Thanawat Anansuthivara 730054400, ta541@exeter.ac.uk

ECMM409 Coursework exercise I: Evolutionary Algorithm for Traveling Salesman Problem

A. Introduction

Evolutionary algorithms are computational methods inspired by biological evolution processes, particularly natural selection. Among these, Genetic Algorithms (GAs) have become prominent for their adaptability and proficiency in navigating complex solution spaces. This report focuses on the application of GAs to the Traveling Salesman Problem (TSP), with the objective of exploring the intricacies of these algorithms, their effectiveness, and their constraints in solving optimization problems and identify the impact of the

The primary goal of this report is to investigate the application of the Genetic Algorithm in solving the TSP and to identify the most effective parameters for this task. The study involves:

- Evaluating the performance of the algorithm's parameters by comparing the solution quality against TSP benchmark results.
- Analysing the impact of each parameter/operator on the TSP solution.
- Assessing the algorithm's convergence and quality across various settings.
- Conducting multiple runs to verify the consistency of the algorithm within a stochastic framework.

B. Algorithm implementation for Traveling Salesman Problem

1. Path representation

In this heuristic method, cities are arranged in a sequence from left to right within an array, signifying the visitation order. The array's first element denotes the tour's starting city, with each subsequent number indicating the next city. The final element represents the tour's last city. This sequence is termed a Genome,' where each city is referred to a 'bit'. The phenome is the representation of fitness value, the inverse of the total cost distance. It evaluates the quality of each path. The Chromosome is the representative of Genome and Phenome of each path solution.

Genome						Phenome	
Order	1st	2nd	3rd	4th	5th	Fitness value (1.0/cost distance)	10.0
City	1	4	5	2	3		

Table 1 Path representation or Chromosome of a tour (2,1,4,5,6,3)

2. Crossover operation

2.1 point crossover: Swap the latter part of the parent's Chromosome at a random cut index and insert the less of the city back into the offsprings.
2.2 ordered crossover: Rebuild offspring by choosing a subtour from the parents and swap them and then completed by - remaining bits in relative order.
2.3 partial map crossover: Rebuild offspring by choosing two random cut indices, filling with non-repetitive members from the parents and then completing with the remaining information.

3. Mutation operation.

3.1 Single swap mutation: randomly swap two elements in the chromosome.
3.2 Multiple swap mutation: randomly swap multiple pairs of bits.
3.4 Inversion swap mutation: randomly reversed the order of range of bits.
3.5 Scramble swap mutation: randomly rearrange the order of range of bits.

4. Replacement operation.

The weakest replacement strategy is introduced in this report, where the least fit candidate is replaced by a superior offspring in each generation passed.

Thanawat Anansuthivara 730054400, ta541@exeter.ac.uk

C. Methodology/Experiment

The experiments are designed to understand how different operators and parameters affect the model's performance. The result is gathered by searching one trial of the genetic evolution to capture the pattern how fitness value is improved via iterations and the speed of convergence while collecting the result of multiple trial of the algorithm to conduct a statistic result to ensure the consistency of the model. The visualization is also used to describe the behavior of the model performance where the fitness value is plotted against the generations and the other tells how certain generations generate the results.

1. Operator performance experiment
   The experiment focuses on studying the impact of the mutation and crossover function on the model performance. The baseline combination of numerical parameters is set for max generation = 1000, population size = 500, and tournament size = 400 where the mutate function is implemented with inversion swap for crossover experiment while the mutate experiment uses one point crossover with fixed as a baseline. The result is referred to Appendix A where the tables and illustrations are located.
2. Parameter performance experiment
   The experiment focuses on studying the impact of the different sets of numerical parameters on model performance and how they affect the model’s behavior. The baseline combination of operator functions is fixed with the code name referred to table 1 in Appendix B to assess the quality of the different set of parameters result in each experiment. The result is referred to Appendix B where the tables and illustrations are located.

D. Result and Analysis

1. Mutation experiment.

Burma Experiment on Mutation operator (30trial)						
Mutation function	500th Gen		1000th Gen		Time (second)	
	Mean	Std	Mean	Std	Mean	Std
Single swap	0.000289	0.000015	0.000291	0.000015	5.1320239	0.8959131
Multiple swap	0.000286	0.000007	0.000293	0.000006	4.4347669	0.5052257
Inversion swap	0.000296	0.000007	0.000298	0.000006	5.1685673	0.8491291
Scramble swap	0.000283	0.000015	0.000286	0.000012	5.5394507	0.9925967

Table 1 Mutation Operator Experiment Result on best fitness value it generates on Burma dataset.

Brazil Experiment on Mutation operator (30trial)						
Mutation function	500th Gen		1000th Gen		Time (second)	
	Mean	Std	Mean	Std	Mean	Std
Single swap	0.000019	0.000001	0.000021	0.000020	6.0317365	0.7204101
Multiple swap	0.000017	1.08871e-6	0.000019	1.10217e-6	6.3707278	1.6137274
Inversion swap	0.000024	0.000001	0.000029	9.81072e-7	6.3956671	0.8827789
Scramble swap	0.000014	1.02887e-6	0.000016	1.08672e-6	7.3150642	1.4871458

Table 2 Mutation Operator Experiment Result on best fitness value it generates on Brazil dataset.

