Dynamic Vehicle Routing with Stochastic Requests

Alexandre Florio

November 2, 2021

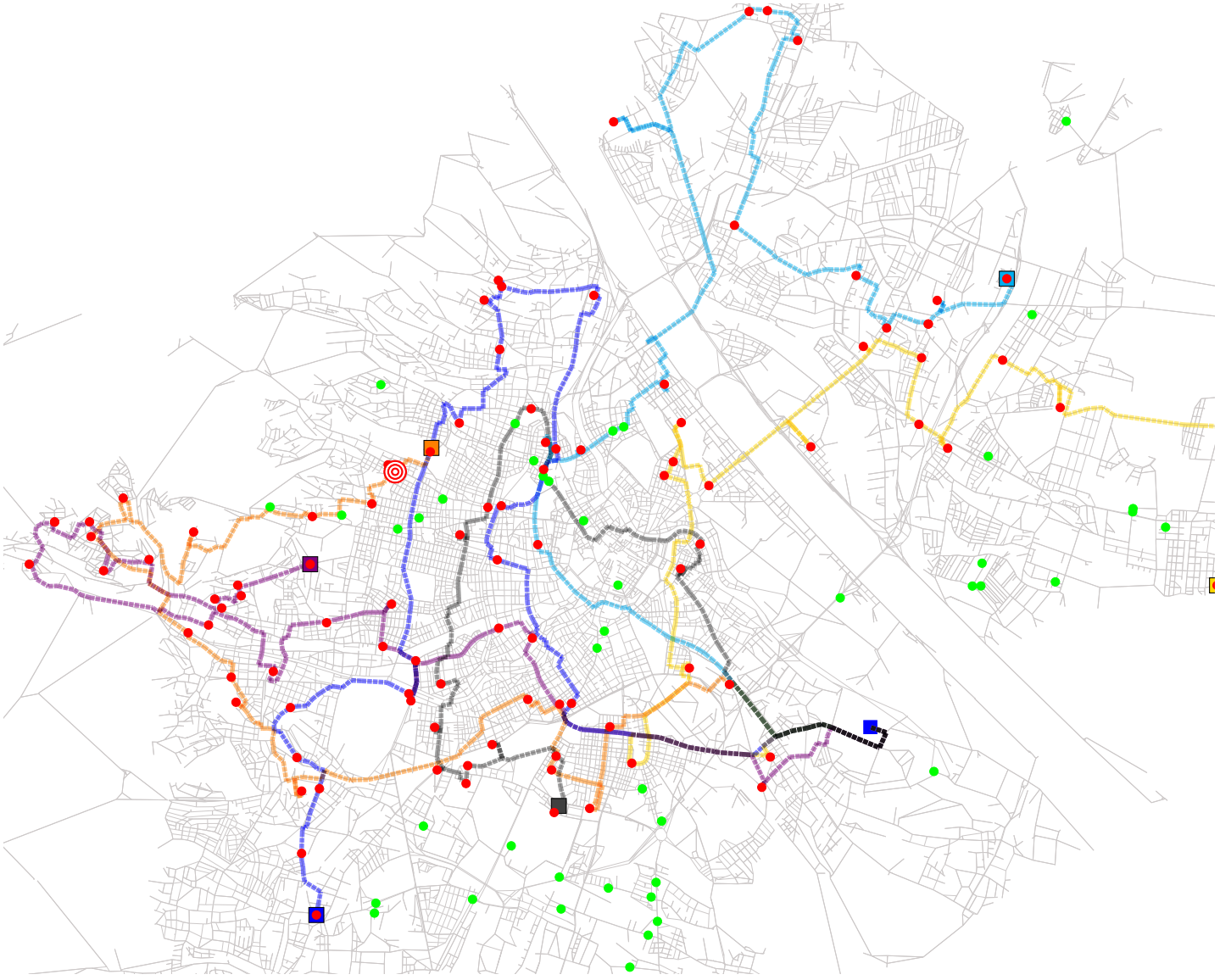
Solving Large-Scale Dynamic Vehicle Routing with Stochastic Requests in Real-Time

