Winning Approach for the EURO-NeurIPS 2022 Dynamic Vehicle Routing Competition

Léo Baty[1], Kai Jungel[2], Patrick Klein[2], Maximilian Schiffer[2], Axel Parmentier[1]

[1]CERMICS, École des Ponts, [2]Technical University of Munich

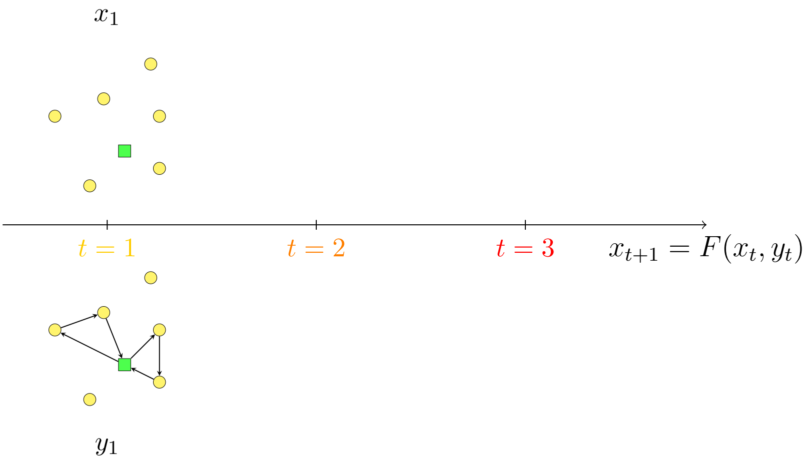
February 21, 2023

École des Ponts
ParisTech

Dynamic Vehicle Routing Problem with Time Windows (Dynamic VRPTW)

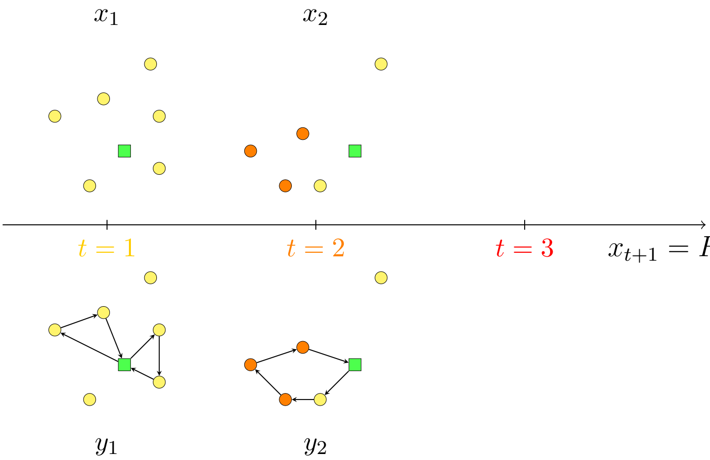
State
$x_t \in \mathcal{X}$
set of customers

Decision
$y_t \in \mathcal{Y}(x_t)$
set of routes



Dynamic Vehicle Routing Problem with Time Windows (Dynamic VRPTW)

State
$x_t \in \mathcal{X}$
set of customers



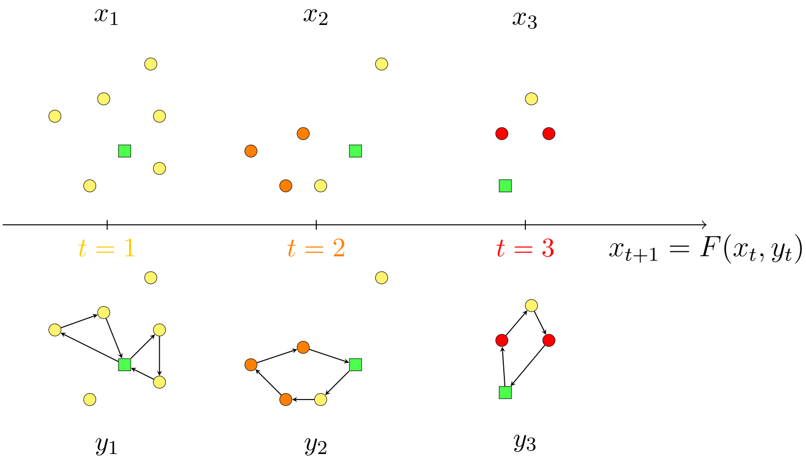
$x_{t+1} = F(x_t, y_t)$

Decision
$y_t \in \mathcal{Y}(x_t)$
set of routes

Dynamic Vehicle Routing Problem with Time Windows (Dynamic VRPTW)