The table 1 and table 2 show the quality of the average result from multiple attempts of genetic algorithms performed on Burma and Brazil respectively. The Inversion swap conducts the best average result in both 500th and 1000th generation with the less variation of error. Although its execution time is not the best, the time consumed is not the worst.

Referring to illustration in Appendix A, the inverse swap also reached the convergence faster than any other mutation function with high consistency according to the Figure 7.1, 7.2, 7.3, and 7.4 compared to the other picture in Mutation Experiment section.

2. Crossover experiment

Crossover experiment on Burma dataset (20 trail)						
Crossover function	500th Gen		1000th Gen		Time (second)	
	Mean	Std	Mean	Std	Mean	Std
Point with fixed	0.000291	0.000008	0.000298	0.000005	0.4995975	0.277170
Partial Map	0.000293	0.000007	0.000299	0.000003	0.344865	0.035729

Ordered	0.000296	0.000004	0.000300	0.000003	0.340050	0.028378

Table 3 Crossover Operator Experiment Result on best fitness value it generates on Burma dataset.

Crossover experiment on Brazil dataset (20 trial)						
Crossover function	500th Gen		1000th Gen		Time (second)	
	Mean	Std	Mean	Std	Mean	Std
Point with fixed	0.000017	8.91835e-7	0.000021	1.28734e-6	0.6663590	4.98601e-2
Partial Map	0.000016	8.96991e-7	0.000020	1.51089e-6	0.6999622	3.34106e-2
Ordered	0.000017	9.23710e-7	0.000021	1.18136e-6	0.7001510	4.77472e-2

Table 4 Crossover Operator Experiment Result on best fitness value it generates on Brazil dataset.

The ordered crossover has performed the best result among the crossover operators with the less time consumed on the Burma dataset, but it took the largest time on the Brazil dataset. This can be caused by the logic of the ordered crossover for the different size of the problem where the size of the Burma is 14! while the Brazil is 58! that it must rebuild the whole remaining bits, so it tends to consume more time for a large dataset.

3. Parameter performance analysis

Option	Parameter Experiment on Burma dataset. (30 trials)					
	Fitness value			Time		
(max_gen, pop_size, tour_size)	Best	Mean	Std	Best	Mean	Std
MICO (1000,100,10)	3.00932e-4	2.9850e-4	3.46430e-6	0.37299	0.61886	0.198851
MICO (3000,100,80)	3.00932e-4	3.00698e-4	4.77099e-6	2.22399	2.515334	0.548705
MICO (3000,500,400)	3.00932e-4	3.00259e-4	2.19711e-6	11.55120	12.14114	0.872544
MICO (5000,100,80)	3.00932e-4	3.00777e-4	4.05456e-7	4.106004	4.687186	0.466008
MICO (10000,100,80)	3.00932e-4	3.00932e-4	0.000000	7.508957	9.1870065	1.7504311
MICO (10000,1000,800)	3.00932e-4	3.00932e-4	0.00000	70.38695	84.789599	10.004181

Table 6 Parameter experiment result based on specific set of operators and parameters on Burma dataset.

From the table 6, it appears that the max generation of 1000 generation can achieve the best fitness value for this dataset where the prove can be found at MICO (10000,100,800) where the standard deviation of the best fitness value is zero which means that every member of the population owns the same genome. The average of the best fitness seems to be improved via the increasing of the max generation capacity where the population size and tournament size increase the stability of the standard deviation of the model. Moreover, the greater tournament size is close to the population size would give the better result refers to the MICO (1000,100,10) and (3000,100,80). In terms of execution time, the max generation has a significant relationship to have a longer time to execute while the population and tournament size have a huge impact on runtime.

Option	Parameter Experiment on Brazil dataset. (30 trials)					
	Fitness value			Time		
(max_gen, pop_size, tour_size)	Best	Mean	Std	Best	Mean	Std
MICO (1000,100,10)	2.33034e-5	2.04311e-5	1.13802e-6	0.792953	1.13133	0.198932
MICO (3000,100,80)	3.74686e-5	3.00698e-5	1.20403e-6	3.189970	3.90234	0.51878
MICO (3000,500,400)	3.71996e-5	3.53253e-5	1.19181e-6	9.956036	10.432424	0.495336
MICO (5000,100,80)	3.85981e-5	3.70461e-5	7.32330e-7	5.834999	6.5565046	0.440093
MICO (10000,100,80)	3.92465e-5	3.80009e-5	7.11729e-7	10.89294	12.38091	2.120824
MICO (10000,1000,800)	3.93778e-5	3.77158e-5	9.13498e-7	75.38202	91.72567	10.38447

Table 7 Parameter experiment result based on specific set of operators and parameters on Brazil dataset.