Joint work with Jian Zhang, Kelin Luo and Tom Van Woensel

General Setting & Motivation

- Dynamic VRP with stochastic requests (DVRPSR)
  - Static (planned) and dynamic (on-demand) requests
  - Applications: technician routing, package collection (first-mile, courier or marketplace sellers)
- **Request:** pickup (or service) location and service time
  - Requests may be rejected
- Multiple drivers/vehicles
- Vehicles must return to the depot by the given deadline
- **Goal:** serve as many requests as possible
- **Contribution:** large-scale, real-time, street networks

General Setting & Motivation



Related Literature

Literature	Problem setting				Initial plan		Online policy	
	RT	Req.	Veh.	Graph	Method	Ant.	Method	Ant.
Bent & Van Hentenryck (2004)	×	100	n/a	Synth.	MSA	✓	MSA	✓
Chen & Xu (2006)	×	100	n/a	Synth.	CGBH	×	CGBH	×
Hvattum et al. (2006)	×	130	n/a	Synth.	DSHH	✓	DSHH	✓
Ichoua et al. (2006)	✓	240	6	Synth.	TS	✓	TS	✓
Thomas (2007)	×	50	1	Synth.	GRASP	×	Heuristics	✓
Azi et al. (2012)	✓	144	5	Synth.	n/a		ALNS	✓
Ferrucci & Bock (2015, 2016)	×	150	12	SSSN	TS	✓	TS	✓
Klapp et al. (2018a)	×	40	1	Synth.	B&C	✓	Rollout	✓
Ulmer et al. (2018a)	×	100	1	Synth.	CI	×	VFA	✓
Ulmer et al. (2018b)	×	100	1	Synth.	CI	×	VFA	✓
Ulmer et al. (2019)	×	100	1	Synth.	CI	×	VFA+rollout	✓
van Heeswijk et al. (2019)	×	400	n/a	Synth., SSSN	CW	×	VFA, rollout	✓
Voccia et al. (2019)	×	192	13	Synth.	n/a		MSA	✓
Ulmer (2020)	✓	180	3	Synth.	n/a		VFA	✓
Ulmer & Thomas (2020)	✓	50	1	Synth.	n/a		VFA	✓
This paper	✓	**947**	**20**	**LSSN**	**PbCGBH**	✓	**PbPs**	✓

Methodology

- Problem formulation: sequential stochastic optimization model
  - Decisions must be taken each time a request arrives
  - Note: alternative approach, "batch and optimize"
- Key contribution: approximation of the reward-to-go (potential) by relaxed knapsack models
- First decision: initial plan: column generation-based petal heuristic
- Remaining decisions: potential-based scheduling policy

Notation

$\mathcal{G} = (\mathcal{V}, \mathcal{A})$	Street network graph
$\mathcal{V}$	Road junctions or intersections
$\mathcal{A}$	Road segments
$0 \in \mathcal{V}$	Depot
d_{ij}	Length of segment $(i, j) \in \mathcal{A}$
$t_{ij} = d_{ij}/\overline{s}$	Travel time along segment $(i, j) \in \mathcal{A}$
$t(i, j)$	Duration of the fastest path from node i to node j
$[0, U]$	Service period
K	Fleet size

Notation

$\mathcal{Z}$	Set of static (or planned) requests
$\mathcal{D}(u)$	(Ordered) set of dynamic requests up to instant $u \in [0, U]$
(u, i, d)	Request, where $u \in [0, U]$ is the arrival/release time, $i \in \mathcal{V}$ is the location and $d \in \mathbb{R}_{>0}$ is the service duration
$T \equiv \lvert\mathcal{D}(U)\rvert$	R.v. indicating the total number of dynamic requests

We assume the spatiotemporal probability distribution of dynamic requests can be sampled from (e.g., data is available)

$\omega = \{r_1^\omega, \ldots, r_{T_\omega}^\omega\}$	Sample path of the request arrival process

MDP Model: Decision Epochs and States

- Decision epoch: moment when a decision must be made
- In our setting, we have:
  - Initial decision: setup a route plan to serve all requests in $\mathcal{Z}$
  - Following decisions: each time a dynamic request arrives, decide whether to accept or reject the request and, if accept, how to serve it
- Hence, $1 + T$ decision epochs, where T is unknown

MDP Model: Decision Epochs and States

Definition (Route)

Sequence of pairs $\theta = ((v_0, \delta_0), \ldots, (v_f, \delta_f))$, where $v_0 = v_f = 0$, $(v_i, v_{i+1}) \in \mathcal{A}$, $i \in \{0, \ldots, f-1\}$, $\delta_0 = \delta_f = \emptyset$, and δ_i, $i \in \{1, \ldots, f-1\}$, is the (possibly empty) set of requests to serve when arriving at the i-th node in θ

