Thesis for Bachelor's Degree

Local Improvement for Large-scale Traveling Salesman Problem

Eungyeol Lee

Department of Electrical Engineering and Computer Science

Gwangju Institute of Science and Technology

2025

학 사 학 위 논 문

대규모 외판원 순회 문제를 위한 지역 개선

이은결

전 기 전 자 컴 퓨 터 공 학 부

광 주 과 학 기 술 원

2025

Local Improvement for Large-scale Traveling Salesman Problem

Advisor: Prof. Heungno Lee

by

Eungyeol Lee

Department of Electrical Engineering and Computer Science

Gwangju Institute of Science and Technology

A thesis submitted to the faculty of the Gwangju Institute of Science and Technology in partial fulfillment of the requirements for the degree of Bachelor of Science in the Electrical Engineering and Computer Science Concentration

Gwangju, Republic of Korea

May 01, 2025

Approved by

Professor Prof. Heungno Lee

Committee Chair

Local Improvement for Large-scale Traveling Salesman Problem

Eungyeol Lee

Accepted in partial fulfillment of the requirements for

the degree of Bachelor of Science

May 01, 2025

Committee Chair _____

Prof. first prof. Heungno Lee

Committee Member _____

Prof. second prof. Chang Wook Ahn

BS/EC 20215155 Eungyeol Lee. Local Improvement for Large-scale Traveling Salesman Problem. Department of Electrical Engineering and Computer Science. 2025. 22p. Advisor: Prof. Prof. Heungno Lee.

Abstract

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.

©2025
Eungyeol Lee
ALL RIGHTS RESERVED

BS/EC 20215155 이은결. 대규모 외판원 순회 문제를 위한 지역 개선. 전기전자컴퓨터공학부. 2025. 22p. 지도교수: 이흥노.

국 문 요 약

신경망 기반 방법은 조합 최적화의 최적에 가까운 근사해를 빠르게 구하기 위한 유망한 방법이다. 물류 최적화, 차량 경로 문제, 일정 할당 문제 등과 같은 조합 최적화 문제는 실생활에서 자주 볼 수 있는 조합할 수 있는 경우의 수가 굉장히 많아 최적해를 구하기 어려운 문제이다. 외판원 순회 문제(Traveling Salesman Problem)은 대표적인 조합 최적화 문제로 효율적으로 고품질의 근사해를 얻기 위한 신경망 기반의 연구가 진행되었다. 본 논문은 대규모 외판원 순회 문제에서 효과적으로 최적에 가까운 근사해를 탐색하는 신경망 기반 모델 훈련 전략 및 모델 추론 휴리스틱 방법을 이용한 지역 개선(Local Improvement) 방법을 제시한다. 소개한 방법들을 통해 대규모 외판원 순회 문제에서 실시간에 적용 가능할 정도로 적은 시간 내 고품질의 근사해를 얻는 성능을 보였다. 이는 짧은 시간 내의 굉장히 복잡한 조합 최적화 문제에서도 신경망 기반 방법이 효과적임을 입증한다.

ⓒ2025
이 은 결
ALL RIGHTS RESERVED

Contents

Abstract (English) **i**

Abstract (Korean) **iii**

List of Contents **iv**

List of Tables **vi**

List of Figures **vii**

1 Introduction **1**

2 Related Works **4**

2.1 Supervised TSP NCO Solvers 4

2.2 Reinforcement TSP NCO Solvers 5

2.3 Large-scale TSP NCO Solvers 5

3 Preliminary **7**

3.1 Definition of TSP 7

3.2 Definition of SHPP 8

4 Methodology **9**

4.1 Local Improvement 9

4.1.1 Initialization 10

4.1.2 d-Decomposition 10

4.1.3 Normalization 10

4.1.4 Solve SHPP 10

4.1.5 Composition 11

4.2 LEHD SHPP Policy 11

5 Experiments **13**

5.1 LI Model Training and Inference 13

5.1.1 Model Training 13

5.1.2 Dataset 13
5.1.3 Inference Strategy 13
5.2 Experimental Setup 14
5.2.1 Baseline Model 14
5.2.2 Matrics 15
5.2.3 Hardware and Software 15
5.3 Experimental Results 15

