Using Simulated Annealing Technique for Scheduling Optimization Problem

Elaman Turekulov
Dept. of Data Science
Innopolis University
Innopolis, Russia
e.turekulov@innopolis.university

Danil Meshcherekov
Dept. of Data Science
Innopolis University
Innopolis, Russia
d.meshcherekov@innopolis.university

Demid Efremov
Dept. of Data Science
Innopolis University
Innopolis, Russia
d.efremov@innopolis.university

***Abstract*—This document describes the Santa's Workshop Tour 2019 problem and introduces the Simulated Annealing algorithm as a means of solving this problem. Additionally, it explores hyper-parameters and fine-tuning and contains the link to the source code for this research project.**

Keywords—Scheduling, Optimization, Nature-Inspired Computing, Simulated Annealing

I. Introduction

Scheduling optimization is a problem about constructing an optimal schedule for tasks with constrained time or resources. Such problems are common in various domains that require to minimize completion time or to maximize resource utilization.

Simulated Annealing is a stochastic global search optimization algorithm. It is a nature-inspired computing algorithm that mimics the annealing process – heating and forging the metal, controlling the temperature and the cooling rate.

This document is focused on applying the Simulated Annealing approach to solve a complex scheduling optimization problem - in particular, the Santa's Workshop Tour 2019 problem – in a way that minimizes the penalty cost. We propose a solution and explore its effectiveness through experiments and evaluation.

II. Related work

Nature has many amazing phenomena, so it is natural for people to use models and computational techniques inspired by nature. In an attempt to understand the world in terms of information processing, we understand how and why these natural models work, get insights about nature, and find deep relationships between familiar yet seemingly unrelated algorithms.

John Holland [1] was the first scientist to present the concept of genetic algorithms, which are a type of algorithms that mimic by natural selection. It has since inspired multiple other algorithms and techniques based on natural processes.

The Simulated Annealing approach was first introduced by S. Kirkpatrick *et al.* in [2], where they have observed a useful connection between statistical mechanics and combinatorial optimization. They provided a detailed analogy with annealing in solids.

Other research [3], [4] has used simulated annealing for benchmark routing scheduling problems and compared the results to other genetic algorithms. The results have shown that the proposed algorithms outperform other algorithms in terms of solution quality and computational time.

III. Methodology

Our main objective is to find an optimal solution to the Santa Workshop 2019 Tour problem [5].

Each of the 5,000 families has listed their top 10 preferences for the dates when they would like Satan to attend them. Dates are integer values representing the days before Christmas. Each family also has a number of people attending.

There are several constraints and conditions Santa has to follow. The main condition for Santa is that he has to attend at least 125 people but no more than 300 people each day. If this condition is not met, then the schedule is considered invalid. Then, Santa must provide consolation gifts to each family, if he was not able to live up to their expectations. Additionally, there is an accounting penalty, which depends on daily occupancy.

The main project idea is to explore how to schedule the families to Santa's workshop in a way that is both valid, and minimizes the penalty cost to Santa.

Our research team has attempted to solve this problem with an algorithm that is similar to Genetic Algorithms. The algorithm flow is as follows:

- Generating a population of random genes;
- Simulating the annealing process to choose genes that will be promoted to the next generation;
- Creating a new generation by the means of mutation and crossover.

The main difference of our algorithm from Genetic Programming is simulating the annealing process. As opposed to choosing several best genes, it has a random chance of choosing a worse gene. This approach increases the time needed for the algorithm to converge, but produces better results by allowing it to escape from local optima.

Simulated Annealing is done by sampling the Beta distribution with parameters a=1 and b=epoch, where epoch is the current epoch number. This approach allows the genes to be chosen quite at random in earlier epochs, and to get progressively better as more epochs pass.

Figures 1.1 and 1.2 show the visualization of the described process on a sorted list of genes. Note that the gene at position 0.0 is the best gene in the population, while the gene at position 1.0 is the worst one.

Figure 1.1 shows the gene selection in epoch #2: they are selected almost at random. This behavior allows the model to explore the search space. Figure 1.2 shows the gene selection in epoch #15: the model obviously favors the genes that show better performance. It introduces search space exploitation in the model.