Definition (Budget of a route θ that started at instant τ)

$$b(\tau, \theta) = U - \tau - \sum_{i=0}^{f-1} t_{v_i, v_{i+1}} - \sum_{(u,i,d) \in \Delta(\theta)} d$$

where $\Delta(\theta) = \cup_{i=1}^{f-1} \delta_i$

MDP Model: Decision Epochs and States

Definition (Vehicle state)

State of vehicle $k \in \{1, \ldots, K\}$ is given by

$$V_k = \begin{cases} \emptyset & \text{if } k \text{ is idle (stationed at the depot)} \\ (\tau_k, \theta_k) & \text{if } k \text{ started to travel along route } \theta_k \text{ at instant } \tau_k \end{cases}$$

Definition (State)

State S_0 (initial state) represents all parameters of the problem

State S_t, $t \in \{1, \ldots, T\}$, is a $(K+1)$-tuple $S_t = (V_1, \ldots, V_K, r_t)$, where V_k are vehicle states, and $r_t = (u_t, i_t, d_t)$ is the t-th element of $\mathcal{D}(U)$

Offline Decision: Initial Route Plan

- The decision at state S_0 is a set of routes $\{\theta_1, \ldots, \theta_{k'}\}$, $k' \leq K$, such that

$$b(0, \theta_1), \ldots, b(0, \theta_{k'}) \geq 0$$

$$\Delta(\theta_1), \ldots, \Delta(\theta_{k'}) \text{ is a partition of } \mathcal{Z}$$

- We let $\mathcal{F}$ represent the set of feasible decisions at state S_0
  - Same feasible space as in a duration-constrained VRP
- Ideally, decision at S_0 should leave the system in a good condition to accept dynamic requests

Online Decisions: Scheduling Policies

At each state S_t, $t \in \{1, \ldots, T\}$, a scheduling policy prescribes:

1. **Acceptance** decision: accept or reject request r_t
2. **Assignment** decision: if r_t is accepted, assign r_t to a vehicle k
3. **Routing** decision: when r_t is assigned to vehicle k, define how route θ_k is adjusted (in case $V_k = (\tau_k, \theta_k)$) or initialized (in case $V_k = \emptyset$) to accommodate r_t

Note: reassignment of requests among vehicles is not allowed

Online Decisions: Routing Policies

Consider a request $r = (u, i, d)$ and a vehicle state $V = (\tau, \theta)$

Definition (Cheapest Insertion (CI) Routing Policy ρ_{CI})

Under the CI routing policy ρ_{CI}, route θ is adjusted into a new route $\rho_{\text{CI}}(\theta, r)$ in such a way that $\Delta_u(\rho_{\text{CI}}(\theta, r)) \setminus \Delta_u(\theta) = \{r\}$, $\Delta_u(\theta)$ is a subsequence of $\Delta_u(\rho_{\text{CI}}(\theta, r))$ and the budget $b(\tau, \rho_{\text{CI}}(\theta, r))$ is maximized

$\rightarrow$ Polynomial time

Definition (Reoptimization Routing Policy ρ_{R})

Under the reoptimization routing policy ρ_{R}, route θ is adjusted into a new route $\rho_{\text{R}}(\theta, r)$ in such a way that $\Delta_u(\rho_{\text{R}}(\theta, r)) \setminus \Delta_u(\theta) = \{r\}$ and the budget $b(\tau, \rho_{\text{R}}(\theta, r))$ is maximized

$\rightarrow$ NP-hard

Online Decisions: State Transitions

- Let $\mathcal{X}_t$ be the set of possible decisions when at state S_t
- A policy π maps each possible state S_t to a decision $X^\pi(S_t) \in \mathcal{X}_t$
- State transition:

$$S_{t+1} = S^M(S_t, x_t, r_{t+1})$$

- Decisions must be taken in real-time (i.e., in seconds, not minutes)

MDP Model: Rewards and Objective

Rewards:

$$R(S_t, x) = \begin{cases} 1 & \text{if } x \in \mathcal{X}_t \text{ is an 'accept' decision} \\ 0 & \text{otherwise} \end{cases}$$

