# Artificial Intelligence Fundamentals
# Project 2 - The Slow and the Calm: Darwin's Edition

André Carvalho
andrecarvalho@student.dei.uc.pt
no. 2019216156

Paulo Cortesão
paulocortesao@student.dei.uc.pt
no. 2019216517

Class PL3

March 2022

# Contents

1 Introduction . . . . . . . . 1
2 Implementation . . . . . . . . 1
3 Experimentation . . . . . . . . 3
3.1 *GapRoad* scenario . . . . . . . . 3
3.2 *HillRoad* scenario . . . . . . . . 5
4 Considerations for possible future work . . . . . . . . 10
5 Conclusion . . . . . . . . 11

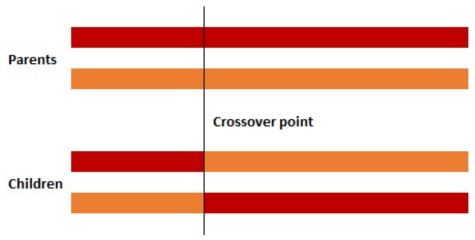
## 1 Introduction

This report aims to describe the work undertaken in the implementation of a genetic algorithm that allows 2-dimensional cars to overcome obstacles in roads with potholes and steep hills. The implementation details will be provided, and the result analysis will be registered here.

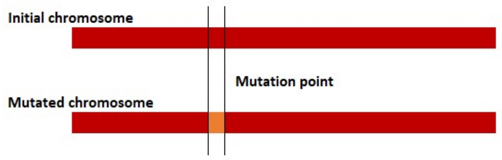
## 2 Implementation

From the statement, it was intended that the students implemented the functions related to the genetic algorithm, in order to assess the impact of their presence on the overall car functionality. Given that the pseudocode provided in the statement was already very complete, the authors adapted it to C# in the respective files, applying only minor changes. The following functions were implemented:

- *Single Point Crossover*: having two cars that constitute parents (parent 1 and 2), the genotype for two children is generated: after the determination of a random crossover point $P$, the genes of the two parents are swapped from that point on, in such a way that one child will have parent 1's genes until $P$ and parent 2's genes after $P$, and the other child will have the remaining genes. This phenomenon occurs with probability $p_{cross}$, allowing certain parents to remain unchanged between generations.



- *Single Point Mutation*: given the genotype representing a given car as a sequence of bits, a mutation is obtained by flipping one of the bits in a random position. This impacts the expression of these bits into features and will generate a car different from the initial one. A mutation will only happen with probability $p$, an algorithm parameter that allows control over the changeability of the population's genotype.



- *Tournament Selection*: given the genotypes of the population members, and their fitness in the previous generation, the parents of the next generation, which will undergo crossover and mutation, are chosen, one by one, from randomly selected pools of size $k$, in which only the genotype corresponding to the highest fitness function value is selected. This parameter allows for the regulation of selective pressure, as the probability that the genes of the least fit will remain decreases as $k$ increases.
- *Elitism*: given the cars ordered by fitness, the top *eliteSize* ones are chosen to move on to the next generation, without undergoing mutation or crossover. This parameter also allows for a control of selective pressure and genetic variability, as it directly affects the size of the population that can bear non-elite genes.

In order to test the algorithms implemented, the parameters described above were added to the class relative to configurations. As well as this, the number of generations was also used, as it allows for a control of the degree of evolution undergone by the population. Finally, a fitness function was defined, so the cars' genotype could be evaluated and all the features could be tested. Considering that the algorithm ran with no implementation errors in *Unity*, the authors continued their work and started experimenting.

## 3 Experimentation

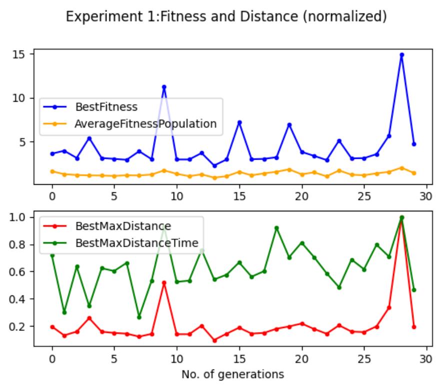
### 3.1 *GapRoad* scenario

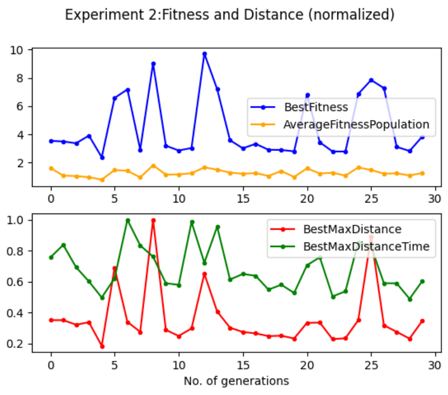
In the beginning, 5 different configurations should be run at least 3 times in a road with potholes, in order to select the best set of parameters, i.e., the one that allows for a fast and continuous rise in the average fitness function. Each one of these experiments took an average of 30 to 40 minutes in the authors' machines. In this phase, the goal was not to find the best fitness function, but the best set of genetic algorithm parameters. Therefore, the fitness function used simply tried to take advantage of the multiple characteristics measured about the cars and somehow normalize them.