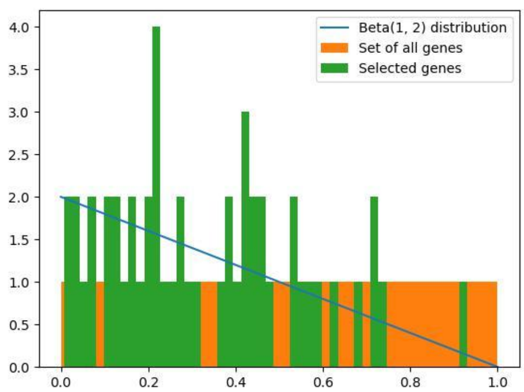

Figure 1.1: gene selection in epoch #2

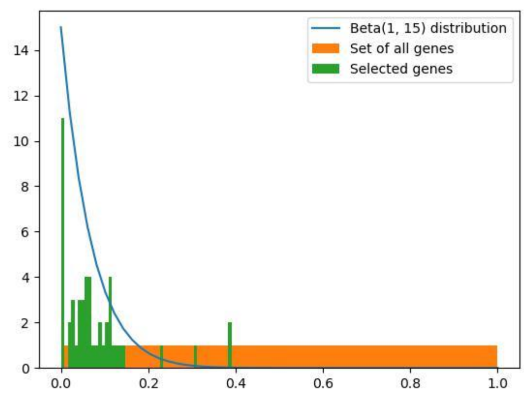

Figure 1.2: gene selection in epoch #15

Moreover, each gene was selected based on its cost value: if it was invalid, it was rejected. So, each generation could only consist of valid genes. While it could have worsened overall finding the optimal solution, this approach could also generate no valid solution whatsoever.

IV. GitHub

The source code, readme file, sample dataset, and our generated submission file are available at [6].

V. Analysis and observations

The model has multiple hyper-parameters.

The first two are survival rate and mutation degree. Survival rate shows the proportion of genes that survive and make it into the next generation. If the survival rate is small, then only a few best of them are selected, which makes convergence faster, but has a high possibility of trapping in a local optimum; however, if the rate is large, then the model effectively explores the search space with a slow convergence speed. Mutation degree simply shows how many families, out of 5,000, change their days after the mutation function.

These two hyper-parameters were selected in a grid search-like manner. Fig.2 shows the results of fine tuning. The optimal parameters were found to be 0.05 for survival rate and 340 for mutation degree.

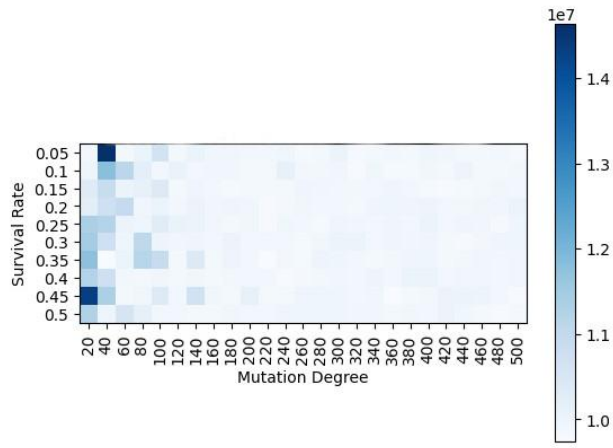

Figure 2: hyperparameters heatmap

As for the next hyper-hyperparameter, we need to go a little deeper into the dataset problem statement. If no preferred days are selected, then the cost of consolation gifts is extremely high. On the other hand, if only the preferred days are selected, genes no longer satisfy the main condition and, therefore, invalid. In our team’s final version of code those two approaches are combined – there is a random chance x that for a certain family the day will be picked randomly, and 1-x chance to choose one of its preferred days.

Since our algorithm is designed to discard and re-generate genes if they are not valid, the choice of x significantly affects time taken to generate each generation. Figure 3 shows how much time it takes to generate a single gene, depending on the choice of x. Hence, it was decided to choose x = 0.1.

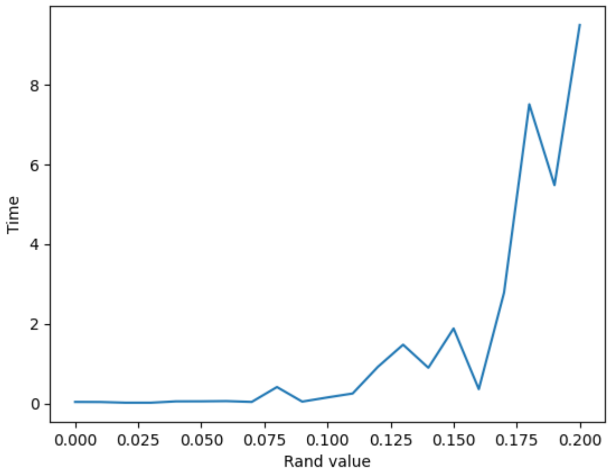

FIgure 3: time taken to generate one gene, in seconds

VI. Experiments and evaluation

Our team has conducted several experiments on the dataset and the code, including:

- Several gene generation techniques
- Different implementations of crossover
- Several ways to implement gene selection for simulated annealing
- Various choices of the population size and epochs.

