# Learning Heuristics to Solve Routing Problems Increasingly Faster

**Marc-Emmanuel Coupvent des Graviers, Jérémy Turi, Christophe Guettier**

SAFRAN ELETRONICS & DEFENSE, MASSY, FRANCE
marc-emmanuel.des-graviers, jeremy.turi, christophe.guettier@safrangroup.com

**Abstract**

Recent research has explored neuro-symbolic approaches that bind Machine Learning and classical combinatorial problem solving to improve efficiency. In particular, Graph Neural Networks (GNNs) have been used to solve routing problems such as the Traveling Salesman Problem (TSP), which is representative of various mission planning applications. However, previous neuro-symbolic approaches remained limited in their ability to generalize beyond training distributions. In this work we propose to improve the generalization of GNN-based guidance for recurring routing problems, where many similar but distinct instances must be solved repeatedly. Our approach shows promising results on realistic instances generated from OpenStreetMap.

Keywords: Neuro-symbolic, Recurring Vehicle Routing Problem, Constraint Programming, Graph Neural Network, Heuristic Learning

## Introduction

In this work, we study recurring routing problems that arise in many real-world applications where vehicles must repeatedly serve a set of geographically distributed locations while minimizing operational costs such as travel distance or time Typical examples include daily mail delivery, waste collection, or fresh food distribution, where one or more vehicles must traverse similar road networks every day, but the specific set of destinations to be visited can vary. These problems belong to the broad family of Vehicle Routing Problems (VRP), a central topic in combinatorial optimization and logistics.

In particular, we are interested in the single-vehicle routing problem, or Recurring VRP (RVRP). Each day, a new set of locations is selected for visitation, requiring the computation of a fresh route. Daily locations are not related to each other in any way, thus each daily instance can be solved independently without loss of optimality. However, unlike classical VRP, this problem exhibits recurrent structure: the underlying road network remains unchanged even if locations number and positions vary daily. This allows learning-based methods to exploit past experiences. This learning capacity becomes even more useful when it generalizes from simpler instances (i.e., days with fewer locations to serve) to more complex ones, as it allows reducing computation time in daily operations. The problem is defined as follows:

- The environment consists of a static road network represented as a graph $G = (V, E)$, where $V$ is the set of vertices (intersections and locations), and $E$ is the set of weighted edges representing roads with associated travel costs (e.g., distance or travel time).
- Each day, a new subset of locations $M \in V$ is selected for mandatory visitation, with the number of locations varying daily.
- A single vehicle must start from a fixed depot $start \in V$, visit all locations in $M$ exactly once, and return to a fixed depot $end \in V$ while minimizing the total travel cost.

Previous approaches addressing routing problems with machine learning often exhibit a lack of generalization through instance sizes, which then requires extensive solving time to create training datasets. The output of Neural Networks (NNs) is also exploited by traditional algorithms such as beam search, which does not guaranty solution optimality, and requires additional algorithmic adaptation when industrial constraints are added in the problem definition.

This paper applies and enhances a hybrid approach using a GNN to capture structures in the routing graph and Constraint Programming (CP) solver to ensure that the generated solutions satisfy all operational constraints. This integration enables efficient and trustworthy route optimization, combining data-driven learning with formal constraint reasoning.

We propose a three step approach. First, we generate problem instances from real road network, and solving small instances optimally. These optimal solutions serve as high-quality training data. Secondly, we train a GNN to predict edge relevance within the optimal solution. This model learns to assign scores to edges, estimating their likelihood of being part of an efficient route. The original NN architecture exhibits strong generalization capabilities across problem instances. Finally, we guide a CP Predicted edge scores are integrated into a guided CP model, prioritizing promising edges and reducing the search space while maintaining optimal guarantees.

