**Solving the Traveling Salesman Problem using a Hybrid Approach Combining Generative Adversarial Networks and Genetic Algorithms**

by

**Zijun Mei**

Department of Computer Science

College of Information Science and Technology

University of Nebraska at Omaha

Omaha, NE 68182-0500

Phone Number: 4025106076

Email address: zmei@unomaha.edu

For

**CSCI 8910, MASTER OF SCIENCE CAPSTONE**

Spring, 2023

Date: Thursday, March 09, 2023

**Abstract**

The traveling salesman problem (TSP) is a famous problem in both mathematical and graph theory where the goal is to find the shortest path visiting a set of cities and returning to the starting city. The current state-of-the-art solutions for TSP include exact algorithms, heuristics, and machine learning algorithms. However, it is possible for improvement in terms of solution quality and time-complexity. This research project aims to explore a hybrid approach that combines Generative Adversarial Networks (GANs) and Genetic Algorithms (GA) to solve the Traveling Salesperson Problem (TSP). The research will involve designing and implementing the hybrid approach and evaluating its performance through experiments. The evaluation will be based on several metrics, including the solution quality, computation time, and convergence rate. The research result will compare to previous state-of-the-art techniques and published in relevant scientific journals.

Table of Contents

[I. Introduction 1](#_Toc127213372)

[II. Overview 4](#_Toc127213373)

[III. Techniques 6](#_Toc127213374)

[IV. Approach 8](#_Toc127213375)

[V. Work Plan 11](#_Toc127213376)

[VI. Mitigations 13](#_Toc127213377)

[VII. Summary 14](#_Toc127213378)

[References 15](#_Toc127213379)

# I. Introduction

**I.1. Problem**

The problem to be researched in this project is the Traveling Salesman Problem (TSP). The TSP is about finding the shortest possible route to visit a set of cities and return to the starting point, visiting each city exactly once. In this project, I will propose a hybrid approach combining a generative adversarial network (GAN) and a genetic algorithm (GA) for solving the TSP.

**I.2. Motivations**

The Traveling Salesman Problem (TSP) is one of a series of combinatorial optimization problem(COP) that has both theoretical and practical significance. It is an NP-hard problem so that there is no polynomial time algorithm that can find the optimal solution. The study of the TSP led to many important contributions to the field of optimization and computer science more broadly. In real-word perspective, TSP can be equated with many practical problems. For example, TSP can be used to model the problem of routing vehicles, such as delivery trucks or taxies, to a set of customers , cargo, or locations. It can also be used on network design such as designing the optimal route for a pipeline or a communication network. Therefore, the study of the TSP is important both from a theoretical and practical perspective.

**I.3. Significance**

As I mentioned, the TSP is a challenging optimization problem, and the development of new and more effective algorithms for solving the TSP can have a significant impact on the field of optimization and a wide range of real-world applications. Finding a new approach of TSP can lead to the development of new optimization methods that can be applied to other combinatorial optimization problems, as well as the improvement of existing optimization methods.

Existing heuristic methods for solving the TSP, such as 2-opt and Lin-Kernighan can be effective in some cases but may not be effective in others. For example, these methods can be slow for large TSP instances or may not be able to find the optimal solution for instances with many local optima.

In this project, I will try to use a hybrid approach, which combines a generative adversarial network (GAN) and a genetic algorithm (GA), to solve TSP. The whole approach is divided into two steps: 1) use the GAN to generate high-quality candidate solutions to the TSP, and 2) use the GA to optimize the candidate solutions generated by the GAN, exploring the search space, and finding improved solutions. The hybrid approach combines the strengths of both the GAN and the GA, it can be more robust to different TSP instances and can work well for a wide range of instances by taking advantage of the ability of the GAN to generate high-quality candidate solutions and the ability of the GA to optimize these solutions.

**1.4. Challenges**

As I mentioned, optimizing the TSP, despite being a classic NP-hard optimization problem, is not a trivial problem and not a problem unsolvable because there are so many researchers have made significant contributions, such as such as branch and bound, branch and cut, and many heuristics, to the field of computer science and operations research in their efforts to solve the TSP.

There are several difficulties associated with the research: Choosing the appropriate network architecture and hyperparameters for the GAN can be challenging and requires careful consideration. Furthermore, training a GAN can be difficult, as the training process can be unstable and prone to convergence problems. Comparing the results of the hybrid approach with those of traditional optimization methods can also be challenging, as it requires a standardized evaluation procedure and a large number of TSP instances to ensure robust results. Since TSP is NP-hard and finding the optimal solution is difficult, it could be very difficult to verify the output solution generated by hybrid approach.