State
$x_t \in \mathcal{X}$
set of customers

Decision
$y_t \in \mathcal{Y}(x_t)$
set of routes



Dynamic VRPTW

A solution of this problem is a **stationary policy**:

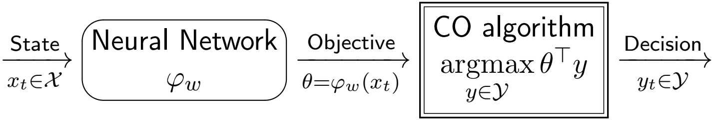
$$\pi : \mathcal{X} \to \mathcal{Y}$$
$$x_t \mapsto y_t$$

Objective: find $\pi^\star$, serving all customers before end of horizon, and minimizing total cost

$$\pi^\star = \underset{\pi}{\operatorname{argmin}} \mathbb{E} \left[\sum_{\text{epochs } t} \text{total cost of routes in decision } y_t = \pi(x_t) \right]$$

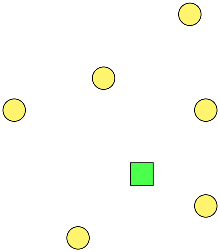
Winner team of the EURO-NeurIPS challenge

- Euro-NeurIPS competition[1]
  - 100 entering customers at each time step
  - maximum 2 minutes per time step
- Our team won first prize of the challenge
- Policy π_w, Machine Learning (ML) and Combinatorial Optimization (CO) pipeline

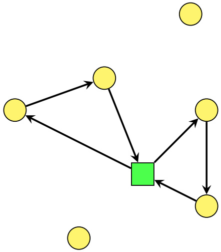


[1] https://euro-neurips-vrp-2022.challenges.ortec.com/

Policy based on a Deep Learning pipeline



State
x_t



Decision
y_t

Policy based on a Deep Learning pipeline

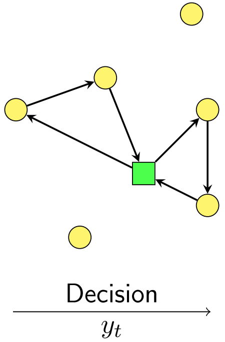
Epoch decisions can be seen as the solution of a Prize Collecting VRPTW:

- Serving customers is optional
- Serving customer v gives **prize** θ_v
- **Objective**: maximize total profit minus routes costs

$$\max_{y \in \mathcal{Y}(x_t)} \underbrace{\sum_{(u,v) \in x_t^2} \theta_v y_{u,v}}_{\text{total profit}} - \underbrace{\sum_{(u,v) \in x_t^2} c_{u,v} y_{u,v}}_{\text{total routes cost}} .$$

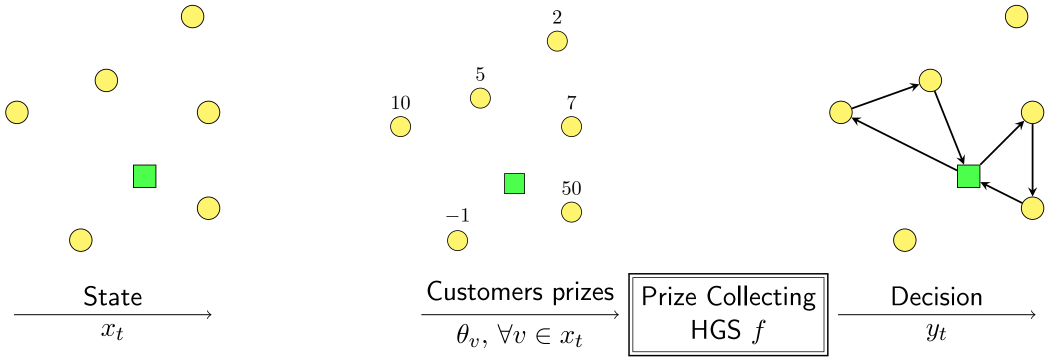
- **Algorithm**: Prize Collecting Hybrid Genetic Search

