Relatório do 2º trabalho prático "Rolling in the Hill Evolutionary Edition"

Fundamentos de Inteligência Artificial

04 de maio de 2023

—

Henrique Costa 2020214120 PL2
uc2020214120@student.uc.pt

Gonçalo Senra 2020213750 PL2
uc2020213750@student.uc.pt

João Coelho 2020235901 PL2
uc2020235901@student.uc.pt

—

Penousal Machado

Nuno Lourenço

Índice

Introdução

A inteligência artificial está cada vez mais presente no nosso dia a dia, sendo um grande exemplo disso a popularidade imediata de serviços como o ***ChatGPT*** e, como uma vasta parte da inteligência artificial é composta por agentes evolutivos iremos, neste relatório, abordar uma simulação de evolução genética que teve como objetivo a criação e evolução de indivíduos capazes de superar vários desafios.

Este relatório irá abordar numa fase inicial a explicação do código que foi necessário implementar, e depois a apresentação, análise e discussão dos resultados obtidos a partir de um grande número de experiências realizadas ao longo de vários meses.

Meta 1 – Modelação de desenvolvimento do algoritmo genético

1.1 – Implementação do pseudocódigo

Numa fase inicial da meta 1, foi implementado o código necessário para a correta execução do programa nos respetivos ficheiros.

No ficheiro ***"Meta1/UniformCrossover.cs"***, implementámos o pseudocódigo necessário de modo a ser possível conceber uma recombinação uniforme através de uma operação de crossover que seleciona aleatoriamente um gene do pai 1 e do pai 2 por cada posição do genótipo do filho, sendo que, para cada posição existe uma probabilidade de 50% do gene selecionado ser de um destes pais. De notar que numa fase inicial do trabalho foram executados alguns testes apenas alterando o ficheiro onde era indicado pelos comentários dos docentes, no entanto durante os testes fomos notando que não havia qualquer semelhança nos veículos de geração para geração, portanto após alguns testes descobrimos que nas linhas 45 e 46, a função ***CreateNew()*** iria criar o ADN do novo veículo a partir do ADN de dois novos veículos criados na hora, no entanto, nós queríamos que o ADN do novo veículo fosse um clone dos dois veículos pais, por isso foi necessário substituir a função ***CreateNew()*** pela ***Clone()***.

De seguida, no ficheiro ***"Meta1/GaussiantMutation.cs"***, implementámos o pseudocódigo necessário de modo a permitir aplicar mutações aleatórias ao genótipo criado pelo ficheiro anteriormente explicado, possibilitando que os filhos obtidos não sejam uma combinação exata dos seus pais.

Após isso, no ficheiro "***Meta1/Roulete.cs***", implementámos o pseudicódigo necessário de modo a permitir selecionar quais os genes serão usados como pais nas próximas gerações, sendo escolhidos do lote de pais, os mais aptos através da função de fitness.

De seguida no ficheiro ***"Meta1/Elitism.cs"***, implementámos o pseudicódigo necessário de modo a permitir transportar das gerações passadas alguns dos melhores casos para preservar as suas características e voltarem a ser testadas.