6 Conclusion and Limitation **17**

Summary **18**

References **19**

List of Tables

5.1 Experimental Result for TSP1000, TSP10000 16

List of Figures

1.1 TSP와 이 문제의 최적해 예시 2

4.1 LI 알고리즘 [1]. 9

4.2 LEHD 훈련 매커니즘 11

Chapter 1

Introduction

외판원 순회 문제(Traveling Salesman Problem, TSP)는 대표적인 조합 최적화(Combinatorial Optimization, CO) 문제로, n개의 노드를 모두 잇는 순회 경로의 길이를 최소화하는 경로를 찾는 문제이다. 본 문제는 NP-hard이기 때문에 최적해를 찾기 위해 수많은 경우의 수를 탐색해야 한다. 이 문제를 해결하기 위한 전통적인 방법으로 선형 계획법이 있다. 또한, 최적해에 가까운 근사해를 찾기 위해 휴리스틱 방법을 사용하기도 한다. k-Opt, Ant Colony Optimization, Genetic Algorithm은 TSP를 위한 전통적인 휴리스틱 기법들이다 [2]. 두 방법은 모두 장단점을 갖고 있는데, 선형 계획법 기반 솔버는 해답의 품질이 좋지만 시간이 매우 오래 걸린다. 휴리스틱 솔버는 빠른 시간 내에 해답을 얻을 수 있지만 품질이 좋지 않거나 시간이 매우 오래 걸린다. 이들은 모두 빠른 시간 내에 근사해를 도출할 수 있다는 장점이 있지만 문제 크기가 커질 수록 근사해의 품질이 매우 떨어진다. LKH3은 Traveling Salesman Problem에서 선형 계획법을 제외한 솔버중 가장 고품질의 근사해를 구하는 휴리스틱이다 [3]. 그러나 선형 계획법 솔버만큼은 아니지만 시간이 매우 오래 걸린다는 단점이 있다.

이처럼 선형 계획법과 휴리스틱 기반 방법은 문제 크기가 커질수록 두드러진다. 선형 계획법의 경우, 문제를 푸는 데에 걸리는 시간이 기하 급수적으로 늘어나며, 휴리스틱의 경우, 최적해와 매우 멀리 떨어진 근사해만 얻어낸다. 이러한 문제를 해결하기 위해 신경망 조합 최적화(Neural Combinatorial Optimization, NCO) 기반 솔버들이 연구되었다 [4].

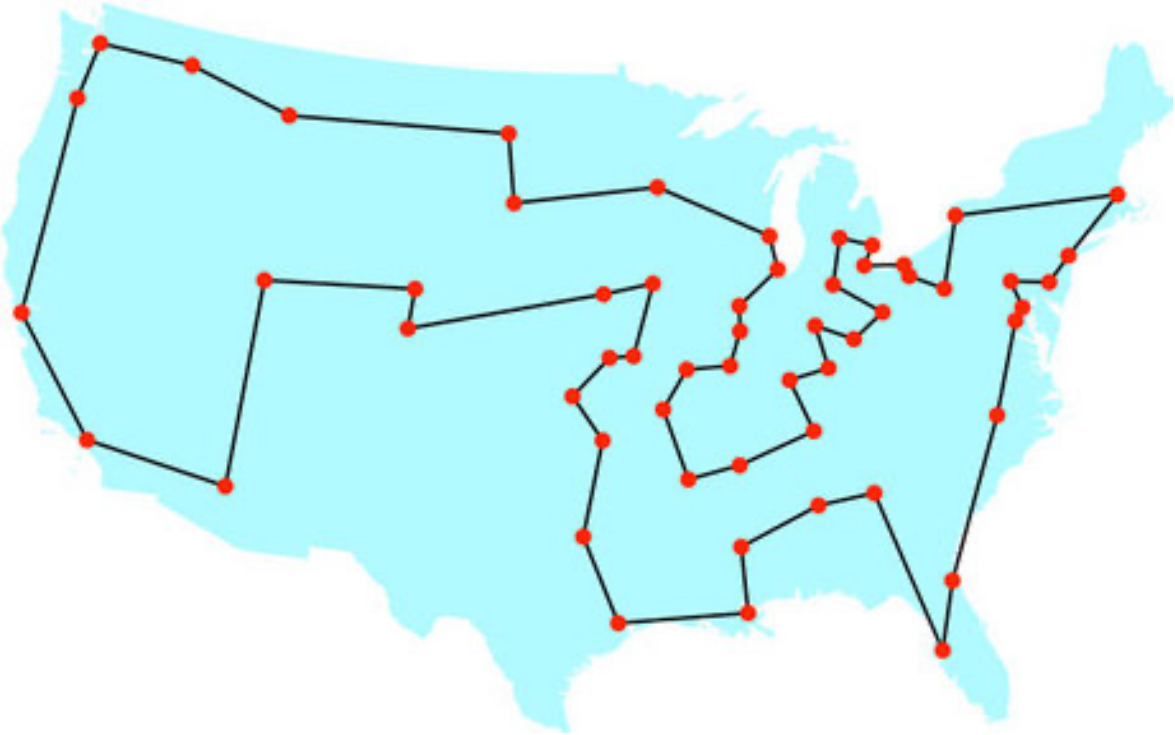

Figure 1.1: TSP와 이 문제의 최적해 예시

신경망 조합 최적화는 외판원 순회 문제를 포함한 여러 조합 최적화 문제를 훌륭하게 풀어냈다. 계산 시간을 매우 줄이면서도 최적해에 가까운 높은 품질의 근사해를 얻어낼 수 있다. NCO는 크게 세 가지 방법으로 분류할 수 있다. 대규모 레이블이 있는 데이터셋을 통한 지도학습 기반 방법들과 레이블이 존재하지 않는 데이터셋을 통한 강화학습 기반 방법, 그리고 초기해를 구한 후 해를 점점 최적해에 가깝게 정확도를 발전시키는 방법들이 제안되었다.

대규모 외판원 순회 문제는 탐색 공간이 너무 방대하여 지도학습으로 학습시키기 어렵다. [5]. 그렇기에 대규모 문제에서는 해의 정확도를 발전시키는 방법 기반으로 연구되었다. 이 방법은 초기해를 구한 후, 더 나은 해를 탐색 및 선택하면서 최적해에 다가가는 것이다. GLOP은 대규모 외판원 순회 문제의 근사해를 구하기 위한 해를 발전시키는 대표적인 모델이다 [1]. GLOP은 높은 품질의 해를 얻으면서도 매우 빠르다는 특징이 있다.

