Coursework Report

Travelling Salesman Problem Using Nature Inspired Computation

Vedant Adawadkar (730042744)

Introduction:

This report tries to explain how genetic algorithms are implemented using concepts learnt in nature inspired computation module. Accordingly, this report analyses various parameters of genetic algorithms and their effectiveness by implementing them in the Travelling Salesman Problem. The Travelling Salesman Problem (TSP) is an optimization problem which describes a scenario where a salesman must travel n-cities in a particular country. Here, one of the most important constraints is that each city which is visited by the salesman can only be visited once and the finishing point of the salesman should be the first city. Each city is associated with a cost and the goal is to minimize the cost it takes for the salesman to complete the journey across n-cities.

To this coursework and its report, various experiment parameters that have been included to reduce the cost of the travelling salesman include Population size, tournament size, and the operators which refer to different implementations of genetic algorithms. Various operators referenced throughout the report are Single Gene Swap Mutation, Inversion Mutation, PMX Crossover, PMX Crossover with Single Gene Swap Mutation and PMX Crossover with Inversion Mutation. Population sizes used in the experiments are increments of 50 starting 50, 100, 150 and 200. Similarly, tournament sizes are in the increments of 5 with 5, 10, 15 and 20. Most importantly though the experiments have been run on cities of 2 countries namely Burma with 14 cities and Brazil with 58 cities. In each of the experiments the goal has been to minimize the costs for each country respectively. The results for both countries for all experiments have been stored separately which provides an overview of the results as well as detailed results of the experiments.

1. Which combination of parameters produces the best results?

From the figure 1.1 which shows the best fitness results for Burma it is clearly visible that the costs easily converge to the optimal value of 3,323 majority of times irrespective of the operator or combination of operators and population and tournament sizes. On the other hand, in figure 2.1 which shows the best fitness results for Brazil, there is a bit more variation with respect to the costs being more spread out and that gives a bit more idea about how different parameters affect the costs. In figure 2.1, single gene swap mutation operator no matter the population and tournament size results in higher costs than the rest of the other operators. Inversion mutation on the other hand produces the best results out of all the other operators with the best result being cost of 25,480 for Inversion mutation, the population size of 100, and tournament size of 20 after being run for 10,000 iterations.

Overall, it can be concluded that for the population sample of Brazil and Burma respectively, Inversion mutation with population higher than or equal to 100 and higher tournament size closer to 20 provides the best results in terms of minimizing the costs.

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

One of the reasons why inversion mutation has performed better than other operators is because mutation has better local search effect. This means that instead of making big changes, mutation makes small changes to order of the genes within the chromosomes. Moreover, it is more effective than single gene swap mutation because rather than swapping only gene per iteration we are inverting a group of genes within a single chromosome, thus preserving important segments of the building blocks in generated solutions. One thing to also consider is that the experiment population and the tournament selection also plays vital role which make inversion mutation a more effective solution than others. One thing that might also affect the result is that crossover or crossover mutations might have introduced too much randomness in the population which could have possibly affected the results.

It is important to note that the effectiveness of inversion mutation can vary depending on the problem and most likely be affected by population and tournament selection, so it is not a universally effective solution.

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

For this report, the performance of the algorithm is calculated based on the execution time and best fitness results of each experiment per 10,000 iterations while considering the operator and population and tournament size combination.

Generally, PMX Crossover causes the algorithm to slow down while still having slightly worse solutions than inversion mutations. One of the major reasons that PMX crossover slows down the run time of the algorithm is the logic that has been applied to run it. Moreover, one the reasons why it still has slightly worse solutions than mutations is as previously mentioned in answer to question two, is that because PMX crossover might be introducing too much randomness into the population sample. This results in the algorithm trying to increase its computational effort to improve bad solutions. Experiments with lesser tournament size produce worse results because lesser tournament size does not allow for diversity in selection where population samples with worse fitness are not selected as much. Similarly, having a lesser population sample also results in worse fitness because as mentioned earlier the population might not have that much diversity. In both the cases it results in increasing selection pressure.