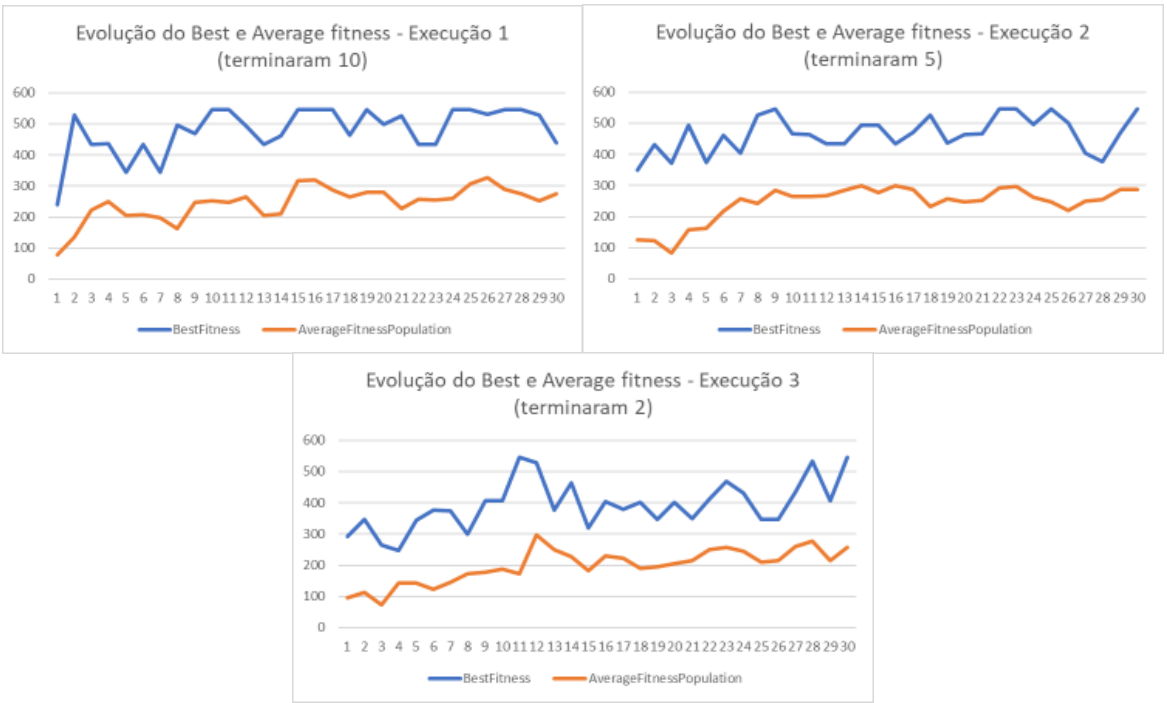
Por fim, o ficheiro ***"GeneticAlgorithmConfigurations.cs"*** permite parametrizar as funções dos ficheiros anteriormente referidos e o ficheiro ***"CarFitness.cs"*** permite definir a função de fitness que será explicada brevemente.

1.2 – Execução de Experiências

Posto isto, após implementarmos todo o código necessário começámos a experimentar diversos valores de mutação, elitismo, crossover e número de gerações, de modo a perceber qual a melhor combinação de valores. De notar que cada combinação foi testada 3 vezes, e que todos os nossos testes foram baseados rigorosamente na **tabela 1 do enunciado do trabalho prático**.

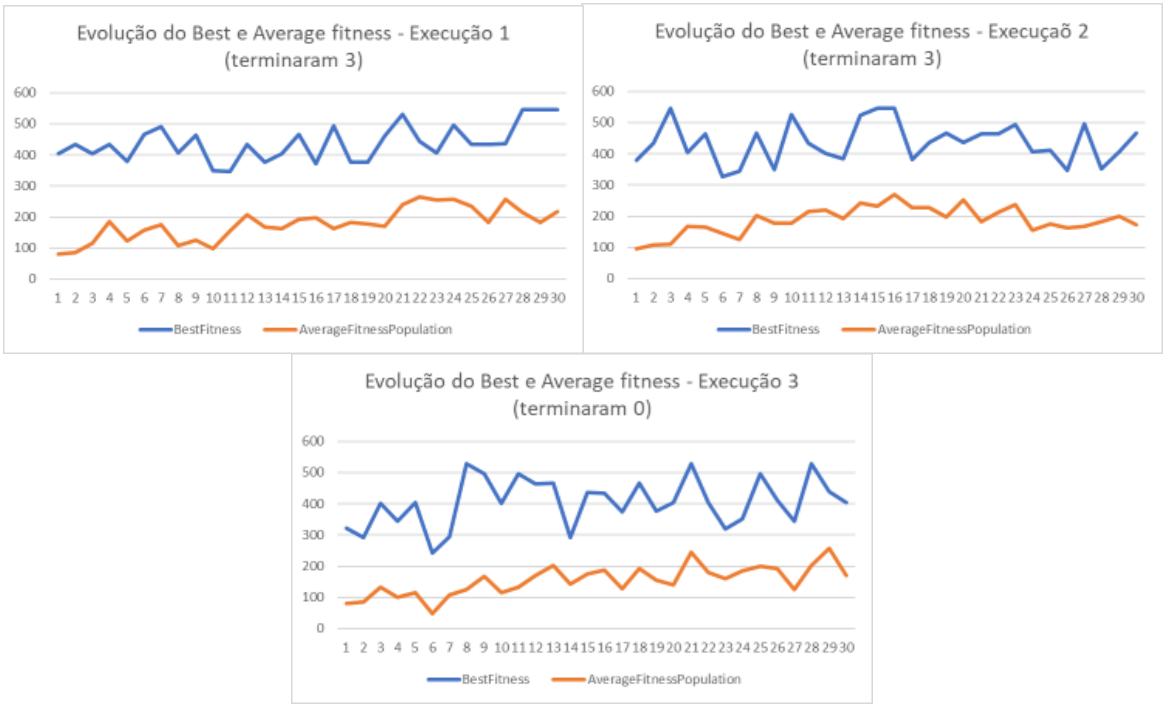
Os resultados obtidos para cada execução de cada experiência encontram-se a seguir:

— Experiência 1 (Mutação - 0.05; Elitismo - 0; Crossover - 0.9; Nº Gerações - 30):



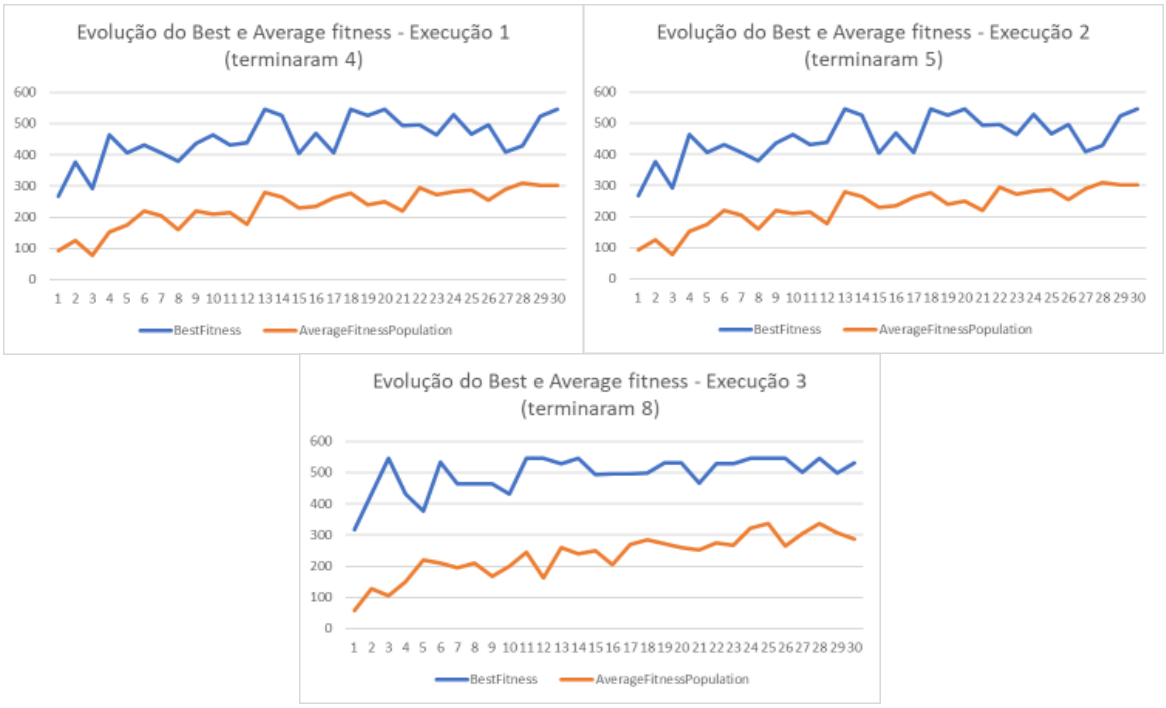
Figuras 1, 2 e 3 – Evolução do Best e Average fitness em 3 execuções; (eixo X - nº de gerações/ eixo Y - fitness)

— Experiência 2 (Mutação - 0.2; Elitismo - 0; Crossover - 0.9; Nº Gerações - 30):



