Machine Learning and Combinatorial Optimization
for the Dynamic Vehicle Routing Problem

Léo Baty\textsuperscript{1}, Kai Jungel\textsuperscript{2}, Patrick Klein\textsuperscript{2}, Maximilian Schiffer\textsuperscript{2},
Axel Parmentier\textsuperscript{1}

\textsuperscript{1}CERMICS, École des Ponts, \textsuperscript{2}Technical University of Munich

January 25, 2022
1 Problem statement
2 Machine Learning pipeline
3 Learning approach
4 Results
Problem statement

- Static problem
- Dynamic problem

Machine Learning pipeline

Learning approach

Results
Vehicle Routing Problem with Time Windows (VRPTW)

**Depot**: vehicles capacity $Q$

**Requests** $v \in V$

- Request 1
- Request 2
- Request 3
- Request 4
- Request 5
- Request 6
Vehicle Routing Problem with Time Windows (VRPTW)

**Depot**: vehicles capacity $Q$

**Requests** $v \in V$

1. Coordinates $p$
   \[ \Rightarrow \] costs $c_{v,v'}$

\[
\begin{align*}
(0.75, 2.5) & \\
(-0.6, 1.5) & \\
(-2.0, 1) & \\
(0, 0) & \\
(1, 1) & \\
(0, 0.75) & \\
(-1.0, -1.0) & \\
(1, -0.5) & \\
\end{align*}
\]
Vehicle Routing Problem with Time Windows (VRPTW)

**Depot**: vehicles capacity $Q$

**Requests** $v \in V$

1. Coordinates $p$
   $\Rightarrow$ costs $c_{v,v'}$
2. Time Windows $[\ell, u]$
Vehicle Routing Problem with Time Windows (VRPTW)

**Depot**: vehicles capacity $Q$

**Requests** $v \in V$

1. Coordinates $p$  
   $\Rightarrow$ costs $c_{v,v'}$
2. Time Windows $[\ell, u]$
3. Demand $q$
Vehicle Routing Problem with Time Windows (VRPTW)

**Problem statement**

**Depot**: vehicles capacity $Q$

**Requests** $v \in V$

1. Coordinates $p$  
   $\Rightarrow$ costs $c_{v,v'}$

2. Time Windows $[\ell, u]$

3. Demand $q$

4. Service time $s$
Vehicle Routing Problem with Time Windows (VRPTW)

**Depot:** vehicles capacity $Q$

**Requests** $v \in V$

1. Coordinates $p$  
   $\Rightarrow$ costs $c_{v,v'}$
2. Time Windows $[\ell,u]$
3. Demand $q$
4. Service time $s$

**Objective:** build feasible routes serving all requests at minimum cost
State-of-the-art algorithm: Hybrid Genetic Search (HGS)

- Genetic algorithm
- Maintains a population of solutions
- Improves it over the iterations using crossover combined with neighborhood searches

See [Vidal, 2021] for details.
1. Problem statement
   - Static problem
   - Dynamic problem

2. Machine Learning pipeline

3. Learning approach

4. Results
Dynamic VRPTW

- Same setting as the static problem
- Discrete time horizon $[T]$, 1-hour epochs
- New requests arrive at the start of each epoch
  $\Rightarrow$ future requests are not known in advance
Dynamic VRPTW: example

Start of epoch 1: requests arrive
Dynamic VRPTW: example

- Decide which request to **dispatch**
- Build routes serving them, other requests are **postponed**
- Each request must be served before end of its time window
  ⇒ some requests must be dispatched
Dynamic VRPTW: example

- Decide which request to **dispatch**
- Build routes serving them, other requests are **postponed**
- Each request must be served before end of its time window
  ⇒ some requests must be dispatched
Dynamic VRPTW: example

► Start of epoch 2: new requests arrive
Dynamic VRPTW: example

- Start of epoch 2: new requests arrive
- Epoch 2 routes
Summary

- At every epoch $t$:
  - Decide which request to **dispatch**
  - Build routes serving them, other requests are **postponed**
  - Each request must be served before end of its time window
    $\Rightarrow$ some requests must be dispatched

- **State** $x_t$ of the system at epoch $t$: set of requests arrived at $t$
  or arrived before but not yet served

- **Objective**: serve all requests, minimize total travel distance

$\Rightarrow$ no state-of-the-art
1 Problem statement
   • Static problem
   • Dynamic problem

2 Machine Learning pipeline

3 Learning approach

4 Results
Policy based on a Deep Learning pipeline

Epoch decisions can be seen as the solution of a Prize Collecting VRPTW:

- Serving requests is optional
- Serving request $v$ gives prize $\theta_v$
- **Objective**: maximize total profit minus costs

$$\max_{y \in \mathcal{Y}(x_t)} \sum_{(u,v) \in x_t^2} (\theta_v - c_{u,v})y_{u,v}.$$ 

- **Algorithm**: Prize Collecting Hybrid Genetic Search
  \[\rightarrow\] Combinatorial Optimization layer
Problem: we have no way of computing meaningful prizes
Policy based on a Deep Learning pipeline

**Solution:** use a neural network to predict request prizes $\theta_v$

**Goal:** find parameters $w$ such that our pipeline is a “good” policy.
1. Problem statement
   - Static problem
   - Dynamic problem