The best fitness value can be observed easily for the Brazil dataset in the table 7. The max generation size significantly has a huge impact on the performance of the model while the population and tournament size also improve the overall performance, based on MICO (3000,100,80) and (3000,500,400). Furthermore, the experiment gives us an insight that the size of the dataset problem has a significant relationship with the max generation, population size, and tournament size which the greater size it is, the greater parameter values must be to perform the best result and display the graph convergence. The ratio of the tournament size and population size also shows a better result when they are close to each other. On the other hand, they also contribute to using more execution time. In term of execution time, the parameter on Brazil dataset has a same behavior as it does on Burma dataset.

Thanawat Anansuthivara 730054400, ta541@exeter.ac.uk

E. Conclusion

In conclusion, this report presents a comprehensive analysis of mutation, crossover, and other parameters in genetic algorithm's performance for traveling salesman problem across Burma, and Brazil dataset. It was found that the size of the problem has a significant relationship with the size of the max generation given in each dataset where the 1000 of generation is completely enough for Burma dataset of which convergence always found in almost experiment while it took more than 5000 generation for Brazil to reach the convergence. Besides, the proportion of the tournament and population also show the significant improvement in the model performance and give them consistency. The experiment also highlights how different mutation and crossover impact on model performance which the inversed mutation and ordered crossover performance is the best combination based on the graph and table in Appendix B.

F. Coursework questions

Question1: Which combination of parameters produces the best results?

Answer: The combination of operation ordered crossover and inversion swap and parameter of max generation = 10,000, population size = 100, tournament size= 80 and 100% chance of occurring a mutation and crossover the best results for Burma and Brazil dataset with cost = 3323.0 and 25480.0 meters respectively.

Question 2: What do you think is the reason for your findings in Question 1?

Answer: refers to the parameter experiment, the best result that I have observed come from this set of combinations where it performs the best fitness and the mean best fitness as shown in table 7 where the best fitness of the Brazil dataset reach the maximum in the entire table and in the table 6, it's entire population has no standard deviation which means it reached the best result.

Question 3: How do each of the parameter settings influence the performance of the algorithm?

Answer: Crossover operator – creating new offsprings by inheriting the information from the previous candidates. It improves the model performance by giving the chance to cross the local peak.
Mutation operator – introduces diversity in creating new offsprings for the model to improve itself.
Max generation capacity – depends on the size of the problem which tends to increase on the bigger problem size. It gives model time to generate the new offspring to improve the entire performance.
Population and tournament size – it controls the pressure selection to increase the chance of selecting the high-quality candidate to generate the better offspring. The closer ratio of tournament size and population is, the result tends to be better for the traveling salesman problem.

Question 4: Can you think of a local heuristic function to add?

Answer: In the genetic algorithm, there are nothing to tell the model to follow the correct path because it is completely randomly generating the new offspring so I believe that there should be a function to define the weight of the correct path where the best route is given the higher weight to track easier.

Question 5: Can you think if any variation for this algorithm to improve your results? Explain your answer.

Answer: I think if there are other functions that can change its effect depending on the model performance would be able to conduct a better result such as dynamic numerical parameters which change align on the model performance and influence the diversity in each experiment.

Question 6: Do you think of any other nature inspired algorithms that might have provided better results?

Answer: Refer from my answer in the question 4, the Ant colony algorithm is the best choice I would answer for this question based on my current knowledge where the qualify path is left with the dense pheromone so the following ant can track these paths easily even though they might get stuck at the local peak, but the function still generate the new solution with hopefully will across this sub peak.

Thanawat Anansuthivara 730054400, ta541@exeter.ac.uk

G. References

[1] L. Yuri and A. Claus, “Experimental Analysis of the Tournament Size on Genetic Algorithms.” In IEEE, 2018. https://www.researchgate.net/publication/330478320_Experimental_Analysis_of_the_Tournament_Size_on_Genetic_Algorithms

[2] Tutorialpoint, “Genetic Algorithms Tutorial”, Tutorialpoint. https://www.tutorialspoint.com/genetic_algorithms/index.htm

[3] A. Otman, A. Jaafar, and T. Chakir, “Analyzing the Performance of Mutation Operators

to Solve the Travelling Salesman Problem”, 2012. https://arxiv.org/ftp/arxiv/papers/1203/1203.3099.pdf

[4] B. Jason, “Clever Algorithms: Nature-Inspired Programming Recipes”, 2012. https://github.com/clever-algorithms/CleverAlgorithms

[5] S Prayudani et al 2020 J. Phys.: Conf. Ser. 1566 012131, “Analysis Effect of Tournament Selection on Genetic Algorithm Performance in Traveling Salesman Problem (TSP)”, 2020. https://iopscience.iop.org/article/10.1088/1742-6596/1566/1/012131/pdf

[6] Wikipedia, “Travelling Salesman Problem”, https://en.wikipedia.org/wiki/Travelling_salesman_problem

[7] Wikipedia, “Genetic Algorithm”, https://en.wikipedia.org/wiki/Genetic_algorithm

https://www.hindawi.com/journals/cin/2017/7430125/

[8] GitHub, emre-kocyigit. Genetic Algorithm TSP. https://github.com/emrekocyigit/genetic_algorithm_tsp/tree/main

[9] Github, clever-algorithms. Clever Algorithm. https://github.com/clever-algorithms/CleverAlgorithms/tree/master/src/algorithms/evolutionary

Thanawat Anansuthivara 730054400, ta541@exeter.ac.uk

H. Appendix

Appendix A: Operator and Parameter Experiment.