**I.5. Objectives**

The goal of this research is to develop a hybrid approach that combines the strengths of a GAN and a GA for solving the TSP. This will involve the implementation of a GAN to generate candidate solutions and a GA to optimize these solutions. I will conduct an experiment to verify the performance of the hybrid approach. The performance of the hybrid approach will be compared to that of traditional optimization methods, such as 2-opt and Lin-Kernighan. The results will be analyzed in terms of the quality of the solutions generated, the computational time required to generate the solutions, and the robustness of the approach to different TSP instances.

# II. Overview

**II.1. History of the problem**

The Traveling Salesman Problem (TSP) is a well-known mathematical optimization problem that was first introduced by mathematician W.R. Hamilton in 1832. The problem is to find the shortest possible route that visits a given set of cities and returns to the starting city. It has been widely studied and has various real-world applications, such as vehicle routing, logistics, and DNA sequencing. Over the years, many different approaches have been proposed to solve the TSP, including exact algorithms, approximation algorithms, and heuristics. With the advent of deep learning, researchers have started to explore the use of neural networks to tackle the TSP.

Previous work on the TSP has focused on developing exact algorithms, heuristics, and approximation algorithms to solve the problem efficiently. Some of the early methods include the nearest neighbor algorithm, the Christofides algorithm, and the Lin-Kernighan heuristic [Dijkstra 1965, Christofides 1976, Lin and Kernighan 1973]. George Dantzig is considered one of the pioneers of linear programming, and his work on the TSP laid the foundations for many of the algorithms and methods used today. In the 1970s, Shen Lin and Brian Kernighan developed a heuristic algorithm for solving the TSP, known as the Lin-Kernighan algorithm, which remains one of the most widely used algorithms for solving the TSP. In the 1990s, researchers developed a number of approximation algorithms for solving the TSP, including the Christofides algorithm, the Held-Karp algorithm, and the Arora-Rao algorithm. After that, genetic algorithms, simulated annealing, and ant colony optimization have been developed for solving the TSP. These methods have proven to be effective for solving the TSP and have been applied to a variety of real-world problems.

**II.2. State of the art**

More recently, researchers have started to explore the use of machine learning methods to solve the TSP. Bengio et. al ‘s article provides valuable insights into the current state of the field and the direction of future research [Bengio et. al 2021].

Vinyals et. al proposed an approach new neural architecture which can be used to solve TSP [Vinyals et. al 2015]. The authors propose the use of a Pointer Network, a type of recurrent neural network (RNN), for generating candidate solutions to the TSP and for optimizing these solutions. The Pointer Network architecture uses an attention mechanism to focus on different parts of the input sequence, allowing the network to make more informed decisions about the solution.

Wu et. al proposed use of a neural network to learn improvement heuristics for solving routing problems and demonstrate the effectiveness of the approach on a range of benchmark problems, including the TSP [Wu et. al 2022]. The authors show that the learned heuristics can be used to improve the solutions generated by traditional optimization methods, such as local search and simulated annealing, and that the approach outperforms traditional methods in terms of solution quality and computational efficiency.

The authors Chaitanya K. Joshi et. al introduce a deep learning method for resolving the Traveling Salesman Problem (TSP) and explores the challenges of generalization in this context [Joshi et. al 2022]. The authors demonstrate that conventional deep learning methods, including RNNs and CNNs, have difficulty generalizing to new instances of the TSP, even when trained on extensive datasets. The authors propose a new approach, based on graph neural networks (GNNs), that is better suited to solving the TSP and demonstrate the effectiveness of the approach on a range of benchmark problems.

Zhang et. al introduces a deep reinforcement learning approach to tackle the TSP and demonstrates the effectiveness of the approach on a range of benchmark problems [Zhang et.al 2020]. The authors approach the TSP as a problem of making sequential decisions and employ reinforcement learning techniques to obtain an optimal solution. The approach is based on a deep neural network that maps a state representation of the TSP to an action, which is then used to update the solution.

# III. Techniques

**III.1. Principles, Concepts, and Theoretical Foundations of the research problem**

The TSP is depicted as a graph G = (V, E), with V being the set of cities represented as nodes, and E representing the edges or path among these cities. The aim of TSP is to identify the shortest Hamiltonian cycle, which indicates that is visits each node can only be visited once and it should be back to the start node in the end of tour, in the graph.