Objective:

$$\max_{\pi \in \Pi, y \in \mathcal{F}} \mathbb{E}\left[\mathbb{E}\left[\sum_{t=1}^{T} R(S_t, X^{\pi}(S_t)) \middle| S_1\right]\right]$$

where Π is the set of feasible policies and $\mathcal{F}$ is the set of feasible initial plans

Methodology

Online decisions:

- **Potential-based policy (PbP)**
- **Simplified PbP (S-PbP)**
- Benchmark policies:
  - Greedy policy with cheapest insertion (CI) and reoptimization (GP_{CI} and GP_{R})
  - PFA policies with CI and reoptimization (PFA_{CI} and PFA_{R})
  - Rollout policies with GP_{CI} and PFA_{CI} as base policies (R_{R}-$GP_{CI}(H)$ and R_{R}-$PFA_{CI}(H)$)

Offline decisions:

- Myopic plan
- **Potential-based plan**

Potential of a State

Given a policy π, the potential of a state S_t is the expected reward-to-go:

$$\Phi_\pi(S_t) = \mathbb{E}\left[\sum_{t'=t}^{T} R(S_{t'}, X^\pi(S_{t'}))\middle| S_t\right] \qquad t \neq 0$$

Optimal Decision Rules

At state S_t, let $x_t^- \in \mathcal{X}_t$ be the decision that rejects request r_t

Decision Rule 1 (Acceptance Rule)

The optimal policy $\pi^* \in \Pi$ accepts request r_t if and only if there exists an 'accept' decision $x_t^+ \in \mathcal{X}_t \setminus \{x_t^-\}$ such that:

$$1 + \mathbb{E}\left[\Phi_\pi(S^M(S_t, x_t^+, r_{t+1}))\right] \geq \mathbb{E}\left[\Phi_\pi(S^M(S_t, x_t^-, r_{t+1}))\right]$$

Optimal Decision Rules

Decision Rule 2 (Assignment Rule)

Let $\{\mathcal{X}_t^k\}$ be a partition of $\mathcal{X}_t \setminus \{x_t^-\}$, in which $\mathcal{X}_t^k$ is the set of decisions where r_t is accepted and assigned to vehicle k. Also, for all $\mathcal{X}_t^k \neq \emptyset$, let

$$x_t^k = \arg\max_{x_t \in \mathcal{X}_t^k} \mathbb{E}\left[\Phi_\pi(S^M(S_t, x_t, r_{t+1}))\right]$$

Then, the optimal policy π^* assigns an accepted request r_t to vehicle

$$k^* = \arg\max_k \mathbb{E}\left[\Phi_\pi(S^M(S_t, x_t^k, r_{t+1}))\right]$$

Idea: approximate potentials and prescribe decisions according to DR1 and DR2

Towards an Approximation Model: Two Remarks

Remark 1 (Late Depot Arrival)

Since the goal is to maximize the number of accepted requests, drivers return to the depot at an instant near the end of the service period

Remark 2 (Request Order Preservation)

The relative order of scheduled requests does not change (CI policy), or changes only slightly (reopt. policy), when a new request is assigned to a route

Modeling Insight: Effective Vehicle Speed

Remarks 1 and 2 suggest the concept of **effective speed**:

Definition (Effective Vehicle Speed)

The effective speed of a vehicle k with state $V_k = (\tau_k, \theta_k)$ is the speed such that k arrives at the depot exactly at instant U:

$$\check{s}_{u_t}(V_k) = \frac{d_{u_t}(V_k)\overline{s}}{d_{u_t}(V_k) + b(\tau_k, \theta_k)\overline{s}}$$

where u_t is the current time instant and $d_{u_t}(V_k)$ is the remaining distance to be traveled along route θ_k at instant u_t

The effective speed allows us to predict the location of a vehicle at any future instant

Modeling Insight: Assignment Costs

$\mathcal{V}_{u_t}(V_k, u')$ (Predicted) set of nodes along θ_k not yet traversed by vehicle $V_k = (\tau_k, \theta_k)$ at a future instant u', obtained by simulating route θ_k under the effective speed

The minimum cost (or budget consumption) when assigning a future request $r' = (u', i', d')$ to vehicle k is therefore:

$$c_{u_t}(V_k, r') = \min_{j \in \mathcal{V}_{u_t}(V_k, u')} t(j, i') + t(i', j) + d'$$