The baseline option for the mutate experiment and crossover experiment is defined in the table below:

Baseline Name	Max gen	Pop size	Tour size	Mutate func	Crossover func
Mutate experiment	1000	500	400	Depends	Point with fixed
Crossover experiment	1000	500	400	Inversion	Depends

Table 0 Option's mutate and crossover combination on defined name. for each group experiment.

Formatted: Font: 9 pt, Underline, Complex Script Font: 9 pt

The Mutation experiments.

Burma Experiment on Mutation operator (30trial)						
Mutation function	500th Gen		1000th Gen		Time (second)	
	Mean	Std	Mean	Std	Mean	Std
Single swap	0.000289	0.000015	0.000291	0.000015	5.1320239	0.8959131
Multiple swap	0.000286	0.000007	0.000293	0.000006	4.4347669	0.5052257
Inversion swap	0.000296	0.000007	0.000298	0.000006	5.1685673	0.8491291
Scramble swap	0.000283	0.000015	0.000286	0.000012	5.5394507	0.9925967

Table 1 Mutation Operator Experiment Result on best fitness value it generates on Burma dataset.

Brazil Experiment on Mutation operator (30trial)						
Mutation function	500th Gen		1000th Gen		Time (second)	
	Mean	Std	Mean	Std	Mean	Std
Single swap	0.000019	0.000001	0.000021	0.000020	6.0317365	0.7204101
Multiple swap	0.000017	1.08871e-6	0.000019	1.10217e-6	6.3707278	1.6137274
Inversion swap	0.000024	0.000001	0.000029	9.81072e-7	6.3956671	0.8827789
Scramble swap	0.000014	1.02887e-6	0.000016	1.08672e-6	7.3150642	1.4871458

Table 2 Mutation Operator Experiment Result on best fitness value it generates on Brazil dataset.

Thanawat Anansuthivara 730054400, ta541@exeter.ac.uk

Single swap visualization

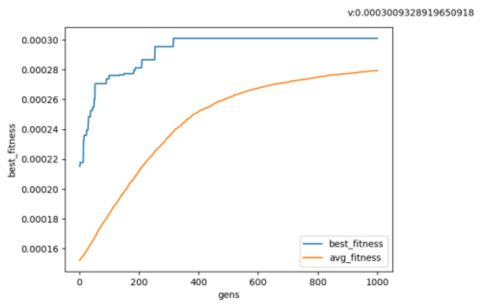

Figure 1.1 Best fitness vs Iterations with Single swap mutation on Burma dataset.

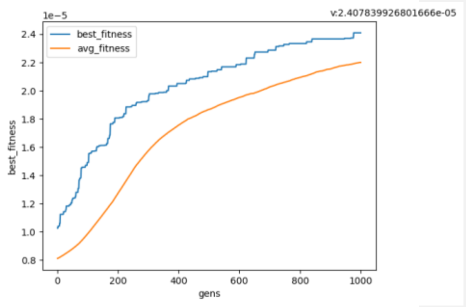

Figure 2.1 Best fitness vs Iterations with Single swap mutation on Brazil dataset.

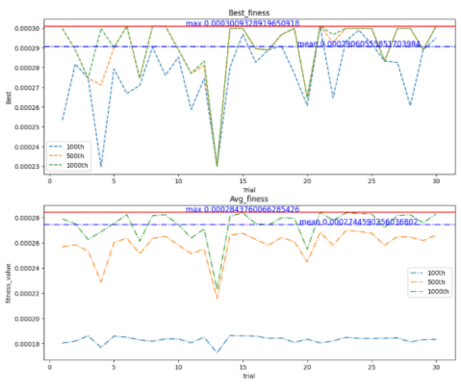

Figure 1.2 fitness value on each trial for 100th, 500th and 100th generation with Single swap mutation on Burma dataset.

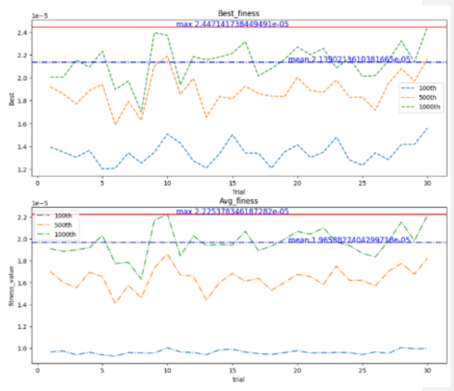

Figure 1.2 fitness value on each trial for 100th, 500th and 100th generation with Single swap mutation on Brazil dataset.

Thanawat Anansuthivara 730054400, ta541@exeter.ac.uk

Multiple swap visualization

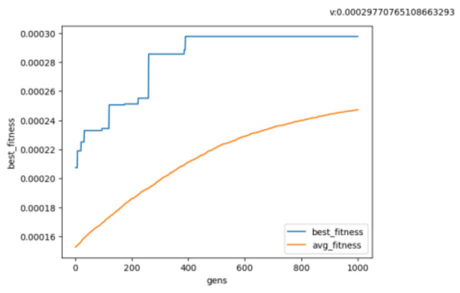

Figure 3.1 Best fitness vs Iterations with Multiple swap mutation on Burma dataset.