그러나, GLOP으로 얻은 근사해는 여전히 지역 최적점에서 벗어나지 못하는 경향이 크다. 이 논문은 GLOP의 유망한 성능을 기반으로 한계를 조금이나마 극복하여 품질이 좋으면서도 더 빠르게 근사해를 얻는 Local Improvement 방법을 제안한다.

본 논문의 구성은 다음과 같다. 1은 본 논문의 개요를 소개하며 2는 관련한 선행 연구를 소개한다. 3은 본 논문에서 풀고자 하는 외판원 순회 문제와 방법론에서 사용될 가장 짧은 해밀토니언 경로 문제(Shortest Hamiltionian Path Problem, SHPP)를 소개한다. 4는 이 논문에서 제안하는 LI 방법을 제시하며, 5는 본 방법을 실험적으로 성능을 검증한다. 6는 결론과 한계점을 설명한다.

본 논문을 통해 다음과 같은 기여를 하였다. 첫째, GLOP의 Attention SHPP Policy 보다 더 적합한 LEHD SHPP Policy 신경망 모델을 훈련하는 방법을 소개하며, 이를 GLOP에 적용한 Local Improvement(LI)를 제안한다. 둘째, LI를 효율적으로 적용하기 위해 실험에서 적용할 Inference Strategy인 분해 스케쥴링 방법을 제안한다. 제안한 방법으로 대규모 외판원 순회 문제에서 짧은 제한시간 내 기존보다 훌륭한 품질의 근사해를 얻을 수 있었다.

Chapter 2

Related Works

Neural Combinatorial Optimization(NCO)은 CO 문제를 풀기 위한 심층학습 방법으로 빠른 시간 내에 근사해를 구하는데 특화되어있어 촉망받는 방법이다 [6]. NCO가 목표로 삼는 것은 TSP를 포함한 Vehicle Routing Problem의 여러 변형 문제들, 배낭 문제, 스케쥴링 문제, 그래프 매칭, 그리고 그래프 색칠 등의 문제들이다. 이 섹션에서는 TSP를 위한 NCO 솔버들의 동향을 종류별로 소개한다. 지도학습 기반 방법과 강화학습 기반 방법이 있으며, 대규모 문제를 위한 방법으로 분류하였다.

2.1 Supervised TSP NCO Solvers

대규모 데이터를 지도학습으로 학습한 TSP Solver가 존재한다 [7, 8, 9, 10]. Pointer Networks [7]는 RNN을 응용하여 가장 처음으로 TSP에 적용한 지도학습 NCO 모델이다. Graph Convolution Network를 훈련시킨 방법도 존재한다 [11, 8]. 그러나 두 모델은 문제의 크기가 커지면 높은 품질의 해를 얻지 못한다는 단점이 존재한다. Light Encoder Heavy Decoder는 Transformer 구조를 활용해 이러한 한계를 극복한 지도학습 모델이다 [12, 9]. 또한 Difusco는 anisotropic graph neural network를 지도학습 시켜 유망한 결과를 보였다 [10].

2.2 Reinforcement TSP NCO Solvers

강화학습을 통해 훈련시킨 TSP NCO 솔버에 관한 연구도 많이 존재한다 [13, 14, 15, 16, 17, 18]. Attention Model은 자가회귀(Auto Regressive) NCO 솔버의 일종으로 Transformer Encoder 모델을 Policy Gradient로 훈련시켜 유망한 결과를 보여주었다 [12, 13]. 이후 본 Attention Model을 기반으로 같은 문제를 다른 형태로 바꾸거나 문제의 시작점을 여러개로 하여 탐색 공간을 넓히는 방법들이 제안되었다 [14, 15, 16]. POMO는 가장 먼저 이러한 시도를 한 모델이다. 문제를 대칭과 뒤집기를 활용해 같은 TSP를 서로 다른 8개의 문제로 만들어 학습시킨다 [14]. 또한 각 노드를 모두 출발점으로 정의하여 추론 과정에서의 성능을 올리기도 하였다. symnco는 POMO에서 영감을 받아 대칭을 비롯한 회전 형태의 문제 또한 손실함수를 통해 학습한 모델이다 [15]. Pointerformer는 POMO를 개선한 Reversible residual network를 통해 메모리 사용량을 줄인 모델로, 보다 큰 크기의 TSP에서 유망한 성능을 보였다 [16]. 이 외의 다양한 탐색을 통해 여러 해를 샘플링하면서 성능을 올린 LCP 모델도 존재한다 [17]. 혹은 효율적으로 탐색 공간을 줄이는 기법을 적용하여 학습한 BQ-NCO 모델이 제안되기도 했다 [18].