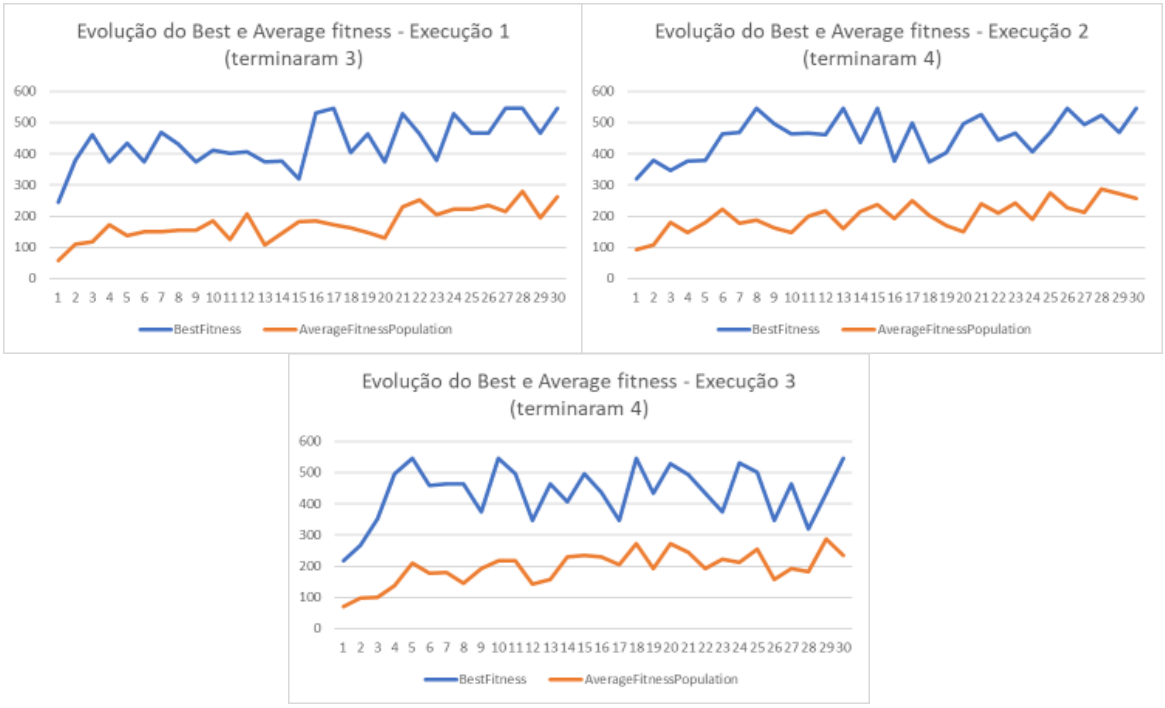
Figuras 4, 5 e 6 – Evolução do Best e Average fitness em 3 execuções; (eixo X - nº de gerações/ eixo Y - fitness)

— Experiência 3 (Mutação - 0.05; Elitismo - 2; Crossover - 0.9; Nº Gerações - 30):



Figuras 7, 8 e 9 – Evolução do Best e Average fitness em 3 execuções; (eixo X - nº de gerações/ eixo Y - fitness)

— Experiência 4 (Mutação - 0.2; Elitismo - 2; Crossover - 0.9; Nº Gerações - 30):



Figuras 10, 11 e 12 – Evolução do Best e Average fitness em 3 execuções; (eixo X - nº de gerações/ eixo Y - fitness)

1.2.1 – Execução de Experiências

Com todas as experiências executadas e resultados apresentados, chegou a altura de analisar os mesmos para escolher a melhor combinação.

A nossa principal preocupação com a realização de todas estas experiências, era conseguir encontrar uma combinação que possibilitasse com que pelo menos um veículo chegasse ao fim do cenário **GapRoad.**

No entanto, mesmo antes de começar a executar testes, o grupo chegou a um consenso que a melhor combinação, de um ponto de vista teórico, seria a experiência 3 (**Mutação - 0.05; Elitismo - 2; Crossover - 0.9; Nº Gerações - 30**), pois em termos de **probabilidade de mutação**, uma probabilidade de 20% iria ser demasiado destrutiva, o que foi confirmado durante uma aula teórica onde o docente referiu que tal percentagem iria deformar demasiado os veículos, por outro lado, achámos que o melhor valor do **elitismo** seria 2, pois se o mesmo fosse 0 os melhores veículos da geração atual não iriam ser testados novamente na geração seguinte, o que não faria muito sentido pois nós queremos encontrar os melhores veículos para cada experiência.

Posto isto, ao olharmos para os resultados das experiências, ou seja, **figuras 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11 e 12**, concluímos que a nossa previsão teórica estaria relativamente certa, pois nas experiências com o valor de **probabilidade de mutação de 20%** os gráficos **são bastante irregulares**,

não apresentando uma pequena "subida" como os gráficos com probabilidade de mutação 0.05 sendo que o mesmo acontece quando o valor do **elitismo é 0,** o que pode ser explicado pela perda dos melhores veículos de geração em geração.

Por fim, concluímos que iríamos realizar os testes da meta 2 com os valores: **Mutação - 0.05; Elitismo - 2; Crossover - 0.9; Nº Gerações – 30,** pois assim não aplicamos um valor de mutação extremamente destrutivo de geração em geração e graças ao valor do elitismo, os melhores veículos iram ser testados mais do que uma vez com pequenas alterações, graças ao baixo valor de mutação.

Nota: É importante mencionar que todas as experiências foram realizadas com a função de fitness que veio definida nos ficheiros disponibilizados, ou seja, ***fitness* = *MaxDistance***.

