An hybrid approach based on Graph Attention Network for the Team Orienteering Problem.

Iván Peña-Arenas, Rym Nesrine Guibadj, Cyril Fonlupt

An hybrid approach based on Graph Attention Network for the Team Orienteering Problem

Iván Peña-Arenas[1] Rym Nesrine Guibadj[1] and Cyril Fonlupt[1]

Université du Littoral Côte d'Opale, EA 4491 – LISIC – Laboratoire d'Informatique Signal et Image de la Côte d'Opale, F-62228 Calais, France
{ivan.pena-arenas, rym.guibadj, cyril.fonlupt}@univ-littoral.fr

1 Introduction

The Team Orienteering Problem (TOP) is a combinatorial optimization problem that has been proven NP-Hard [1], and which has different practical applications ranging from logistics to telecommunications. End to end machine learning models have been proven effective to solve discrete optimization problems [2]. Advancements in deep learning techniques designed for managing data structures of varying sizes, such as attention mechanisms and sequence to sequence approaches, have significantly increased their usage. Nevertheless, as the scale of problem instances grows, deep learning models experience a decline in performance, leading to a challenge in generalization (see, [3], [4]).

Heuristics methods are efficient solution algorithms designed for particular problems, which rely on their programmed behaviour. Several contributions involve merging optimization algorithms with machine learning [5], wherein the machine learning model aids the optimization algorithm. Following the same idea, we propose leveraging both solution methods, although in this case, the heuristic supports the learning algorithm, guiding the learning process. While the concept has been previously suggested [6], as far as we are aware, this is the first implementation of this idea in the context of the TOP.

We present a learning framework that integrates an efficient splitting algorithm applied to the TOP [7] within a deep learning model. This hybrid approach works in two steps. Initially, a *giant tour* (a sequence of customers/locations) is generated at once using a deep neural network, and subsequently, it is evaluated using the split algorithm (see Fig. 1). The main objective is to narrow down the solution space in which the deep learning model works (i.e. one optimal set of sub-tours for each *giant tour*), expecting an overall better performance. In addition, two handcrafted solution methods are used to compare the performance and the quality of the results. Section 2, briefly describes the method, Section 3 shows results and comparisons, and conclusions are presented in Section 4.

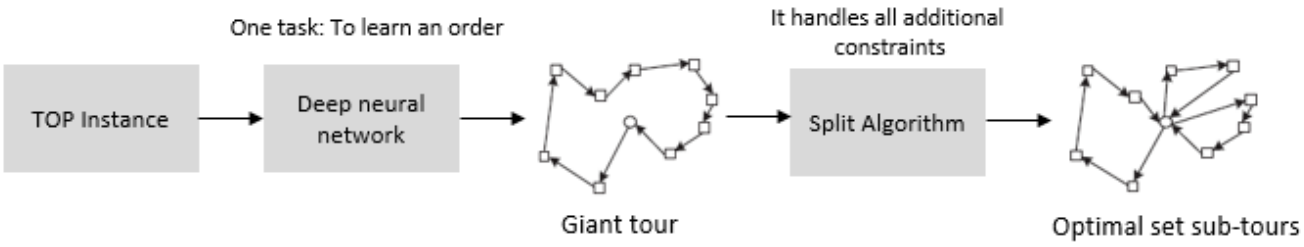

Fig. 1. Solution schema.

2 Hybrid Graph Attention Model

During the first stage, a reinforcement-trained model is used to generate a *'giant tour'*. Our deep learning procedure is based on a Graph Attention Network that operates as an encoder-decoder system [4]. The encoder is responsible for processing the input instance and generating embeddings for all input nodes, including clients and the depot. This transformation of the instance's representation aims to capture underlying structures beneficial for problem-solving. Subsequently, the decoder utilizes the encoder embeddings along with the history of previously visited clients to compute a singular vector summarizing the current solution. Following this, a multi-head attention module uses this vector to establish a probability distribution function for selecting the subsequent client. The decoder progressively constructs candidate solutions, sequentially selecting the next node until all clients have been visited.

In the following stage, we employ an optimal splitting algorithm to identify the collection of sub-tours that maximizes the total sum of their profits. This method ensures that if a series of sub-tours, constituting an optimal solution for the Team Orienteering Problem, exists as sub-sequences within the *'giant tour'* $\boldsymbol{\pi}^*$, applying the splitting process to $\boldsymbol{\pi}^*$ will yield the optimal TOP solution. Consequently, the neural network operates within a more confined solution space, determined by the collection of giant tours resulting from the splitting procedure.

During the learning process, the Graph Attention Model defines a stochastic policy $p_\theta(\boldsymbol{\pi}|s)$ for selecting a *giant tour (sequence)* $\boldsymbol{\pi}$ given a TOP instance s and the parameters θ. Later, by applying the split procedure this sequence $\boldsymbol{\pi}$ is decomposed in a set of optimal sub-tours considering the maximum length duration for a tour L. Moreover, we define $J(\theta|s)$ as the policy objective function, which is the total expected score of the sub-tours evaluated by the split procedure given the instance s.

$$J(\theta|s) = \mathbb{E}_{p_\theta(\boldsymbol{\pi}|s)}\left[(split(\boldsymbol{\pi}|s)\right]$$

We use policy gradients methods to search for a local maximum in $J(\theta|s)$ by ascending gradient policy, w.r.t parameters θ., defined as:

$$\nabla_\theta J(\theta|s) = \mathbb{E}_{p_\theta(\boldsymbol{\pi}|s)}\left[\nabla_\theta \log p_\theta(\boldsymbol{\pi}|s)(split(\boldsymbol{\pi}|s) - b(s))\right]$$

Where $b(s)$ is the baseline, which is used to reduce the gradient variance. The value of the gradient function $\nabla_\theta J(\theta|s)$ is equal to the expected value of the multiplication between the functions score and advantage. The gradient $(\nabla_\theta \log p_\theta(\boldsymbol{\pi}|s))$ is a measure of the movement of the function in the solution space *(score function)*. While, the difference between the score $split(\boldsymbol{\pi}|s)$ of the tour π, and the baseline $b(s)$ is the *advantage function*. To optimize the expected score we use REINFORCE gradient estimator, and an exponential moving average with a decay $\beta = 0.2$ as a baseline.

