# Inverse Optimization for Dynamic Vehicle Routing

Pedro Zattoni Scroccaro[1], Peyman Mohajerin Esfahani[1], and Bilge Atasoy*[1]

[1]Faculty of Mechanical Engineering, Delft University of Technology, The Netherlands

February 14, 2025

Keywords: inverse optimization, dynamic routing, prize-collecting VRP, dispatching policy

## 1 INTRODUCTION

In Inverse Optimization (IO) problems, our goal is to model the behavior of an expert agent, which given an exogenous signal, returns a response action. It is assumed that to compute its response, the expert agent solves an optimization problem that depends on the exogenous signal, and with an unknown cost function. Therefore, given a dataset of examples of signals and corresponding expert responses, we use IO to learn the cost function used by the expert agent (Zattoni Scroccaro *et al.*, 2024b). The assumption is that we have access to corresponding expert responses (aka decisions) or that we can generate those based on available data.

We have earlier developed a tailored IO methodology to efficiently solve static routing problems by learning the cost of edges on the graph. We applied and showed the promising performance of IO on the *Amazon Last Mile Routing Research Challenge*, proposed by Amazon.com, Inc. in 2021. The final score of our IO approach is **0.0302**, which ranks 2nd compared to the 48 models that qualified for the final round of the Amazon Challenge (Zattoni Scroccaro *et al.*, 2024a).

In this paper, we extend the methodology to handle dynamic routing problems. The inverse optimization approach in this case learns a dispatching policy from data given a dynamic vehicle routing problem with time windows (DVRPTW) where requests appear in the system dynamically. The contribution of our work is the inverse optimization methodology for learning the dispatching policy instead of a deep learning structure proposed in the literature.

## 2 PROBLEM STATEMENT

The considered problem in this paper is inspired by the dynamic variant of the *EURO Meets NeurIPS 2022 Vehicle Routing Competition*, which concerns a DVRPTW with unlimited fleet (see Kool *et al.* (2022)). For this problem, each day is divided into 1-hour epochs. At the start of each epoch, a certain number of requests are received, where each request corresponds to a customer with a certain demand and time windows that need to be respected. Each request can either be dispatched at the end of the current epoch or postponed to the next one. At the final epoch of the day, all requests must be dispatched. Naive policies such as "greedy" (i.e., dispatching all requests as soon as they arrive) or "lazy" (i.e., dispatching requests only when their time windows make them infeasible for the next epoch) are in most cases sub-optimal, and coming up with good dispatching policies is a challenging problem due to the uncertainty and stochasticity of future requests.

We formalize the problem as given in Algorithm 1. The goal is to learn a dispatching policy that for each epoch $t \in [T]$, dispatches a subset of the currently available customers $x_t \subseteq A_t := \{z_{1t}, \dots, z_{m_t t}\}$, and postpones the rest. At the beginning of each epoch, a set of new customers $w_t$ is added to the set of available customers $A_t$. We call $\{w_t\}_{t=1}^{T}$ a DVRPTW instance which

in a sense represents the dynamic aspect of the problem in terms of the arrival of customers over time. Let $c(x_t) \in \mathbb{R}$ be the cost of the routes to serve the customers in $x_t$. Then, given a DVRPTW instance, the goal is to minimize the total cost (e.g., total driven distance of the vehicles) for the whole day).

**Algorithm 1** DVRPTW scenario

1: **for** $t = 1, \ldots, T$ **do**
2: Set of available customers: $A_t$
3: Choose $x_t \subseteq A_t$ customers to dispatch
4: New customers: $w_{t+1}$
5: Update set of available customers: $A_{t+1} = (A_t \setminus x_t) \cup w_{t+1}$
6: **end for**
7: **Total cost:** $\sum_{t=1}^{T} c(x_t)$.