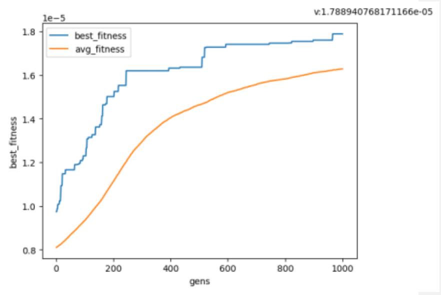

Figure 4.1 Best fitness vs Iterations with Multiple swap mutation on Brazil dataset.

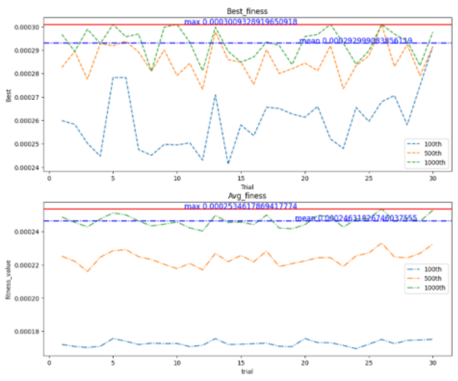

Figure 3.2 fitness value on each trial for 100th, 500th and 100th generation with Multiple swap mutation on Burma dataset.

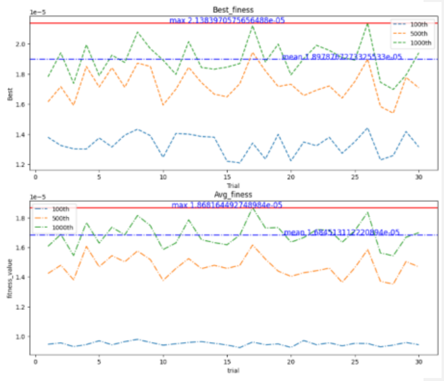

Figure 4.2 fitness value on each trial for 100th, 500th and 100th generation with Multiple swap mutation on Brazil dataset.

Thanawat Anansuthivara 730054400, ta541@exeter.ac.uk

Inversion swap visualization

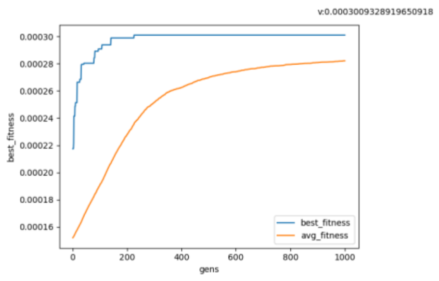

Figure 5.1 Best fitness vs Iterations with Inversion swap mutation on Burma dataset

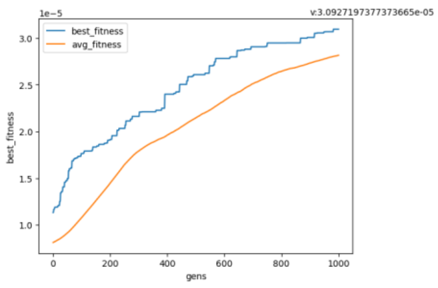

Figure 6.1 Best fitness vs Iterations with Inversion swap mutation on Brazil dataset

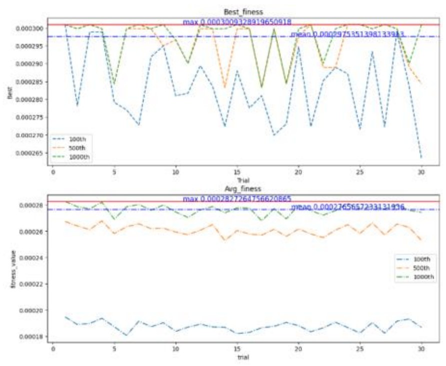

Figure 5.2 fitness value on each trial for 100th, 500th and 100th generation with Inversion swap mutation on Burma dataset.

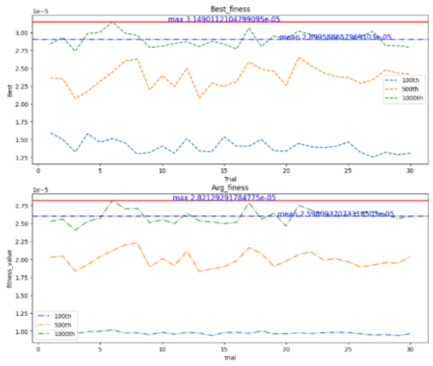

Figure 6.2 fitness value on each trial for 100th, 500th and 100th generation with Inversion swap mutation on Brazil dataset.

