Computational Intelligence Coursework

Conner Weatherston (40167111)

ABSTRACT

The purpose of this report is to implement and evaluate a lunar landing evolutionary algorithm to determine how effective the evolutionary algorithm is at landing space-ship on a platform using a test data set.

This report can be separated into four main sections. In the first section, the purpose of the report is given in detail, this is then followed the terminology that will be used throughout this report.

In the second section, the methodology of the evolutionary algorithm was be described. This will cover the selection, crossover, mutation and replacement operators used within the EA. After that, the testing approached for evaluating the project will be described.

Following on from the methodology, the results obtained from the testing will be covered. This will provide justification on the results and will comment on possible future improvements.

The final section contains the conclusion, this will briefly sum up the overall project and will make final comments regarding its success.

1 INTRODUCTION

This aim of this report is to implement an evolutionary algorithm to evolve the weights of a neural network for the landing of a spacecraft. For the objective to be achieved, the fitness value of the test and training must be close to zero, as that indicates that the shuttles has landed successfully on the platform.

1.1 Background

The background of this report will cover the terms required for the undertaking of this project as well as background reading covered to allow for the project to be planned and created.

1.1.1 Convergence. Convergence is defined as tending to come together, in other words merging. In evolutionary algorithms this is a factor that needs to be accounted for. If the evolution's converge too quickly, then changes in generations will not occur due to crossover, as both parents will be identical. As a result, possible modifications of new children will only come from the mutation stage. This can slow down or possibly stop changes in new generations.

1.1.2 Individual. In this project, an individual can be classified as an permutation of the weights for the neural network which drives the space-ships. The individual also contains a fitness value. This value represents how well it meets the criteria set out in the fitness function which is briefly described in section 2.2.6.

1.1.3 Population. The population of an evolutionary algorithm is defined as the collection of individuals which make up a group of possible permutations. It can be considered the section of the sample space of the problem. In the evolutionary algorithm for this report, the population is a set of individuals which represents a possible permutations as a set of weights for a neural net to land a lunar shuttle. The larger the population the more permutations that are able to be sampled.

1.1.4 Mutation Rate. The mutation rate is a value which determines how common it is for a chromosome segment of an individual to mutate.

1.1.5 Mutation Change. The mutation change is a value which can shift the chromosome segment in a direction. It is common to keep the mutation change a small value, as it causes jumps within the search space. If the mutation change is too large it may possibly skip the optimal or close to optimal result.

1.1.6 Overfitting. When an neural network is being trained on a training set, it is important to ensure that the neural network is not trained to work explicitly on only the training set. If this occurs then it is called overfitting. If the neural network is overfitted then it will not able to adapt to new data sets.

2 METHODOLOGY

In this section, the methodology of the approach taken will be described. This will cover the selection, crossover and replacement operators of the evolutionary algorithm. Furthermore, it will describe the testing approach planned to obtain the results and allow for an evaluation of the project.

2.1 Approach

In this project, a base evolutionary algorithm was supplied to evolve the neural weights. However, the main operators were still be be implemented. The approach taken to finish the evolutionary algorithm was identify methods which would try to reduce convergence from occurring too early during the iterations of evolution.

2.2 Design

2.2.1 Selection. The selection operator determines which two parents are to be selected within a population. Two different selection methods are implemented and evaluated, these are fitness proportionate and tournament. In fitness proportionate selection, each individual in a population to have an opportunity to pass its genes onto the next generation. However, there is a bias towards individuals which have a higher fitness. Tournament selection consists of sampling a random sample of the population and determining which two out of the selection are the best of the sample size. These are selected as the parents for crossover. One concern of tournament is convergence. If the tournament size is too large, there is a greater chance for the best individuals of the entire population being selected each time, which would reduce the diversity in the population.

2.2.2 Crossover. The crossover method is required for the new chromosome structure for the children produced for the new generation. This method determines which chromosomes are selected from each parent. There are two different crossover methods that

have been implemented to determine which is more effective. In both cases, two children are created.

In one point cross over, one section of the first parent is inherited by the child, then the rest of the chromosome comes from the second parent. There can be a bias on towards a particular parent. However, for the purposes of this implementation, the one point crossover will occur roughly in the middle of the chromosome (dependent on chromosome length).