3 Results

We consider three types of TOP instances with number of clients n equal to 20, 50 and 100. For each problem size we generate 800000 instances for training, and two sets of 10000 instances for validation and test. The depot location as well as n node locations are sampled uniformly at random in the unit square. We consider a constant distribution of prices $p_i = 1$, thus the goal becomes to visit as many nodes as possible within the length constraint. The number of teams m is set to 2, and the maximum length T is fixed to 1.

We initialize parameters Uniform$(-1/\sqrt{d}, 1/\sqrt{d})$, where $d = 128$ represents the embedding dimension. We train 100 epochs, using a fixed batch size B equal to 500 instances and the training and validation data were generated on the fly. We use $K = 3$ layers in the encoder and a constant learning rate $\eta = 10^{-5}$. All the experiments were conducted on a single Nvidia RTX-3500 Ada with 12 GB of VRAM. The Hybrid graph attention model (HGAM) was implemented with Python, using the Pytorch libraries, while the approximation algorithms were coded in C++.

To evaluate the effectiveness of the HGAM, we compare it with a tailored heuristic known as Construction Heuristic (CH) and with the metaheuristic Iterative Destruction/Construction Heuristic (IDCH)[8]. To apply the acquired policies, we utilize both a sampling decoding method,

applied at each decoding step, and a sampling strategy that generates 100 candidate solutions for each test instance. We then select the best solution based on their scores. These solutions are sampled from the probability distributions provided by the models. This evaluation was conducted using test instances that had never been seen during the training and validation processes. We report in Table 1 the average solution score (Objective), the average GAP (in percentage) to the best average solution score and the running time (in seconds) to solve a test instance.

Table 1. Performance comparison between solution methods.

	n=20			n=50			n=100		
Method	Objective	GAP(%)	Time(s)	Objective	GAP(%)	Time(s)	Objective	GAP(%)	Time(s)
IDCH	6.71	0.00	0.0057	13.73	0.00	0.0959	22.76	0.00	1.0158
CH	6.21	7.31	0.0002	12.07	12.09	0.0008	19.12	15.66	0.0035
HGAM	6.45	2.38	0.0097	12.70	7.50	0.0129	19.38	14.85	0.0829

IDCH obtains the highest average solution score for all types of instances. The relative gap between the solution methods widens as the instance size increases, reaching up to 14.85% and 15.66% for HGAM and CH, respectively, in instances with 100 clients. On the other hand, CH is the fastest solution method, averaging 0.0035 seconds to solve an instance with 100 clients.

On scenarios involving 20 customers, there is a 2.38% variation in performance between IDCH and HGAM. For instances with 50 and 100 customers, this difference increases to an average of 12.09% and 15.66%, although HGAM is almost 7.4 ($\approx 0.0959/0.0129$) and 12.2($\approx 1.0158/0.0829$) times faster, respectively. In contrast, while CH is two orders of magnitude faster than HGAM, HGAM achieves better scores for all types of instances. In this sense, HGAM presents a good trade-off between the quality of the solutions and their running times.

4 Conclusion

Our work is part of a research trend aiming to integrate machine learning and optimization approaches to effectively solve combinatorial optimization problems. For this purpose, we proposed a hybrid architecture in which a deep neural network based on attention mechanisms seeks to prioritize clients/nodes. An optimal split procedure is then applied to (1) incorporate the constraints, (2) identify the routes, and (3) produce an optimal solution considering the order generated by the GAM. Preliminary findings suggest that our method is competitive when compared to specialized heuristics. Furthermore, unlike heuristic approaches, our model could be used with minimal modifications to solve several other problem variants with benefits.

References

1. Chao, I., Golden, B. and Wasil, E.: The team orienteering problem. European Journal of Operational Research **88**(3), 464-474 (1996).
2. Bengio, Y., Lodi, A., Prouvost, A.: Machine learning for combinatorial optimization: a methodological tour d'horizon. European Journal of Operational Research **290**(2), 405–421 (2021)
3. Vinyals, O., Fortunato, M., and Jaitly, N.: Pointer Networks. In: Advances in Neural Information Processing Systems (28), 2692–2700 (2015)
4. Kool, W., Van Hoof, H., Welling, M.: Attention, learn to solve routing problems! arXiv:1803.08475 (2018)
5. Lodi, A. and Zarpellon, G. On learning and branching: A survey. TOP **25**(2), 207–236 (2017)
6. Yaddaden, A., Harispe, S., Vasquez, M.: Neural Order-First Split-Second Algorithm for the Capacitated Vehicle Routing Problem. In: Dorronsoro, B., Pavone, M., Nakib, A., Talbi, EG. (eds) Optimization and Learning. OLA 2022. Communications in Computer and Information Science, vol 1684. Springer, Cham. (2022)
7. Duc-Cuong D., Guibadj, R. and Moukrim, A.: An effective PSO-inspired algorithm for the team orienteering problem. European Journal of Operational Research, **229**(2), 332-344(2013)
8. H. Bouly, D.-C. Dang, and A. Moukrim.: A memetic algorithm for the team orienteering problem. 4OR, **8**(1),49–70(2010)