Multiple-Knapsack Approximation of the Potential

Given H sample paths $\Omega = \{\omega_1, \ldots, \omega_H\}$ for the remaining horizon:

$$\mathbb{E}[\Phi_{\pi^*}(S_{t+1})|x_t] \approx \hat{\Phi}_{\pi^*}(S_{t+1}|x_t) = \frac{1}{H} \sum_{\omega \in \Omega} \phi^{\omega}_{\pi^*}(S_{t+1}|x_t)$$

where

$$\begin{aligned}
\phi^{\omega}_{\pi^*}(S_{t+1}|x_t) = \max \quad & \sum_{k \in \overline{K}} \sum_{r \in \omega} z_{kr} \\
\text{s.t.} \quad & \sum_{r \in \omega} c_{u_t}(V_k, r) z_{kr} \leq b(\tau_k, \theta_k) && k \in \overline{K} \\
& \sum_{k \in \overline{K}} z_{kr} \leq 1 && r \in \omega \\
& 0 \leq z_{kr} \leq 1 && k \in \overline{K}, r \in \omega
\end{aligned}$$

Potential-based Policy (PbP, short version)

Given routing policy $\rho \in \{\rho_{\text{CI}}, \rho_{\text{R}}\}$. Upon arrival of $r_t = (u_t, i_t, d_t)$:

1. If there is an idle vehicle k that can serve request r_t, accept r_t and assign it to vehicle k **(same step for all policies)**
2. Otherwise, initialize a set of candidate decisions $\tilde{\mathcal{X}}_t = \{\tilde{x}_t^-\}$ with the 'reject' decision $\tilde{x}_t^-$ related to request r_t
3. If it is feasible to serve r_t with vehicle k under a given routing policy ρ, add the corresponding decision $\tilde{x}_t^k$ to set $\tilde{\mathcal{X}}_t$
4. Compute $\hat{\Phi}_{\pi^*}(S_{t+1}|\tilde{x}_t)$ for each $\tilde{x}_t \in \tilde{\mathcal{X}}_t$, and select the best decision according to Decision Rules 1 and 2

Potential-based Policy (PbP, long version)

Given routing policy $\rho \in \{\rho_{\text{CI}}, \rho_{\text{R}}\}$. Upon arrival of $r_t = (u_t, i_t, d_t)$:

1. If there is an idle vehicle k that can serve request r_t, r_t is accepted and assigned to vehicle k; route θ_k is initialized to the fastest route from the depot to node i_t, and back **(same step for all policies)**
2. Otherwise, a set $\tilde{\mathcal{X}}_t = \{\tilde{x}_t^-\}$ of candidate decisions is initialized, where $\tilde{x}_t^-$ is the 'reject' decision related to request r_t
3. For each non-idle vehicle $V_k = (\tau_k, \theta_k)$, we compute the adjusted route $\rho(\theta_k, r_t)$. If $b(\tau_k, \rho(\theta_k, r_t)) \geq 0$, then the decision $\tilde{x}_t^k$ of accepting and assigning r_t to k under routing policy ρ is feasible and is added to $\tilde{\mathcal{X}}_t$
4. A decision is selected from $\tilde{\mathcal{X}}_t$ by evaluating Decision Rules 1 and 2 based on approximations $\hat{\Phi}_{\pi^*}(S_{t+1}|\tilde{x}_t)$ of $\mathbb{E}[\Phi_{\pi^*}(S_{t+1})|\tilde{x}_t]$, $\tilde{x}_t \in \tilde{\mathcal{X}}_t$, computed by the multiple-knapsack model

Simplified Potential-based Policy (S-PbP)

- Policy PbP requires solving the multiple-knapsack model up to $H(K+1)$ times
  - Depending on our real-time requirements, this may take too long
  - Note: algorithm can be easily parallelized (not implemented)
- S-PbP: trades-off approximation accuracy for computational efficiency

Simplified Potential-based Policy (S-PbP)

Consider the following single-knapsack model:

$$
\begin{aligned}
p^{\omega}_{u_t}(V_k) = \max \quad & \sum_{r \in \omega} z_r \\
\text{s.t.} \quad & \sum_{r \in \omega} c_{u_t}(V_k, r) z_r \leq b(\tau_k, \theta_k) \\
& 0 \leq z_r \leq 1 \qquad\qquad r \in \omega
\end{aligned}
$$