## Related Works

Routing problems (Braekers, Ramaekers, and Nieuwenhuyse 2016) represent a central topic in combinatorial optimization, encompassing variants such as TSP (Steiglitz and Weiner 1968) and the Capacitated VRP (CVRP) (Dantzig and Ramser 1959). The TSP seeks the shortest tour visiting all cities once, while CVRP introduces vehicle capacity constraints. Both are NP-hard (Karp 1972) and have been addressed through exact algorithms: branch-and-bound, dynamic programming (Held and Karp 1961)), heuristics (Nilsson 2003), and metaheuristics such as genetic algorithms, simulated annealing, and ant colony optimization (Johnson and McGeoch 1997). Although these methods can solve small to medium-sized instances efficiently, their computational cost grows rapidly with problem size.

Recent advances in Machine Learning (ML) and GNNs (Scarselli et al. 2009; Chen et al. 2020) have enabled data-driven heuristics that learn structural patterns from graph representations of combinatorial problems (Vesselinova et al. 2020; Peng, Choi, and Xu 2021). Early models such as Pointer Networks (Vinyals, Fortunato, and Jaitly 2017) and attention-based solvers (Kool, van Hoof, and Welling 2019) have shown promising results for TSP and CVRP by leveraging training data to generalize across similar instances. Nevertheless, generalization remains challenging: models trained on small benchmarks (e.g., TSP20–TSP100) often degrade in performance when applied to larger or more complex instances (Hudson et al. 2022; Joshi et al. 2022).

## Graph Neural Network for Edge Scoring

The training of NNs to extract structural information from TSP has been demonstrated multiple time, such as in (Osanlou et al. 2019) and (Joshi et al. 2022). We extend this line of work with an architecture exhibiting stronger generalization capabilities. To achieve this, our GNN processes simultaneously two level of problem instance representations. The *first representation* is the road network graph $G = (V, E)$, which is static and represents the road network itself. Its edge structure $E$ is invariant through instances, while mandatory vertices $M$ are identified in $V$ features. The *second representation* is an instance-specific graph $G' = (M, L)$, which uses only mandatory locations $M \in V$ as vertices, and shortest paths between $M$ in $G$ as edges $L$. This architecture is graph-size agnostic while leveraging information from $G$ to guide path selection in $G'$.

The GNN structure reflects this two level of representations. First a stack of graph convolutions, implemented with *GraphConv* (Morris et al. 2021) takes as input the $G$ nodes and edges. Its 11 layers interleaved with sigmoid activations progressively transforms these inputs into higher-dimensional node embeddings. Subsequently, the network predicts a score for each edge of the candidate graph $G'$ using a shared Multi-Layer Perceptron (MLP). The predicted score $pred_{i,j} \in [0, 1]$ indicates the likelihood that edge $(i, j)$ belongs to the optimal tour $\Psi$. Unlike auto-regressive approaches, multiple edges in $G'$ may simultaneously receive high scores. An ideally trained model would assign a score of 1 to all edges included in $\Psi$ and 0 otherwise. The overall network architecture for the prediction of a given edge score is presented in Figure 1.

### Dataset Generation

Our dataset is built from real-world road networks extracted from OpenStreetMap. Intersections are captured as vertices, and road distances between them are accumulated as weighted edges, resulting in partially connected graphs with inherently non-uniform distributions of vertex degrees and edge lengths. The partially connected graph $G$ extracted nearby Reykjavik city is displayed on Figure 2.

We generate multiple problem instances by randomly selecting a subset of vertices $M$ among $V$, designated as mandatory visitation points. Training dataset consists in 10,000 instances, with 10 to 30 mandatory vertices per instance. Testing datasets are multiple sets of 100 instances, with 5, 30, 50, 60, 80 and 100 mandatory vertices per instance. Shortest path distances between all pairs of mandatory vertices $M$ are stored as a distance matrix $L$. Labels are generated by finding optimal solutions for all instances using a standard TSP formulation (1), (2), (3), solved with OR-Tools SAT Solver (Perron and Furnon 2022) running in multi-thread.

$$circuit(s) \tag{1}$$

$$alldifferent(s) \tag{2}$$

$$D(\Psi) = \sum_{i \in \Psi} L_{i,s(i)} \tag{3}$$

### Training Procedure

For each problem instance defined by $M$ mandatory locations, $|M| \times 2$ directed edges among $|M|^2$ pertain to the optimal tour $\Psi$. Consequently, expected edge scores become largely imbalanced when $|M|$ increases. Thus, the prediction loss for a given instance is computed with weighted binary cross entropy through instance edges $L$.

