Local Improvement for Large-scale Traveling Salesman Problem

Eungyeol Lee
Advisor: Heung-No Lee
School of Electrical Engineering and Computer Science, Gwangju Institute of Science and Technology

Abstact

Neural network-based approaches are promising methods for quickly finding near-optimal approximate solutions to combinatorial optimization problems. Combinatorial optimization problems, such as logistics optimization, vehicle routing, and scheduling, frequently arise in real-world applications and they are challenging due to the vast number of possible combinations. The Traveling Salesman Problem (TSP) is a representative example of combinatorial optimization problems, and various neural network-based methods have been studied to efficiently obtain high-quality approximate solutions. In this paper, I proposes a Local Improvement(LI) methods including neural network training strategy and model inference heuristics to effectively search for near-optimal solutions in large-scale TSPs. The proposed methods demonstrate the ability to obtain high-quality approximate solutions for large-scale TSP instances in real-time. This demonstrates that neural network-based methods are effective even for large-scale combinatorial optimization problems within short time.

Introduction

The Traveling Salesman Problem (TSP[1]) is a classic problem of a Combinatorial Optimization (CO), where the goal is to find the shortest possible route that visits all *n* nodes exactly once and returns to the starting point. Since TSP is NP-hard, solving the TSP requires exploring a vast number of possible combinations to find the optimal solution. Traditional approaches include linear programming, while heuristic methods are also employed to find near-optimal solutions more efficiently. Recently, neural combinatorial optimization has demonstrated remarkable success in solving TSP and other combinatorial problems, producing high-quality approximate solutions with significantly reduced computation time.

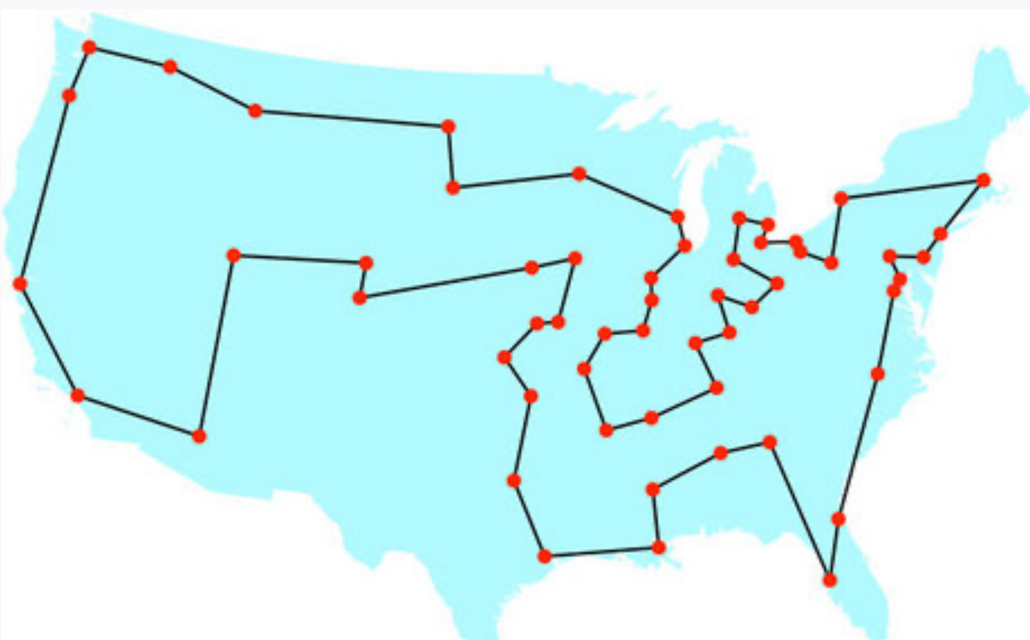

Fig 1. Example of TSP and exact solution.

For large-scale Traveling Salesman Problem, the search space becomes prohibitively vast, and it makes supervised learning approaches difficult to apply. As a result, research has focused on improving solution accuracy through iterative methods. These methods begin with an initial solution and repeatedly search for better ones, aiming to get closer to the global optimum. GLOP[3] is a representative model that improves solutions by decomposing large-scale TSPs into smaller sub-problems, and it achieves high-quality results with remarkable speed. However, the approximate solutions produced by GLOP often remain limitations that solutions fall into local optima. Building on GLOP's promising performance, this paper proposes a **Local Improvement** that aims to overcome these limitations and generate even higher-quality solutions more efficiently.

[1] TSP : Traveling Salesman Problem is a problem that finding a N combinations route which can minimize the length of cycle.
[2] SHPP : Shortest Hamiltonian Path Problem is a problem that finding a N combinations route which can minimize the length of route.
[3] GLOP: Learning Global Partition and Local Construction for Solving Large-scale Routing Problems in Real-time, AAAI 2024
[4] LEHD : Neural Combinatorial Optimization with Heavy Decoder: Toward Large Scale Generalization, NeurIPS 2023.

Methods

Local Improvement. Local Improvement (LI) is an algorithm for solving the TSP[1] by dividing an existing solution into several smaller subproblems and re-solving them to improve the overall solution. The core idea of LI is to apply a divide-and-conquer approach to TSP. Given a TSP instance, an initial solution is first quickly generated using Random Insertion heuristic. Then, the following process is repeated. The current solution is divided into *d* segments, and *d* new SHPP[2] subproblems of size $\left\lfloor \frac{N}{d} \right\rfloor$ are solved by LEHD SHPP solver neural network model. If any newly obtained solution is shorter than current solution, it is replaced. Finally, the solutions of all SHPP segments are merged to form an updated solution to the entire TSP[1] instance.