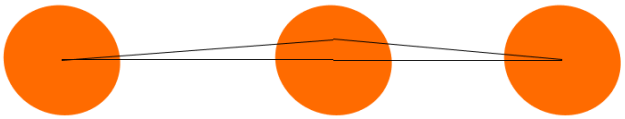
Meta 2 – Experimentação e análise

2.1 – GapRoad

Os veículos têm um cenário composto por buracos que os mesmos têm de ultrapassar tendo como objetivo alcançar a meta de forma consecutiva para que se note uma evolução de geração em geração.

Analisando as características que poderíamos utilizar e as características do cenário, verificamos que seria interessante desde início a "**MaxDistance**" (distância máxima percorrida) porque o que nos interessa efetivamente é que o veículo chegue o mais longe possível, e, também o "**CarMass"** (massa do carro), para que as rodas e as ligações do veículo ficassem maiores, adquirindo assim uma forma mais "achatada", acabando por reduzir a probabilidade de o mesmo cair em buracos.

A incorporação do "**NumberOfWheels**" (número de rodas) foi desde logo descartado visto que, o que pretendíamos era aumentar as rodas, e não a sua quantidade. Por outro lado, e apesar de resultados pouco consistentes, foi estudado o uso da "**MaxVelocity**" (velocidade máxima) de modo que o carro não fosse lento e caísse nos buracos devido a isso.

