1,1,0,0,0,0,0,0,1,1,1,0,0,0,0,1,0,0,1,0,0,
1,0,1,0,0,1,0,1,1,0,1,0,0,0,1,0,0,1,0,0,0,1,0,0,1,0,
1,0,1,0,0,1,0,0,1,0,0,1,1,0,0,1,0,0,1,1,0,1,1,0,1,1,
0,1,0,1,1,1,1,0,1,1,1,0,0,1,1,1,1,0,0,1,0,0,1,1,1,1,
1,0,0,0,0,1,1,1,0,0,0,1,0,0,0,1,0,0,1,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,1,1,0,0,1,0,1,0,1,1,0,1,1,1,0,0,1,
1,0,1,0,0,1,0,0,0,0,0,1,0,1,1,0,1,1,0,0,0,1,0,0,0,0,
0,1,0,0,0,1,1,0
```

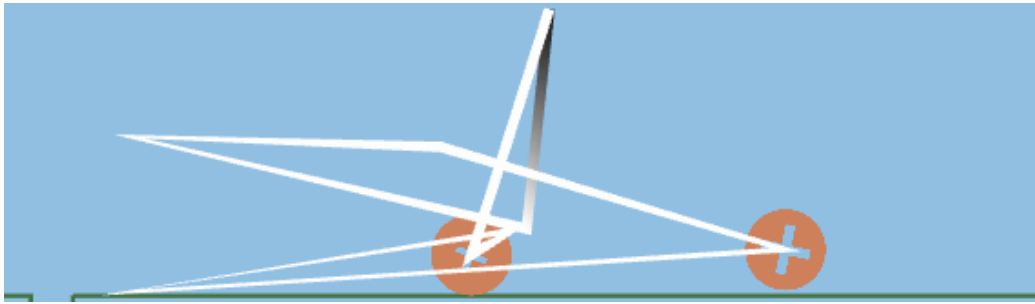


Figura 2: Fenótipo

De forma a permitir avaliar a qualidade de cada veículo são retiradas várias informações do ambiente de simulação, nomeadamente:

- Distância máxima percorrida;
- Velocidade Máxima atingida;
- Número de rodas;
- Massa do carro;
- Conseguiu chegar ao fim do percurso.

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

1. Meta 1 – Modelação e desenvolvimento do Algoritmo Genético.
2. Meta 2 – Experimentação e análise.

2.1 Meta 1 – Modelação e desenvolvimento do Algoritmo Genético

A representação escolhida, os operadores genéticos, os mecanismos de selecção, a atribuição de aptidão (*fitness*), são componentes essenciais para o bom funcionamento de um algoritmo genético. Desta forma, a etapa de **modelação** desempenha um papel fundamental no sucesso do seu algoritmo. Relativamente à representação de cada veículo, i.e., solução, o código fornecido considera que cada cromossoma é uma lista de 0 e 1's. No entanto, terá de desenvolver as restantes funcionalidades básicas do Algoritmo Genético, alterando os ficheiros que se encontram na pasta `EvolvingCars/Assets/EvolvingCars/TP2/`:

Recombinação Deve desenvolver o operador de recombinação de **1-ponto**, descrito pelo Algoritmo 1. Para isso deverá editar o ficheiro `Meta1/SinglePointCrossover.cs`;

Mutação Desenvolver o operador de mutação de *bit-flip* descrito pelo Algoritmo 2. Para isso deverá editar o ficheiro `Meta1/SinglePointMutation.cs`;

Seleccção de Pais Deve implementar o mecanismo de **selecção por torneio**, descrito pelo Algoritmo 3. **Lembre-se que é um problema de maximização**. Para isso deverá editar o ficheiro `Meta1/Tournament.cs`;

Seleccção de Sobreviventes Deve implementar o mecanismo de **Elitismo**, descrito pelo Algoritmo 4. Para isso deverá editar o ficheiro `Meta1/Elitism.cs`;

Parameterização Os parâmetros do algoritmo evolucionário – p.ex. especificar a probabilidade de mutação por gene, probabilidade de recombinação, tamanho do torneio, deverão ser configurados e alterados no ficheiro `GeneticAlgorithmConfigurations.cs`;

Aptidão A aptidão de um indivíduo está relacionada com a capacidade dos veículos conseguirem chegar o mais longe possível nos cenários onde são inseridos. É por isso um **componente essencial** ao sucesso do algoritmo, e encontra-se programada no ficheiro `CarFitness.cs`. Devem ser exploradas várias funções de aptidão, tendo em conta o conjunto de informações do ambiente. Para conseguir desenvolver a sua função de aptidão terá ao dispôr, em cada simulação, as seguintes informações:

1. `MaxDistance` - Distância máxima percorrida pelo veículo;

2. MaxDistanceTime - Tempo (em segundos) que o veículo demorou a percorrer a distância máxima;
3. MaxVelocity - Velocidade máxima atingida pelo veículo;
4. NumberOfWheels - Número de rodas que foram utilizadas pelo veículo;
5. CarMass - Peso do veículo;
6. IsRoadComplete - Indica se o veículo completou o cenário, isto é, se chegou ao fim da estrada. Tem o valor 1 se completar o cenário, e 0 caso contrário.