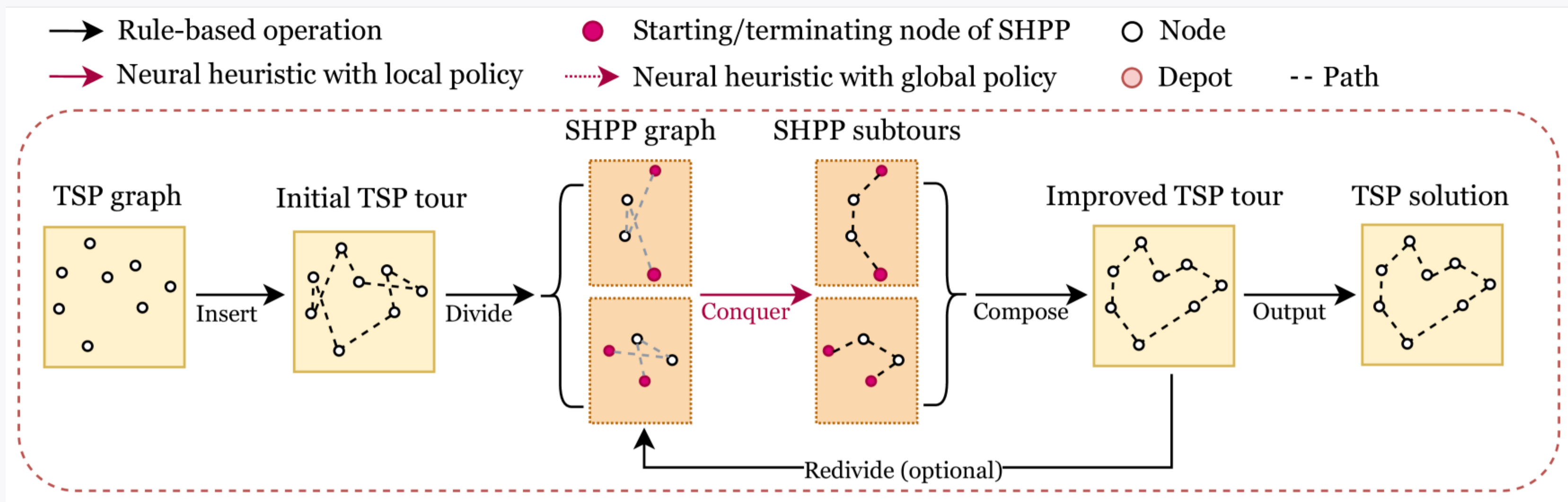

Fig 2. Overall progress of Local Improvement(LI).

SHPP Solver. The LEHD[4] SHPP[2] policy adopts the model architecture of Light Encoder Heavy Decoder (LEHD), a Neural Combinatorial Optimization (NCO) solver for solving TSP[1]. To train LEHD, a dataset with optimal solution labels is randomly partitioned to create SHPP sub-problems along with their corresponding labels. Since a sub-route of an optimal solution is also optimal, the sampled sub-problems retain valid optimal labels. Supervised learning is then performed using the cross-entropy loss function. In addition, for each SHPP instance, both the forward and backward SHPPs are considered, and the higher of the two loss values is used as the final loss value.

Inference Strategy. To make the Local Improvement (LI) method more efficient during inference, I propose **Sigmoid Scheduling**, a dynamic strategy for controlling the number of partitions d when applying LI. This method gradually reduces the d according to a sigmoid function as the LI of large d begins to converge. It makes solver faster because it initially addresses the global problem and gradually shifts to solving local problem.

$$d(t; d_{max}, d_{min}, \alpha) = (d_{max} - d_{min})sigmoid(-t + \alpha) + d_{min}$$

This scheduling can be tuned via hyper-parameters. d_{max} denotes the maximum number of partitions used in d-LI, while d_{min} represents the minimum. he parameter α determines how many iterations the maximum partition size is applied before the reduction begins. These values can be empirically tuned based on the problem size.

Experiments

Method	Type	TSP-1000			TSP-10000		
		Obj.	Gap	Time	Obj.	Gap	Time
Concorde	Exact	23.12	0.00%	2.21h	N/A	N/A	N/A
LKH-3	Heuristic	23.12	0.00%	4.41h	71.77	0.00%	6h
Random Insertion	Heuristic	26.15	13.1%	1s	81.75%	13.9%	3s
H-TSP	RL	24.66	6.62%	3.1m	55.31	8.39%	2.4m
LEHD	SL	23.84	3.11%	1.64m	91.33	27.2%	3.0h
Att-GCN	SL+MCTS	23.67	2.37%	15.2m	74.50	3.80%	20.9m
DIMES	RL+MCTS	23.73	2.64%	6.9m	74.63	4.0%	29.4m
distMCTS	MCTS	23.63	2.20%	3.3m	74.03	3.13%	16.8m
DIFUSCO	SL+MCTS	**23.38**	1.12%	13.7m	**73.62**	2.58%	1.07h
SO	RL+H	23.77	2.80%	**24s**	74.32	3.55%	6.8m
GLOP	RL+H	23.84	3.11%	3.0m	75.29	4.90%	1.8m
LI	SL+H	23.51	1.73%	4.1m	74.00	3.11%	9.6m
LI_{fast}	SL+H	23.67	2.38%	1.1m	74.75	4.15%	**1.0m**

Table 1. Experimental Results of Local Improvement(LI).

The LI_{fast} solver achieved a result of **23.51** on TSP1000 in **4.1 minutes**, and **74.75** on TSP10000 in **1.0 minute**. In comparison, the LI solver achieved **23.67** on TSP1000 in **1.1 minutes**, and **74.00** on TSP10000 in **9.6 minutes**.