$\Rightarrow$ Combinatorial Optimization layer f



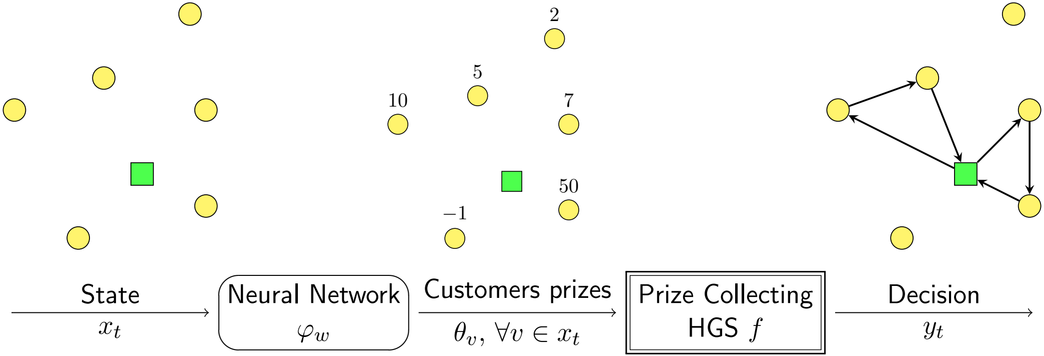
Policy based on a Deep Learning pipeline

Difficulty: no natural way of computing meaningful prizes



Policy based on a Deep Learning pipeline

Solution: use a neural network to predict request prizes $\theta = \varphi_w(x_t)$

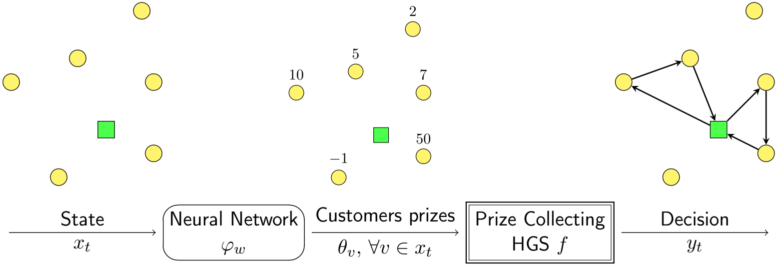


Parameterized policy: $\pi_w : x_t \longmapsto f(\varphi_w(x_t))$

1. Policy encoded as a Deep Learning pipeline
2. Training the policy
3. Results

Learning problem

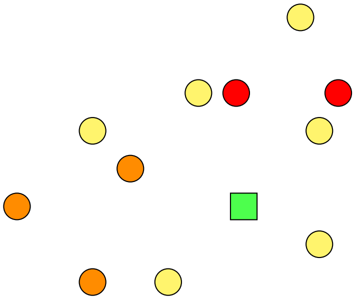
Goal: find parameters w such that our pipeline is a "good" policy.



$$\hat{w} = \underset{w}{\operatorname{argmin}} \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}(\varphi_w(x^i), \bar{y}^i)$$

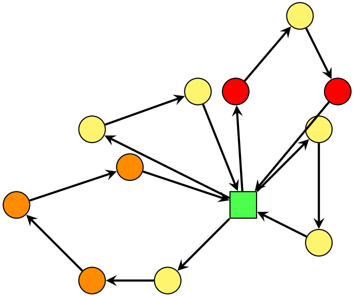
We need to build a labeled dataset $\mathcal{D} = \{(x^1, \bar{y}^1), \dots, (x^n, \bar{y}^n)\}$.