Problem: considers vehicles in isolation; does not take into account competition of vehicles for requests, so the potential is overestimated

Simplified Potential-based Policy (S-PbP)

To compensate for the overestimation bias, we define the ratio α^ω:

$$\alpha^\omega = \frac{\phi^\omega_{\pi^*}(S_{t+1}|\tilde{x}_t^-)}{\sum_{V_k \neq \emptyset} p^\omega_{u_t}(V_k)}$$

Policy S-PbP approximates the potential as follows:

$$\mathbb{E}[\Phi_{\pi^*}(S_{t+1})|\tilde{x}_t^k] \approx \frac{1}{H} \sum_{\omega \in \Omega} \alpha^\omega \left(p^\omega_{u_t}(V_k^+) + \sum_{V_l \neq \emptyset, l \neq k} p^\omega_{u_t}(V_l) \right)$$

where V_k^+ is the state of vehicle k after 'accept' decision $\tilde{x}_t^k$

Complexity: requires solving the multiple-knapsack model H times, and solving the single-knapsack model up to $2HK$ times

Benchmark Policies: Greedy Policy

Given routing policy $\rho \in \{\rho_{\text{CI}}, \rho_{\text{R}}\}$. Upon arrival of $r_t = (u_t, i_t, d_t)$:

1. If there is an idle vehicle k that can serve request r_t, accept r_t and assign it to vehicle k **(same step for all policies)**
2. Otherwise, request r_t is accepted whenever there is a non-idle vehicle k, $V_k = (\tau_k, \theta_k)$, such that $b(\tau_k, \rho(\theta_k, r_t)) \geq 0$
3. If r_t is accepted, assign r_t to the non-idle vehicle k, $V_k = (\tau_k, \theta_k)$, such that $b(\tau_k, \theta_k) - b(\tau_k, \rho(\theta_k, r_t))$ is minimum

Benchmark Policies: Policy Function Approximation

PFA-based policy approximates the potential as follows:

$$\hat{\Phi}_{\text{PFA}}(S_{t+1}|x_t) = \begin{cases} \gamma \sum_{V_k \neq \emptyset} b(\tau_k, \theta_k) & \text{if } x_t = x_t^- \in \mathcal{X}_t \\ \gamma \Big(b(u_t, \rho(\theta_k, r_t)) + \sum_{i \neq k, V_i \neq \emptyset} b(\tau_i, \theta_i) \Big) & \text{if } x_t \in \mathcal{X}_t^k \end{cases}$$

Parameter γ:

- Interpretation: expected reward per unit of budget
- Trained offline by grid search and simulation

Benchmark Policies: Rollout Policies

Upon arrival of request $r_t = (u_t, i_t, d_t)$:

1. If there is an idle vehicle k that can serve request r_t, accept r_t and assign it to vehicle k **(same step for all policies)**
2. Otherwise, initialize a set of candidate decisions $\tilde{\mathcal{X}}_t = \{\tilde{x}_t^-\}$ with the 'reject' decision $\tilde{x}_t^-$ related to request r_t
3. If it is feasible to serve r_t with vehicle k under a given routing policy ρ, add the corresponding decision $\tilde{x}_t^k$ to set $\tilde{\mathcal{X}}_t$
4. Simulate each candidate decision over H sample paths under a given base policy; then, prescribe the decision with the highest simulated reward

Offline Decision: Initial Route Plan

Set-partitioning formulation:

$$\max_{\{\theta_1,\ldots,\theta_{k'}\}\subset\Theta, k'\leq K} \quad g(S_1) \equiv \mathbb{E}\left[\Phi_{\pi^*}(S_1)\right]$$

$$\text{s.t.} \quad \sum_{k\in\{1,\ldots,k'\}} \left[(u,i,d)\in\Delta(\theta_k)\right] = 1 \qquad (u,i,d)\in\mathcal{Z}$$

$$V_k = (0,\theta_k) \qquad k\in\{1,\ldots,k'\}$$

$$V_k = \emptyset \qquad k\in\{k'+1,\ldots,K\}$$

where $S_1 = (V_1,\ldots,V_K,r_1)$, and $\Theta = \{\theta : b(0,\theta)\geq 0\}$ is the set of feasible routes

