SPIDER MONKEY ALGORITHM IN SOLVING TRAVELLING SALESMAN PROBLEM

Dawid Płudowski
Faculty of Mathematics and Information Science
Warsaw University of Technology
dawid.pludowski.stud@pw.edu.pl

Antoni Zajko
Faculty of Mathematics and Information Science
Warsaw University of Technology
anotni.zajko.stud@pw.edu.pl

Franciszek Szczepaniak
Faculty of Mathematics and Information Science
Warsaw University of Technology
franciszek.szczepaniak.stud@pw.edu.pl

Kamil Kisiel
Faculty of Mathematics and Information Science
Warsaw University of Technology
kamil.kisiel.stud@pw.edu.pl

Maciej Szpetmański
Faculty of Mathematics and Information Science
Warsaw University of Technology
maciej.szpetmanski.stud@pw.edu.pl

Keywords Spider Monkey · TSP · swarm intelligence

1 Summary

This report introduces the Spider Monkey Optimization (SMO) algorithm as a solution to the Asymmetric Traveling Salesman Problem (ATSP). Inspired by the foraging behavior of spider monkeys, the algorithm mimics their fission-fusion social structure and information sharing. The goal is to find the shortest route that visits each city exactly once and returns to the origin city. The algorithm utilizes swarm intelligence principles and employs operations such as Swap Sequence (SS) and Swap Operator (SO) to interact among monkey representatives of TSP solutions.

The report presents the methodology followed, including the initialization of the monkey population, local leader and global leader phases, learning phases, and decision phases. It also discusses the hyperparameters of the algorithm, such as the maximum number of groups, perturbation rate, local leader limit, global leader limit, and population size.

To evaluate the algorithm's performance, experiments were conducted on benchmark TSPs, and the results were compared to other metaheuristic methods. The Spider Monkey algorithm demonstrated its ability to outperform other algorithms in terms of dependability, effectiveness, and precision. However, a key limitation was identified in the algorithm's reliance on user-dependent parameters, which require significant time investment for optimal tuning.

The report concludes by discussing the potential for further development and improvement of the Spider Monkey Optimization algorithm. It highlights the algorithm's advantages, such as robustness, distributed abilities, fast convergence, and compatibility with hybridization. The report suggests exploring self-adaptive parameter tuning techniques to enhance the algorithm's reliability and recommends considering the growing number of publications dedicated to the algorithm's development and applications.

Overall, the Spider Monkey Optimization algorithm shows promise as an efficient optimizer for the ATSP and offers opportunities for future research and innovation in optimization techniques.

2 Introduction

The following paper is a report that familiarizes the reader with our solution to the Asymmetric Traveling Salesman Problem. It is based on the evolutionary algorithm called Spider Monkey Optimization, inspired by the foraging behavior of Spider Monkeys. Spider Monkeys search for food in groups led by a single individual and share information among themselves. When monkeys are unable to find food, the group splits into subgroups. Subgroups also have their own leaders and venture in different directions in search of food. At some point, all subgroups merge into one large main group. The social behavior of spider monkeys is an example of a fission-fusion social structure.

We will demonstrate how the behavior of monkeys utilized in the algorithm relates to the Traveling Salesman Problem (TSP) and how we ultimately strive for an optimal result in our code. To achieve this, we will present the results of our work for different parameters and conduct their analysis. Our goal was to reach a relatively optimal solution within a time frame of 5 minutes. By paying attention to the parameters accepted by our algorithm and analyzing the returned results, we were able to find the most satisfying solution. The reader will have the opportunity to become acquainted with this analysis and our conclusions drawn from it, which summarize our work on the algorithm.

The TSP optimization project is conducted as part of the Research Workshops in the second and third year of studies in Data Engineering and Analysis. To verify the reliability of our results, they will be compared with those of other teams composed of students participating in the same workshops. They have worked on the same problem but utilizing different algorithms.

Continuing the reading, the reader will have a chance to learn about our ideas for further development of our approach to this solution. We are aware that despite the extensive time dedicated to writing the algorithm and conducting the analysis, there is always room for improvement. Therefore, if someone would like to expand on our idea, they will be able to draw inspiration from our suggestions.

Algorithms based on animal behavior are continuously improved. Increasingly innovative approaches and attempts to tackle problems from different perspectives can lead to finding better solutions even for challenging problems. As the authors of this report, we hope that presenting one of the attempts to utilize an evolutionary algorithm can yield fruitful results and inspire self-improvement and exploration of innovative optimization techniques.

3 Methodology

Asymmetric Traveling Salesman Problem asks the following question: "Given a list of cities and the distances between each pair of cities, what is the shortest possible route that visits each city exactly once and returns to the origin city?" It is an NP-hard problem in combinatorial optimization, important in theoretical computer science and operations research.

The problem was first formulated in 1930 and is one of the most intensively studied problems in optimization. It is used as a benchmark for many optimization methods. Even though the problem is computationally difficult, many heuristics and exact algorithms are known, so that some instances with tens of thousands of cities can be solved completely and even problems with millions of cities can be approximated within a small fraction of 1%.

To tackle this problem we decided to turn our eyes to swarm intelligence based optimization algorithms, specifically to an algorithm known as Spider Monkey Optimization. Although we were firstly drawn to it by its rather peculiar name, the more we read about it, the better of a solution it appeared.

4 Algorithm

In the proposed discrete Spider Monkey Optimization (SMO), every spider monkey represents a TSP solution; Swap Sequence (SS) and Swap Operator (SO)-based operations are considered for interaction among monkeys to find the optimal TSP solution. An SS is a group of SOs. Each contains a pair of indexes on the TSP tour. A new tour is generated transforming a TSP tour by swapping cities indicated by SOs of an SS. The SS generation to update a monkey for an improved solution is the key feature of the method and SS of a particular spider monkey is generated with interaction with other members of the group. In the case of updating a solution of spider monkey with generated SS, a Partial Search (PS) technique is deployed to achieve the best outcome with full or partial SS. The proposed method has been tested on a suite of benchmark TSPs and is proved to outperform other well-known metaheuristic methods applied to TSP.