2.3 Large-scale TSP NCO Solvers

대규모 TSP의 근사해를 찾기 위한 방법들이 연구되었다 [5, 19, 20, 10, 21, 22, 1]. NCO 솔버로 대규모 TSP의 근사해를 얻어낸 첫 시도는 Monte Carlo Tree Search(MCTS)를 이용한 방법이다 [5]. 대규모 TSP에서 문제를 샘플링하여 작은 규모의 여러 문제를 만든다. 그후 작은 규모의 문제로 학습된 TSP 솔버를 활용하여 엣지 선택 확률을 MCTS로 샘플링한 기법으로, 대규모 문제에서 효과적인 성능을 보였다. DIMES는 해의 확률 분포를 샘플링한 방법으로, 대규모 TSP에서 훌륭한 성능을 보였다 [20]. 그러나, 시간이

오래 걸린다는 단점이 존재한다. 이를 개선하기 위해 Diffusion을 도입한 Difusco 모델이 제안되었다 [23, 10]. 이 외에 Ant Colony Optimization 알고리즘을 심층 강화학습으로 설계한 DeepACO모델 또한 유망한 결과를 보였다 [22]. 해의 품질보다 시간을 줄이기 위한 방법들도 많이 연구되었는데, 대규모 TSP를 영역별로 나누어 작은 문제로 만들어서 풀어 품질이 낮지만 매우 빠르게 근사해를 얻어낸 H-TSP가 제안되었다 [19]. SO와 GLOP은 초기해를 구한 후 다른 해를 탐색하여 더 좋은 해가 있으면 좋은 해로 개선시켜 얻어내는 방법이다 [21, 1].

Chapter 3

Preliminary

3.1 Definition of TSP

본 논문에서 풀고자 하는 문제인 TSP는 차량 경로 문제(Vehicle Routing Problem)의 가장 기본적인 형태로, 노드를 한번씩 방문한 후 되돌아오는 가장 짧은 해밀토니언 순환(Hamiltonian Cycle)을 찾는 것이 목표다. 입력은 모든 i에 대해 $x_i \in \mathbb{R}^2$일 때, n개의 시퀀스로 구성된 2차원 유클리드 공간 $\mathbf{X} = [x_0, x_1, x_2 \ldots, x_{n-1}]$으로 나타낸다. 이때, 문제의 해는 아래 3.1처럼 시퀀스 π로 나타낼 수 있다.

$$\pi = [\pi_0, \pi_1, \pi_2, \ldots, \pi_{n-1}], \quad \pi_i \in \{0, 1, 2, \ldots, n-1\}, \quad \pi_i \neq \pi_j \quad \text{if} \quad i \neq j \tag{3.1}$$

문제의 목표는 π의 순서대로 만들어진 순환의 길이를 최소화하는 것이며, 목적 함수는 아래 식 3.2와 같이 정의된다.

$$L(\pi|\mathbf{X}) = \|x_0 - x_{n-1}\|_2 + \sum_{i=1}^{n} \|x_{\pi_i} - x_{\pi_{i-1}}\|_2 \tag{3.2}$$

3.2 Definition of SHPP

SHPP는 TSP의 변형 문제로, 노드를 한번씩 방문한 후 되돌아오는 가장 짧은 해밀토니언 경로(Hamiltonian Path)을 찾는 것이 목표다. 입력은 TSP와 마찬가지로 모든 i에 대해 $x_i \in \mathbb{R}^2$ 일 때, n개의 시퀀스로 구성된 2차원 유클리드 공간 $\mathbf{X} = [x_1, x_2, \ldots, x_n]$으로 나타낸다. TSP와 조금 다른 점이 있는데, SHPP는 항상 $\pi_0 = 0$, $\pi_{n-1} = n-1$으로 각각 출발점과 도착점으로 고정된다. 이때, 문제의 해는 아래 3.3처럼 시퀀스 π로 나타낼 수 있다.

$$\pi = [\pi_1, \pi_2, \pi_3, \ldots, \pi_{n-2}], \quad \pi_i \in \{1, 2, 3, \ldots, n-2\}, \quad \pi_i \neq \pi_j \quad \text{if} \quad i \neq j \tag{3.3}$$

문제의 목표는 π의 순서대로 만들어진 경로의 길이를 최소화하는 것이며, 목적 함수는 아래 식 3.4와 같이 정의된다.

$$L(\pi|\mathbf{X}) = \sum_{i=1}^{n} \|x_{\pi_i} - x_{\pi_{i-1}}\|_2 \tag{3.4}$$

Chapter 4

Methodology

본 챕터에서는 대규모 외판원 순회 문제를 위한 Local Improvement(LI)를 설명한다. LI는 GLOP의 Local Construction을 발전시킨 알고리즘으로, GLOP에서 Attention SHPP Policy 대신 성능을 개선한 LEHD SHPP Policy를 적용하였다 [1].

4.1은 LI를, 4.2는 LEHD SHPP Policy을 설명한다.

4.1 Local Improvement

Local Improvement(LI)는 TSP를 푸는 알고리즘으로, 기존 해를 여러 개의 더 작은 문제들로 나눈 뒤 새롭게 풀어내 기존 해를 개선시키는 알고리즘이다. LI는 GLOP의 Local Construction과 동일한 매커니즘이다.