To generate a training dataset of requests and expert routes, we use the *best in hindsight* solution to the DVRPTW, that is, the optimal dispatching policy with knowledge of future requests. In practice, when choosing $x_t$, we do not have the access to $w_k$ for $k > t$. Nonetheless, given instances of DVRPTWs, i.e., a sequence of $\{w_t\}_{t=1}^{T}$, we can use (1) to compute the best dispatching sequence $x_1, \ldots, x_T$ in hindsight.

$$\begin{aligned} \text{DVRPTW}(\{w_t\}_{t=1}^{T}) := \min_{x_1,\ldots,x_T} \quad & \sum_{t=1}^{T} c(x_t) \\ \text{s.t.} \quad & x_t \subseteq A_t, && \forall t \in [T] \\ & A_{t+1} = (A_t \setminus x_t) \cup w_{t+1}, && \forall t \in [T-1] \\ & A_1 = w_1. \end{aligned} \tag{1}$$

Notice: Although not explicit in (1), the *order* of the customers in $x_t$ matters, since it defines the vehicle routes, which affect the total cost of the route $c(x_t)$.

# 3 INVERSE OPTIMIZATION APPROACH

The idea behind the IO approach is illustrated in Figure 1. Given a DVRPTW instance $\{w_t\}_{t=1}^{T}$, we compute the best solution in hindsight $\{x_1^\star, \ldots, x_T^\star\}$ by solving (1). We then use this data to create the signal-response IO dataset $\{(A_t, x_t^\star)\}_{t=1}^{T}$. As a Forward Optimization Problem, we use a Prize-collecting VRP (PCVRPTW):

$$\begin{aligned} \text{FOP}(\theta, A_t) := \arg\min_{x_t} \quad & c(x_t) + \sum_{z_{it} \in A_t \setminus x_t} p_\theta(z_{it}) \\ \text{s.t.} \quad & x_t \subseteq A_t, \end{aligned} \tag{2}$$

where $p_\theta(z_{it})$ is the prize of the customer with feature vector $z_{it}$. For instance, we can model it linearly as $p_\theta(z_{it}) = \langle \theta, z_{it} \rangle$.

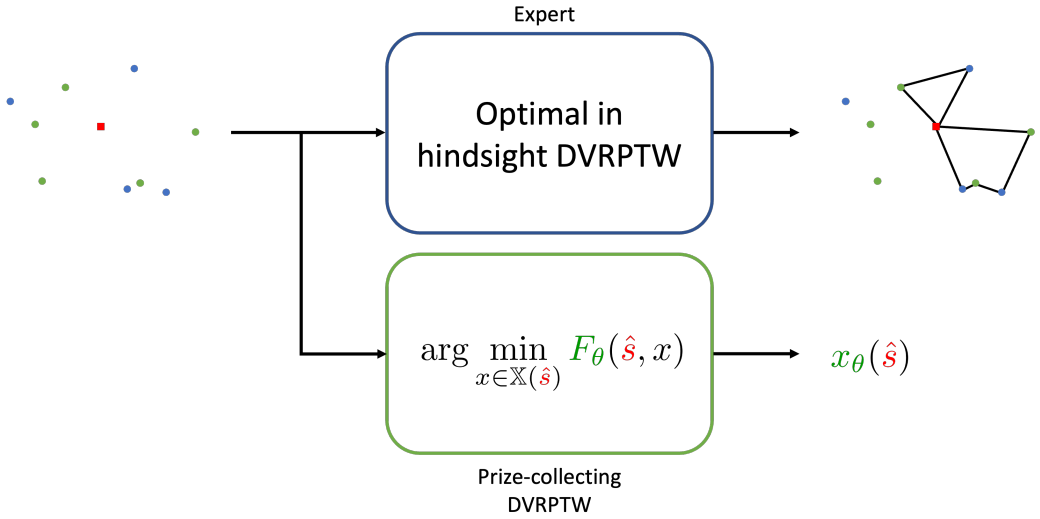


Figure 1 – *Illustration of the IO approach on dynamic VRP*

A PCVRPTW is a routing problem where each request is associated with a "prize" and not all requests need to be served (also used as the underlying formulation by Baty *et al.* (2024) who won the competition). The goal is to collect the maximum amount of prizes by dispatching its respective request, while also minimizing the travel distance. This way, requests with low prizes are not served, which in a dynamic routing problem are postponed to the next epoch. In this case, using the optimal dispatches computed by the hindsight expert to learn the cost function of a PCVRPTW is equivalent to learning a "prizing function" that assigns a score to each request, and then dispatches the requests with a large prize and postpones the ones with a small prize, according to the PCVRPTW solution.