4.1 General overview of the algorithm

Initialization:
  Initialize the monkey population
  Find the global leader of the population
  Create a monkey group g_1 consisting of the entire population
  Set the group's global leader as the local leader
Algorithm:
  While the termination criterion is not met:
    Local Leader Phase
    Global Leader Phase
    Local Leader Learning Phase
    Global Leader Learning Phase
    Local Leader Decision Phase
    Global Leader Decision Phase

4.2 Hyperparameters of the algorithm

- G_{max} - maximum allowed number of groups
- p - perturbation rate
- LC_{max} - local leader limit
- GC_{max} - global leader limit
- N - population size

4.3 Markings

- f - cost function that maps a monkey representing a Hamiltonian cycle to its cost
- LL_g - local leader of group g
- LC_g - counter of the local leader of group g
- GL - global leader of the entire population
- GC - counter of the global leader
- G_c - number of groups

4.4 Step by Step

4.4.1 Local leader phase

For each monkey group g:
  For each monkey m in group g:
    If $U(0,1) \geq p$:
      m_r = random monkey from g
      $m_{new} = m + U(0,1)(LL_g - m) + U(0,1)(m_r - m)$
      If $f(m_{new}) \leq f(m)$:
        $m = m_{new}$

4.4.2 Global leader phase

For each monkey group g:
  For each monkey m in group g:
    $p_m = 0.9 * \frac{f(GL)}{f(m)} + 0.1$
    If $U(0,1) \leq p_m$:
      m_r = random monkey from the population
      $m_{new} = m + U(0,1)(GL - m) + U(0,1)(m_r - m)$
      If $f(m_{new}) \leq f(m)$:
        $m = m_{new}$

4.4.3 Local leader learning phase

For each monkey group g:
  $LL_{g,new}$ = monkey in group with the smallest value of f
  If $f(LL_{g,new}) < f(LL_g)$:
    $LL_g = LL_{g,new}$
    $LC_g = 0$
  Otherwise:
    $LC_g = LC_g + 1$

4.4.4 Global leader learning phase

GL_{new} = Local leader with the smallest f
If $f(GL_{new}) < f(GL)$:
  $GL = GL_{new}$
  $GC = 0$
Otherwise:
  $GC = GC + 1$

4.4.5 Local leader decision phase

For each monkey group g:
  If $LC_g > LC_{max}$:
    $LC_g = 0$
    For each monkey m in g:
      If $U(0,1) \geq p$:
        m = random solution
      Otherwise:
        $m = m + U(0,1)(GL - m) + U(0,1)(LL_g - m)$

4.4.6 Global leader decision phase

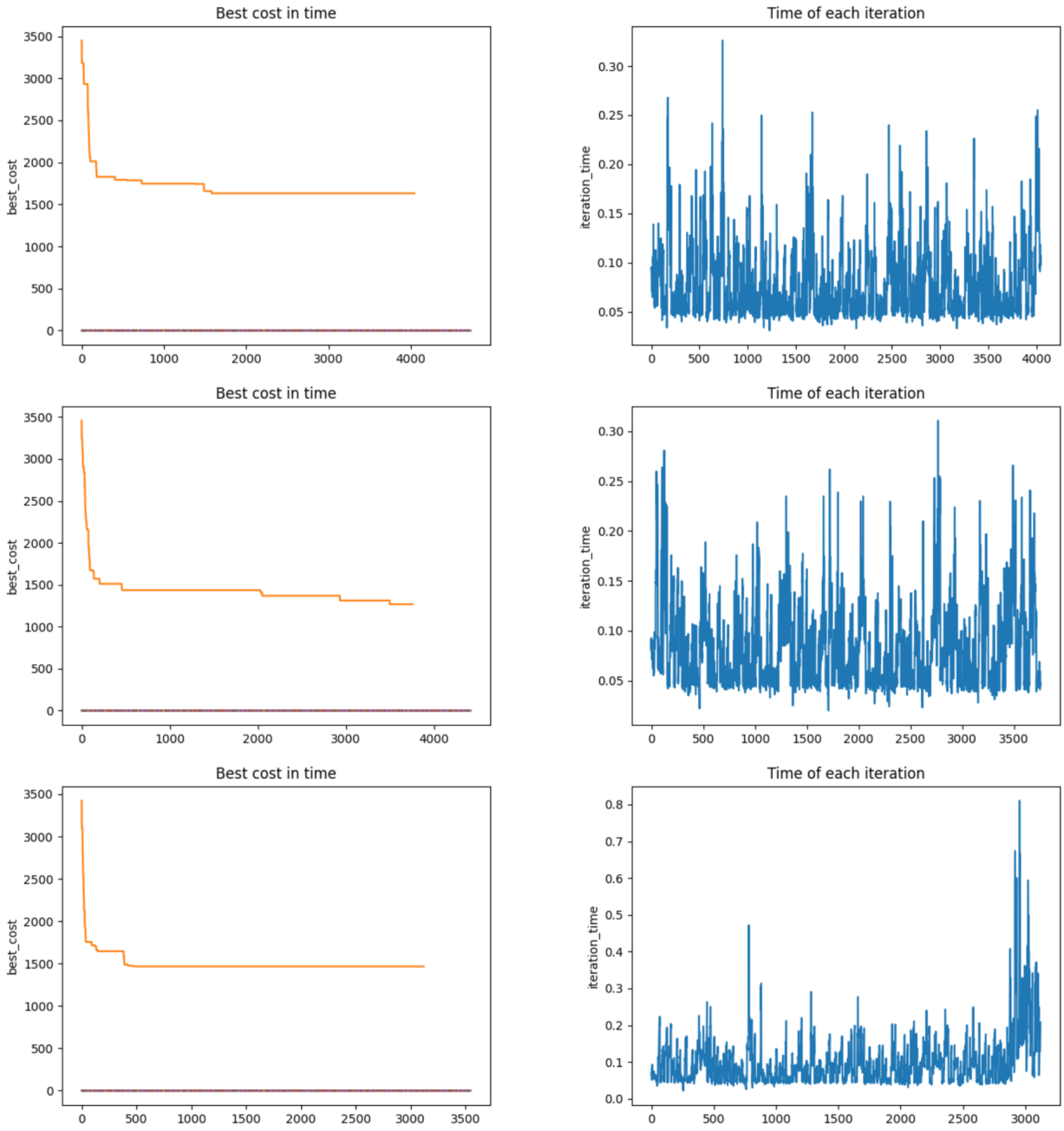
If $GC > GC_{max}$:
  $GC = 0$
  If $G_c < G_{max}$:
    Split a random monkey group into 2 groups
    $G_c = G_c + 1$
  Otherwise:
    Merge all groups into a single group
    $G_c = 1$

5 Experiments

Hyperparameters of the algorithm will be tested on two problems, where in the first one, the average cost obtained is significantly lower than in the second one.