The uniform crossover method goes along the child chromosome and determines at each segment which parent the chromosome segment will inherit from. As this project uses two parents, there is a binary chance at each segment for which parent is inherited from.

Between the two approaches described, the initial assumption is the uniform crossover will perform better in terms of producing the best individual. This is due to the approach allowing for more permutations within the children chromosome compared to the one point crossover. Furthermore, it is more likely for convergence to occur quicker when using the one point crossover, thus relying on the mutation operator to obtain a better fitness.

2.2.3 Mutation. The mutation operator is designed to modify new children that are created each generation to allow for adaptation within the solution. This is done by modifying chromosome segments once certain conditions are achieved.

The mutation operator that is implemented is a conditional shift operator. This moves the chromosome segment value by a set value if the conditions are met. The mutation stage is conditional, so there is a chance that the child will not undergo mutation of their chromosome. This segment is important for obtaining a better solution later on as the population converges. For this project, only the mutation rate and mutation change will be modified.

2.2.4 Replacement. The replacement operation looks at how the new children created in the generation are inserted into the new population. Typically, members in the population are removed to allow for the new children to entire the pool of individuals.

For this evolutionary algorithm implementation, the approach that has been adopted is the replacement of the worst individual. For each child created within the new generation, it is inserted into the population by removing the worst individual. Therefore, with each generation created, the overall population fitness decreases towards zero, meaning they are more adapt to solving the solution.

2.2.5 Parameters. There are four main parameters in the evolutionary algorithm that can be modified. These are presented in the following list:

- gene boundaries - represents the limits the weights can be set too.
- population - number of individuals which make up the sample space.
- mutation rate – how often a chromosome segment can mutate.
- mutation change - how much the chromosome can shift by.

The population, and mutation parameters will be experimented with to determine their impact upon the evolutionary algorithm. There are hidden nodes and gene length which can be modified. However, this report is not investigating the effects of modifying these.

2.2.6 Fitness Function. The fitness function is a method of determining how 'strong' an individual is in a population. A 'strong' individual is one which is close to achieving the fitness value. The fitness value is obtained by calculating the parameters of the problem into a single value.

For this project, an individual can be declared as strong if it is at the landing location, with little speed and has used up all its fuel. As such the following equation is used to determine the fitness:

$$Fitness = (D + V + F)/3$$

where

- D is the normalised distance from landing position.
- V is the normalised velocity.
- F is the normalised amount of remaining fuel.

2.3 Testing Approach

For the evaluating of this evolutionary algorithm, several parameters will be experimented with to determine the impact is has on the final fitness value produced. A successful implementation is one which has been trained on the training data set and can land all eight spaceships successfully on the platform using the test data set.

For each variation of parameters, a total of 10 iterations will be undertaken. This allows for a more reliable result, as there is a random feature to the evolutionary algorithm, an average must be undertaken to allow for a more fairer comparison.

Once each of the main operators (selection, crossover and mutation) have been investigated, a final solution will be created using the results gathered. The final outcome of this will also be investigated.

3 RESULTS

In this section, the results obtained from testing the evolution algorithm will be displayed and discussed. The results can be broken down into four main sections.

In the main first section, an evaluation of the two selection operators will be undertaken,

The second section looks at the differences of using one point or uniform crossover operators. This comparison is carried out against various population sizes to determine if the population size has an effect on the cross over. Both crossover operations followed the same selection and mutation operators to ensure a fair comparison.

In the third section, the mutation operator is investigated. This looks at modifying the mutation rate and mutation change parameters in regards to its effect on the evolutionary algorithm. To keep the experiment fair, only roulette selection and uniform cross over is used during the gathering of these results.

Lastly, the fourth section will look at the final solution produced to solve the lunar landing problem. This will used to results obtained from previous testing to shape the solution.

3.1 Selection

Parameter	Value
Gene boundaries	5
Mutation rate	0.2
Mutation chance	0.6
Population size	200

Table 1: Constant parameters used for tournament testing

Table 1 represents the constant parameters set throughout the testing of the selection methods. This was done to attempt to improve the reliability of the results.