The training of the IO approach is handled through a training algorithm given in Algorithm 2 using a linear prize function $p_\theta(z_{it}) = \langle \theta, z_{it} \rangle$. The algorithm starts with an initial set of parameters $\theta_1$ and the set of computed best solutions in hindsight $\mathcal{D} = \{(A_t, x_t^\star)\}_{t=1}^T$. With the current $\theta$ parameters and for a given example from the sample, the set of dispatched customers is optimized with the FOP. The "gradient" $g_k$ then represents the difference between this solution and the best solution in hindsight in terms of the features of the selected customers. Based on this gradient, the parameters are updated and the algorithm continues until a predefined maximum number of iterations to generate the learned $\theta$ parameters. This algorithm is based on the suboptimality loss and stochastic gradient descent ideas (see Zattoni Scroccaro *et al.* (2024b) and Zattoni Scroccaro *et al.* (2024a)).

**Algorithm 2** Stochastic first-order algorithm

**Input:** $\theta_1 \in \Theta$ and $\mathcal{D} = \{(A_t, x_t^\star)\}_{t=1}^T$.
**for** $k = 1, \ldots, N$ **do**
  Sample example: $(A_k, x_k^\star)$
  $x_k \in \text{FOP}(\theta_k, A_k)$
  $g_k = \sum_{z_{ik} \in A_k \setminus x_k^\star} z_{ik} - \sum_{z_{ik} \in A_k \setminus x_k} z_{ik}$
  $\theta_{k+1} = \theta_k - \eta_k g_k$
**end for**

# 4 RESULTS

A dataset consisting of real-world historical DVRPTW instances was provided in the context of the *EURO Meets NeurIPS 2022 Vehicle Routing Competition*. Different methods were evaluated by a test dataset which we also use in this study.

Based on the training algorithm (Algorithm 2) we learn the $\theta$ parameters that relate the features of the customers to the prize to be collected upon dispatching them. These learned parameters, i.e., learned dispatching policy, are then used on the test data to make decisions on dispatching or delaying the available customers.

The current results we obtained demonstrate intuitive learned parameters of the price function such that the collected prize of a customer increases if (i) the demand of the customer is larger, (ii) the time window starts sooner, (iii) its location is further to the depot and (iv) the service time is longer.

The best score we obtained on the test data is around **354000** which is between the first and the second score (for the dynamic variant) published as a result of the competition [1]. Therefore, our proposed approach is promising to tackle dynamic vehicle routing problems without relying on deep learning techniques which necessitate a thorough tuning of the involved parameters. This is ongoing work and at the conference, we plan to present a comprehensive analysis of the results including computational efficiency.

There are various possibilities to further improve the performance of the method. Algorithm 2 is a quite standard algorithm in terms of the gradient updates and could be further tested with more iterations or with different sampling strategies. The price function we have now is linear in features and can be investigated with further specifications and also possibly with Kernel approaches relating the price to the already seen set of customers.

## Acknowledgements

This work is co-funded by the European Union (ERC, ADAPT-OR, No 101117675). 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 Research Council. Neither the European Union nor the granting authority can be held responsible for them.

## References

Baty, Léo, Jungel, Kai, Klein, Patrick S, Parmentier, Axel, & Schiffer, Maximilian. 2024. Combinatorial optimization-enriched machine learning to solve the dynamic vehicle routing problem with time windows. *Transportation Science*, **58**(4), 708–725.

Kool, Wouter, Bliek, Laurens, Numeroso, Danilo, Zhang, Yingqian, Catshoek, Tom, Tierney, Kevin, Vidal, Thibaut, & Gromicho, Joaquim. 2022. The EURO meets NeurIPS 2022 vehicle routing competition. *Pages 35–49 of: NeurIPS 2022 Competition Track*. PMLR.

Zattoni Scroccaro, Pedro, van Beek, Piet, Mohajerin Esfahani, Peyman, & Atasoy, Bilge. 2024a. Inverse Optimization for Routing Problems. *Transportation Science*.

Zattoni Scroccaro, Pedro, Atasoy, Bilge, & Mohajerin Esfahani, Peyman. 2024b. Learning in Inverse Optimization: Incenter Cost, Augmented Suboptimality Loss, and Algorithms. *Operations Research*.

[1] https://euro-neurips-vrp-2022.challenges.ortec.com/