Hyper-parameters are tuned on testing datasets of specific instance sizes: 5, 30 and 60. Generalization behavior is confirmed with testing datasets of other sizes: 50, 80. The following hyper-parameters are retained: Adam optimizer with learning rate $10^{-3}$ for epochs 0 to 35, then learning rate $10^{-4}$. Training is stopped early at epoch 95. Training the GNN takes 1.5 hours on RTX 3060 / Ryzen 3 computer.

### Generalization Capabilities

Generalization to larger instances is critical for recurring routing problem, and notoriously difficult to achieve. While the network was trained exclusively on problem instances containing between 10 and 30 mandatory points, we evaluated its inference performance on test datasets spanning sizes from 10 up to 60. Remarkably, the network maintained predictive accuracy well beyond the training range. Its performance was already observable during training, where monitoring prediction accuracy across epochs revealed a clean learning curve not only for instances within the training range (10–30), but also for larger instances. The property is clearly visible in Figure 3, where inference loss converges

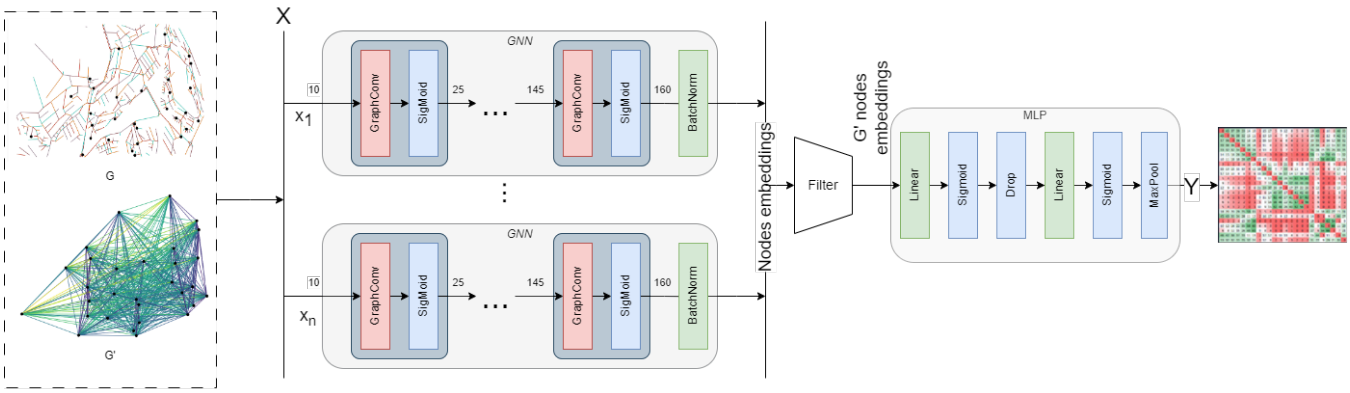


Figure 1: *Architecture of the NN that consists of a node-embedding stack (GNNs) followed by an edge-scoring head (MLP). The node-embedding stack processes the input graph $G$ and assigns to each node an embedding vector of 160 values. For every edge in $G'$, the MLP head receives as input the concatenation of the two embeddings corresponding to its mandatory nodes and outputs a score. Collectively, this produces a two-dimensional score matrix of size $|M|^2$.*