Thanawat Anansuthivara 730054400, ta541@exeter.ac.uk

Scramble swap visualization

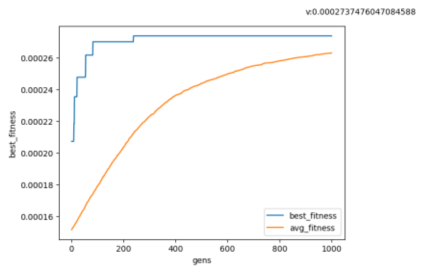

Figure 7.1 Best fitness vs Iterations with Scramble swap mutation on Burma dataset

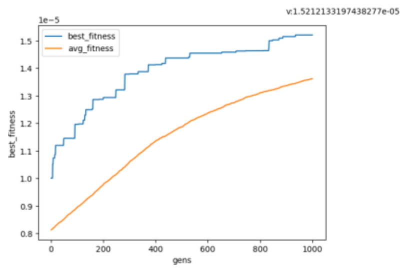

Figure 8.1 Best fitness vs Iterations with Scramble swap mutation on Brazil dataset.

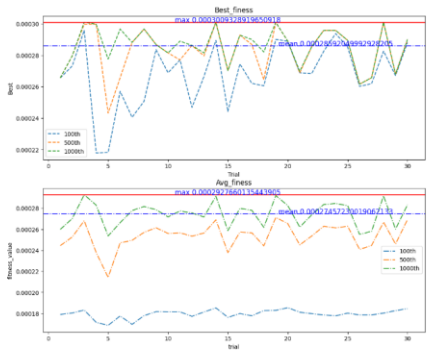

Figure 7.2 fitness value on each trial for 100th, 500th and 100th generation with Scramble swap mutation on Burma dataset.

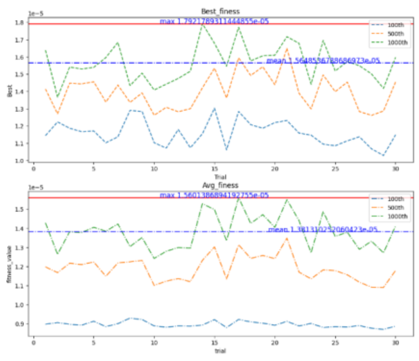

Figure 8.2 fitness value on each trial for 100th, 500th and 100th generation with Scramble swap mutation on Brazil dataset.

Thanawat Anansuthivara 730054400, ta541@exeter.ac.uk

The Crossover experiments.

Crossover experiment on Burma dataset (20 trail)						
Crossover function	500th Gen		1000th Gen		Time (second)	
	Mean	Std	Mean	Std	Mean	Std
Point with fixed	0.000291	0.000008	0.000298	0.000005	0.4995975	0.277170
Partial Map	0.000293	0.000007	0.000299	0.000003	0.344865	0.035729
Ordered	0.000296	0.000004	0.000300	0.000003	0.340050	0.028378

Table 3 Crossover Operator Experiment Result on best fitness value it generates on Burma dataset.

Crossover experiment on Brazil dataset (20 trial)						
Crossover function	500th Gen		1000th Gen		Time (second)	
	Mean	Std	Mean	Std	Mean	Std
Point with fixed	0.000017	8.91835e-7	0.000021	1.28734e-6	0.6663590	4.98601e-2
Partial Map	0.000016	8.96991e-7	0.000020	1.51089e-6	0.6999622	3.34106e-2
Ordered	0.000017	9.23710e-7	0.000021	1.18136e-6	0.7001510	4.77472e-2

Table 4 Crossover Operator Experiment Result on best fitness value it generates on Brazil dataset.

Point crossover with fixed.

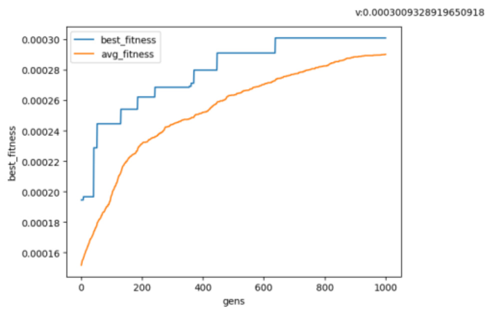

Figure 9.1 Best fitness vs Iterations with Point crossover on Burma dataset.

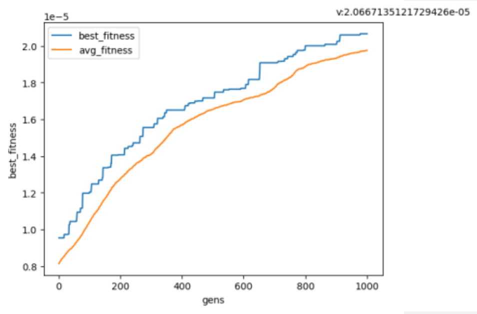

Figure 10.1 Best fitness vs Iterations with Point crossover on Brazil dataset.

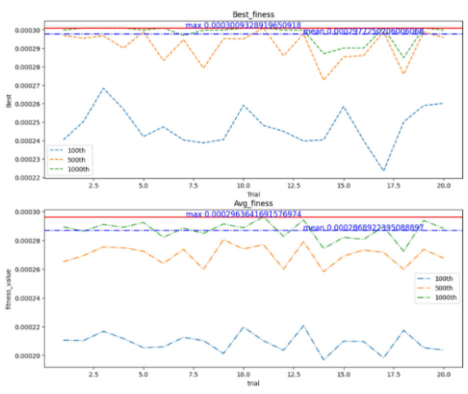

Figure 9.2 fitness value on each trial for 100th, 500th and 100th generation with Point crossover on Burma dataset.