Offline Decision: Initial Route Plan

Myopic plan: maximize budget

$$\tilde{g}_{\text{myo}}(S_1) = \sum_{V_k \neq \emptyset} b(0, \theta_k) + \sum_{V_k = \emptyset} U$$

- Rational: more budget means more time to serve dynamic requests
- Problem reduces to a duration-constrained VRP
- Column generation-based petal heuristic

Offline Decision: Initial Route Plan

Potential-based plan: maximize potential

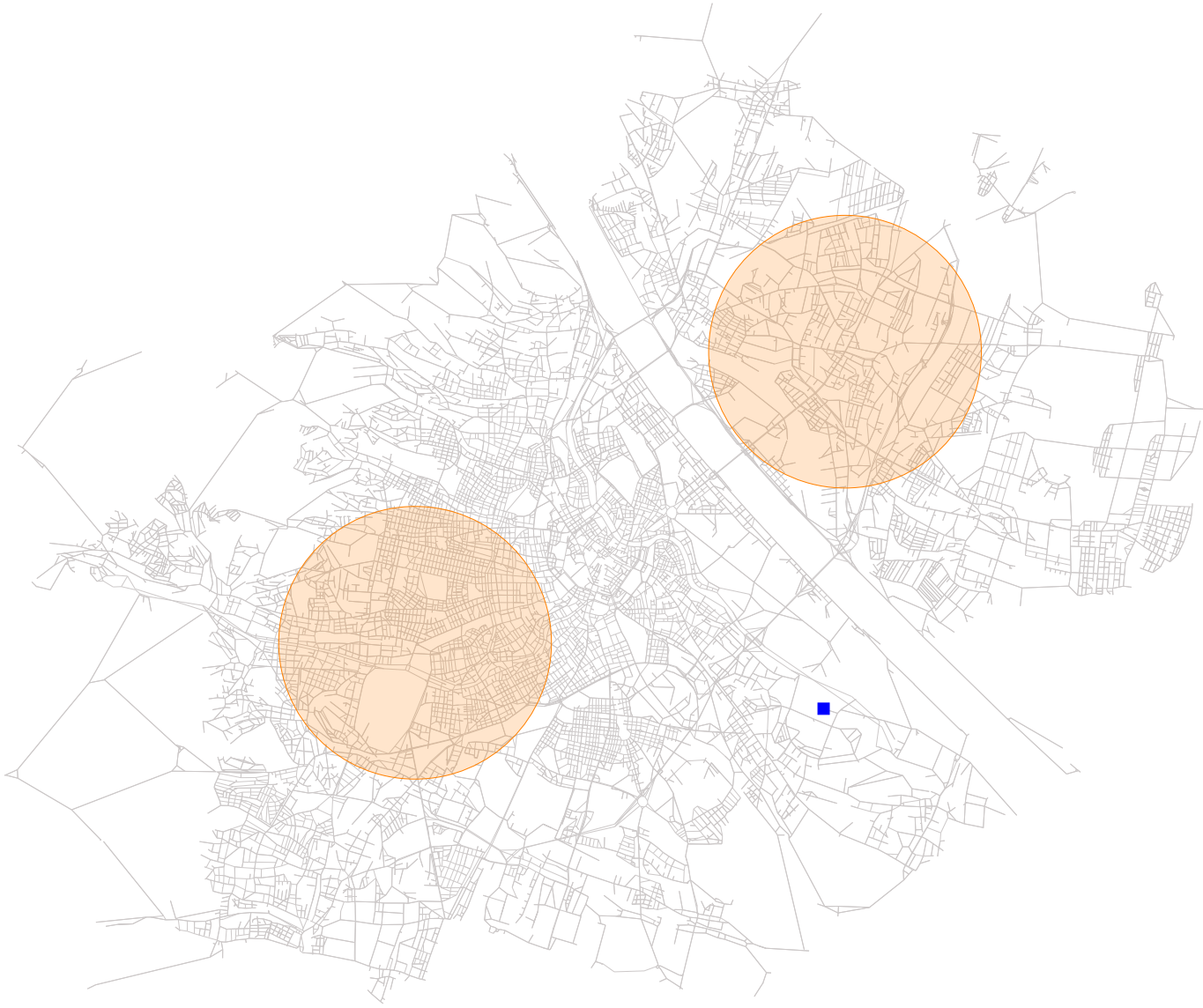
$$\tilde{g}_{\text{pb}}(S_1) = \frac{1}{H} \sum_{\omega \in \Omega} \sum_{V_k \neq \emptyset} p_0^{\omega}(V_k)$$