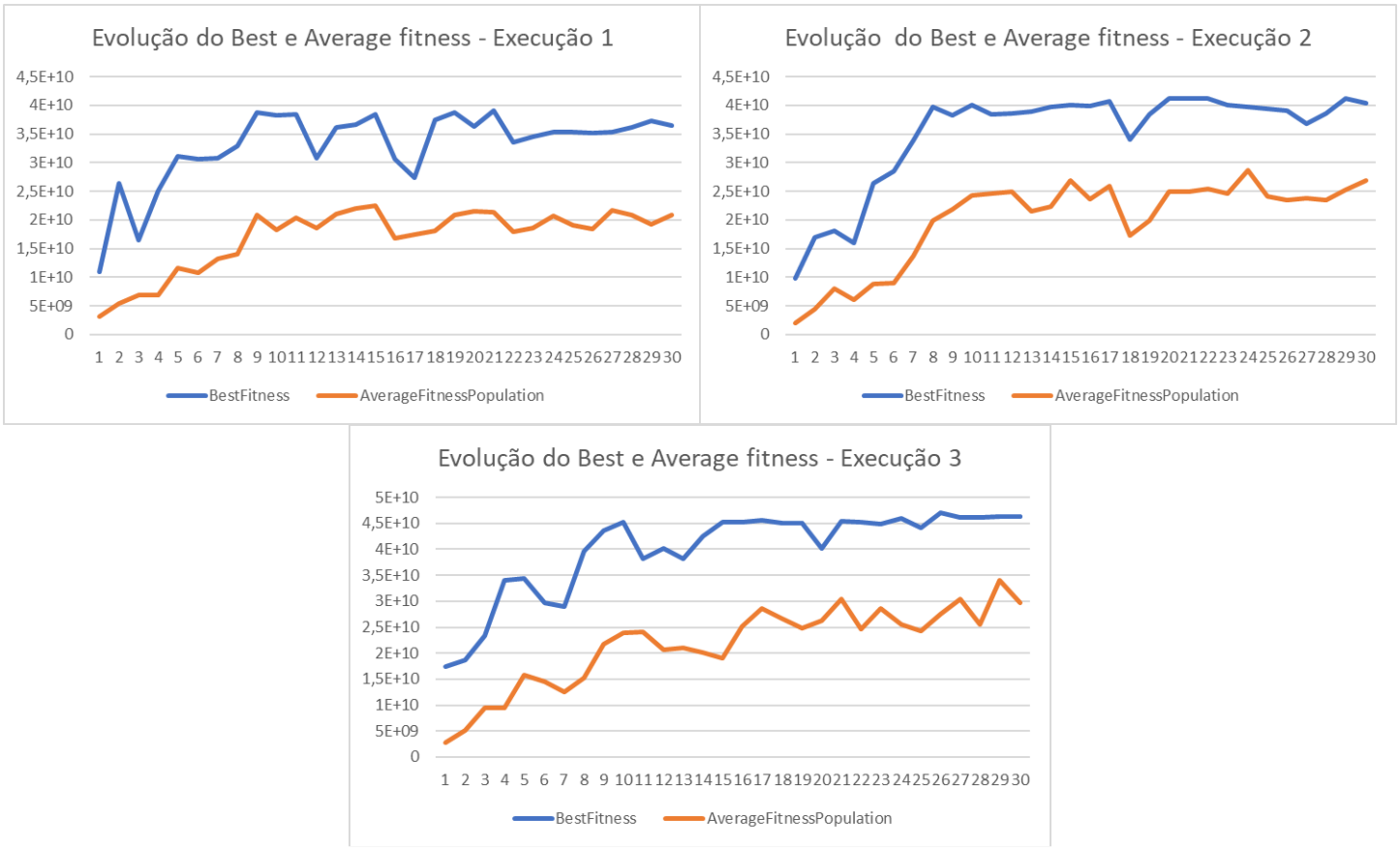


Figuras 13 – Modelo conceptual do melhor veículo;

Deste modo, a função onde obtivemos mais gerações com pelo menos um veículo a acabar, e de forma consistente de geração em geração foi a seguinte:

fitness = (float) (CarMass * CarMass * CarMass * MaxDistance)

Como é percetível pela função de fitness, a mesma encontra-se exponenciada porque, uma vez que utilizamos seleção por roleta, precisamos de criar alguma pressão evolutiva para que os veículos mais aptos tenham um melhor aproveitamento nas seguintes gerações.



Figuras 14, 15 e 16 – Evolução do Best e Average fitness em 3 execuções; (eixo X - nº de gerações/ eixo Y - fitness)

Analisando os resultados obtidos, ou seja, as **figuras 14, 15 e 16**, é possível afirmar que a função de fitness está adequada ao cenário uma vez que pelos gráficos denota-se uma evolução ao longo das gerações e através da **tabela 17**, verifica-se que em bastantes gerações, pelo menos um carro chega ao final do percurso, havendo de geração em geração um número crescente de carros a chegar à meta.

Por outro lado, pela **figura 18**, podemos notar que o melhor genótipo da melhor experiência não ficou completamente igual ao nosso modelo conceptual, tal pode ser explicado pelo facto das experiências estarem a ser executadas apenas com 30 gerações, logo os nossos veículos podem não ter tido gerações suficientes para evoluir para a sua forma mais eficiente.

	Número de gerações com pelo menos 1 veículo a terminar
1ª Execução	17
2ª Execução	18
3ª Execução	14

Tabela 17 – Número de veículos que terminaram por geração;

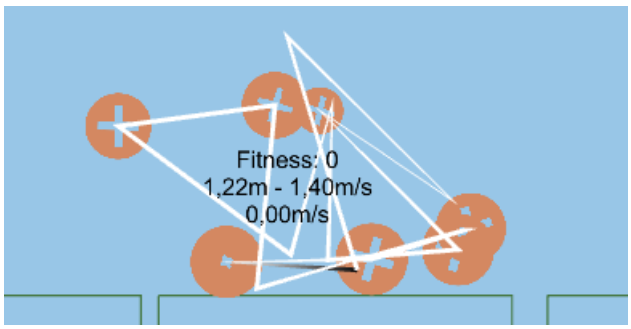


Figura 18 – Melhor veículo da 2ª execução;

2.2 – ObstacleRoad

Neste cenário, os carros foram desafiados a completar um percurso com obstáculos cujo tamanho aumenta ao longo da estrada. Numa fase inicial, deduzimos que para ser capaz de terminar este percurso, o veículo teria de enfrentar os obstáculos com uma velocidade elevada, sendo que o mesmo iria precisar de uma espécie de ancora na parte traseira para o impedir de rolar ao embater nos

obstáculos, pois após visualizar várias experiências notámos que caso o veículo começa-se a rolar seria uma questão de tempo até parar (modelo conceptual do veículo na **figura 19**).

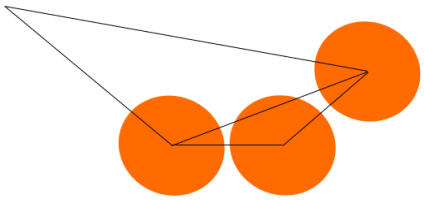


Figura 19 – Modelo conceptual do melhor veículo;

Tendo isto em conta, decidimos que as melhores funções de fitness para testar numa fase inicial seriam uma combinação da **velocidade máxima do veículo** (quanto mais depressa o veículo enfrentar o obstáculo melhor) e da **massa do veículo** (quanto maior a massa, mais achatado o veículo será). Por isso algumas das funções testadas foram:

1. **fitness = (float) (MaxVelocity * MaxVelocity * MaxVelocity)**
2. **fitness = (float) (MaxVelocity * MaxVelocity * MaxDistanceTime * CarMass)**
3. **fitness = (float) (MaxVelocity * MaxVelocity*MaxDistanceTime * MaxDistance*CarMass)**
4. **fitness = (float) (MaxVelocity * MaxVelocity *CarMass* CarMass)**

	Funções de Fitness											
	1.			2.			3.			4.		
Média da MaxDistance em 3 Execuções	98,86764 m			116,1618 m			93,4703 m			81,89547 m		
Desvio Padrão	3,170815 m			58,61074 m			12,3070 m			4,807629 m		
Nº de gerações com veículos a acabar	0	0	0	2	0	0	0	0	0	0	0	0

Figura 20 – Tabela de resultados das diferentes funções de fitness;

Analisando a **tabela 20,** podemos observar que mesmo com as melhores funções de fitness que encontrámos, os resultados não são satisfatórios. Mesmo durante a experiência 2, onde duas gerações conseguiram ter veículos a chegar ao final do percurso, a evolução das linhas do gráfico da **figura 21** mostra que talvez tinha sido apenas sorte. No entanto, se olharmos para o gráfico da **figura 22** já vemos uma evolução gradual do fitness da população, logo se o número de gerações da experiência fosse maior talvez conseguíssemos obter veículos capazes de terminar o percurso.

Para concluir, este percurso foi sem dúvida o que apresentou mais dificuldade na procura de bons veículos, no entanto mesmo com essa dificuldade, foram testadas inúmeras funções de fitness (apenas se encontram no relatório as funções que apresentaram o mínimo de sucesso) e a conclusão a que chegámos foi que a função de fitness mais adequada é **fitness = (float) (MaxVelocity * MaxVelocity *CarMass* CarMass).**

Figura 21 – Evolução do Best e Average fitness;

Figura 22 – Evolução do Best e Average fitness;

(Eixo X – nº de gerações / Eixo Y - fitness)

2.3 – HillRoad

Neste último percurso os nossos veículos depararam-se com uma estrada composta apenas por montanhas e vales, este tipo de percurso ofereceu-lhes uma dificuldade bastante acentuada pois, nem os melhores veículos dos percursos anteriores foram capazes de acabar esta estrada.

Após uma análise do percurso e alguma pesquisa, chegámos à conclusão de que o melhor veículo para este cenário teria um aspeto triangular e seria composto por apenas 3 rodas, uma em cada

ponta e outra no centro, tal como podemos observar na **figura 23**. Desta forma o veículo nunca iria ficar preso em vales ou subidas e a velocidade perdida nos impactos em vales seria mínima.

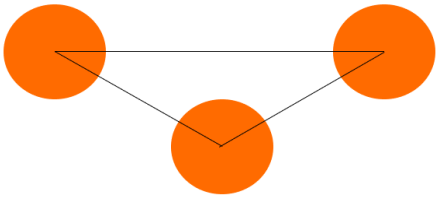


Figura 23 – Modelo conceptual do melhor veículo;

Tendo isto em conta, decidimos que as melhores funções de fitness para testar numa fase inicial seriam uma combinação da **distância máxima percorrida** (quanto mais longe o veículo chegar melhor), da **massa do veículo** (quanto maior a massa, mais achatado o veículo será) e da **velocidade máxima do veículo** (é necessária uma velocidade elevada para ultrapassar as montanhas). Por isso algumas das funções testadas foram:

1. **fitness = (float) (((MaxDistance * MaxDistance * MaxDistance) + (MaxVelocity*MaxVelocity)))**
2. **fitness = (float) (MaxDistance * MaxDistance * MaxDistance + CarMass * CarMass)**
3. **fitness = (float) (MaxDistance * MaxDistance * MaxDistance) * (CarMass * CarMass) * (1 / (NumberOfWheels * NumberOfWheels))**

	Funções de Fitness								
	1.			2.			3.		
Média da MaxDistance em 3 Execuções	1060,221 m			1516,953 m			1023,951 m		
Desvio Padrão	14,737716 m			28,34461 m			27,5639 m		
Gerações com pelo menos um veículo a terminar em cada execução	2	0	0	22	17	18	0	1	0

Tabela 24 – Resultados das diversas funções de fitness;

Figura 25 – Melhor veículo da 1ª execução com a 2ª função de fitness;