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

One of the most common implementations of a local heuristic that can be used, and in fact most used is the local search heuristic. Local search with the help of genetic algorithm operators like mutation and crossover operators as used in the experiments can help with providing better results because it helps in global exploration and local exploitation (Merz & Freisleben et al., 1997). Local search can be implemented after the combination of operators have been implemented to improve the time it takes to reach the optimal solution. Local search starts off with selecting an initial solution on which operators are used to get a set of solutions called neighborhood of solutions. From this neighborhood, using local search procedures a new current solution is selected that improves the value of the objective function. This process is repeated until there are no improving solutions about the current new solution. This is called as a local optimum. There are different types of local search heuristics that have been researched and implemented such as the first improvement local search and its opposite the

delayed improvement local search, Genetic Local Search procedures and so on (Amaral et al., 2021; Zhang, 2010).

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

The experiments carried out in this report are carried out in such a way that crossover and mutation occur in every iteration. Here, crossover and mutation probabilities are not considered when performing the experiments on the travelling salesman problem. Introducing these probabilities can help in improving the results of the algorithm.

According to existing research, there is evidence that crossover and mutation probabilities are quite critical to the success of genetic algorithms. But it is important to note however that correctly identifying the crossover or mutation rates can differ from problem to problem of vary even in different stages of genetic process. Applying the correct crossover or mutation rates can help increase the search space there by improving the genetic algorithm. In fact, the success of the genetic algorithm largely depends on maintaining an acceptable level of productivity throughout the process of evolution (Lin et al., 2003). Which is why it is important to make sure that genetic algorithms correctly adapt themselves to crossover or mutation rates.

6. Do you think of any other nature inspired algorithms that might have provided better results? Explain your answer.

One of more common nature inspired algorithms that can be used to provide better results is the Ant Colony Optimization (ACO). In ACO, ants convey a solution to a problem by leaving an artificial pheromone trail to communicate indirectly with other ants, which are associated with edges. The ants construct solutions by identifying tours iteratively from current city to next city based on some proportional probability. With the inclusion of local search algorithm to solutions provided by a traditional ACO it is possible to improve the results than those of the genetic algorithm simply using crossover and mutations. One more variation of the ACO in the form of MIN-MAX Ant System (MMAS), provides one of the best performing ACO algorithms. Here processes such pheromone trail-reinitialization are used to improve the results. One of the reasons why MMAS is considered a robust variant of MMAS is its ability to not fall into the local optimum by identifying premature convergence to suboptimal solutions early on (Stützle et al., 2000).

Conclusion:

In summary, based on the experiments performed it can be concluded that the for the given population size as well as the tournament size inversion mutation performance was much better than the other operators in the case of Brazil. Even though with Burma majority of experiments were able to reach the optimal solution, due to the population sample size, Brazil was not able to reach its optimal solution and the results were varied based on the parameters applied at the of experimentation. Improvements such as adding crossover rates or mutation rates can help with providing much better results, whereas if applied correctly an Ant Colony Optimization can provide even better results.

Graphs and figures:

Burma - Best Fitness:

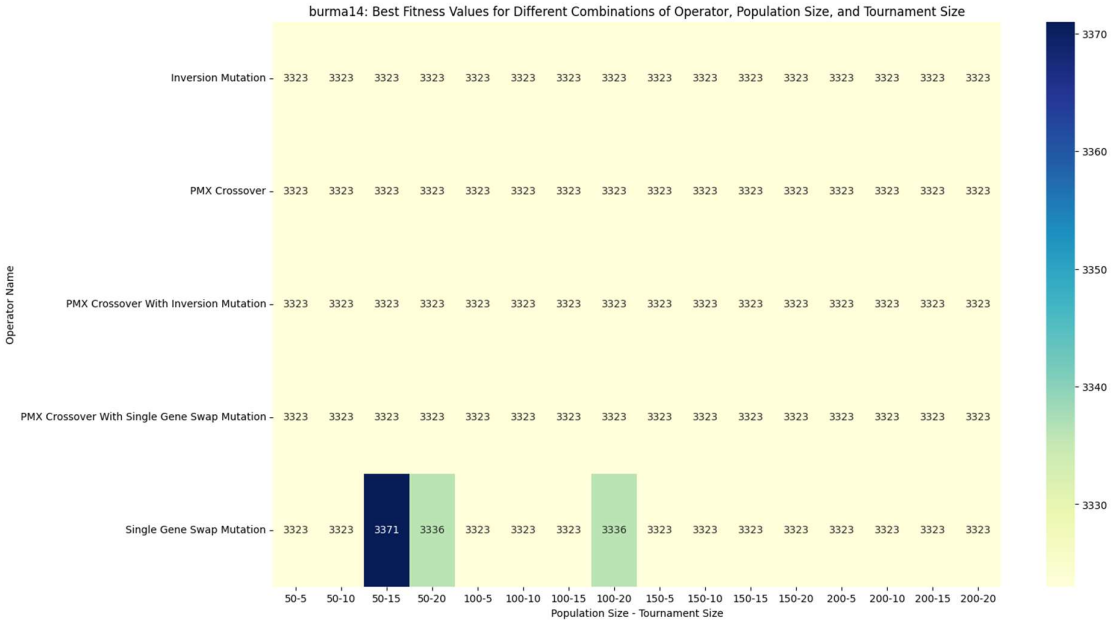

Fig. 1.1. - A heatmap of Burma where along the X-axis we have a combination of population size – tournament size. Y-axis has the names of the operators used in the experiment. The lowest values represent the best results.

Brazil - Best Fitness:

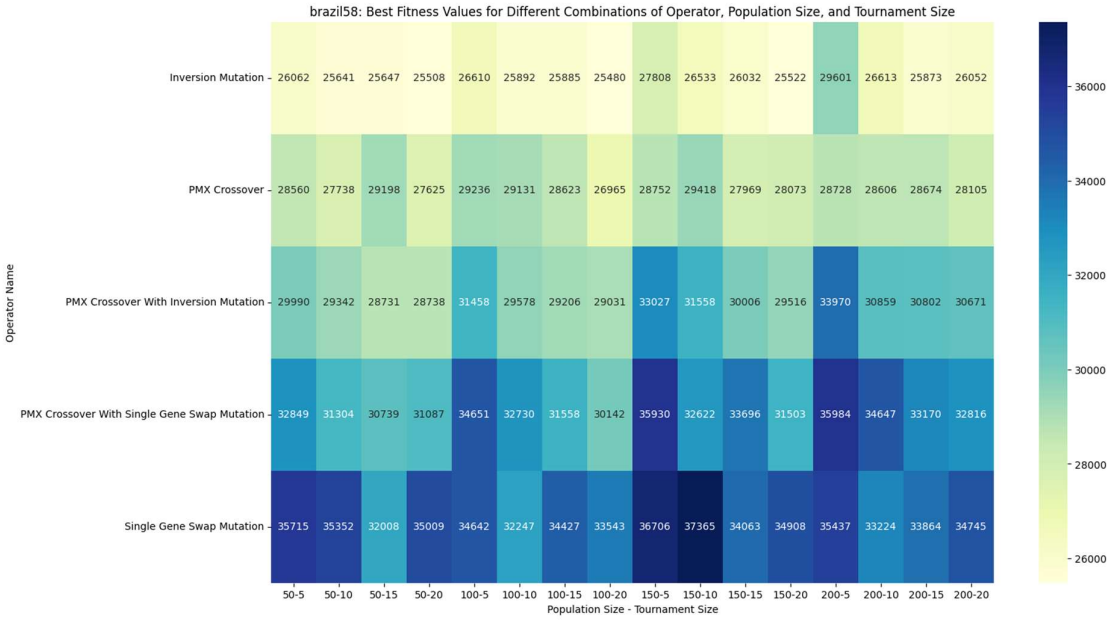

Fig. 2.1. - A heatmap of Brazil where along the X-axis we have a combination of population size – tournament size. Y-axis has the names of the operators used in the experiment. The lowest values represent the best results.

