Letters and Numbers Assignment

Michael Novak n10272224, Sebastian Young n9952438, Mathew Haywood n10012320

Introduction

The aim of this project was to find the most optimal trade-off between population size and the maximum number of generations to produce the highest success rate, using a genetic algorithm (GA) for the Numbers game from the Australian TV show, Letters and Numbers. The GA randomly "chose" a total of 6 numbers in which it needed to find a solution with the highest fitness while being limited to a computational time of approximately 2 seconds.

Methodology

The approach to finding the trade-off between population size and the maximum number of generations was found by first finding the maximum population that could still be processed within a 2 second budget. A lower and upper limit for population size was selected in which several values between the boundaries were chosen. Population sizes were chosen by semi-periodically increasing the gap between each value.

To calculate the maximum generations for each population size that kept to the 2 second time budget, an experiment function first runs the evolution function using the population size, and a max generation of 1. This value is saved and then used to determine the maximum number of generations that can be used in the given 2 second period. An example of this would be, if a given population took 0.25 seconds for that one generation, it would be able to process 8 generations in the 2 second time budget.

Using the sample of population sizes, a test on each population size was performed over a total of 30 games. This was done by first generating a set of 30 game conditions, so that each population size could be tested fairly. Then for each population, each game was tested, in which the statistics such as the running time, the distance between solution and goal, and if the solution was reached. The statistics were then aggregated to get the average performance of each population size and was then graphed for analysis. This functionality was stored in the*run_experiments()* function and with a run time of approximately 5-6 minutes.

An important note is that it generates 30 random games, so the data in this report will be slightly different to the data that would be generated. However, the overall trend should remain the same.

Results

Using: Q = [50,75,9,10,2,2]

Target = 533

Population Size	Max Generations
7	937
10	815
25	348
50	209
100	122
150	105
200	71
400	39
500	34
750	18
1000	14
1500	8

2000	8
3000	5
4000	4
5000	3
6000	3
7000	2
8000	2
15000	1

Table 1: Population vs Max generations table

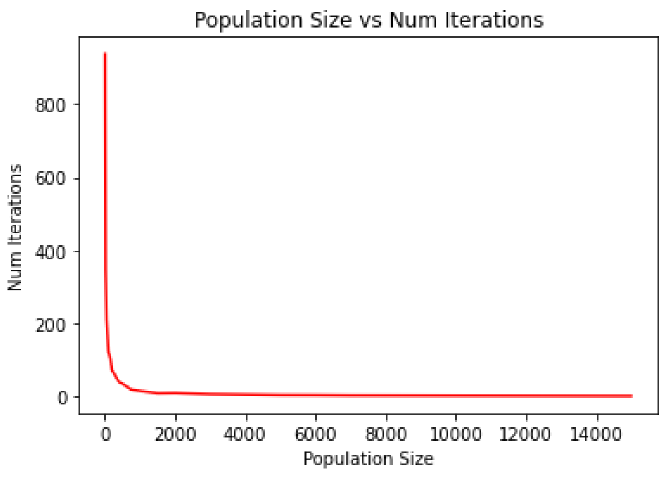

Figure 1: Graph depicting Population Size and Generations

Population Size	Avg Times [s]	Avg Final Cost	Avg Success Rate [%]
7	1.34	60	13
10	1.16	52	6
25	1.35	26	16
50	1.06	23	30
100	1.34	20	33
150	1.15	20	33
200	1.10	19	43
400	1.15	19	40
500	0.903	19	50
750	0.954	19	43
1000	1.04	19	46
1500	0.991	19	56
2000	**0.934**	**18**	**70**
3000	1.10	18	63
4000	0.932	18	63
5000	1.03	18	66
6000	1.20	18	63
7000	1.27	18	63
8000	1.24	19	50
15000	2.00	19	50

Table 2: Population vs Average Times vs Average Final Cost vs Average Succeses Rate