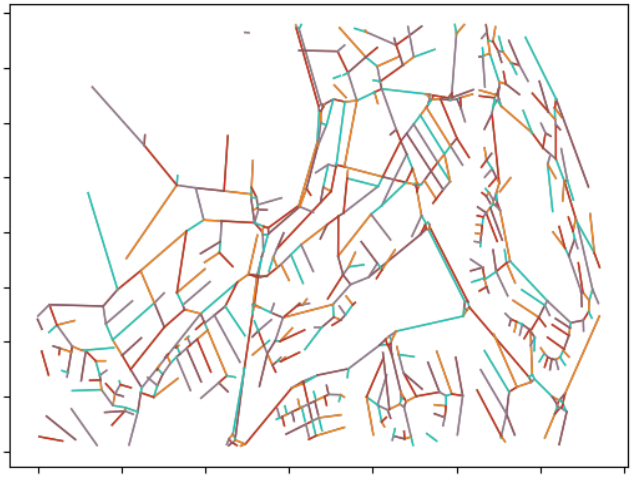

Figure 2: *Intersections graph extracted from the road network with 676 intersections and 3414 localized nodes.*

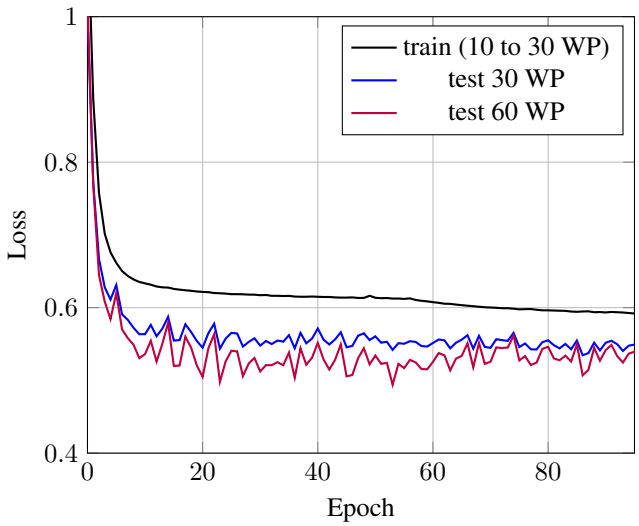


Figure 3: *Edge score loss over training and testing datasets, during training procedure. Experiment performed on the single city area 'Reykjavik'. Training dataset embeds a mix of problems with 10 to 30 mandatory points.*

uniformly across both seen and unseen instance sizes. This provides a strong validation of our neural architecture design, highlighting that the proposed GNN is not only effective within the training domain, but also scalable and robust to significantly larger problem instances.

## Search Guidance by Neural Network Scoring

We propose to guide a CP solver with single-pass GNN inference, using predicted edge scores $pred_{i,j}$ into the search as guidance signals. The CP model incorporates these scores via a cutoff parameter $Cut$. Only edges with $pred_{i,j} > Cut$ are initially considered. To ensure completeness and optimality, $Cut$ is iteratively relaxed, which gradually expands the search domain by enabling all edges. This balances efficiency (targeted search when predictions are good) and completeness (full search recovery if needed).

The problem is formulated as a successor-based model, where the solution is expressed through a successor function. Let $s(i)$ denote the successor of location $i \in M$ such that $s(i) = j$ means that location $j \in M$ is visited immediately after location $i$. The solution validity is enforced by circuit constraint, and the optimal tour length is obtained by minimizing $D(s)$:

Variables:

$$s : M \to M \quad (4)$$

*Assign the next location to each $i \in M$*

Objective: (5)

$$D(\Psi) = \min_s D(s) = \min_s \sum_{i \in M} L[i, s(i)]$$

*Minimize total travel distance*

Subject to:

$$\texttt{circuit}(s) \tag{6}$$

*Ensures the sequence forms a valid tour*

$$\forall i, j \in M,\ i \neq j: \quad C_{i,j} = (pred_{i,j} > Cut) \tag{7}$$

$$\forall i \in M: \quad C_{i,s(i)} \vee C_{s(i),i} \tag{8}$$

$$\forall i, j \in M,\ j > i: \quad \neg(C_{i,j} \vee C_{j,i}) \Rightarrow s(i) \neq j \wedge s(j) \neq i \tag{9}$$

We implemented this formulation with MiniZinc, using the OR-Tools CP-SAT solver (Perron and Furnon 2022) as backend.