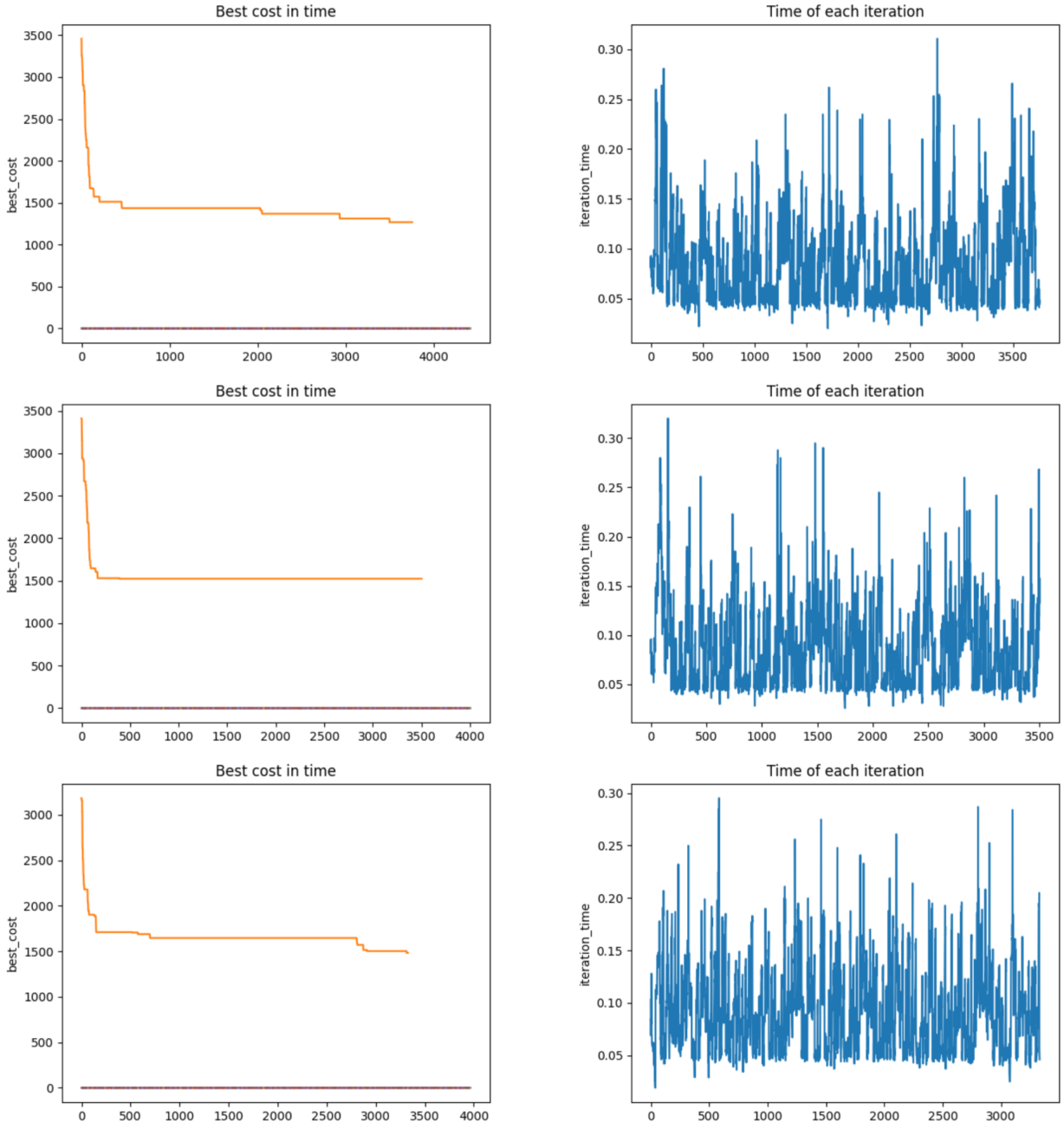
5.1 1st Problem

5.1.1 Allowed maximum groups = [3,4,5]



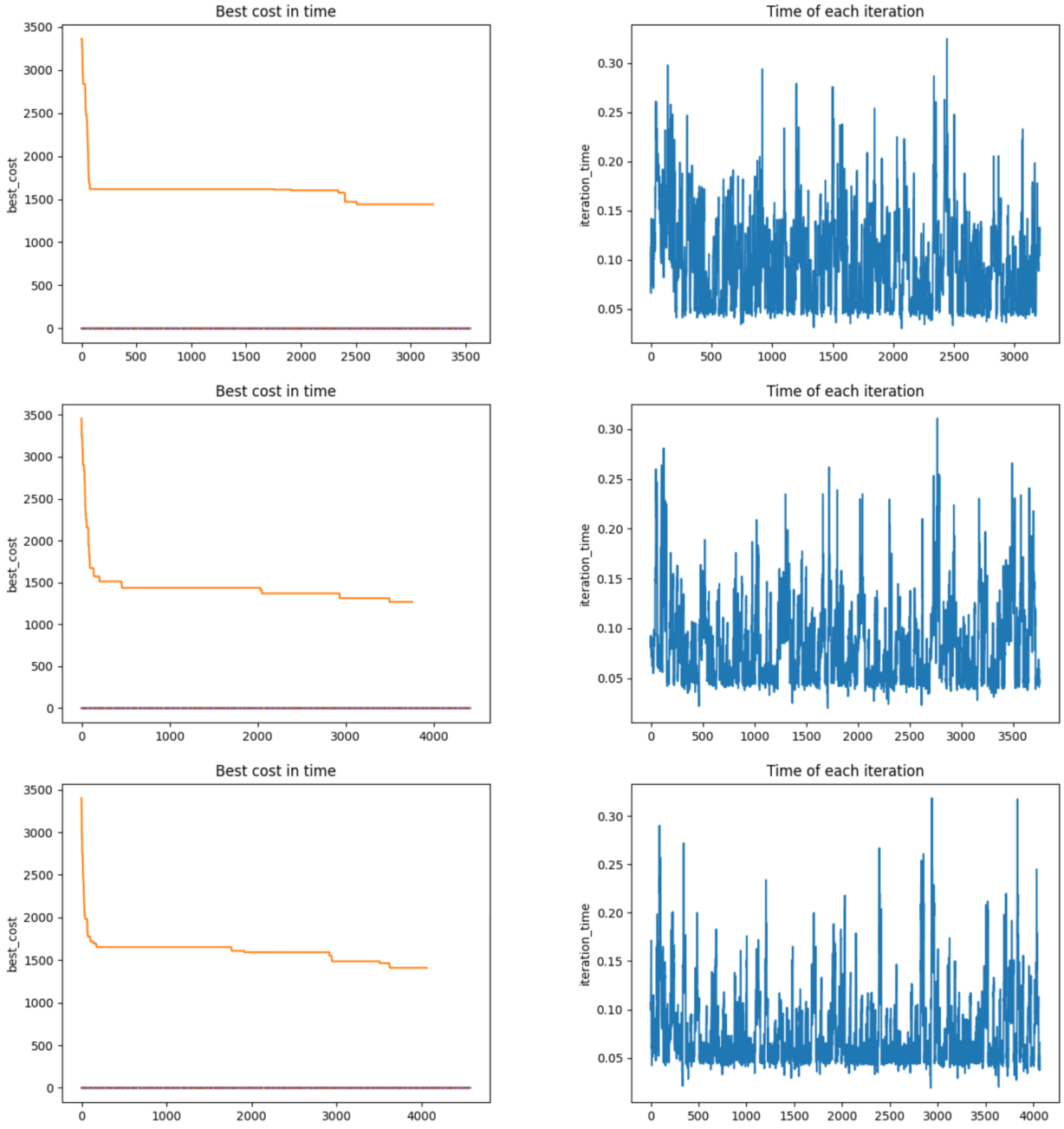
Conclusion - for 5, the algorithm achieved the best result in the best time.