LI는 TSP를 divide and conquer로 해결하는 것이 핵심 아이디어이다. 먼저 TSP가 주어지면 RI로 빠르게 초기 해를 구한다. 이후, 다음 과정을 반복한다. 먼저, 기존 해를

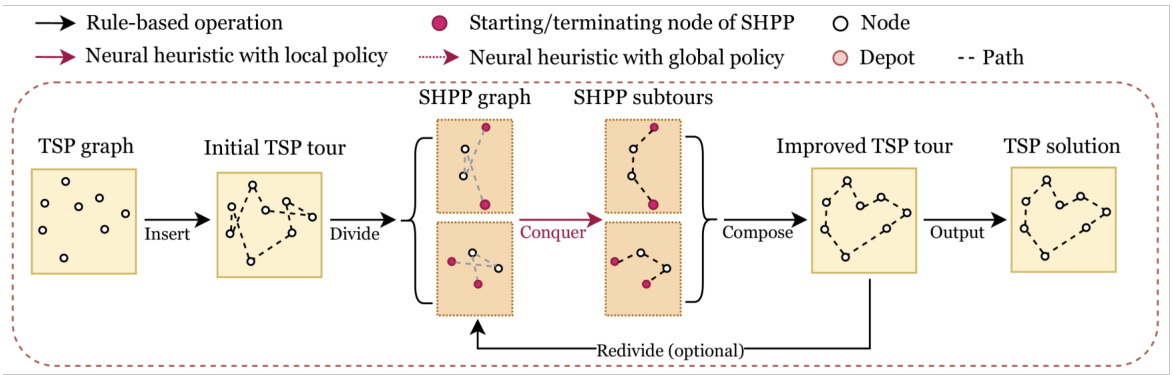

Figure 4.1: LI 알고리즘 [1].

d개로 나눈 뒤, $\lfloor \frac{N}{d} \rfloor$ 크기의 SHPP d개를 새롭게 풀어낸다. 이때, 새롭게 얻은 해가 기존 해보다 더 짧은 경로이면 새롭게 얻은 해로 대체한다. 이후, 나뉘어진 모든 SHPP의 해를 합쳐 전체 TSP 해를 형성한다. 아래는 각 단계별로 자세히 방법을 설명한 것이다.

4.1.1 Initialization

Random Insertion(RI)는 솔루션의 품질이 낮지만 매우 빠르게 얻을 수 있는 휴리스틱 알고리즘이다. RI를 통해 TSP의 초기 해를 찾는다.

4.1.2 d-Decomposition

크기가 n인 TSP는 더 작은 문제로 나눌 수 있다. d-Decomposition은 $\lfloor \frac{n}{d} \rfloor$ 크기의 노드가 중복되지 않는 SHPP d개로 문제를 재정의한다. 이때, n이 d로 나누어 떨어지지 않는 부분은 병렬 연산이 불가능하므로 그대로 놔둔다.

4.1.3 Normalization

모델의 성능을 향상시키기 위해 문제의 위치를 정규화한다. TSP의 크기가 커질수록 부분해 위치의 최댓값과 최솟값 차이가 적어지기 때문에 x 좌표에 대해 최댓값을 1, 최솟값을 0으로, 그리고 y 좌표에 대해 최댓값을 1, 최솟값을 0으로 정규화한다.

4.1.4 Solve SHPP

새롭게 정의된 d개의 SHPP 문제를 LEHD SHPP Policy를 통해 새 해를 구한다. 본 모델은 신경망으로 이루어진 솔버이기 때문에 굉장히 빠르게 병렬로 문제를 풀어낼 수 있다. LEHD SHPP Policy는 4.2에 자세히 설명되어 있다.

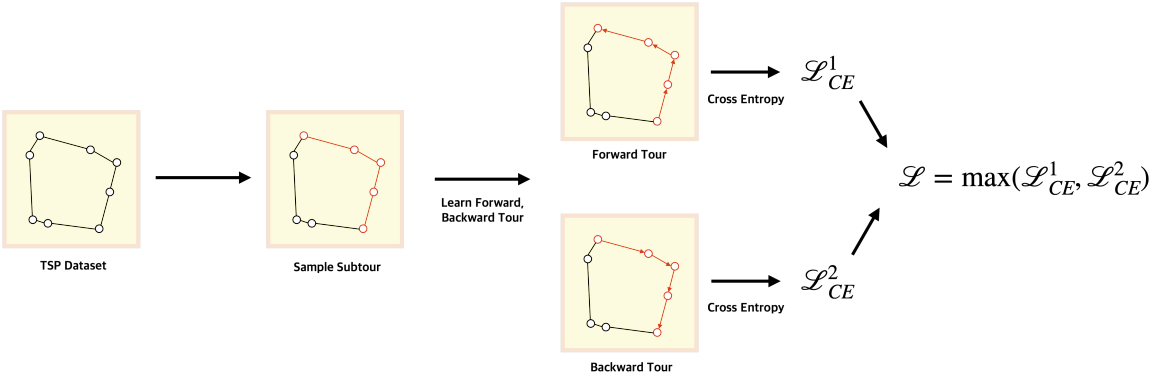

Figure 4.2: LEHD 훈련 매커니즘

4.1.5 Composition

이전 단계로부터 얻어낸 d개의 새 해의 경로 길이가 기존 해보다 개선되었다면 각각 새 해로 채택한다. 만약 그렇지 않다면 기존해와 동일한 경로를 채택한다. 이후, 채택된 부분의 경로를 모두 다시 연결하여 전역 해를 얻는다.