Generative Adversarial Networks (GANs) are a deep neural network architecture utilized for generating models [Goodfellow et.al 2014]. The GAN architecture comprises of two crucial parts - a generator network and a discriminator network. The generator network is tasked with creating new samples from a noise distribution, while the discriminator network is trained to differentiate between the generated samples and actual samples from the desired distribution. In the context of solving TSP with GANs, the desired distribution is that of valid tours, which are sequences of cities that visit each city only once. The generator network is trained to produce candidate tours from a noise distribution, while the discriminator network is trained to distinguish between the generated tours and actual tours from the desired distribution.

The purpose of the generator network is to produce tours that closely resemble the target distribution of valid tours. Conversely, the objective of the discriminator network is to accurately differentiate between the created tours and the real tours from the target distribution. The training process can be viewed as a minimax optimization problem where the generator network aims to minimize a loss function that evaluates the dissimilarity between the generated tours and the target distribution, while the discriminator network strives to maximize the same loss function. The loss function employed in this scenario is commonly the cross-entropy loss between the predicted probabilities of the discriminator and the actual labels.

**III.2. Techniques that have been used by other researchers for the research problem**

**Exact algorithms**

Exact algorithms are a category of algorithms that are capable of finding an optimal solution to the TSP within a finite time frame. Some examples of exact algorithms include branch-and-bound and branch-and-cut algorithms as well as dynamic programming algorithms. They are quite expensive and hard to apply on solving real-word problem.

**Approximation algorithms**

Approximation algorithms is a series of algorithms that provide a solution to the TSP that is close to the optimal solution, but not necessarily optimal. Approximation algorithms for the TSP include, among others, the Christofides algorithm and the greedy algorithm.

**Meta-heuristic algorithms**

Meta-heuristic algorithms are a series of algorithms that use heuristics to find a good solution to the TSP, but do not guarantee an optimal solution. Meta-heuristic algorithms for the TSP include simulated annealing, tabu search, and genetic algorithms, to name a few.

**Deep learning**

These are algorithms that use deep neural networks to solve the TSP. Deep learning algorithms that have been applied to solve the TSP include the use of recurrent neural networks (RNNs), convolutional neural networks (CNNs), and graph neural networks (GNNs). These algorithms exhibit great versatility, as they are able to be trained on diverse data and can be utilized to address a broad spectrum of combinatorial optimization problems, such as the TSP. Furthermore, they can handle large amounts of data, making them well-suited for solving large TSP problems.

**Hybrid approach**

These are algorithms that combine two or more of the above techniques to solve the TSP. In this project, I will combine GAN and GA as a hybrid algorithm for the TSP. Therefore, this approach theoretically not only keep the feature of solving large TSP problems from GAN but also improved the performance of individual optimization by using GA.

**III.3. Relevant technologies that would be useful to this research**

Since this research is based on the GAN, deep learning frameworks such as TensorFlow and PyTorch would be useful for developing and training deep learning models for the TSP. other useful technologies could be Matplotlib and Seaborn for visualizing the solutions to the TSP and evaluating the performance of the hybrid approach.

# IV. Approach

**IV.1. Methodologies I am going to apply in this research**

First of all, I will conduct a literature review of existing research on the TSP and the hybrid approach to solving the TSP to gain an understanding of the state-of-the-art and identify gaps in the existing research. Second, I will define the generator and discriminator networks and train the GAN using the TSPLIB dataset. Then I will we will implement GA algorithm to further improve the candidate solutions generated by the GAN. After that, I will design an experiment to evaluate this hybrid approach. Finally, I will visualize the result of the experiment.

**IV.2. Techniques for addressing the problem**

In this research, deep learning algorithms will play a key role in the hybrid approach for solving the Traveling Salesman Problem (TSP). GAN will be used to generate candidate solutions to the TSP, and these solutions will be used to guide the search performed by genetic algorithms. Based on Chaitanya K. Joshi et.al ‘s research, GAN have been shown to be effective at generating high-quality candidate solutions to combinatorial optimization problems (Joshi, 2022). Therefore, this hybrid approach is not totally new but based on Chaitanya K. Joshi et.al ‘s research idea to further explore the new approaches.