5.1.2 Global leader limit = [4,5,6]



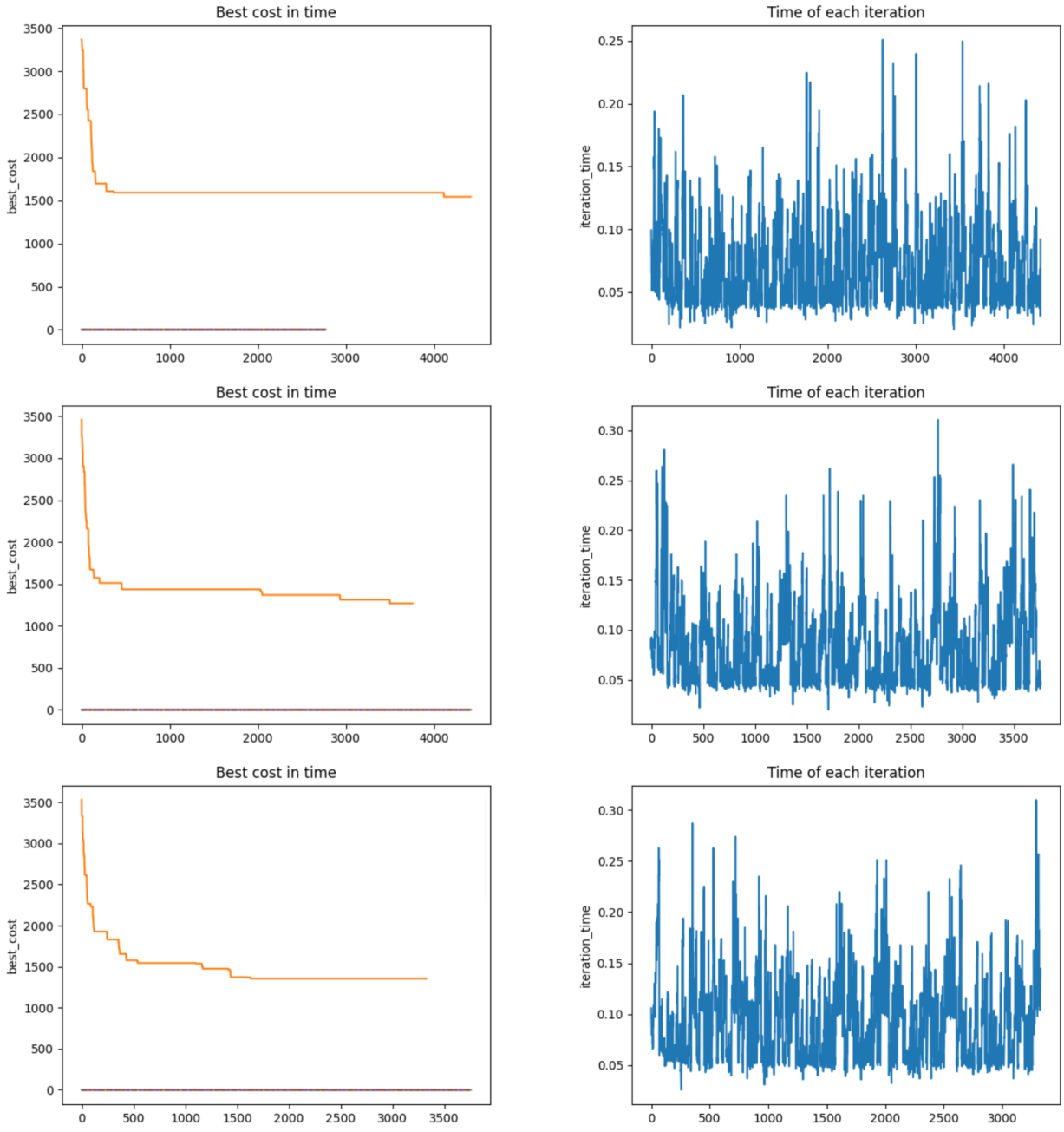
Conclusion - we obtained the best result for 4, but it took significantly more time than for 5, where the result was slightly weaker.