Min	Max	Average
0.02	0.098	0.066

Table 2: Training fitness values from fitness proportionate selection.

For testing the fitness propropionate function, there was not variables that could be modified to determine the impact it had upon the method. This is due to the method relying on a random weighted sum which is decritmented to zero to select the individual. As such, the results obtained from running 10 iterations of the ea has been compiled into tables 2 and 3.

As can be seen in the tables, the training set has an low average of 0.066, however in the test set, the average is higher with a value of 0.127. As the training value is not close to zero, the selection method requires further training for it to have feasible results for the test set. As such, currently, the weights have not been overfitted for the training set.

Min	Max	Average
0.072	0.191	0.127

Table 3: Test fitness values from fitness proportionate selection.

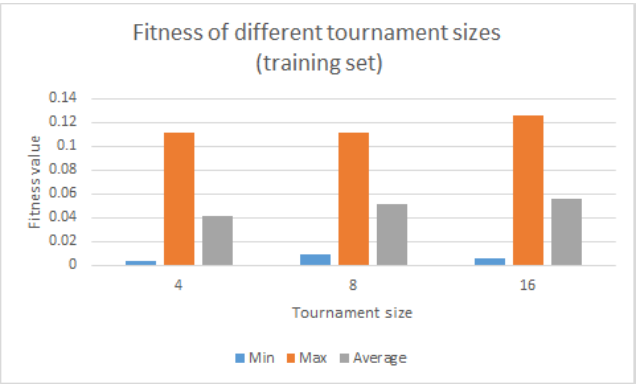

Figure 1: Training fitness values with different tournament sizes

From investing the effect of increasing the tournament size for the tournament selection, it can be seen in figure 1 that the fitness value increases as the tournament size increases. The can be due to convergence occurring quicker with larger tournaments as their is a greater likelihood of the best individuals of the population being selected each time. The average fitness value for a tournament size of four was 0.114. This is slightly lower than the average obtained from the fitness proportionate testing. However for larger tournaments the fitness proportionate method is better for obtaining a lower fitness value on the test data.

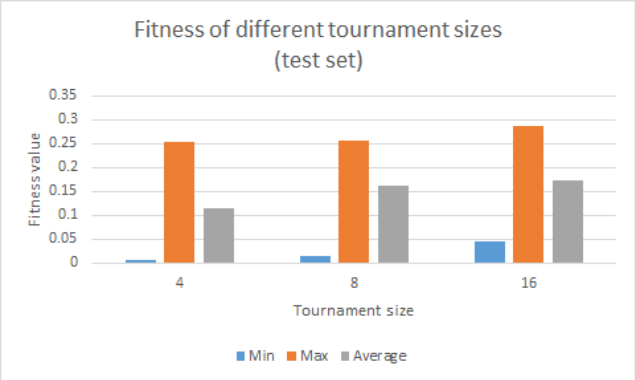

Figure 2: Test fitness values with different tournament sizes

3.2 Crossover

Parameter	Value
Gene boundaries	5
Mutation rate	0.05
Mutation change	0.01
Population size	200

Table 4: Constant parameters used for testing crossover

Figure 3 and 4 displays the minimum, maximum and average fitness values obtained from testing the neural network on the training set.

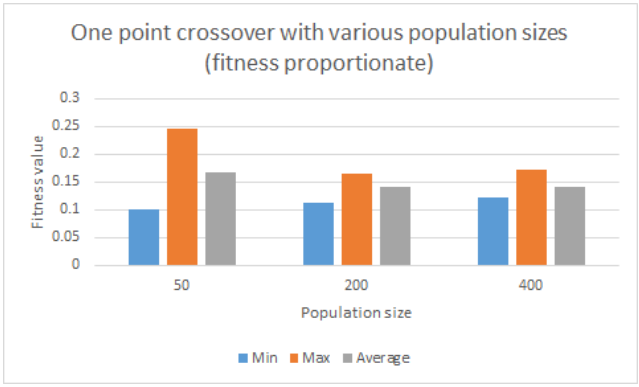

Figure 3: Min, max and average values of one point crossover with different population sizes