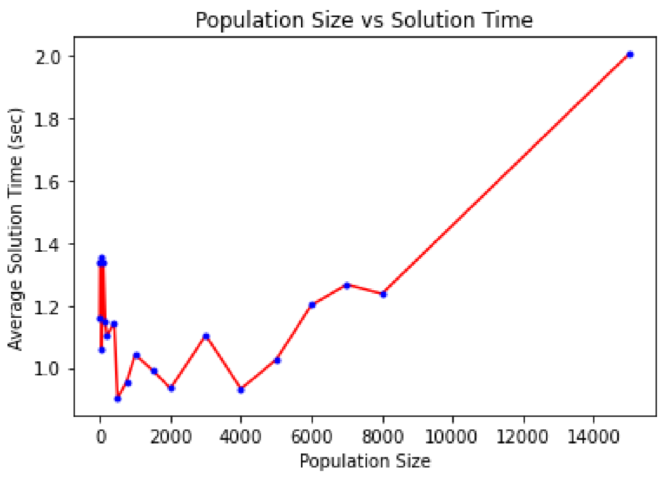

Figure 2: Graph depicting Population Size and Times

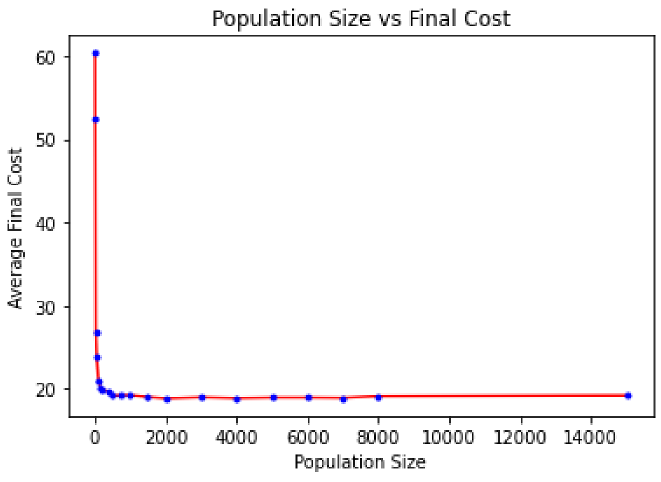

Figure 3: Graph depicting Population Size and Cost

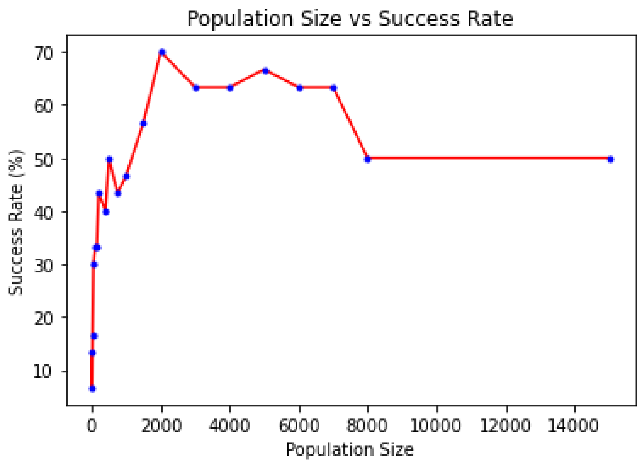

Figure 4: Graph depicting Population Size and Success Rates

Discussion

As per the *Population Size vs Final Cost* graph, a large population can produce more accurate solutions than a small population. Small populations are not able to create a lot of diversity, meaning it is harder to explore new solutions and harder to get more accurate solutions. With large populations, there is a lot of diversity simply due to the large number of solutions undergoing mutation and crossover.

However, having a larger population size does not this does not mean you should just make the populations size as large as possible and be done with it. Large population sizes also take longer to calculate, as more solutions means more crossovers and mutations to calculate, and it takes longer to get from one generation to the next. As seen in the *Population Size Vs Solution Time* graph, there are minimum points around a population of 100, 2000 and 4000 have a lower average time to get to a solution. Before and after these points the time increases towards 2s. This shows that there is an optimal balance between population size and time. As you increase the population size, you decrease time initially, then at a certain point it begins to increase again.

And finally looking at the Population Size vs Success Rate graph. This shows how often it was able to get to the exact solution. The maximum of this graph is at a population size of 2000. It shows a trend of having low success rates at small population sizes, and again low success rates at high populations. The small populations would likely not have enough diversity, and therefore unable to get to the target. The large populations would have enough diversity, but not enough generations to be able to get to the target.

Conclusion

It was shown that as the population size increased the accuracy of the solutions increased. It was also shown that time taken to solve was initially high at a low population, decreased to a minimum around 200-4000, and then increased again as the population increased. Finally, it was seen that the success rates would be low for low populations, but would increase as population increased to 2000, but then decreased after this point. This can be attributed to the smaller populations not having enough diversity, and the larger populations not being able to process enough generations to be able to get to the target within the time budget. The optimal population of 2000 was able to balance the diversity with the time budget, scoring the highest accuracy as well as the quickest time to solution. In the end a population size of ~2000 with a maximum of ~8 generations resulted in the highest success rate, while remaining inside the 2 second time budget.