5.1.3 Local leader limit = [2,3,4]



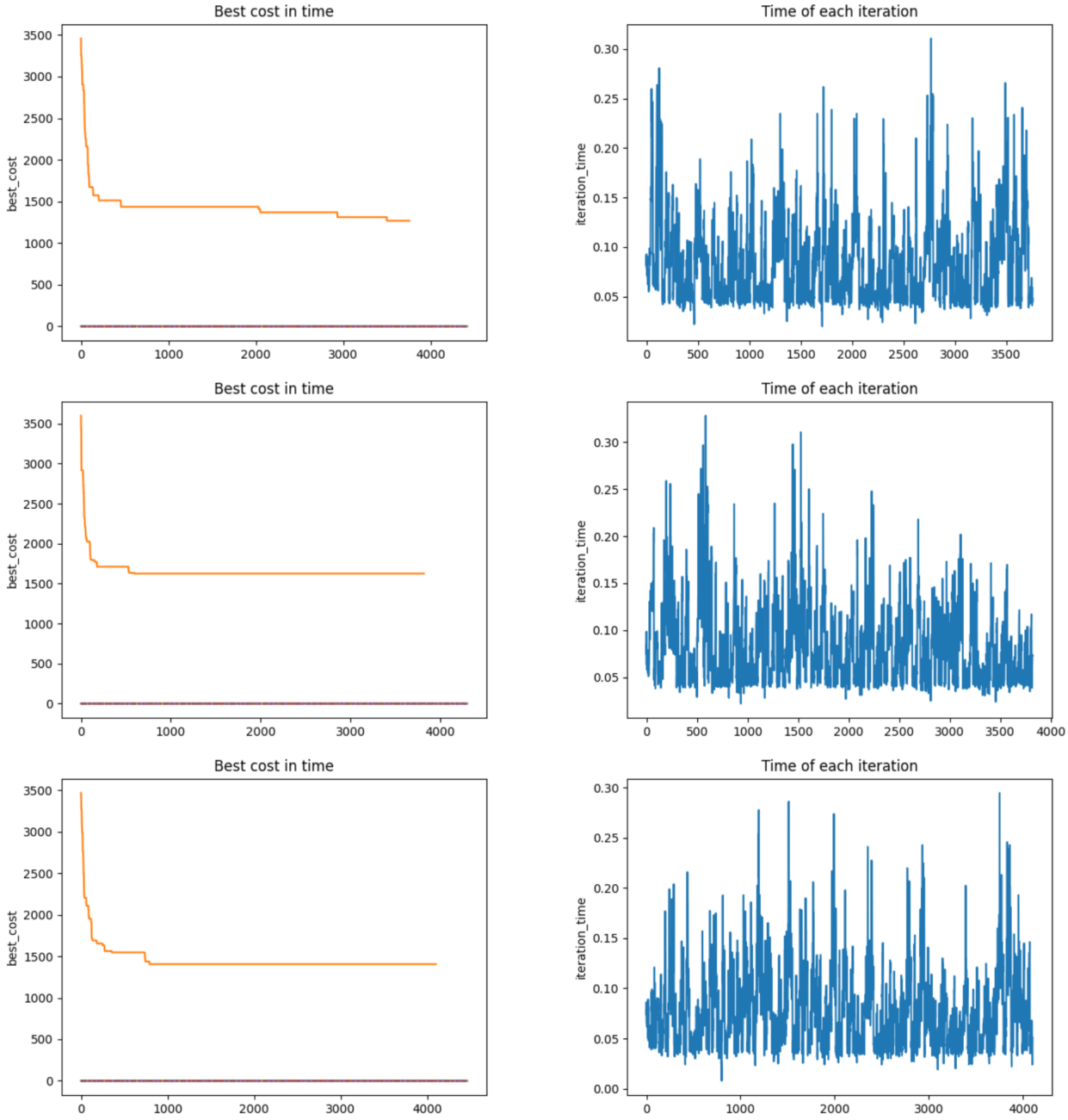
Conclusion - all 3 options yielded similar results in a similar amount of time.

5.1.4 Population size = [70,80,90]



Conclusion - we achieved the best results for 80 and 90, with a slight advantage for the iteration count of the latter.

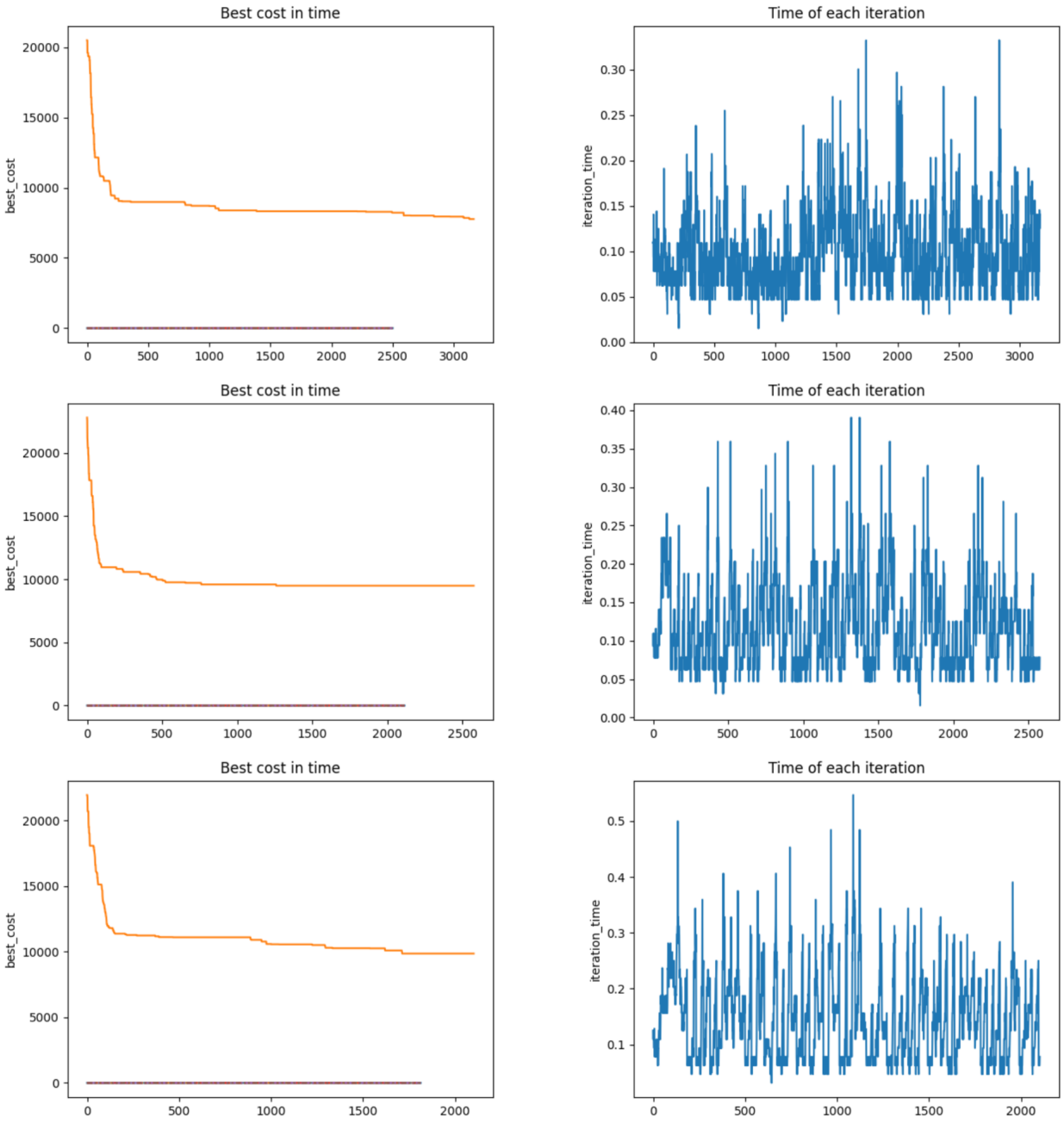
5.1.5 Perturbation rate = [0.3,0.4,0.5]



Conclusion - we obtained the best result for 0.3, but it took significantly more time than for 0.5, where the result was slightly weaker.