```

1 Function SinglePointCrossover(parents):
2   i  $\leftarrow$  0 ;
3   if RandomizationProvider.Current.GetDouble()  $\leq$ 
      crossoverProbability then
4     cutPoint = RandomizationProvider.Current.GetInt(1,
      parent1.Length)
5     for i < cutPoint do
6       offspring1.ReplaceGene(i, parent2.GetGene(i))
7       offspring2.ReplaceGene(i, parent1.GetGene(i))
8       i  $\leftarrow$  i + 1
9     end
10  end

```

Algoritmo 1: Pseudocódigo do Algoritmo de Single Point Crossover. As funções RandomizationProvider.Current.GetDouble e RandomizationProvider.Current.GetInt já se encontram implementadas, e devem ser chamadas de forma como se encontra no pseudocódigo.

```

1 Function SinglePointMutation(chromosome, probability):
2   i  $\leftarrow$  0
3   for i < chromosome.Length do
4     if RandomizationProvider.Current.GetDouble()  $\leq$  probability
5       then
6         geneValue = chromosome.GetGene(i).Value
7         if geneValue == 1 then
8           | chromosome.ReplaceGene(i, new Gene(0))
9         else
10          | chromosome.ReplaceGene(i, new Gene(1))
11        end
12      i  $\leftarrow$  i + 1
13    end

```

Algoritmo 2: Pseudocódigo do Algoritmo de Single Point Mutation. A função *RandomizationProvider.Current.GetDouble* já se encontra implementada, e devem ser chamada de forma como se encontra no pseudocódigo.

```

1 Function TournamentSelection(number, generation):
2   parents  $\leftarrow$  []
3   i  $\leftarrow$  0
4   for i < number do
5     randomIndexes =
      RandomizationProvider.Current.GetUniqueInts(Size, 0,
      population.Count);
6     winner  $\leftarrow$  null
7     winnerFitness = -1
8     for k < Size do
9       /* Maximização */
10      if (winner == null) or population[k].Fitness >
      winnerFitness then
11        winner  $\leftarrow$  population[k]
12        winnerFitness  $\leftarrow$  population[k].Fitness
13      end
14      k  $\leftarrow$  k + 1
15    end
16    parents.Add(winner)
17    i  $\leftarrow$  i + 1
18  end

```

Algoritmo 3: Pseudocódigo do Algoritmo de Selecção por Torneio. Este algoritmo permite seleccionar o melhor indivíduo num torneio de *Size* indivíduos, e adicionar à lista de progenitores. A função `RandomizationProvider.Current.GetUniqueInts` já se encontra implementada, e devem ser chamada de forma como se encontra no pseudocódigo.