2. Machine Learning pipeline

3. Learning approach

4. Results
Learn to imitate an anticipative policy

Anticipative policy

▸ At $t = 0$, we assume that we know all future requests.
▸ Optimal solution by solving a VRPTW with release times
⇒ Hybrid Genetic Search
Learn to imitate an anticipative policy

**Anticipative policy**

- At $t = 0$, we assume that we know all future requests.
- Optimal solution by solving a VRPTW with release times

$\Rightarrow$ Hybrid Genetic Search

Dataset labeled with anticipative decisions:

$$\mathcal{D} = \{(x^1, y^1), \ldots, (x^n, y^n)\}$$
Learn to imitate an anticipative policy

**Anticipative policy**

- At $t = 0$, we assume that we know all future requests.
- Optimal solution by solving a VRPTW with release times

⇒ Hybrid Genetic Search

Dataset labeled with anticipative decisions:

$$\mathcal{D} = \{(x^1, \bar{y}^1), \ldots, (x^n, \bar{y}^n)\}$$

Can we apply classical supervised learning techniques?
A natural loss function

Combinatorial Optimization (CO) layer:

\[ f: \theta \mapsto \arg\max_{y \in \mathcal{Y}(x_t)} \sum_{(u,v) \in x_t^2} (\theta_v - c_{u,v})y_{u,v}. \]

\[ f: \theta \mapsto \arg\max_{y \in \mathcal{Y}(x_t)} \theta^\top g(y) + h(y) \]

with \( g(y) = (\sum_{u \in V} y_{u,v})_{v \in V}, \) and \( h(y) = -\sum_{(u,v) \in x_t^2} c_{u,v}y_{u,v} \)
A natural loss function

\[ f : \theta \mapsto \arg\max_{y \in \mathcal{Y}(x_t)} \theta^\top g(y) + h(y) \]

When we apply Automatic Differentiation (AD) to a CO oracle:

- It usually doesn’t work (lack of compatibility with solver)
- Even when it does, the Jacobian is either zero or undefined
A natural loss function

\[ f: \theta \longmapsto \arg\max_{y \in \mathcal{Y}(x_t)} \theta^\top g(y) + h(y) \]

When we apply Automatic Differentiation (AD) to a CO oracle:
- It usually doesn’t work (lack of compatibility with solver)
- Even when it does, the Jacobian is either zero or undefined

Natural loss function

Non-optimality of target routes \( \bar{y} \) as a solution of \( f \)

\[ \mathcal{L}(\theta, \bar{y}) = \max_{y \in \mathcal{Y}(x)} \{ \theta^\top g(y) + h(y) \} - (\theta^\top g(\bar{y}) + h(\bar{y})) \]

**Problem:** \( \theta = 0 \) is a trivial solution that minimizes \( \mathcal{L}(\cdot, \bar{y}) \)
Regularization through perturbation

Perturb the objective with an additive noise [Berthet et al., 2020]:

$$
\hat{f}_\varepsilon : \theta \mapsto \mathbb{E} \left[ \arg\max_{y \in \mathcal{Y}(x_t)} (\theta + \varepsilon Z)^\top g(y) + h(y) \right] = \mathbb{E}[f(\theta + \varepsilon Z)]
$$

with $Z \sim \mathcal{N}(0, 1)$, and $\varepsilon \in \mathbb{R}_+$. 

Intractable expectation $\Rightarrow$ Monte-Carlo sampling approximation
Fenchel-Young loss

Perturbed Fenchel-Young loss

\[
\mathcal{L}_\varepsilon(\theta, \bar{y}) = \mathbb{E} \left[ \max_{y \in \mathcal{Y}(x_t)} (\theta + \varepsilon Z) \top g(y) + h(y) \right] - (\theta \top g(\bar{y}) + h(\bar{y})) ,
\]

\[
g(\hat{f}_\varepsilon(\theta)) - g(\bar{y}) \in \partial_\theta \mathcal{L}_\varepsilon(\theta, \bar{y}).
\]

Learning problem:

\[
\arg\min_w \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}_\varepsilon(\varphi_w(x^i), \bar{y}^i)
\]
How to implement this pipeline?

Our package InferOpt.jl [Dalle et al., 2022], written in Julia:

- Open source: https://github.com/axelparmentier/InferOpt.jl
- Easy to use
- Works with any CO oracle, independent of the implementation
- Compatible with Julia ML and AD ecosystem (through ChainRules.jl)
1 Problem statement
   - Static problem
   - Dynamic problem

2 Machine Learning pipeline

3 Learning approach

4 Results
Results: 4.4% average gap

Benchmark on 2252 instances-seed combinations:
Winner team of Euro-NeurIPS competition

<table>
<thead>
<tr>
<th>Rank</th>
<th>Team name</th>
<th>Dynamic cost</th>
<th>Static rank</th>
<th>Dynamic rank</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Kléopatra</td>
<td>348831.56</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>OptiML</td>
<td>359270.09</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>Team_SB</td>
<td>358161.36</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>HustSmart</td>
<td>361803.57</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>Miles To Go Before We Sleep</td>
<td>369098.13</td>
<td>4</td>
<td>7</td>
</tr>
</tbody>
</table>