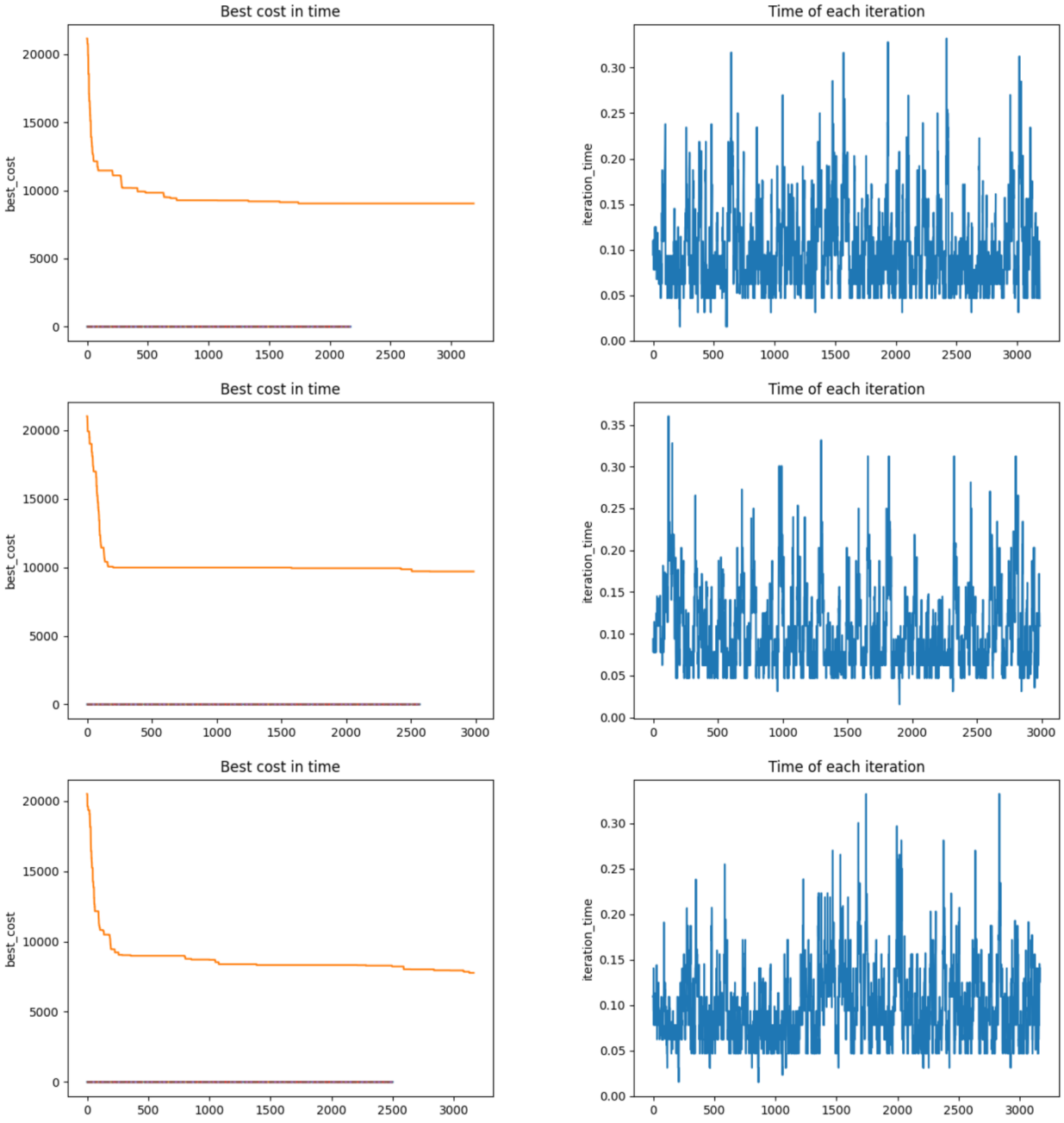
5.2 2nd Problem

5.2.1 Allowed maximum groups = [3,4,5]



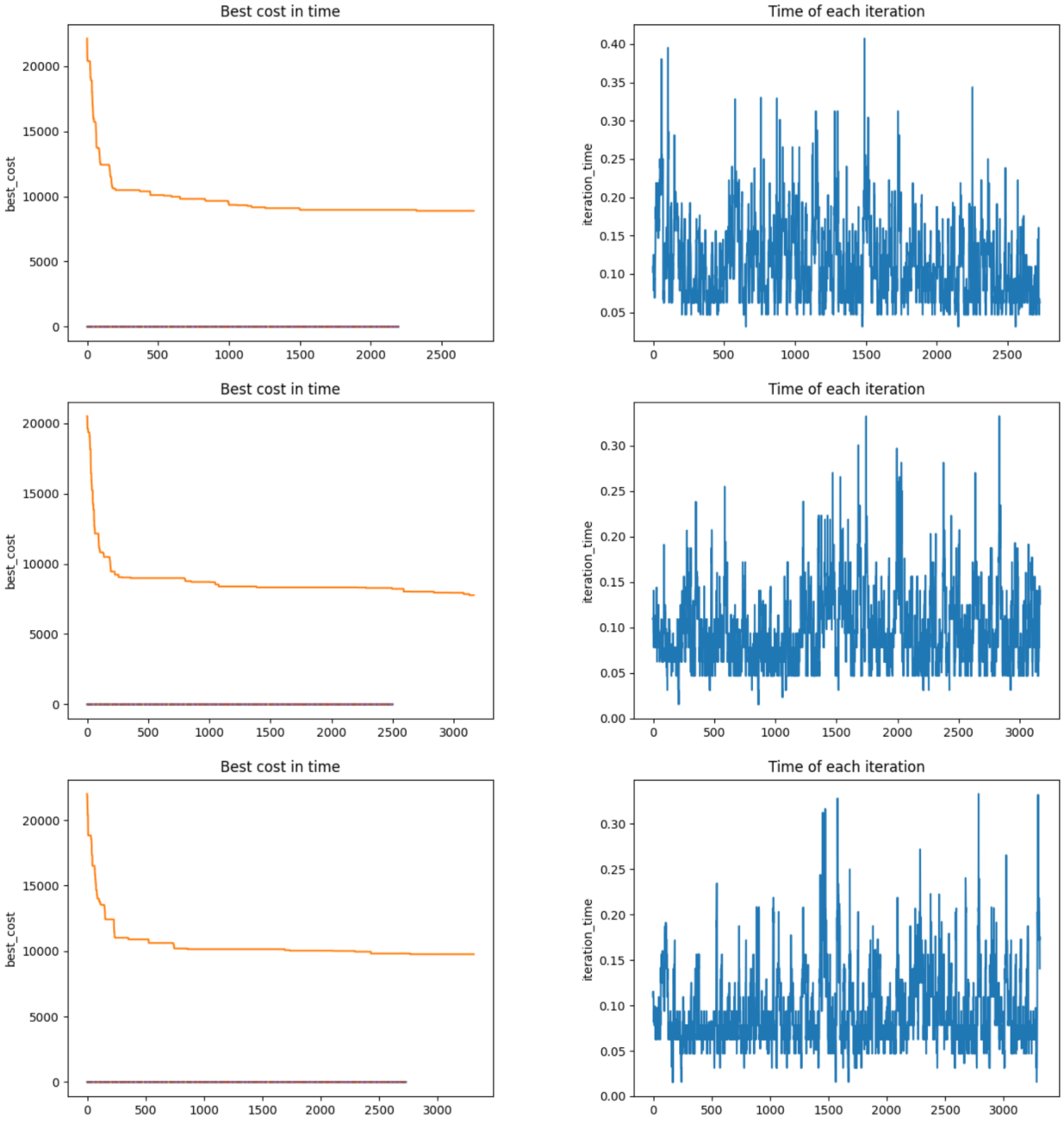
Conclusion - clearly, we achieved the best result for 3, but it took longer compared to the other options.