Learn to imitate anticipative decisions



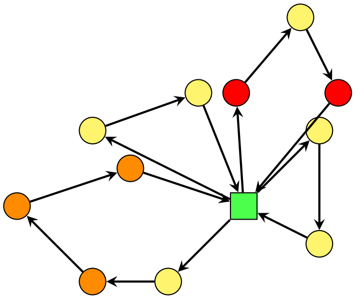
- Full instance with all future customers
- Release times:
  - $t = 1$
  - $t = 2$
  - $t = 3$

Learn to imitate anticipative decisions

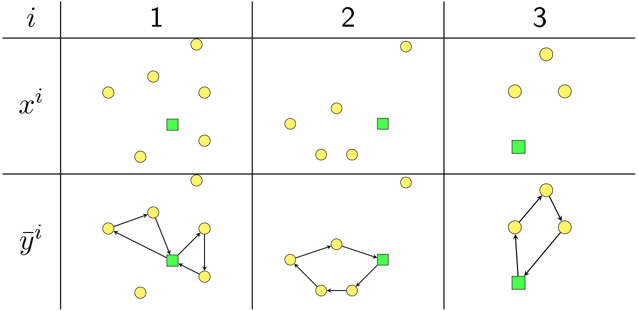


- Full instance with all future customers
- Release times:
  - $t = 1$
  - $t = 2$
  - $t = 3$
- Hybrid Genetic Search [Vidal, 2021]
- Anticipative lower bound

Learn to imitate anticipative decisions



We rebuild the anticipative decisions a posteriori



A natural loss function

$(x, \bar{y}) \in \mathcal{D}$, $\theta = \varphi_w(x)$

$$f : \theta \longmapsto \underset{y \in \mathcal{Y}(x)}{\operatorname{argmax}} \, \theta^\top g(y) + h(y)$$

with $g(y) = \left(\sum_{u \in x} y_{u,v}\right)_{v \in x}$ and $h(y) = -\sum_{(u,v) \in x^2} c_{u,v} y_{u,v}$

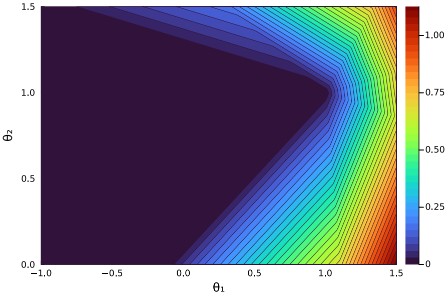
Non-optimality of target routes $\bar{y}$ as a solution of f

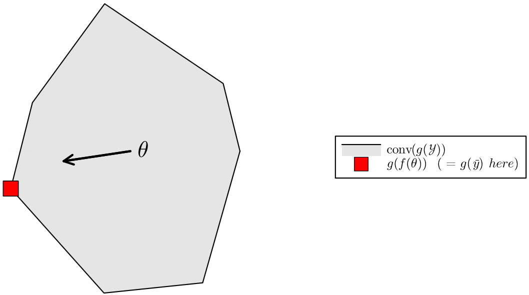
$$\mathcal{L}(\theta, \bar{y}) = \max_{y \in \mathcal{Y}} \{\theta^\top g(y) + h(y)\} - (\theta^\top g(\bar{y}) + h(\bar{y}))$$

A natural loss function

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

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





$$\theta \in \mathbb{R}^2 \mapsto \mathcal{L}(\theta, \bar{y})$$

$\Rightarrow$ Non smooth + degeneracy

Building a differentiable loss

Theorem [Berthet et al., 2020, Baty et al., 2023]