4.2 LEHD SHPP Policy

LEHD SHPP Policy는 TSP와 CVRP를 풀기 위한 NCO 솔버인 Light Encoder Heavy Decoder(LEHD) [9]의 모델 구조를 그대로 사용한다. LEHD를 훈련시키기 위해, GLOP의 Local Policy와 LEHD의 훈련 매커니즘을 모두 참조한다. LEHD는 최적해 레이블 있는 데이터셋을 임의로 나누어 SHPP 문제와 최적해 레이블을 만든다. 최적해의 부분해는 역시 최적해이므로 임의로 나누어 샘플링된 데이터셋 역시 최적해 레이블을 가진다. 이렇게 구성된 데이터셋을 통해 Cross Entropy loss function으로 지도학습을 한다.

GLOP은 SHPP 정책을 학습하기 위해 POMO [14]에서 영감을 받아 같은 SHPP의 시작점, 도착점을 뒤집어 순방향, 역방향 문제를 만들었다. 이후, 두 문제를 모두 풀어 더 최적의 값을 얻어낸 시퀀스를 채택한다. 그러나, 훈련 단계에서는 양방향의 loss

function의 최댓값들을 loss function으로 정의한 후 이를 최소화하도록 학습된다. 본 논문의 LEHD SHPP Policy 또한 이 아이디어를 채택해 SHPP의 시작점, 도착점을 뒤집어 순방향, 역방향 문제를 모두 푼다. 이때, 두 문제 모두 Cross Entropy loss function을 정의한 후, 두 함수의 최댓값을 loss function으로 정의해 이를 최소화하는 방향으로 학습한다.

GLOP에서는 Attention SHPP Policy를 사용하는데, 이에 비해 LEHD SHPP Policy가 가지는 장점은 크게 두 가지가 있다. 첫째, LEHD SHPP Policy는 더 좋은 품질의 근사해를 얻어낼 수 있다. 둘째, Attention SHPP Policy는 SHPP의 문제 크기에 따라 각각 다른 모델 구조와 가중치가 필요하나 LEHD는 모든 문제 크기에서 적용 가능하다. 이는 LI의 추론 과정을 더 효율적으로 적용할 수 있게 해주는 요인이 된다.

Chapter 5

Experiments

5.1 LI Model Training and Inference

5.1.1 Model Training

4.2의 방법을 그대로 따라 LEHD 모델을 학습시켰다. 모델은 128차원의 선형 임베딩과 1개의 Attention Layer로 구성된 Encoder, 6개의 Attention Layer로 구성된 Decoder 구조로 구성되어 있다.

Attention Layer의 구조는 [13]의 Transformer의 Attention Layer 구조를 따른다. 128차원의 attention layer, 8개의 attention head, 512차원의 feed forward layer, 그리고 batch normalization을 포함한다.

5.1.2 Dataset

훈련 데이터셋은 100만개의 TSP100($1,000,000 \times 100 \times 2$)과 이 데이터에 대응되는 최적해 레이블($1,000,000 \times 100$)로 구성되어있다.

5.1.3 Inference Strategy

LI를 보다 효율적으로 추론 과정에서 사용하기 위해 LI를 적용할때 나누는 개수를 계획해야 한다. 분해 스케쥴링은 추론 과정에서 효율적인 d-LI를 적용하기 위한 계획법이다. d-LI는 d-Decomposition의 LI를 적용한 것이다.

본 섹션에서는 분해 스케쥴링으로 사용할 Sigmoid Scheduling을 제안한다. 반복적인 d-LI를 통해 최적에 가까운 해에 다가갈 수 있지만 매 단계마다 d를 설정하는 것은 어려운 문제다. 작은 d-Decomposition의 LI는 큰 d-Decomposition보다 높지만 전역 최적에서 멀어지는 경향이 있다. 그러므로 적절한 d-LI를 적용하는 것은 중요하다. GLOP의 경우, Attention SHPP Policy 특성상 고정된 크기의 문제만 풀 수 있기 때문에 Scheduling이 불가능하다 [1]. 이는 local optimal에 빠지거나, 불필요한 시간을 소모하는 결과를 초래한다. Sigmoid Scheduling은 LEHD SHPP Policy는 모든 문제 크기에 대해 풀 수 있다는 특성을 이용해 큰 d-LI가 수렴할 때, Sigmoid 함수를 따라 d를 줄여가는 방법이다. 아래 식 5.1은 d를 구하는 방법이다.

$$d(t; d_{\max}, d_{\min}, \alpha) = (d_{\max} - d_{\min}) sigmoid(-t + \alpha)) + d_{\min} \tag{5.1}$$

이때, hyperparameter를 통해 스케쥴링을 조절할 수 있다. $d_{\max}$는 d-LI를 적용할 d의 최댓값이며, $d_{\min}$는 d-LI를 적용할 d의 최솟값이다. α는 최대치의 $d_{\max}$ 를 적용할 횟수를 나타내며 문제 크기에 따라 실험적으로 설정 가능하다.