## Preliminary results

To compare the guided solver, we implemented a baseline approach solves each routing instance independently, without relying on experience gathered from previously solved instances. Therefore, the models is limited to equations (5) and (6) for unguided solving. Each instance is solved directly in the reduced graph $G'$, where the goal is to determine both the optimal tour length and at least one corresponding optimal visitation order $\Psi$ of visited locations in $M$. The complete route in the original road network $G$ can later be reconstructed using precomputed shortest paths between visited locations. The efficiency of this direct solving approach, executed on a single CPU thread, is adopted as our performance baseline, against which we later compare the GNN-guided solving strategy.

Solver performances are presented for instances of 80 mandatory waypoints on the Reykjavik area. To enable comparisons across instances of heterogeneous optimal length, solution quality is assessed using the normalized gap to optimality, defined in (10).

$$Performance = \frac{D(\Psi) - D(\Psi_{Opt})}{D(\Psi_{Opt})} \tag{10}$$

Here, $D(\Psi)$ denotes the length of the candidate solution and $D(\Psi_{Opt})$ the length of the optimal solution. $D(\Psi_{Opt})$ is obtained through unguided multi-threaded solving without time limitation. We observe the guidance effect on single-threaded solving by tracking the quality of intermediate solutions over solving time, both for unguided and guided solver, thereby characterizing specifically the convergence behavior of the guided search compared to unguided version.

Figure 4 reports solution quality trajectories for single threaded unguided and guided solvers on 100 problem instances. The guided solver rapidly outperforms the unguided baseline across all instances. This sharp improvement demonstrates that neural network guidance effectively steers the search toward promising edges, yielding higher-quality solutions more quickly overall. Using instance of 80 waypoints also demonstrates that the advantage of guided search is efficient well beyond training distribution.

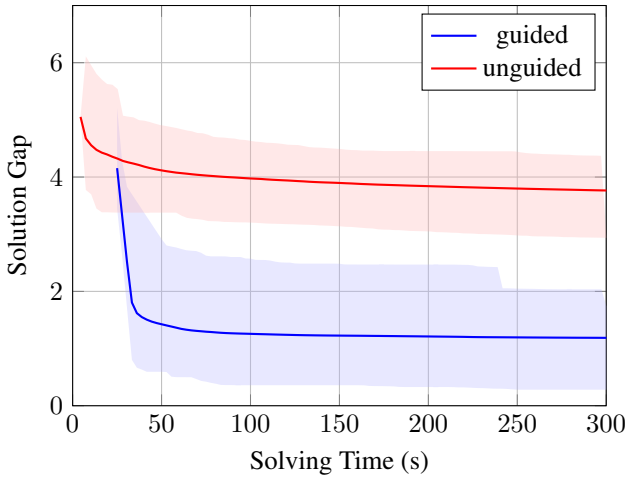


Figure 4: *Results for both guided and unguided solving for instances of 80 waypoints in Reykjavik city area for 100 problem instances. The average and complete range (i.e. from min to max) of intermediate solution quality are represented. Intermediate solution for guided solver starts after 25 seconds of initial domain contraction*

## Conclusion

We presented a hybrid neuro-symbolic approach combining GNN with CP to address recurring routing problems on real road networks. A central idea of this work is to learn from past experiences to accelerate new problem instance solving, instead of treating each problem independently as in classical operational research. Often, solving a small part of NP-hard problem does not help addressing the global problem. However, our methodology is based on repeatedly solving small instances to generate training data, then train a neural network to score edges between waypoints.

Our experiments show that the proposed GNN architecture generalizes across instances of varying larger sizes.

Although the GNN is trained on a specific city area and must be retrained for each new city, the problem setting is relevant to many real-world applications where the navigation network is fixed but the tasks positions vary dynamically, such as parcel delivery, inspection, or taxi dispatching.