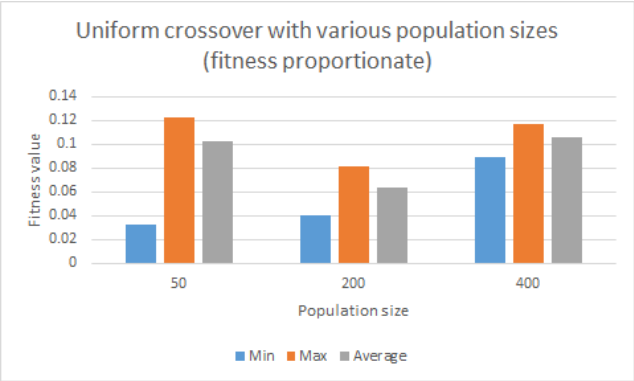

Figure 4: Min, max and average values of uniform crossover with different population sizes

In the one point crossover, the average fitness is significantly larger at lower population sizes compared against the larger population. This occurs due to the the solution converging quickly when there is a low set of individuals to select from. As one point crossover do not allow for much variation in the children, the new children created each generation will be similar to those of its parents, unless mutation occurs.

The uniform crossover works more effectively when there is a moderate population size. If the population is too small, convergence starts to occur, although less than one point cross over. If the individual pool is too large, then the number of potential variations from this technique can hinder the overall progress towards a better fitness value. From the graph seen in figure 4 the ideal population size is around 200 individuals. This allows for the best average fitness value and will ensure the variations in new children are not extreme.

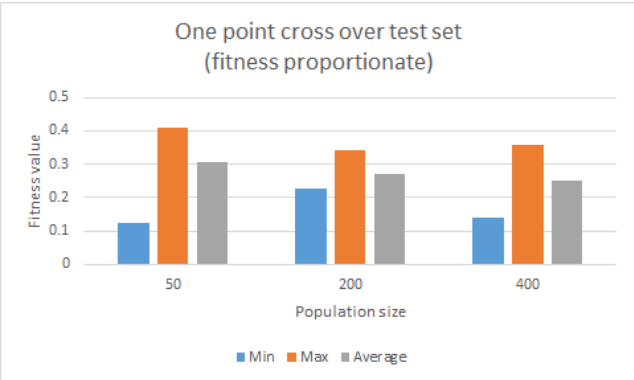

Figure 5: Fitness values of one point crossover on test set

Figure 5 and 6 displays the average fitness values from executing the neural network trained on the training set on the test set. As can be seen both approaches have a higher fitness value than occurred on the training set. This is due to the test set having cases not in the training set which results in the neural network having to make its own decision. However, as the fitness values are relatively high for both training and test sets, it can be seen that the neural network is not yet overfitted.

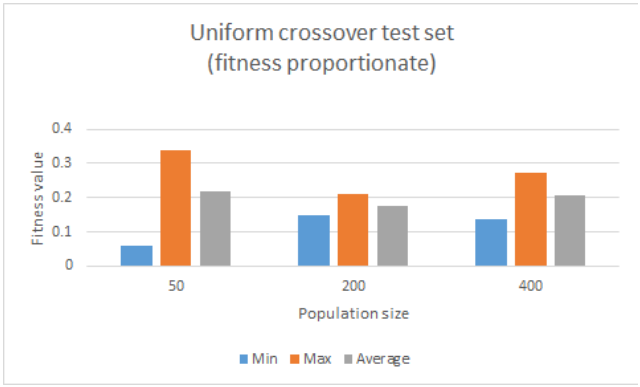

Figure 6: Fitness values of uniform crossover on test set

3.3 Mutation

Parameter	Value
Gene boundaries	5
Population size	200

Table 5: Constant parameters used for tournament testing

The results from the mutation stage of the algorithm is looking at the impact mutation rate and mutation change has upon the algorithm. It is expected that a higher mutation would be better on tournament selection as it would minimise the rate of convergence compared to fitness proportionate selection. For the evaluation the effects on changing the mutation value has only been tested on fitness proportionate selection. It may be worthwhile repeating the test using tournament selection in future work.