However; all of them have shown little to no improvement, so it was decided not to include these options as tunable hyperparameters.

The only intriguing part of experimentation was the choice of whether or not to always retain the best gene in the next generation. If the best gene was not kept, the loss function shows an interesting behavior of skyrocketing to some high value with no intention of going back and plateauing there. Figure 4 illustrates such typical behavior.

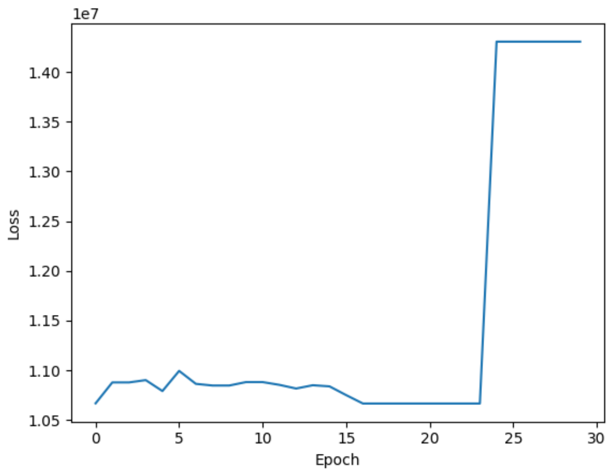

Figure 4: typical optimization process for model without retaining the best gene

Finally, after fine-tuning all parameters and experimenting with different models, we decided to run the model with the population size of 1000 genes and for 50 epochs. The best gene our model could possibly generate had a score of laughable 8.9 million points. Figure 5 shows the timeline of how the loss decreased as the generations passed.

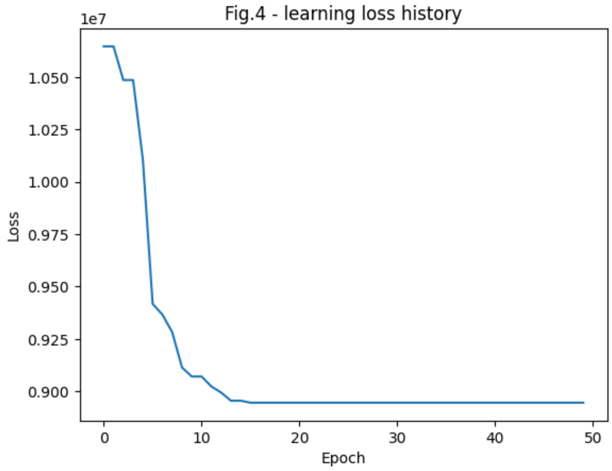

FIgure 5: loss function for the fine-tuned model

VII. Conclusion

Santa’s Workshop 2019 Tour is a challenging problem that requires finding an optimal schedule for Santa, such that he has to attend at least 125 people but no more than 300 people each day, and the costs for accounting and the consolation gifts for families are minimized.

In this paper, our research team presents a solution to the Santa’s Workshop 2019 Tour problem using the Simulated Annealing approach. We have explored various implementations of gene generation and selection, mutations, and crossover. We fine-tuned these hyperparameters and have come up with the most appropriate model for the described problem.

Although we have successfully implemented the Simulated Annealing algorithm and have fine tuned it; due to the lack of time, decent programming skills, and computing power we have not achieved a successful result. Our best result has the 8.9 million loss, while the most quality solutions have a score of 66 thousands.

Despite this setback, the team plans to conduct more in-depth research on the dataset and find better and optimized ways to generate and choose solutions in future works. The Santa's Workshop 2019 Tour problem presents a complex challenge that requires unusual solutions and advanced computational techniques.

VIII. References

[1] John H. Holland, Adaptation in Natural and Artificial Systems

[2] S. Kirkpatrick, C. D. Gelatt, Jr,. M. P. Vecchi, “Optimization by Simulated Annealing”, *Science,* vol. 220, p. 671

[3] Y. Li, X. Li, Y. Liu, “A Hybrid Simulated Annealing Algorithm for Solving the Vehicle Routing Problem with Time Windows”, Applied Soft Computing, p.20764

[4] S. Kavitha, S. Arumugam, “A SImulated Annealing Algorithm for the Capacitated Vehicle Routing Problem”, International Journal of Industrial Engineering Computations

[5] https://www.kaggle.com/competitions/santa-workshop-tour-2019/overview

[6] https://github.com/balian729/NIC2023_Santas_workshop_tour