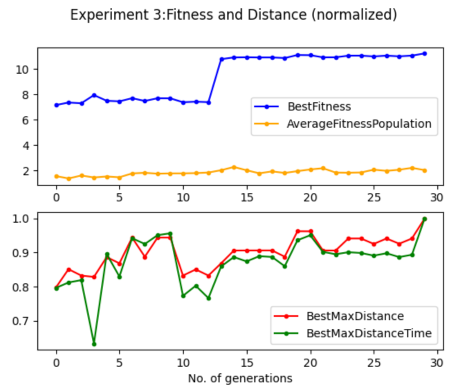
$$\begin{split} Fitness = \frac{MaxDistance}{200} + \frac{NumberOfWheel}{15} + \\ \frac{MaxVelocity}{10} + IsRoadComplete \times 10 \end{split} \tag{1}$$

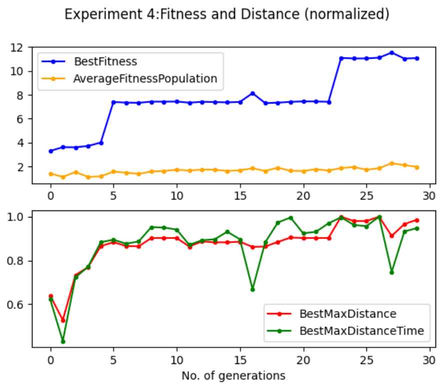
This fitness function allows for a progressive evolution, as it is not based on excessively discrete values (this would happen if the only parameter that mattered to the fitness were if the car had finished the race). Because of this, it is suitable to test the algorithm parameters.

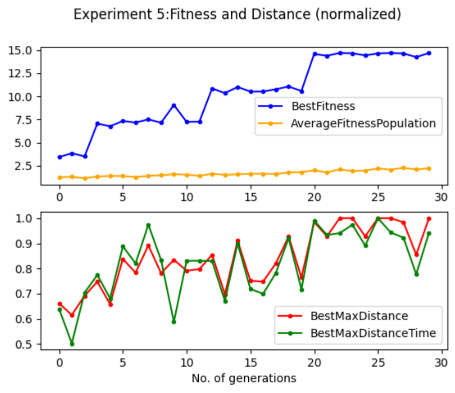
In compliance with the statement, the sets of parameters were put to the proof three times. The average results per generation are the following:











Some metrics were extracted from the data presented here, including the mean and standard deviation of the differences of consecutive values of best maximum distance ($\Delta best$ ):

| $\Delta best$ $(\mu \pm \sigma)$ | Exp 1 | Exp 2 | Exp 3 | Exp 4 | Exp 5 |
|---|---|---|---|---|---|
| | $-0.24 \pm 501$ | $-0.16 \pm 360$ | $4.5 \pm 26$ | $7.4 \pm 34$ | $7.7 \pm 70$ |

Some values in this table appear to be too high for what is physically possible in this experiment (the path length was 656 meters). This is due to the fact that some cars did not comply with the physical rules, as they did not touch the ground and their wheels simply orbited their body, while moving at absurdly high speeds. This allowed them to run distances greater than the total path length and led to this kind of errors.

As can be seen from the graphs, the presence of elitism has a positive impact on the evaluation of the fitness function, as it resembles a monotonically increasing function over time in experiments 3 to 5, as opposed to what occurs in the other experiments.

Also, the tournament with a pool of size 5 is superior in maintaining constancy in the evolution: this is evident when comparing experiments 3 and 4 to experiment 5: the standard deviation of $\Delta best$ is more than twice in 5 than in 3 and 4. This is positive as the outcomes are more predictable, and function evaluation becom

Furthermore, a mutation probability of 0.2 allows for a faster evolution than 0.05. This can be seen by comparing the distance deltas in experiments 4 and 3, where it is evident that the increase in distance is on average two thirds greater in 4 than in 3, even though that leads to a slightly higher standard deviation.