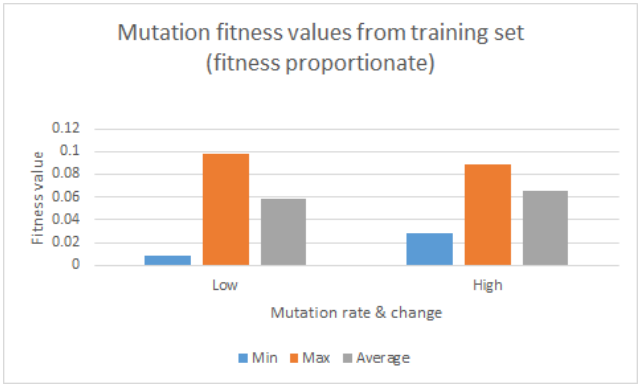

Figure 7: Fitness values on training set

Figure 7 displays the minimum, maximum and average of the effect different mutation rates and mutation changes has on the evolutionary algorithm. The results shown are from the training set.

Low mutation is a mutation rate of 0.01 and the mutation change was a value of 0.05. The high mutation is a mutation rate of 0.1 and a mutation change of 0.5.

The results obtained shows that the average fitness was improved by using a higher mutation factor compared to using a lower mutation factor. This is due to the ability for the solution to jump around in its search space to find a better solution. A drawback with this solution is it will not find the optimal solution. As it is skipping sections of the search space, if the currently permutation is close to the optimal result, the mutation may make it skip the peak of the local optimum, thus making it more difficult for obtain finer values as the evolution's progress.

The gene boundaries are another factor of the algorithm that could be modified. By evaluating the effect of different gene boundaries, there may be a possibility of a consistent improvement by setting them to other bounds.

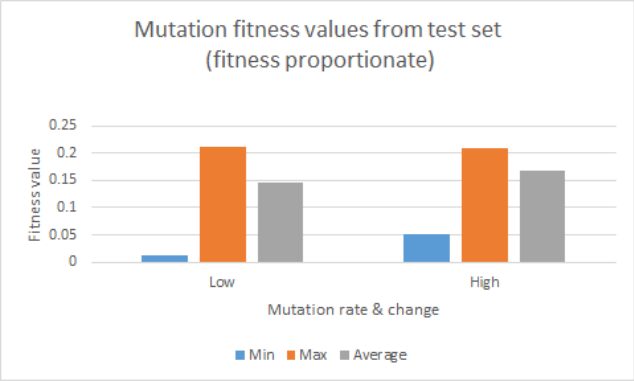

Figure 8: Fitness values on test set

When testing the neural networks with the high and low mutation factors, there is little difference in average fitness for the test set.

3.4 Final solution

From the testing done with different methods and different parameters, a final solution to the evolutionary algorithm was created. This consisted of using tournament selection, uniform crossover and a high mutation rate.

By using this combination, the population will be constantly changing preventing the population from converging too quickly. Once it was implemented, the final solution was tested with the following parameters.

Parameter	Value
Gene boundaries	5
Mutation rate	0.7
Mutation change	0.86
Population size	200
Tournament size	6

Table 6: Constant parameters used for final solution

Using this combination of methods, a final average of the training and test sets was obtained. These are displayed in table 7.

	Min	Max	Average
training			
test			

Table 7: Final solution averages

4 FUTURE WORK

The results from the project displays the current technique tried cannot guarantee a solution that will be suitable for the test test consistently. This could be improved on by using other crossover techniques or look into different approaches to the mutation function, for example Gaussian mutation. Furthermore, a deeper analysis on the results obtained could be useful in furthering identifying the techniques that were more suitable to give more evidence to the claims made in the results section. For example the use of t-tests or more iterations executed.

5 CONCLUSION

To conclude this report, out of the two selection methods available, tournament is better as long as the tournament size is restricted to a low value. If the value is too large, then convergence will occur quicker. For the crossover operator, the ideal method is uniform crossover as it consistently provided lower fitness values compared to one point crossover over a series of population samples. The mutation

Overall, this project was a success as it identified limitations or certain techniques and provided a solution which is almost capable of constantly having strong results in terms of the fitness value on the test set.