Neuroevolution – Deliverable 1

Filipe Marçal Dias (r20181050), Inês Santos (20191184), Manuel Marreiros (r20191223)

Code Explanation:

In our *double_tournament* function, defined in the *GeneticOperators.py* file, we start by looking at the *fitness_first* flag, that determines which tournament (fitness or size) will be performed first. If this Boolean variable is set to True, then the fitness tournament will be the first one, and the size tournament the second one. If it is set to False, it will be the other way around. At this stage, we also create a list with the tournament order that will be used later on to pick the correct tournament, and we check for exceptions that may jeopardize the good functioning of the code. More specifically, if fitness is determined to be the first tournament, then we must ensure that the parsimony tournament size is smaller or equal than the number of fitness tournaments, as it wouldn't make sense to select more individuals than the resulting winners of the first tournament for the second tournament. Moreover, if the order is reversed, then we need to ensure that the number of individuals in the fitness tournament is smaller or equal than the parsimony tournament size. If these conditions are not met, then the algorithm presents a warning message and automatically terminates the execution of the program.

Moving on, now that we have the order in which the tournaments will be performed, it is perform them. For that, we first take a look at the previously defined *tournament_order* variable at the position 0. If it matches a fitness tournament, then the function *tournament* is called *fitness_tournament_size* times, each time passing a random argument, the entire population, and the *number_of_individuals*, which corresponds to how many individuals of the population will be participating in the tournament (this variable is independent of whether the first tournament is for fitness or size). The function *tournament* returns the individual with the greatest fitness, with the help of the getFitness() method from the class *Individual*. Contrarily, if the first tournament is the size one, we will instead call the function *size_tournament* the number of times defined by the variable *parsimony_tournament_size*. The function *size_tournament* uses the *getSize()* method defined on the class *Individual* to return the smallest individual. Both these functions are contained in the *GeneticOperators.py* file and they return the fittest/smallest individual from the random individuals that were passed to them.

No matter what tournament was done first, we will end up with as many contestants for the second tournament as the number of tournaments we had in the first stage, since each tournament will only be won by one individual.

Having these contestants established, that is, the winners of the respective 1st tournaments, it is time to move to the second tournament. Here, we must once again understand which type of tournament ought to be done, and it will always be the opposite of the 1st tournament. To ensure that, we check once again the variable *tournament_order*, this time at position 1. Then, we just call the function *tournament* or *size_tournament* one time (unlike before, where a loop was performed) with the random argument, the contestants selected from the first tournament, and either the *parsimony_tournament_size* or the *fitness_tournament_size*, which represent how many of the contestants will actually go to the second tournament. This works as a sort of a filter that picks a certain number of individuals to go to the second tournament from those who won the first one. The tournament will then be performed with these random individuals that won the first tournament and return the final winner based on the criteria that is being applied (size or fitness).