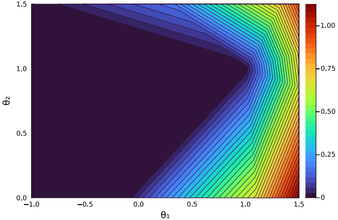
The perturbed loss function

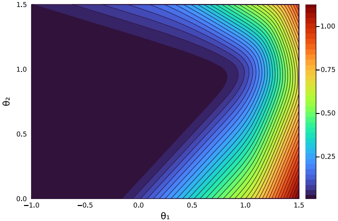
$$\mathcal{L}_\varepsilon(\theta, \bar{y}) = \mathbb{E}\left[\max_{y \in \mathcal{Y}} (\theta + \varepsilon Z)^\top g(y) + h(y)\right] - (\theta^\top g(\bar{y}) + h(\bar{y}))$$

with $\varepsilon \in \mathbb{R}_+$, and $Z \sim \mathcal{N}(0, I_d)$, is convex and differentiable in θ

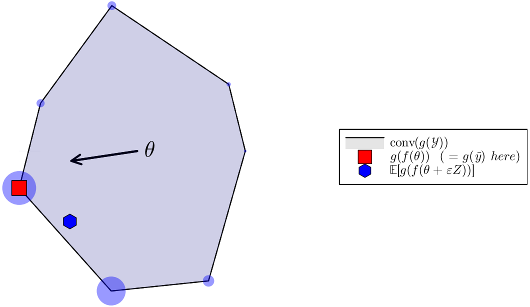
$$\begin{aligned}\nabla_\theta \mathcal{L}_\varepsilon(\theta, \bar{y}) &= \mathbb{E}\left[g\left(\operatorname*{argmax}_{y \in \mathcal{Y}} (\theta + \varepsilon Z)^\top g(y) + h(y)\right)\right] - g(\bar{y}) \\ &= \mathbb{E}[g(f(\theta + \varepsilon Z))] - g(\bar{y})\end{aligned}$$

$$\theta \mapsto \mathcal{L}(\theta, \bar{y})$$





$$\theta \mapsto \mathcal{L}_\varepsilon(\theta, \bar{y})$$



Our team Kléopatra wins the Euro-NeurIPS competition

Team name	Dynamic cost	Improvement over 9th team
Kléopatra	348831.56	5.9%
Team_SB	358161.36	3.3%
OptiML	359270.09	3.1%
HustSmart	361803.57	2.4%
ORberto Hood and the Barrymen	362481.13	2.2%
UPB	367007.49	1%
Miles To Go Before We Sleep	369098.13	0.4%
HowToRoute	369797.03	0.2%
Kirchhoffslaw	370670.53	0%

Comparison to baseline policies

Policy	Kléopatra	Rolling-horizon	Monte-Carlo
Runtime	90s	600s	5400s

Table: Runtime for each time step

Conclusion

Contributions:

- Deep Learning pipeline for the Dynamic VRPTW
- Generalization of the learning approach
  - Julia open source implementation in `InferOpt.jl`[2] [Dalle et al., 2022]

Perspectives:

- There is still room for improvement, especially on the policy to imitate

[2] https://github.com/axelparmentier/InferOpt.jl

References

Baty, L., Jungel, K., Klein, P., Parmentier, A., and Schiffer, M. (2023).
Combinatorial optimization enriched machine learning to solve the dynamic vehicle routing problem with time windows.

Berthet, Q., Blondel, M., Teboul, O., Cuturi, M., Vert, J.-P., and Bach, F. (2020).
Learning with Differentiable Perturbed Optimizers.
arXiv:2002.08676 [cs, math, stat].

Dalle, G., Baty, L., Bouvier, L., and Parmentier, A. (2022).
Learning with Combinatorial Optimization Layers: A Probabilistic Approach.

Vidal, T. (2021).
Hybrid Genetic Search for the CVRP: Open-Source Implementation and SWAP* Neighborhood.