One aspect to note is the fact that only in experiments 3 to 5 was it possible to develop cars that would consistently reach the end of the circuit within the 30 available generations, which was somewhat expected, given the impact of moderate elitism in genetic algorithms.

Despite the multiple successful attempts to complete this scenario with a relatively simple fitness function, it was possible to draw some conclusions on the properties of strong cars in this scenario. These cars would need a very large body, with some part of it to the left and/or right of the two main wheels. This extra weight would help balance the car when the first wheel would enter a gap, as the vehicle would otherwise get stuck in it. This is an example of one such car:

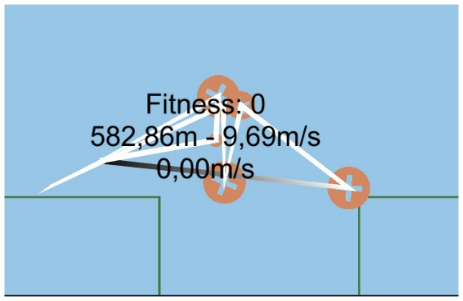


Figure 1: Successful car in gap road

As can be seen from the picture, this car has some weight in its back, which balances it after the front wheel encounters a gap. Despite having its two main wheels inside the gap at the same time, the car's weight distribution compensates for this and allows getting the car back on track.

Considering all the empirically obtained results, it was decided to use the parameters corresponding to experiment 4 for the *HillRoad* scenario, as they allowed for a fast and consistent evolution of the fitness function.

## 3.2 *HillRoad* scenario

At first, the authors tested the new scenario with the fitness function used in the previous one, which proved rather inefficient, as no car would get close to climbing one of the last hills. Apart from that, the bugs reported in the previous experiments persisted, which negatively affected the cars' evolution.

To avoid this issue, it was henceforth decided that the fitness functions would return zero if the maximum distance was greater than that of the circuit. This solution was not perfect, since a bugged car could sometimes have high speeds and not finish the race.

In order to identify what features a successful car would have in this scenario, the genetic algorithm was run using only the maximum distance as fitness, with 500 generations at 10x speed (using Unity's *Time.timeScale*). The first successful car appeared in the $462nd$ generation, and its features were registered: the mass was 240 grams and it had 8 wheels (despite the fact that not all were visible: it is inferred that some were superposed). In another experiment, a 243-gram, 8-wheel car was found after 71 generations. Even though an increased speed was used, the results were validated by using the *EvaluationHill* scenario, in which the generated cars could be tested in normal conditions.

From this, it was inferred that light cars with a high number of wheels (and normally two or 3 sets of superposed wheels) would probably be the fittest to solve the problem. As such, the next step for the *HillRoad* scenario completion was to adapt the fitness function to enhance the most important features a car needs to climb hills. Of course, this approach was limited to the metrics that were already calculated. Being able to assess more features, such as the number of sets of superposed wheels, would have eased the task of devising a suitable fitness function.

Before this conclusion, other fitness functions were tried using only 30 generations, even though did not produce any interesting results. Some of those functions were the following:

- $fitness = \frac{MD}{656} + \frac{MV}{100} + \frac{10}{CM} + \frac{2}{NOW} + IRC \times 10$
- $fitness = \frac{MD}{MDT} + MV + \frac{1}{CM} + \frac{1}{NOW} + IRC \times 1000$
- $fitness = \frac{MD}{MDT \times MV + 1} + \frac{1}{CM} + IRC \times 1000$
- $fitness = \frac{MD}{MDT \times MV + 1} + \frac{MD}{MDT + 1} + IRC \times 1000$
- $fitness = MD - NOW \times 10 - \frac{CM}{50} + \frac{MD}{MDT \times MV + 1} + IRC \times 1000$

Where $MD = MaxDistance$, $MV = MaxVelocity$, $CM = CarMass$, $MDT = MaxDistanceTime$, $NOW = NumberOfWheels$ and $IRC = IsRoadComplete$.

These functions failed to normalize correctly the multiple available attributes, or mixed attributes that did not belong together, which could explain the uninteresting results.

Knowing that car mass was an important parameter, the authors decided to focus their fitness functions on this parameter, as well as the car velocity, as it was believed that this was also a very relevant piece of information in this scenario, due to the hill-climbing involved. Both *MaxVelocity* and *AverageVelocity* (calculated by dividing *MaxDistance* by *MaxDistanceTime*) could be relevant, so these values were also used in the fitness functions.

One of the first fitness functions tried as a result of these conclusions was:

$$fitness = MD + 0.6 \times \frac{MD}{MDT + 1} + 0.4 \times MV - \frac{CM}{100} + IRC \times 1000$$

Apart from using the relevant attributes mentioned above, this function also uses MD, as it shows how far in the road the cars travel. However, the coefficients used were not exactly correct, so the cars still would not finish the circuit.

