Improving subtour constraints generation in Branch-and-Cut algorithms for TSP with Machine Learning

VO Thi Quynh Trang[1], NGUYEN Viet Hung[1], BAIOU Mourad[1], WENG Paul[2]

[1]LIMOS Université Clermont Auvergne, France
[2]UM-SJTU Joint Institute, Shanghai Jiao Tong University
thi_quynh_trang.vo,viet_hung.nguyen,mourad.baiou@uca.fr, paul.weng@sjtu.edu.cn

Mots-clés : *Traveling Salesman Problem, Subtour Elimination Constraints, Branch-and-Cut, Machine Learning*

1 Introduction

Branch-and-Cut (B&C) is a widely-used algorithm for solving integer programming (IP) problems. In recent years, applying Machine Learning (ML) to improve fundamental decision problems of B&C algorithms is an active research domain. While much of ML research focuses on variable selection [1], node selection [2], and cut selection [3], less attention has been paid to the question of how to design a cutting plane generation strategy in B&C algorithms. This question is crucial since generating cutting planes might become a computational burden when the instance's size increases.

In this work, we focus on improving the subtour elimination constraints (SEC) generation in B&C algorithms for Traveling Salesman Problem (TSP) by ML. SEC is a well-known constraint used to exactly solve TSP. In the IP formulations for TSP, SEC is dynamically generated as cutting planes in the course of B&C algorithms due to its exponential cardinality. Our purpose is to take advantage of ML to address two questions before launching the separation process to find violated SEC cuts on a node of the B&C search tree : 1) Does there exist violated SEC cuts ? 2) If yes, is it worth generating them ? To do this, we propose a ML-based method consisting of two parts : a cut detector to predict the cut existence and a cut decider to decide whether generate cuts. Our method not only can leverage the geometric structure of optimal solutions but it also offers the flexibility of the instance's size.

2 Formulation for TSP

Given an undirected graph $G = (V, E)$ and a cost vector $\boldsymbol{c}$ associated with E, an IP formulation using SEC to formulate TSP is

$$\min \ \boldsymbol{c}^T \boldsymbol{x} \tag{1a}$$

$$\text{s.t} \sum_{e \in \delta(v)} x_e = 2 \qquad \forall v \in V \tag{1b}$$

$$\sum_{e \in \delta(S)} x_e \geq 2 \qquad \forall S \subset V \tag{1c}$$

$$x_e \in \{0, 1\} \qquad \forall e \in E \tag{1d}$$

where $\delta(S) = \{(i, j) \in E | |\{i, j\} \cap S| = 1\}$, $\forall S \subset V$, $\delta(v) = \delta(\{v\})$, $\forall v \in V$. Due to its exponential cardinality, SEC will be generated progressively during the solving process. In particular, when we obtain a feasible integer solution, we verify and add violated SEC to the formulation. Moreover, SEC can be used as cutting planes for optimal fractional solutions at

nodes of the B&C search tree. The process that identifies SEC cuts in TSP, a.k.a separation, is usually performed by solving a minimum cut problem. The main challenge is that solving separation problems can become a dominant task when the instance's size increases. Furthermore, the cuts' effectiveness might not be worth their finding cost, especially in cases where no violated SEC is found. Hence, learning a deft policy to generate SEC is essential.

3 Methodology

To improve the SEC generation in B&C algorithms, we propose a ML-based method that consists of two parts : a cut detector and a cut decider. The former is a classifier to predict the SEC cut existence from optimal fractional solution at nodes. The cut detector aims to reduce the time spent on redundant separation processes that do not find any cut. In detail, we consider detecting SEC cuts at a search tree's node as a classification problem with two labels : *positive* if there exist cuts and *negative* otherwise. We first solve separation problems over the search tree of training instances to collect a dataset. To train the classifier, we represent node's solutions as graphs and use a Graph Neural Network (GNN) to embed the graphs.

The latter, i.e., the cut decider, is a Reinforcement Learning (RL) agent used to decide whether to generate cuts. More precisely, we formulate the sequential decisions making of cut generation as a Markov decision process (MDP) and use Deep Q-learning to train an agent. The action space of the MDP comprises two actions : generate cuts and branch. The state space contains information about fractional solutions, the TSP instance, and the B&C search tree. To help the agent learn a wise policy, we define a reward function based on decreasing the IP relative gap. The larger the IP relative gap drops, the faster B&C solves the IP. By setting the discount $\gamma < 1$, we encourage the agent to perform actions that rapidly reduce the relative gap and terminate the optimization process.

4 Numerical results

To prove the efficiency of our proposed method, we evaluate Branch-and-Cut with the help of ML on random and TSPLIB instances. Numerical results show that our approach can substantially reduce the running time on all instance groups. Table 1 presents the results on several TSPLIB instances.

	Branch-and-Cut			**Branch-and-Cut with ML**		
	Time	**nNodes**	**nCuts**	**Time**	**nNodes**	**nCuts**
ch150	91.66	311	455	**15.25**	536	270
kroA200	2032.56	18167	2101	**921.18**	29180	1670
gil262	1073.25	20383	2169	**586.30**	34253	1306

TAB. 1 – Numerical results on several TSPLIB instances

Références

[1] M Gasse, D Chételat, N Ferroni, L Charlin, and A Lodi. Exact combinatorial optimization with graph convolutional neural networks. *Advances in neural information processing systems*, 2019.

[2] H He, H Daume III, and J Eisner. Learning to search in branch and bound algorithms. *Advances in neural information processing systems*, 2014.

[3] Yunhao Tang, Shipra Agrawal, and Yuri Faenza. Reinforcement learning for integer programming : Learning to cut. In *International conference on machine learning*, 2020.