5.2.2 Global leader limit = [4,5,6]



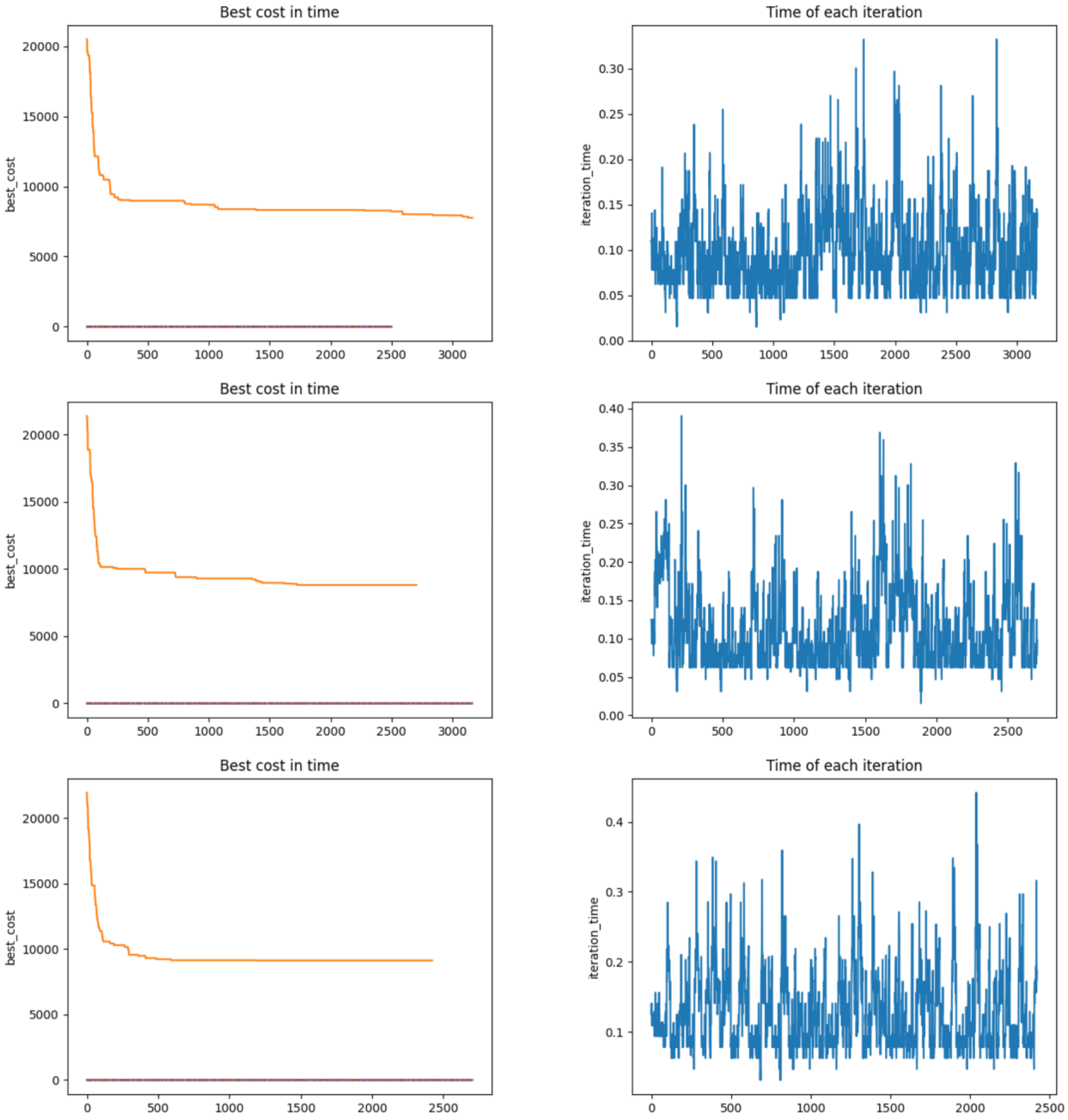
Conclusion - 6 performed the best without significant differences in time compared to other options.