A new variant of this function, which did not consider $MD$, was also tested, but the results were the same in the 30 generations tried. Then it was decided to try to experiment with more than 30 generations once again: one of the tests ended with a car finishing the circuit in the $81st$ generation, with 133 grams and 8 wheels. However, this car was lost in the process and its genotype was not recovered. The resulting genotype resembled a winning car [2], but needed more speed when climbing the last hill. This was probably the result of a mutation after the first car finished the circuit, as the logs say that it has completed the race at least once.

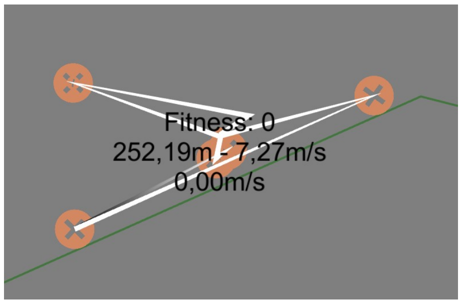


Figure 2: Car extracted from the experiment

A very similar shaped car as [2] was achieved by a different fitness function. This time, in generation number 299, a successful 7-wheeled, 159-gram car was generated using this fitness function:

$$fitness = \frac{MD}{100} + MV - (CM - 200) \times 0.1 + IRC \times 1000$$

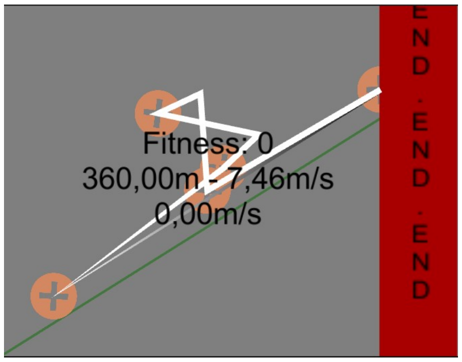


Figure 3: Car extracted from this experiment

This car [3], despite very similar to [2] but somewhat heavier, has a much more favorable weight distribution in the center of the vehicle, allowing it to overcome the last hill and finish the circuit.

After achieving another two winning cars, with 133 and 159 grams respectively, the authors tried to replicate this value for the mass, trying to obtain cars with weight within the interval $[133, 243]g$, considering the successful cars obtained. In order to make this possible, the fitness functions used both absolute values and quadratic terms, such as:

$$|CM - 150| \text{ or } (CM - 240)^2$$

For instance, this fitness function was used:

$$fitness = \frac{MD}{MDT+1} + MV - |CM - 150| + IRC \times 1000$$

Having run this experiment multiple times with no success, it was decided to make some tests with different mutation probabilities, as low as 0.05 and as high as 0.5. After some tries, a 146-gram, 8-wheeled car was found finishing the circuit at the $240th$ generation with mutation probability 0.05. However, the best car changed a little until the last generation (500) and the resulting genotype turned out to be bugged: despite finishing the race, some parts of the car would not touch the floor, making this experiment invalid [4].

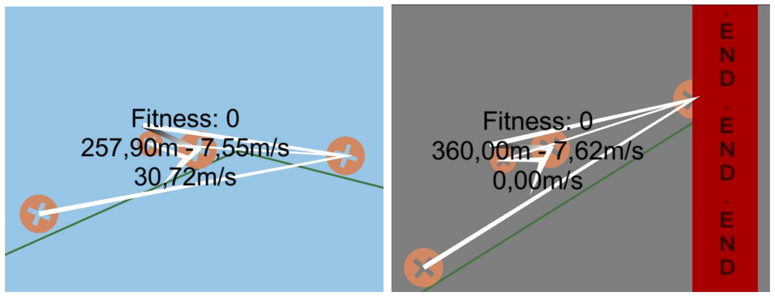


Figure 4: Genotype extracted from the bugged experiment

Using the same 0.05 mutation probability, the following fitness function was tested:

$$f_a = MV \times (MD/360) + IRC \times 1000$$

$$fitness = \begin{cases} f_a, & 0 \leq MD \leq 250 \\ 2 \times f_a, & 250 < MD < 370 \\ 0, & MD \geq 370 \end{cases} \quad (2)$$

This function, despite being very simple and not using the weight of the car, had interesting outcomes in the experiments that tested it. There were multiple

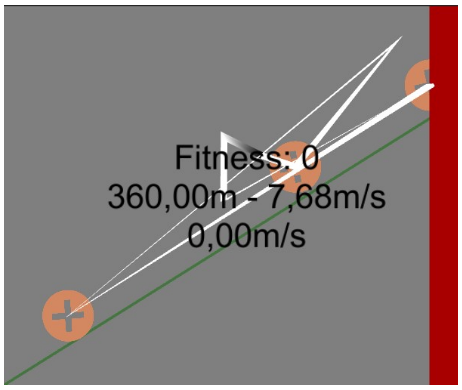


Figure 5: Car that finished *HillRoad* in the 13$^{th}$ generation

cars finishing the scenario in the 500-generation trial, but in one experiment, the first car ending the circuit appeared in the $13th$ generation [5].