Theorem

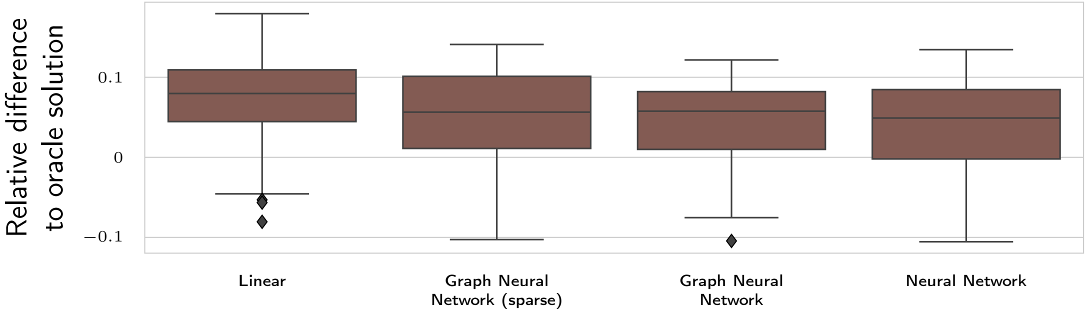
Let $x \in \mathcal{X}$, $\bar{y} \in \mathcal{Y}(x)$. Let $\mathcal{C}(\bar{y}) = \{\theta \in \mathbb{R}^{d(x)} : \theta^\top g(\bar{y}) \geq \theta^\top g(y), \forall y \in \mathcal{Y}(x)\}$ the *normal cone* associated to $\bar{y}$.

1. $\theta \mapsto \mathcal{L}(\theta, \bar{y})$ is piecewise linear and convex, with subgradient $g(f(\theta)) - g(\bar{y}) \in \partial_\theta \mathcal{L}(\theta, \bar{y})$.
2. $\theta \mapsto \mathcal{L}_\varepsilon(\theta, \bar{y})$ is $\mathcal{C}^\infty$ and convex with gradient $\nabla_\theta \mathcal{L}_\varepsilon(\theta, \bar{y}) = \mathbb{E}[g(f(\theta + \varepsilon Z))] - g(\bar{y})$.
3. $\mathcal{L}_\varepsilon(\theta, \bar{y}) \geq \mathcal{L}(\theta, \bar{y})$.
4. $\mathcal{C}(\bar{y})$ is the recession cone of $\mathcal{P}(\bar{y})$.
5. Let $\theta \in \mathbb{R}^{d(x)}$. If η is in $\mathcal{C}(\bar{y}) \backslash \{0\}$, then $\lambda \mapsto \mathcal{L}(\theta + \lambda\eta, \bar{y})$ is non increasing. If in addition $\mathcal{C}(\bar{y}) \neq \mathbb{R}^{d(x)}$, then $\lambda \mapsto \mathcal{L}_\varepsilon(\theta + \lambda\eta, \bar{y})$ is decreasing.
6. Let $\theta \in \mathbb{R}^{d(x)}$. If η is in the interior $\mathring{\mathcal{C}}(\bar{y})$ of $\mathcal{C}(\bar{y})$, then $\lim_{\lambda \to \infty} \mathcal{L}(\theta + \lambda\eta, \bar{y}) = \lim_{\lambda \to \infty} \mathcal{L}_\varepsilon(\theta + \lambda\eta, \bar{y}) = 0$.

Features

Observed		Distribution knowledge	
x coordinate	x_r	*Quantiles from distribution of travel time to all locations:*	
y coordinate	y_r	1% quantile	$Pr[X < x] \leq 0.01, X \sim t_{r,:}$
demand	q_r	5% quantile	$Pr[X < x] \leq 0.05, X \sim t_{r,:}$
service time	s_r	10% quantile	$Pr[X < x] \leq 0.1, X \sim t_{r,:}$
time window start	l_r	50% quantile	$Pr[X < x] \leq 0.5, X \sim t_{r,:}$
time window end	u_r	*Quantiles from distribution of slack time to all time windows:*	
time from depot to request	$t_{d,r}$	0% quantile	$Pr[X < x] \leq 0, X \sim u_{:} - (l_r + s_r + t_{r,:})$
relative time depot to request	$t_{d,r}/u_r - s_r)$	1% quantile	$Pr[X < x] \leq 0.01, X \sim u_{:} - (l_r + s_r + t_{r,:})$
time window start / rem. time	$l_r/(T_{max} - \tau_e)$	5% quantile	$Pr[X < x] \leq 0.05, X \sim u_{:} - (l_r + s_r + t_{r,:})$
time window end / rem. time	$u_r/(T_{max} - \tau_e)$	10% quantile	$Pr[X < x] \leq 0.1, X \sim u_{:} - (l_r + s_r + t_{r,:})$
is must dispatch	$\mathbb{1}_{\tau_e + \Delta + t_{d,r}} > u_r$	50% quantile	$Pr[X < x] \leq 0.5, X \sim u_{:} - (l_r + s_r + t_{r,:})$