References:

Amaral, H. F., Urrutia, S., & Hvattum, L. M. (2021). Delayed improvement local search. *Journal of Heuristics*, *27*(5), 923–950. https://doi.org/10.1007/s10732-021-09479-9

Lin, W.-Y., Hong, T.-P., Lin, W., Lee, W., & Hong, T. (2003). Adapting Crossover and Mutation Rates in Genetic Algorithms. Adapting Crossover and Mutation Rates * in Genetic Algorithms. In *Article in Journal of Information Science and Engineering* (Vol. 19). https://www.researchgate.net/publication/220587952

Merz, P., & Freisleben, B. (1997). Genetic local search for the TSP: New results. *Proceedings of the IEEE Conference on Evolutionary Computation, ICEC*, 159–164. https://doi.org/10.1109/icec.1997.592288

Stützle, T., ¡£¢, Grün, A., Linke, S., & Rüttger, M. (2000). *A Comparison of Nature Inspired Heuristics on the Traveling Salesman Problem*. https://www.researchgate.net/publication/2456070

Wei, J.-D. (2008). Approaches to the Travelling Salesman Problem Using Evolutionary Computing Algorithms. In *Traveling Salesman Problem*. InTech. https://doi.org/10.5772/5584

Zhang, Q. (2010). Reactive Search and Intelligent Optimization (Battiti, R., et al.; 2008) [Book Review. *IEEE Computational Intelligence Magazine*, *5*(1), 59–60. https://doi.org/10.1109/mci.2009.935312

Andrea, S., Shahrooz AndreaShahrooz Andrea 6711 silver badge77 bronze badges and j4reyj4rey 2 (2016) Sorting by genetic algorithm, duplicate data in cross over, Stack Overflow. Available at: https://stackoverflow.com/questions/37285603/sorting-by-genetic-algorithm-duplicate-data-in-cross-over (Accessed: 13 November 2023).

Crossover of enumerated chromosomes (no date) Appendix A - Genetic Algorithm Internals and Advanced Topics > Crossover of Enumerated Chromosomes. Available at: https://www.wardsystems.com/manuals/genehunter/crossover_of_enumerated_chromosomes.htm (Accessed: 13 November 2023).

Eldamir et al. (2015) Beautifulsoup - search by text inside a tag, Stack Overflow. Available at: https://stackoverflow.com/questions/31958637/beautifulsoup-search-by-text-inside-a-tag (Accessed: 13 November 2023).

KalecKalec 2, deongdeong 3 and Winston EwertWinston Ewert 44.2k1010 gold badges6868 silver badges8383 bronze badges (2020) Trying to find a good data representation for a genetic (evolutionary) algorithm, but I just can't imagine One, Stack Overflow. Available at: https://stackoverflow.com/questions/16590806/trying-to-find-a-good-data-representation-for-a-genetic-evolutionary-algorithm (Accessed: 13 November 2023).

MahlerFiveMahlerFive 5 et al. (2010) Why does adding crossover to my genetic algorithm gives me worse results?, Stack Overflow. Available at: https://stackoverflow.com/questions/2439391/why-does-adding-crossover-to-my-genetic-algorithm-gives-me-worse-results (Accessed: 13 November 2023).

Mutation algorithms for string manipulation (GA) (2022) GeeksforGeeks. Available at: https://www.geeksforgeeks.org/mutation-algorithms-for-string-manipulation-ga/ (Accessed: 13 November 2023).

VincylicVincylic 2333 bronze badges and Mayowa AyodeleMayowa Ayodele 54922 silver badges1111 bronze badges (2020) Handling duplicates when using partially matched crossover for genetic algorithm, Stack Overflow. Available at: https://stackoverflow.com/questions/60320147/handling-duplicates-when-using-partially-matched-crossover-for-genetic-algorithm (Accessed: 13 November 2023).