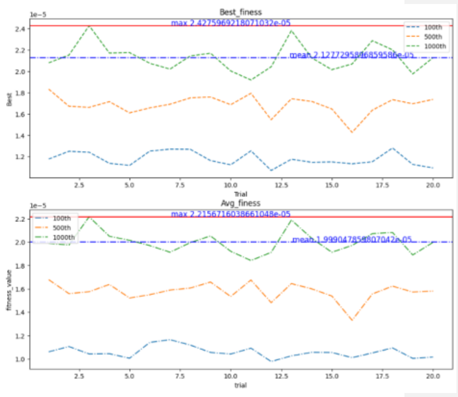

Figure 10.2 fitness value on each trial for 100th, 500th and 100th generation with Point crossover on Brazil dataset.

Thanawat Anansuthivara 730054400, ta541@exeter.ac.uk

Partial map crossover

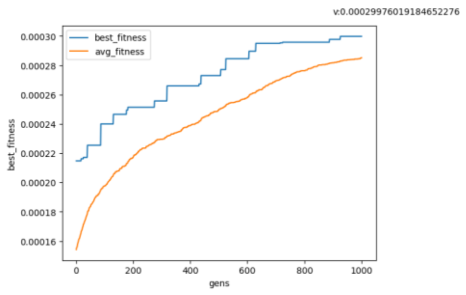

Figure 11.1 Best fitness vs Iterations with Partial map crossover on Burma dataset.

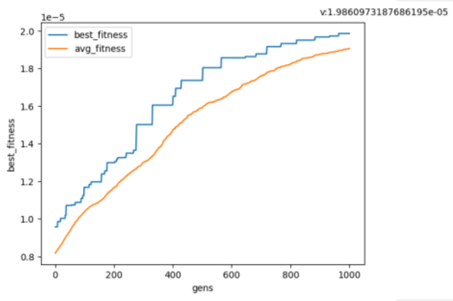

Figure 12.1 Best fitness vs Iterations with Partial map crossover on Brazil dataset.

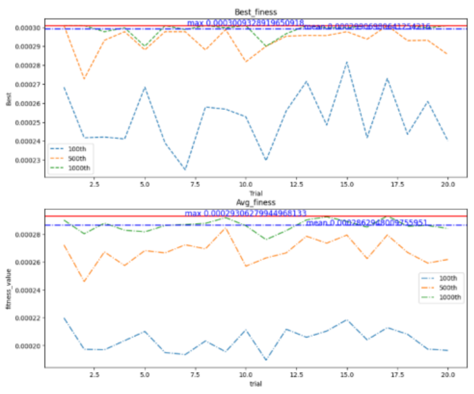

Figure 11.2 fitness value on each trial for 100th, 500th and 100th generation with Partial map crossover on Burma dataset.

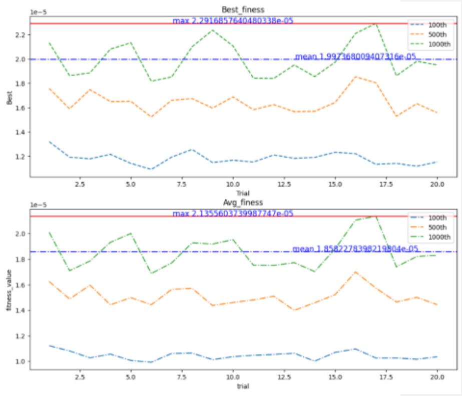

Figure 12.2 fitness value on each trial for 100th, 500th and 100th generation with Partial map crossover on Brazil dataset.

Thanawat Anansuthivara 730054400, ta541@exeter.ac.uk

Ordered Crossover

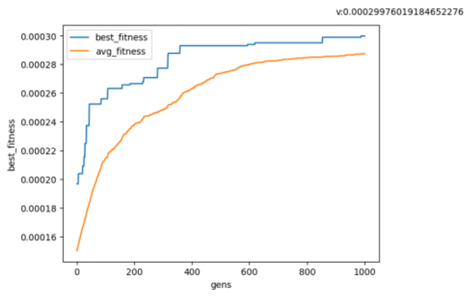

Figure 13.1 Best fitness vs Iterations with Ordered crossover on Burma dataset.

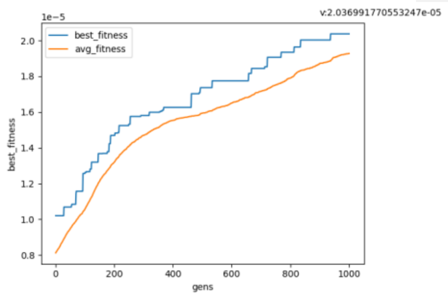

Figure 14.1 Best fitness vs Iterations with Ordered crossover on Brazil dataset.