5.2.3 Local leader limit = [2,3,4]



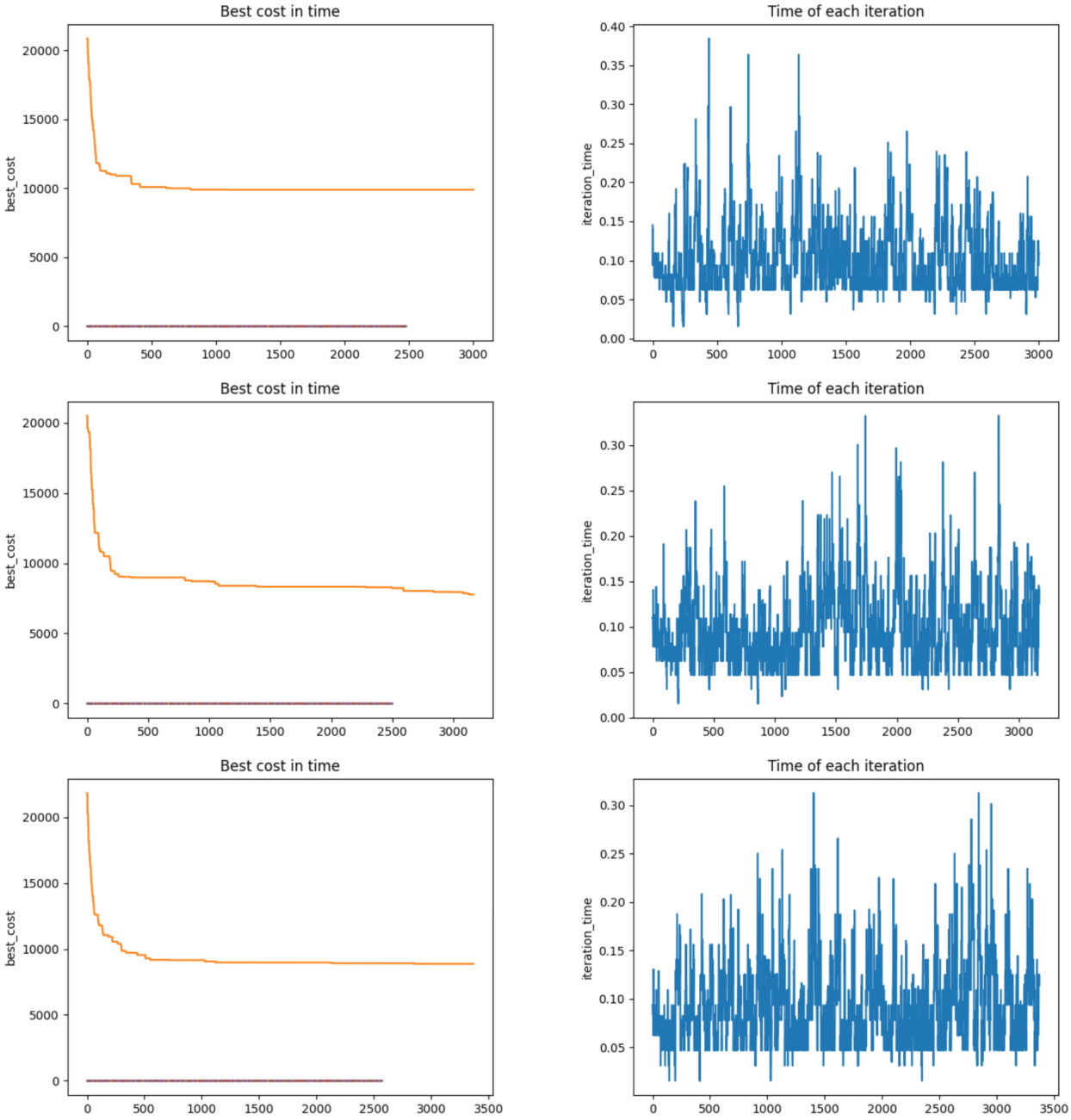
Conclusion - we obtained the best result for 3 in a decent amount of time.

5.2.4 Population size = [70,80,90]



Conclusion - the best result was achieved for 70.

5.2.5 Perturbation rate = [0.3,0.4,0.5]



Permutation rate - 0.4 performed the best, although it took slightly more time than 0.3, it yielded significantly better results.

6 Results

The Spider Monkey algorithm enables finding an optimal solution to the Asymmetric Traveling Salesman Problem, but its Achilles' heel lies in its hyperparameters. Of course, with the right choices, they can yield optimal solutions. However, it is impossible to find a parameter set that guarantees optimal solutions for every given problem. In the above examples, the best results were obtained using completely different parameter sets (except for the local leader limit). This makes it difficult to rely on the algorithm, as tuning the hyperparameters for each problem requires a significant amount of time.

7 Comparison with other algorithm

The Traveling Salesman Problem is something that can be tackled with many different approaches. Alternatively to our Spider Monkey algorithm it can be solved by for example Simulated Annealing. It is a probabilistic technique for approximating the global optimum of a given function. Specifically, it is a metaheuristic to approximate global optimization in a large search space for an optimization problem. For large numbers of local optima, SA can find the global optima. It is often used when the search space is discrete.

SA algorithm proposed by other group, although relatively simpler in implementation has proven to be worse in finding the desired solution. In order to determine that we have run dozens of test. Both of the algorithms were given five minutes to find the optimal path for our salesman to travel. Below are some of the results given by respectively our SMO algorithm on the left and on the right the SA algorithm. (Solution for the problem both algorithms were tested on is 6905).

SMO	SA
10040	21128
10073	21043
10031	21173
10120	21585
10070	21220
10059	21743
10088	21542
10044	21205
10040	21441
10067	21064

As we can clearly see Spider Monkey Optimization has proven to be undoubtedly better in finding a solution closest to the optimal one.

8 Discussion

As a technique appearing not long time, SMO algorithm has been receiving gradually wider attention over recent years. Advantages of SMO algorithm can be summarized as follows: [2] It has excellent robustness and can be used in different application environments with a little modification. [4] It has strong distributed ability, because the algorithm is essentially the swarm evolutionary algorithm, so it is easy to realize parallel computation. [3] It can converge to the optimization value quickly. [1] It is easy to hybridize with other algorithms to improve its performance.

In SMO, the local leader phase and the global leader phase help in exploitation of the search space, while exploration is done through the local leader decision phase and global leader decision phase. SMO performance analyses show that SMO outpaces ABC, DE and PSO algorithms, in terms of dependability, effectiveness and precision, provided that its parameters are chosen wisely. However, the presence of a large number of user dependent parameters in SMO is a matter of concern for further research. Self-adaptive parameter tuning may help to improve the robustness and reliability of the algorithm. Just in 4 years, a large number of publications on development and applications of SMO show that it has a great potential to be an efficient optimizer.

References

[1] Juan José Miranda-Bront Agustín Montero, Isabel Méndez-Díaz. An integer programming approach for the time-dependent traveling salesman problem with time windows. *Computers Operations Research*, 88, December 2017.

[2] S.A. Shahriyar N. Siddique H. Adeli M.A.H. Akhand, Safial Islam Ayon. Discrete spider monkey optimization for travelling salesman problem. *Applied Soft Computing*, 86, January 2020.

[3] Shengwu Xiong Prabhat R. Singh, Mohamed Abd Elaziz. Modified spider monkey optimization based on nelder–mead method for global optimization. *Expert Systems with Applications*, 110, November 2018.

[4] Yaroslav Salii. Revisiting dynamic programming for precedence-constrained traveling salesman problem and its time-dependent generalization. *European Journal of Operational Research*, 272, January 2019.