```

1 Function Elitism(population, offspring, parents):
2   i  $\leftarrow$  0
3   for i < eliteSize do
4     offspring[i]  $\leftarrow$  old_population[i]
5     i  $\leftarrow$  i + 1
6   end

```

Algoritmo 4: Pseudocódigo do Algoritmo de Selecção de Sobreviventes com Elitismo.

	Mutação	Elitismo	Torneio	Crossover	Número Gerações
Experiência 1	0.05	0	5	0.9	30
Experiência 2	0.2				
Experiência 3	0.05	2	5	0.9	30
Experiência 4	0.2				
Experiência 5	0.05		2		

Tabela 1: Experiências a realizar no cenário **GapRoad**. Note que, para cada experiência deverá realizar 3 execuções da mesma.

Após implementadas, é vital **testar** as funcionalidades do algoritmo evolucionário por forma a garantir o seu bom funcionamento. Para isso faça uso do cenário **GapRoad**. Aqui deverá conseguir desenvolver uma função de fitness que permita ao algoritmo evolucionário encontrar um veículo capaz de chegar ao fim da estrada.

2.2 Meta 2 – Experimentação e análise

Nesta meta deve utilizar a aplicação desenvolvida para encontrar uma solução para os ambientes disponibilizados. Deve utilizar o código desenvolvido na meta anterior para evoluir os veículos. Os veículos podem ser evoluídos usando diferentes funções de aptidão adoptando assim comportamentos distintos. Deve criar diversas funções de aptidão utilizando a informação disponibilizada pelo ambiente, e tentando otimizar o veículo tendo em conta diferentes parâmetros. Durante a evolução, o genótipo dos melhores veículos é guardado na pasta do projeto. **Deve usar os cenários “Evaluation” para validar e testar os veículos evoluídos.**

Dada a natureza estocástica das abordagens evolucionárias não é possível tirar conclusões a partir de uma única execução. Para cada combinação de parâmetros deverá realizar, pelo menos, **3** repetições da experiência para que a comparação efectuada tenha significado estatístico. **Desta forma, é importante reservar o tempo adequado para esta meta.**

Deve conduzir um conjunto alargado de experiências no cenário GapRoad considerando, de forma sistemática, diferentes combinações de parâmetros, nomeadamente as combinações apresentadas nas Tabela 1.

Não basta enumerar resultados experimentais, deve fazer uma análise dos mesmos procurando explicar as diferenças encontradas e os comportamentos apresentados. Para o auxiliar nesta tarefa, o simulador guarda um conjunto de informações sobre o processo evolucionário na pasta com o nome Results que se encontra dentro da pasta projecto Unity. Em concreto, tem acesso a

2 ficheiros:

- **EvolutionLog.csv** - Guarda, a cada geração, as seguintes informações:
 - **Generation**: Número da geração;
 - **BestFitness**: Qualidade do melhor indivíduo;
 - **AverageFitnessPopulation**: Qualidade média da população;
 - **BestMaxDistance**: Distância máxima percorrida pelo melhor indivíduo;
 - **BestMaxDistanceTime**: Tempo que o melhor indivíduo demorou a percorrer a distância máxima;
 - **BestNumberOfWheels**: Número de rodas do melhor indivíduo;
 - **BestCarMass**: Massa do melhor indivíduo;
 - **BestIsRoadComplete**: Indica se o melhor indivíduo conseguiu, ou não, chegar ao fim da estrada.
- **OverallBestGenotype.txt** - Genótipo do melhor indivíduo. Este ficheiro poderá ser carregado nos cenários **EvaluationGap** ou **EvaluationHill** de forma a verificar o comportamento do indivíduo.

Após a avaliação das componentes do algoritmo genético deverá escolher a(s) melhor(es) combinações e aplicar as mesmas no cenário **HillRoad**, de forma a encontrar um veículo que o permita resolver.

3 Datas e Modo de Entrega

Os grupos têm uma dimensão máxima de 3 alunos. A defesa é obrigatória, bem como a presença de todos os elementos do grupo na mesma.