The proposed method in this research employs a hybrid approach that utilizes a Generative Adversarial Network (GAN) and a Genetic Algorithm (GA) to solve the Traveling Salesman Problem (TSP). The GAN is tasked with generating candidate solutions for the TSP, which are then optimized by the GA. The GAN consists of two components: a generator network and a discriminator network. The generator network is trained to produce candidate solutions that are indistinguishable from actual solutions, while the discriminator network is trained to differentiate between true solutions and generated candidate solutions. The candidate solutions generated by the GAN are then optimized using the GA to produce high-quality solutions to the TSP.

The GA is a heuristic optimization algorithm that emulates the process of natural selection. In the GA, solutions are represented as chromosomes and a population of chromosomes is evolved through operations such as selection, crossover, and mutation. The effectiveness of each chromosome is determined by a fitness function that evaluates the quality of the solution represented by the chromosome. The GA runs for a specified number of generations, and the optimal chromosome in the final population is considered the solution to the TSP.

**IV.3. Steps Involved in this Research Project**

A set of TSP instances will need to be selected for use in the experiments. This will likely involve using a widely used benchmark dataset, my idea so far is to use TSPLIB, that includes a variety of TSP instances with varying sizes and characteristics. Then, I will build a GANs model based on TensorFlow framework. The GAN model (trained) and GA components of the hybrid approach will need to be implemented and integrated into a single program. This will involve training the GAN on the selected TSP instances, and implementing the operations of selection, crossover, and mutation in the GA. This program will be the experimental program.

The experiments will be controlling for factors such as the size of the population, the number of generations, and the hyperparameters of the GAN and GA components of the hybrid approach. The experiments will then be executed, using the program to solve each TSP instance in the dataset. The results of the experiments will be collected, including the solutions generated by the hybrid approach and their corresponding objective values. The results of the experiments will be compared with Tabu, 2-opt and Lin-Kernighan. I will also build a program for these algorithms as control programs.

**IV.4. How the results (outcomes) of the research will be demonstrated?**

There are two main experiments result I will demonstrate: the quality of the solution and the time cost of the computation. The quality of solution indicates the total distance of the path. The time cost of computation indicates the total time of getting the output from the experimental program and control programs. The comparison result and solution generated will be visualized in form of table and plot.

**IV.5. Evaluation, Analysis, and Comparison of Research Outcomes with Previous Studies**

The purpose of this project is to find an approach that can provide a high-quality solution of shortest path and meaningfully reduced the time cost. Therefore, keep the solution path be shortest and the time lowest will be the main factor affect the evaluation result. The Convergence rate evaluation will also be observed to compare with the previous work.

**IV.6. Required resources for the research project**

**Hardware:**

A computer with a high-performance GPU for training deep learning models.(or Google Cloud Computation)

**Software:**

IDE: VScode

A deep learning framework: TensorFlow

A programming language such as Python for implementing the algorithms.

A data visualization tool such as Matplotlib for visualizing the results.

**Dataset**: TSPLIB

# V. Work Plan

**V.1 Steps to be Undertaken in this Study**

1. Literature review
2. Program design: Design of the hybrid approach combining GAN and genetic algorithm (GA) for solving the TSP. This will use TensorFlow framework.
3. Train the model
4. Adjust the parameter avoid training results that do not fall into local optima or any other negative situation.
5. Design the three control programs by using Tabu, 2-opt and Lin-Kernighan algorithms.
6. Conduct the experiment and collect the experiment results.
7. Visualize the experiment results. (Matlib or other visualization tool will be used)
8. Evaluate the experiment results.

**V.2. The schedule of the research**

Table 1: The research schedule

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Order | Dates | Task | Prerequisites | Expected results |
| 1 | From 02/13/2023 to 02/19/2023 | Literature review | UNO library resource, Google scholar | Gain an understanding of the state-of-the-art and identify gaps in the existing research |
| 2 | From 02/20/2023 to 02/26/2023 | Program design | Python, TensorFlow, GAN, GA knowledge, VSCODE | Gain a initial program that integrates the GAN architecture and GA algorithm. |
| 3 | From 02/27/2023 to 03/12/2023 | Train the model and adjust the parameter. | Only need time to change parameters or hyper-parameters | A well-trained model |
| 4 | From 03/13/2023 to 03/19/2023 | Design the three control programs | Python, and knowledge of three algorithms | Three control programs |
| 5 | From 03/20/2023 to 04/12/2023 | Conduct, Visualize, and Evaluate the experiment | Knowledge of visualization tool or matlib | The experiment result |

# VI. Mitigations

**VI.1. Expected Challenges and Concerns**

In my opinion, whether a GAN will be able to generate high-quality solutions to the TSP will depend on several factors, such as the quality of the data, the choice of network architecture, the training procedure, the choice of hyperparameters, and the specific problem instance. Not being able to complete the set tasks as expected can also lead to a tight project timeline, which is a highly likely occurrence. Only by strictly following the schedule and seeking help from the advisor promptly when facing difficulties, can help reduce potential problems.