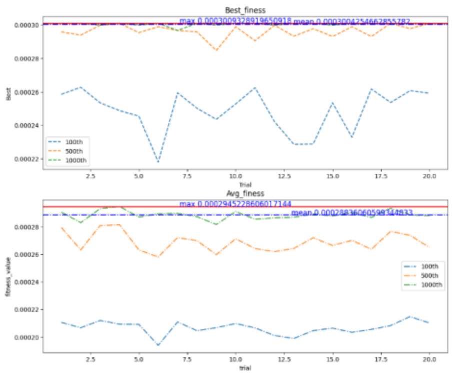

Figure 13.2 fitness value on each trial for 100th, 500th and 100th generation with Ordered crossover on Burma dataset.

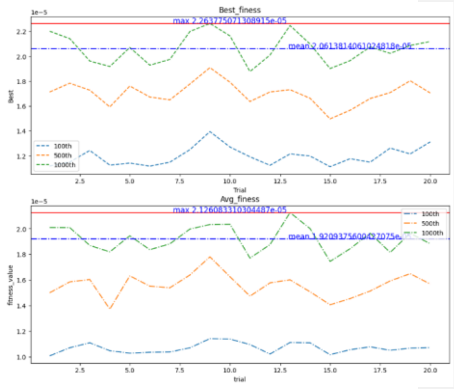

Figure 14.2 fitness value on each trial for 100th, 500th and 100th generation with Ordered crossover on Brazil dataset.

Thanawat Anansuthivara 730054400, ta541@exeter.ac.uk

Parameter Experiments.

Option name	Mutate function	Crossover function
MICO	Inverse swap	Ordered crossover
MICPF	Inverse swap	Point crossover with fixed
MMCO	Multiple swap	Ordered crossover
MMCPF	Multiple swap	Point crossover with fixed

Table 5 Option's mutate and crossover combination on defined name.

Option	Parameter Experiment on Burma dataset. (30 trials)					
	Fitness value			Time		
(max_gen, pop_size, tour_size)	Best	Mean	Std	Best	Mean	Std
MICO (1000,100,10)	3.00932e-4	2.9850e-4	3.46430e-6	0.37299	0.61886	0.198851
MICO (1000,100,80)	3.00932e-4	2.9779e-4	4.06620e-6	1.07702	1.53130	0.351018
MICO (1000,1000,800)	3.00932e-4	2.98830e-4	3.75162e-6	8.81779	11.58821	1.564603
MICO (1000,500,400)	3.00932e-4	2.96271e-4	5.79899e-6	3.387336	3.9614997	0.5687136
MICO (1000,500,100)	3.00932e-4	2.97459e-4	4.89333e-6	1.286007	1.615701	0.222085
MICO (3000,100,10)	3.00932e-4	3.00816e-4	3.57824e-6	0.892035	0.934767	0.029550
MICO (3000,100,80)	3.00932e-4	3.00698e-4	4.77099e-6	2.22399	2.515334	0.548705
MICO (3000,500,400)	3.00932e-4	3.00259e-4	2.19711e-6	11.55120	12.14114	0.872544
MICO (5000,100,80)	3.00932e-4	3.00777e-4	4.05456e-7	4.106004	4.687186	0.466008
MICO (10000,100,80)	3.00932e-4	3.00932e-4	0.000000	7.508957	9.1870065	1.7504311
MICO (10000,1000,800)	3.00932e-4	3.00932e-4	0.00000	70.38695	84.789599	10.004181

Table 6 Parameter experiment result based on specific set of operators and parameters on Burma dataset.

Option	Parameter Experiment on Burma dataset. (30 trials)					
	Fitness value			Time		
(max_gen, pop_size, tour_size)	Best	Mean	Std	Best	Mean	Std
MICO (1000,100,10)	2.33034e-5	2.04311e-5	1.13802e-6	0.792953	1.13133	0.198932
MICO (1000,100,80)	3.14386e-5	2.89275e-5	1.32762e-6	1.420937	2.124081	0.555807
MICO (1000,1000,800)	3.24212e-5	2.86201e-5	1.58266e-6	8.209049	10.24702	0.913754
MICO (1000,500,400)	3.19611e-5	2.87288e-5	1.67991e-6	3.322969	3.634835	0.247919
MICO (1000,500,100)	2.51294e-5	2.32481e-5	1.32705e-6	1.839968	2.069742	0.172222
MICO (3000,100,10)	3.07862e-5	2.76289e-5	1.35156e-6	1.872003	1.95503	0.054269
MICO (3000,100,80)	3.74686e-5	3.00698e-5	1.20403e-6	3.189970	3.90234	0.51878
MICO (3000,500,400)	3.71996e-5	3.53253e-5	1.19181e-6	9.956036	10.432424	0.495336
MICO (5000,100,80)	3.85981e-5	3.70461e-5	7.32330e-7	5.834999	6.5565046	0.440093
MICO (10000,100,80)	3.92465e-5	3.80009e-5	7.11729e-7	10.89294	12.38091	2.120824
MICO (10000,1000,800)	3.93778e-5	3.77158e-5	9.13498e-7	75.38202	91.72567	10.38447

Table 7 Parameter experiment result based on specific set of operators and parameters on Brazil dataset.