5.2 Experimental Setup

5.2.1 Baseline Model

제안한 LI를 실험을 통해 비교하기 위해 Concorde OR 솔버, LKH-3 휴리스틱, Random Insertion 휴리스틱, H-TSP, LEHD, Att-GCN, DIMES, distMCTS, DIFUSCO, SO, GLOP 베이스 라인 모델로 선택하였다 [3, 19, 9, 5, 20, 24, 10, 21, 1]. 이후 같은 조건 하에서 실험을 진행했다. 각 모델은 NCO 모델이 아닌 휴리스틱 혹은 OR 솔버, 샘

플링 없이 greedy policy를 적용해 얻어낸 솔버, MCTS기반 솔버(MCTS), 그리고 NCO 모델과 휴리스틱을 적용한 솔버(H)로 분류하였다. 실험에 기입되지 않은 Greedy policy가 적용된 다양한 솔버가 존재하지만 대부분 TSP500 미만의 문제에 효과적이고 대규모 문제에 최적해를 도출하기 어렵기 때문에 제외하였다.

5.2.2 Matrics

우리는 본 실험을 위해 [13]의 데이터 생성 규칙을 그대로 따라 128개의 TSP1000, 16개의 TSP10000 실험 데이터셋을 제작하였다. 이 데이터셋은 $[0,1]^2$안에서 n개의 균등한 랜덤 위치 좌표를 샘플링해 TSP 데이터를 형성한다. 이 실험 데이터셋을 풀어낸 경로의 길이의 평균, LKH3 알고리즘과의 격차, 그리고 추론 과정에서 걸린 시간을 평가 지표로 삼는다 [3].

5.2.3 Hardware and Software

본 실험을 위해 1장의 Nvidia A100 그래픽 카드를 사용하였다. 소프트웨어 환경은 Python 3.11, Torch 12.1이다.

5.3 Experimental Results

Table 5.1은 각 모델들이 TSP1000과 TSP10000의 문제를 풀어내어 얻어낸 해의 정확도와 시간을 정리한 표다. 실험으로 LI_{fast} 솔버와 LI 솔버를 사용했는데, 이는 같은 모델에 스케쥴링 계수만 다르게 조절한 솔버다. 공통적으로 $d_{max} = 200$, $d_{min} = 50$ 이며, LI_{fast}의, $\alpha = 10$, LI의 $\alpha = 25$로 설정하였고, 종료 지점은 $t = \alpha$인 지점이다.

LI_{fast} 솔버는 TSP1000에서 4.1m의 시간동안 23.51, TSP10000에서 1.0m의 시간동안 74.75의 결과를 얻었다. 이는 목표로 삼던 GLOP보다 더 빠르면서도 정확하다. 시간보

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
$\mathbf{LI}_{\text{fast}}$	SL+H	23.67	2.38%	1.1m	74.75	4.15%	**1.0m**

Table 5.1: Experimental Result for TSP1000, TSP10000

다는 성능에 초점을 맞춘 *LI* 솔버는 TSP1000에서 1.1m의 시간동안 23.67, TSP10000에서 9.6m의 시간동안 74.00의 결과를 얻었다. 이 결과는 GLOP에 비해 시간이 더 걸리지만 성능을 크게 향상시켰다. 시간이 매우 오래 걸린 MCTS 기반 방법인 DIFUSCO를 제외하면 더 적은 시간으로 더 정교한 해를 얻은 것이다. 이는 본 논문에서 제안한 LI 방법과 분해 스케쥴링이 매우 높은 품질의 해를 구하기는 어렵지만 짧은 시간 안에 상당히 높은 품질의 해를 구하기 적합하다는 것을 알 수 있다.

Chapter 6

Conclusion and Limitation

이 논문에서는 조합 최적화의 대표적 문제인 TSP를 다룬다. 대규모 TSP의 근사해를 보다 빠르고 정확하게 구하는 방법을 제시한다. 본 논문은 기존 LEHD 모델을 응용해 GLOP의 방법을 발전시킨 LI를 소개한 아이디어로, TSP의 해를 찾기 위한 LI 방법을 제시한다. 또한, LI에서 추론 과정을 나타내는 분해 스케쥴링 방법을 실험적으로 제시한다. 이를 통해 GLOP에 비해 시간, 정확도 측면에서 모두 우위를 점하는 성과를 보였다. 그러나 본 방법은 중요한 한계점이 존재하는데, 해를 개선하는 휴리스틱 방법 특성상 대규모 TSP에서 전역 최적화에 가까워질 뿐, 지역 최적화에서 벗어나는 해결책을 제시하지 못했다. 이는 GLOP을 포함한 모든 조합 최적화 솔버가 공통적으로 극복해야 할 문제로 추후 이를 다루기 위한 효과적인 해결책이 필요하다.

Summary

Local Improvement for Large-scale Traveling Salesman Problem

신경망 기반 방법은 조합 최적화의 최적에 가까운 근사해를 빠르게 구하기 위한 유망한 방법이다. 물류 최적화, 차량 경로 문제, 일정 할당 문제 등과 같은 조합 최적화 문제는 실생활에서 자주 볼 수 있는 조합할 수 있는 경우의 수가 굉장히 많아 최적해를 구하기 어려운 문제이다. 외판원 순회 문제(Traveling Salesman Problem)은 대표적인 조합 최적화 문제로 효율적으로 고품질의 근사해를 얻기 위한 신경망 기반의 연구가 진행되었다. 본 논문은 대규모 외판원 순회 문제에서 효과적으로 최적에 가까운 근사해를 탐색하는 신경망 기반 모델 훈련 전략 및 휴리스틱 방법을 이용한 지역 개선(Local Improvement) 방법을 제시한다. 본 지역 개선 방법을 통해 대규모 외판원 순회 문제에서 실시간에 적용 가능할 정도로 적은 시간 내 고품질의 근사해를 얻는 성능을 보였다. 이는 짧은 시간 내의 굉장히 복잡한 조합 최적화 문제에서도 신경망 기반 방법이 효과적임을 입증한다.