A entrega da meta 1 é opcional, chama-se no entanto a atenção dos alunos para a importância de concluir atempadamente esta meta. Para efeitos de nota apenas será considerada a entrega final e a defesa.

3.1 Meta 1 – Modelação e desenvolvimento

Material a entregar:

- Scripts onde implementaram e/ou alteraram código, que deve estar 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

Modo de Entrega:

Entrega electrónica através do Inforestudante.

Data Limite: 01 de Maio de 2022

3.2 Meta 2 – Experimentação e análise

Tal como indicado anteriormente, esta entrega será a única que tem um impacto directo na nota. O relatório deve conter informação relativa a **todo** o trabalho realizado. Ou seja, o trabalho realizado no âmbito das metas 1 e 2 deve ser **inteiramente descrito**, por forma a possibilitar a avaliação.

Material a entregar:

- Scripts onde implementaram e/ou alteraram código, que deve estar devidamente comentado.
- Um relatório (max. 20 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.

Conforme pode depreender do enunciado, **experimentação** e **análise** são parte fundamental deste trabalho prático. Assim, deve descrever de forma sucinta mas detalhada as experiências realizadas, os resultados obtidos, analisar os resultados e extrair conclusões.

O relatório deve conter informação relevante tanto da perspectiva do utilizador como do programador. Não deve ultrapassar as 20 páginas, formato A4. Todas as opções tomadas deverão ser devidamente justificadas e explicadas.

Modo de Entrega:

Entrega electrónica através do Inforestudante.

Data Limite: 22 de Maio de 2022

4 Bibliografia

- **Inteligência Artificial: Fundamentos e Aplicações**
Ernesto Costa, Anabela Simões
- **Artificial Intelligence: A Modern Approach**
Stuart Russel, Peter Norvig

Nuno Lourenço, Tiago Baptista, João Correia, Sérgio Rebelo e Pedro Silva – 2021/2022

Checklist

Nesta secção fornece-se uma breve checklist que visa minimizar as probabilidades de lacunas graves no trabalho e relatório. Importa no entanto salientar que esta checklist **não substitui** a validação das opções tomadas, que deverá ser efectuada preferencialmente durante as aulas Práticas Laboratoriais, **nem garante** a obtenção de uma classificação final positiva.

- Implementação:
 - Implementou operador(es) de recombinação?
 - Implementou operador(es) de mutação?
 - Implementou operador(es) de selecção de progenitores?
 - Implementou operador(es) de selecção de sobreviventes?
 - Os operadores de variação preservam a validade dos indivíduos?
 - Implementou o mecanismo de selecção por torneio?
 - Teve em conta o facto de ser um problema de maximização?
 - É possível parametrizar o algoritmo?
- Experimentação:
 - As experiências realizadas têm em conta a natureza estocástica da abordagem (i.e. efectua várias repetições da experiência usando os mesmos parâmetros e seeds aleatórias distintas)?
 - Tendo em conta as opções implementadas, os resultados realizadas permitem indicar:
 - * O melhor individuo para cada problema?
 - * A melhor taxa de mutação?
 - * O impacto do elitismo na performance do algoritmo?
 - * O impacto do tamanho do torneio na performance do algoritmo?
 - Tendo as experiências realizadas consegui evoluir agentes que permitam:
 - Terminar o dois cenários?
 - Qual o o tempo máximo que cada agente demora?
 - Qual o peso do veículo?
 - Qual o número de rodas do veículo?

- As respostas às perguntas anteriores constam do relatório? Estão devidamente justificadas e suportadas em resultados experimentais?
- No relatório, descreveu o que foi feito na meta 1 e meta 2?