Overall, this hybrid approach provides simultaneously practical performance improvements and strong theoretical guarantees: generated solutions remain valid according to problem definition, and optimality can always be reached given enough time in contrast with purely ML-based approaches where solutions do not yield guarantees. This is a step toward more reliable and transferable ML-guided optimization methods in realistic routing contexts. While the single-vehicle use case is a reasonable first step, practical logistics applications typically involve fleets of vehicles (CVRP-like problems). Therefore, extending the current approach in this way seems like a natural step.

## Acknowledgements

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Commission. Neither the European Union nor the granting authority can be held responsible for them. This work was supported by the European Union under the EDF Project FaRADAI (grant number 101103386).

## References

Braekers, K.; Ramaekers, K.; and Nieuwenhuyse, I. V. 2016. The vehicle routing problem: State of the art classification and review. *Computers & Industrial Engineering*, 99: 300–313.

Chen, F.; Wang, Y.-C.; Wang, B.; and Kuo, C.-C. 2020. Graph representation learning: a survey. *APSIPA Transactions on Signal and Information Processing*, 9.

Dantzig, G. B.; and Ramser, J. H. 1959. The Truck Dispatching Problem. *Management Science*, 6(1): 80–91.

Held, M.; and Karp, R. M. 1961. A dynamic programming approach to sequencing problems. In *Proceedings of the 1961 16th ACM National Meeting*, ACM '61, 71.201–71.204. New York, NY, USA: Association for Computing Machinery. ISBN 9781450373883.

Hudson, B.; Li, Q.; Malencia, M.; and Prorok, A. 2022. Graph Neural Network Guided Local Search for the Traveling Salesperson Problem. ArXiv:2110.05291 [cs, stat].

Johnson, D. S.; and McGeoch, L. A. 1997. The traveling salesman problem: A case study in local optimization. *Local search in combinatorial optimization*, 1(1): 215–310.

Joshi, C. K.; Cappart, Q.; Rousseau, L.-M.; and Laurent, T. 2022. Learning the Travelling Salesperson Problem Requires Rethinking Generalization. ArXiv:2006.07054 [cs, stat].

Karp, R. 1972. Reducibility Among Combinatorial Problems. volume 40, 85–103. ISBN 978-3-540-68274-5.

Kool, W.; van Hoof, H.; and Welling, M. 2019. Attention, Learn to Solve Routing Problems! In *International Conference on Learning Representations (ICLR)*.

Morris, C.; Ritzert, M.; Fey, M.; Hamilton, W. L.; Lenssen, J. E.; Rattan, G.; and Grohe, M. 2021. Weisfeiler and Leman Go Neural: Higher-order Graph Neural Networks. ArXiv:1810.02244 [cs, stat].

Nilsson, C. 2003. Heuristics for the Traveling Salesman Problem.

Osanlou, K.; Bursuc, A.; Guettier, C.; Cazenave, T.; and Jacopin, E. 2019. Optimal Solving of Constrained Path-Planning Problems with Graph Convolutional Networks and Optimized Tree Search. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 3519–3525. Macau, China: IEEE. ISBN 978-1-72814-004-9.

Peng, Y.; Choi, B.; and Xu, J. 2021. Graph Learning for Combinatorial Optimization: A Survey of State-of-the-Art. *Data Science and Engineering*, 6(2): 119–141.

Perron, L.; and Furnon, V. 2022. OR-Tools.

Scarselli, F.; Gori, M.; Tsoi, A. C.; Hagenbuchner, M.; and Monfardini, G. 2009. The Graph Neural Network Model. *IEEE Transactions on Neural Networks*, 20(1): 61–80.

Steiglitz, K.; and Weiner, P. 1968. Some Improved Algorithms for Computer Solution of the Traveling Salesman Problem.

Vesselinova, N.; Steinert, R.; Perez-Ramirez, D. F.; and Boman, M. 2020. Learning Combinatorial Optimization on Graphs : A Survey With Applications to Networking. *IEEE Access*, 8: 120388–120416. QC 20201118.

Vinyals, O.; Fortunato, M.; and Jaitly, N. 2017. Pointer Networks. ArXiv:1506.03134 [cs, stat].