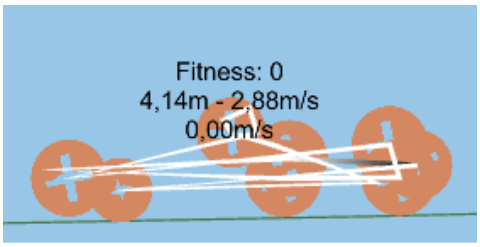


Figura 26 – Melhor veículo da 3ª execução com a 2ª função de fitness;

Analisando a **Tabela 24** e **as figuras 25 e 26**, concluímos que a melhor função de fitness que conseguimos encontrar foi ***fitness = (float) (MaxDistance * MaxDistance * MaxDistance + CarMass * CarMass)***, o que está completamente de acordo com as observações teóricas feitas no início deste capítulo pois, neste cenário o mais importante foi de facto a distância máxima percorrida pelo veículo e a sua massa. Outro ponto bastante positivo é que podemos ver semelhanças entre o modelo conceptual da **figura 23** e os veículos das **figuras 25 e 26**, o que confirma que, os melhores veículos para este cenário necessitam de uma disposição em triângulo com rodas nas pontas.

Para finalizar, podemos também confirmar que a função de fitness escolhida influência de uma maneira bastante positiva a evolução dos nossos veículos, pois tal como podemos ver na **figura 27**, de geração em geração os veículos foram evoluindo de modo a acabar o percurso.

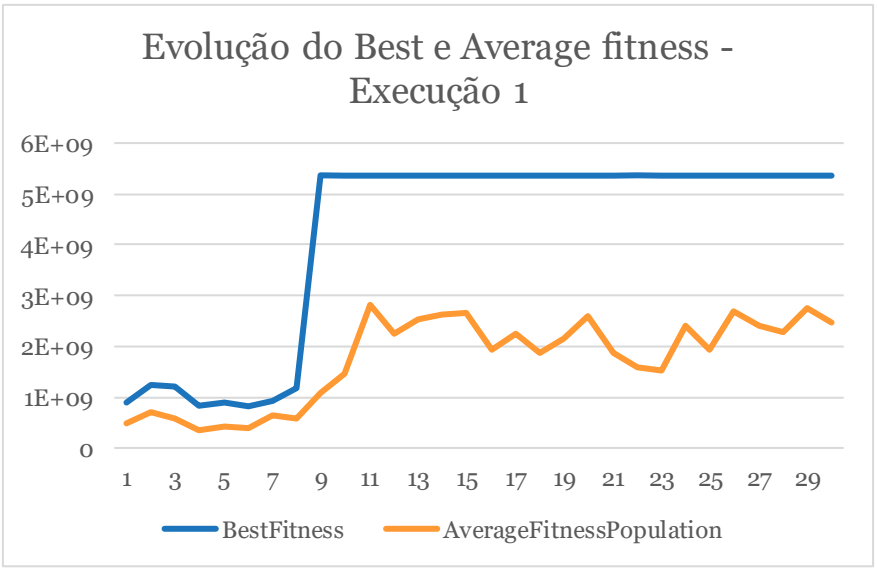


Figura 27 – Evolução do Best e Average fitness dos veículos da primeira execução; (Eixo X – nº de gerações / Eixo Y - fitness)

Conclusão

Achámos o tema deste trabalho prático bastante interessante. Os testes intensos e regulares podem tornar o processo um pouco cansativo, mas quando são encontrados os melhores valores e as melhores funções de fitness, a execução das experiências torna-se extremamente interessante pois conseguimos ver todo o processo de evolução de um "ser" criado a partir de um programa informático.

De notar também, que este trabalho permitiu aprender e consolidar temas teóricos bastante importantes que foram abordados durante o semestre, como a importância de ter uma função de fitness adequada para garantir a evolução correta das gerações.

No entanto, queremos também salientar que as experiências que são necessárias executar no cenário "HillRoad", consomem bastante tempo e recursos computacionais, pois existiram experiências que demoraram mais de 1 hora, deixando o computador onde a experiência estava a ser executada extremamente quente.

Para finalizar, mesmo não tendo conseguido encontrar a melhor função de fitness para todos os cenários, achámos ter atingido a grande maioria dos objetivos propostos no enunciado, acabando consequentemente por aprender bastante sobre processos de evolução, algoritmos evolucionários, a importância de realizar múltiplas experiências com os mesmos valores e funções.

Bibliografia

- Slides fornecidos pelo docente [4/05/2023];
- https://pt.wikipedia.org/wiki/Aptid%C3%A3o [4/05/2023] – Funções de fitness;
- https://www.figma.com/ [04/05/2023] – Modelos conceptuais;