**VI.2. Restrictions and limitations of the research**

Since there are large amounts of high-quality training data, the computational resources required to train complex models, and the need for specialized expertise in deep learning and optimization algorithms. Additionally, the choice of network architecture, training procedure, and hyperparameters can significantly impact the performance of the models, and care must be taken to choose these carefully. Finally, it is important to carefully evaluate the performance of the models using appropriate metrics, and to be aware of the limitations of the models in terms of generalization and robustness to real-world problems.

# VII. Summary

This research project aims to address the Traveling Salesperson Problem (TSP) and to find an effective solution for it. This project proposes a hybrid approach under the guidance of previous research to generate higher quality optimization solutions for TSP in a shorter time. The use of deep learning algorithms for finding optimal solutions for the TSP has always been a hot area of research. Wu et al. improved the solutions generated by traditional optimization methods using neural networks. Joshi et.al’s article provides insight into the current state of the field and the challenges that need to be addressed in order to improve TSP solvers. Understanding these challenges is crucial for developing effective and innovative solutions to the TSP. Vinyals et. al’s research helped me with understanding how to use attention mechanisms to solve TSP. The study by Zhang et.al sheds light on the efficiency of utilizing deep reinforcement learning for addressing combinatorial optimization challenges that come with additional restrictions and limitations, making it valuable for my research.

The hybrid approach will combine the GAN and GA to solve TSP. The well-trained GAN model can generate a candidate solution, and the GA will improve the candidate solution as the result. The expect results of this approach is to generate a high quality, low time cost solution. The innovation of this research is to combine two techniques to solve TSP. This approach are expected to improved efficiency and accuracy of the solution generation process, compared to the existing methods.

# References

Joshi, Chaitanya K., et al. “Learning TSP Requires Rethinking Generalization.” Schloss Dagstuhl - Leibniz-Zentrum Für Informatik, Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021, doi:10.4230/LIPICS.CP.2021.33.

Bengio, Yoshua, et al. “Machine Learning for Combinatorial Optimization: A Methodological Tour d’horizon.” European Journal of Operational Research, no. 2, Elsevier BV, Apr. 2021, pp. 405–21. Crossref, doi:10.1016/j.ejor.2020.07.063.

Vinyals, Oriol & Fortunato, Meire & Jaitly, Navdeep. (2015). Pointer Networks. Advances in Neural Information Processing Systems. 28.

Zhang, Rongkai, et al. “Deep Reinforcement Learning for Traveling Salesman Problem with Time Windows and Rejections.” 2020 International Joint Conference on Neural Networks (IJCNN), IEEE, July 2020. Crossref, doi:10.1109/ijcnn48605.2020.9207026.

Wu, Yaoxin, et al. “Learning Improvement Heuristics for Solving Routing Problems.” *IEEE Transactions on Neural Networks and Learning Systems*, no. 9, Institute of Electrical and Electronics Engineers (IEEE), Sept. 2022, pp. 5057–69. *Crossref*, doi:10.1109/tnnls.2021.3068828.

Y. Wu, W. Song, Z. Cao, J. Zhang and A. Lim, "Learning Improvement Heuristics for Solving Routing Problems," in IEEE Transactions on Neural Networks and Learning Systems, vol. 33, no. 9, pp. 5057-5069, Sept. 2022, doi: 10.1109/TNNLS.2021.3068828.

Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2 (NIPS'14). MIT Press, Cambridge, MA, USA, 2672–2680.

Lin, S., & Kernighan, B.W. (1973). An Effective Heuristic Algorithm for the Traveling- Salesman Problem. Oper. Res., 21, 498-516.

Christofides, N. (1976). Worst-Case Analysis of a New Heuristic for the Travelling Salesman Problem. Oper. Res. Forum, 3.

E. W. Dijkstra. 1965. Solution of a problem in concurrent programming control. Commun. ACM 8, 9 (Sept. 1965), 569. https://doi.org/10.1145/365559.365617