Predictors


Relative difference
to oracle solution
0.1
0
−0.1
Linear
Graph Neural
Network (sparse)
Graph Neural
Network
Neural Network

Other experiments

Num. of training instances							
1	2	5	10	15	20	25	30
9.29%	6.64%	5.95%	4.56%	3.79%	4.48%	3.91%	3.84%

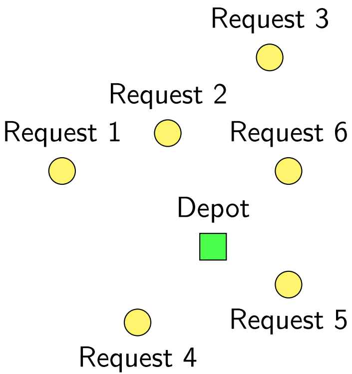
Size of training instances				
10	25	50	75	100
8.05%	5.78%	4.00%	5.06%	10.39%

Imitated upper-bound strategies			
best seed	60 min	15 min	5 min
6.68%	5.97%	4.79%	3.49%

Vehicle Routing Problem with Time Windows (VRPTW)

Depot: vehicles capacity Q

Requests $v \in V$

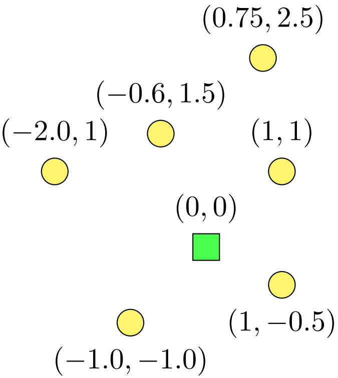


Vehicle Routing Problem with Time Windows (VRPTW)

Depot: vehicles capacity Q

Requests $v \in V$

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

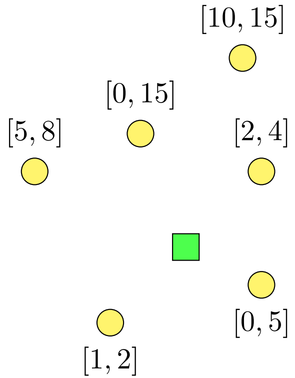


Vehicle Routing Problem with Time Windows (VRPTW)

Depot: vehicles capacity Q

Requests $v \in V$

1. Coordinates p
   $\Rightarrow$ costs $c_{v,v'}$
2. Time Windows $[\ell, u]$

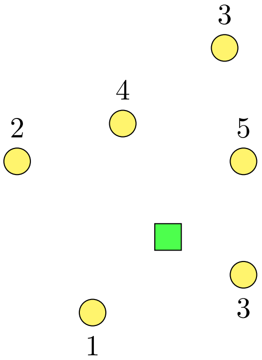


Vehicle Routing Problem with Time Windows (VRPTW)

Depot: vehicles capacity Q

Requests $v \in V$

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

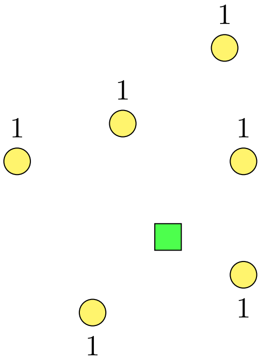


Vehicle Routing Problem with Time Windows (VRPTW)

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

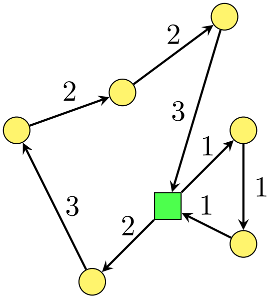


Vehicle Routing Problem with Time Windows (VRPTW)

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.