References

1. H. Ye, J. Wang, H. Liang, Z. Cao, Y. Li, and F. Li, "Glop: Learning global partition and local construction for solving large-scale routing problems in real-time," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 38, pp. 20284–20292, 2024.

2. B. A. Julstrom, "Coding tsp tours as permutations via an insertion heuristic," in *Proceedings of the 1999 ACM symposium on Applied computing*, pp. 297–301, 1999.

3. K. Helsgaun, "An effective implementation of the lin–kernighan traveling salesman heuristic," *European journal of operational research*, vol. 126, no. 1, pp. 106–130, 2000.

4. X. Wu, D. Wang, L. Wen, Y. Xiao, C. Wu, Y. Wu, C. Yu, D. L. Maskell, and Y. Zhou, "Neural combinatorial optimization algorithms for solving vehicle routing problems: A comprehensive survey with perspectives," *arXiv preprint arXiv:2406.00415*, 2024.

5. Z.-H. Fu, K.-B. Qiu, and H. Zha, "Generalize a small pre-trained model to arbitrarily large tsp instances," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 35, pp. 7474–7482, 2021.

6. A. Bogyrbayeva, M. Meraliyev, T. Mustakhov, and B. Dauletbayev, "Learning to solve vehicle routing problems: A survey," *arXiv preprint arXiv:2205.02453*, 2022.

7. O. Vinyals, M. Fortunato, and N. Jaitly, "Pointer networks," *Advances in neural information processing systems*, vol. 28, 2015.

8. C. K. Joshi, T. Laurent, and X. Bresson, "An efficient graph convolutional network technique for the travelling salesman problem," *arXiv preprint arXiv:1906.01227*, 2019.

9. F. Luo, X. Lin, F. Liu, Q. Zhang, and Z. Wang, "Neural combinatorial optimization with heavy decoder: Toward large scale generalization," *Advances in Neural Information Processing Systems*, vol. 36, pp. 8845–8864, 2023.

10. Z. Sun and Y. Yang, "Difusco: Graph-based diffusion solvers for combinatorial optimization," *Advances in neural information processing systems*, vol. 36, pp. 3706–3731, 2023.

11. T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2016.

12. A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

13. W. Kool, H. Van Hoof, and M. Welling, "Attention, learn to solve routing problems!," *arXiv preprint arXiv:1803.08475*, 2018.

14. Y.-D. Kwon, J. Choo, B. Kim, I. Yoon, Y. Gwon, and S. Min, "Pomo: Policy

optimization with multiple optima for reinforcement learning," *Advances in Neural Information Processing Systems*, vol. 33, pp. 21188–21198, 2020.

15. M. Kim, J. Park, and J. Park, "Sym-nco: Leveraging symmetricity for neural combinatorial optimization," *Advances in Neural Information Processing Systems*, vol. 35, pp. 1936–1949, 2022.

16. Y. Jin, Y. Ding, X. Pan, K. He, L. Zhao, T. Qin, L. Song, and J. Bian, "Pointerformer: Deep reinforced multi-pointer transformer for the traveling salesman problem," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 37, pp. 8132–8140, 2023.

17. M. Kim, J. Park, *et al.*, "Learning collaborative policies to solve np-hard routing problems," *Advances in Neural Information Processing Systems*, vol. 34, pp. 10418–10430, 2021.

18. D. Drakulic, S. Michel, F. Mai, A. Sors, and J.-M. Andreoli, "Bq-nco: Bisimulation quotienting for efficient neural combinatorial optimization," *Advances in Neural Information Processing Systems*, vol. 36, pp. 77416–77429, 2023.

19. X. Pan, Y. Jin, Y. Ding, M. Feng, L. Zhao, L. Song, and J. Bian, "H-tsp: Hierarchically solving the large-scale traveling salesman problem," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 37, pp. 9345–9353, 2023.

20. R. Qiu, Z. Sun, and Y. Yang, "Dimes: A differentiable meta solver for combinatorial optimization problems," *Advances in Neural Information Processing Systems*, vol. 35, pp. 25531–25546, 2022.

21. H. Cheng, H. Zheng, Y. Cong, W. Jiang, and S. Pu, "Select and optimize: Learning to solve large-scale tsp instances," in *International Conference on Artificial Intelligence and Statistics*, pp. 1219–1231, PMLR, 2023.

22. H. Ye, J. Wang, Z. Cao, H. Liang, and Y. Li, "Deepaco: Neural-enhanced ant systems for combinatorial optimization," *Advances in neural information processing systems*, vol. 36, pp. 43706–43728, 2023.

23. J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," *Advances in neural information processing systems*, vol. 33, pp. 6840–6851, 2020.

24. Y. Xia, X. Yang, Z. Liu, Z. Liu, L. Song, and J. Bian, "Position: Rethinking post-hoc search-based neural approaches for solving large-scale traveling salesman problems," *arXiv preprint arXiv:2406.03503*, 2024.