Although this was certainly a result of luck, due to the stochastic nature of the experiment, it was believed that this function was quite appropriate for this scenario. Because of that, adjustments were made, adding some dependencies to the mass of the vehicle and changing the multiplier according to MD, as well as the mutation probability (0.3). The result was the following:

$$f_a = MV \times (MD/360) - (CM - 240)^2/1000 + IRC \times 1000$$

$$fitness = \begin{cases} f_a, & 0 \leq MD \leq 245 \\ 2 \times f_a, & 245 < MD < 260 \\ 4 \times f_a, & 260 < MD < 370 \\ 0, & MD \geq 370 \end{cases} \tag{3}$$

During these 3 100-generation experiments, the there were no cars ending the circuit. The maximum distance reached was 280 meters, meaning that a vehicle was successful in surpassing the last hill but did not reach the end goal after that.

Lastly, another fitness function was tried:

$$f_a = MD/5 - (CM - 240)^2/1000 + MV/2$$

$$fitness = \begin{cases} f_a, & 0 \leq MD \leq 250 \\ 2 \times f_a, & 250 < MD < 400 \\ -200, & MD \geq 400 \end{cases} \tag{4}$$

Using these parameters, from three attempts, one successfully produced a car that completed the circuit within 500 generations. One aspect that was noticed was the fact that this function was very quick in leading to cars that would be trapped in the last hill, so it was considered that it was of good quality. In order to produce cars that would complete the circuit earlier, the probability of mutation should increase, so the changes would occur faster and the generation genotype would converge more quickly. The presence of elitism would allow the preservation of the best genotypes, so this combination appeared to be advantageous. As a consequence, this fitness function was tested with a mutation probability of 0.4. However, the results were approximately the same, so this probability was changed to 0.3, and no successful cars were found. Given the amount of conclusions already reached from experimenting with this scenario and the limited time to solve it, the authors decided to end this phase and to summarize what was learned.

From the empirical activity developed, it is inferred that the cars to develop to solve this scenario have a somewhat strict set of features, and that the fitness functions tried are somewhat successful in producing part of those features, as usually, in the first few generations, there are already cars progressing until the last peak (at 250 meters) and becoming stuck there. However, the mutations necessary to make the population have members that complete the circuit do not occur consistently, which means that these functions require further tuning. One phenomenon that was noted in these experiments was the fact that there would be cars with 3 sets of wheels that appeared to be close to the solution, but would be supplanted by lighter cars, with only 2 sets. This points to the necessity of changing the fitness function to treat both types of cars equally.

## 4 Considerations for possible future work

When considering all the experiments performed, including some that are not mentioned in this report, the approximate total time for their execution was 30 hours, divided in an equal way among the authors. As such, given the limited time to devote to this project and the limitations on the way the code could be parameterized, there are some experiments that could lead to interesting results and could be carried out in the future.

Considering the configurations used so far and the current implementation, it would be interesting to adapt simulated annealing to this genetic algorithm: in some generations, there were many cars with good performance, but due to the fact that elite size was constant, they would undergo changes in the subsequent generations. As well as this, in these cases, lowering the probability of mutation would make these populations more stable over time. An algorithm to determine these two parameters as a function of the generation behavior (for example, analyzing the number of vehicles whose fitness function surpasses a given threshold) could have better performance and achieve the desired results faster.

Furthermore, assuming it was possible to define a limited search space for the parameters of the fitness function, applying some kind of search (such as grid search or Bayesian optimization) to that space to optimize the values used would be another way to reach a valid solution to the problem.

## 5 Conclusion

Overall, it is considered that the goals proposed for this project were, in general, achieved. Despite the fact that no fitness function that consistently allowed solving the *HillRoad* scenario was found, this project allowed the authors to understand the mechanism behind genetic algorithms, the influence that several factors, including tournament and elitism, have on their performance; the importance of fitness functions to quickly solve a problem; and the stochastic nature of these methods. The utility of these approaches when solving problems with a very large search space ($2^{216}$ in this case) also became evident, considering that solving these scenarios by applying brute-force algorithms would only produce results after a much longer time.