- Rational: serve all static requests by dispatching drivers along promising routes
- Non-linear cost function due to competition among drivers for requests; approximation via single-knapsack model
- Petal heuristic: reevaluate costs of routes generated by the DCVRP algorithm and resolve set-partitioning model

Computational Study

- Real street network of Vienna (16,080 nodes and 36,424 arcs)
- Service period: 10 hours
- Number of vehicles: $K \in \{2, 3, 5, 6, 10, 12, 20\}$
- Request rate (per minute): $\Lambda \in \{0.2, 0.4, 0.8, 1.5\}$
  - Smallest instances larger than most instances from previous works
- Degree of dynamism: $\eta \in \{75\%, 85\%, 90\%, 95\%\}$
- Three spatiotemporal request distributions:
  - Uniform, time-independent (UTI)
  - Clustered, time-independent (CTI)
  - Clustered, time-dependent (CTD)
- A modular DVRPSR simulator is implemented in C++

Real World Street Network



Instances, Policies and Offline Planners Simulated

Instance parameters					Algorithms		Simulations
Λ	η	K	Distribution(s)	Scenario(s)	Offline	Online	
0.2	0.75	2, 3	UTI, CTI, CTD	5	MY, PB	GP_{CI}, GP_{R}, PFA_{CI}, PFA_{R}, R_{CI}-GP_{CI}(10, 25, 50, 100), R_{R}-GP_{CI}(10, 25, 50, 100), R_{R}-PFA_{CI}(25, 50, 100)*, S-PbP, PbP	12,270
0.4	0.85	3, 5	UTI, CTI, CTD	5	MY, PB	GP_{CI}, GP_{R}, PFA_{CI}, PFA_{R}, R_{CI}-GP_{CI}(10, 25, 50, 100), R_{R}-GP_{CI}(10, 25, 50, 100), R_{R}-PFA_{CI}(25, 50, 100)*, S-PbP, PbP	12,270
0.8	0.90	6, 12	UTI, CTI, CTD	5	PB	GP_{CI}, GP_{R}, PFA_{CI}, PFA_{R}, R_{CI}-GP_{CI}(10), R_{R}-GP_{CI}(10), R_{R}-PFA_{CI}(10), S-PbP, PbP	3,210
1.5	0.95	10, 20	UTI	1	PB	GP_{CI}, GP_{R}, PFA_{CI}, PFA_{R}, R_{CI}-GP_{CI}(10), R_{R}-GP_{CI}(10), R_{R}-PFA_{CI}(10), S-PbP, PbP	214

Notes. MY, myopic planner; PB, potential-based planner; the online algorithms marked by * are tested with the potential-based planner only.

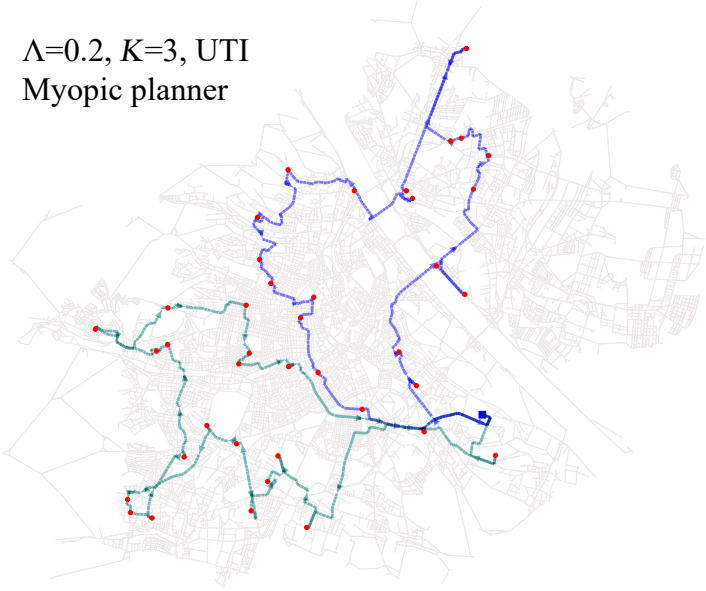
Potential-based vs Myopic Plans

Average number of accepted requests:

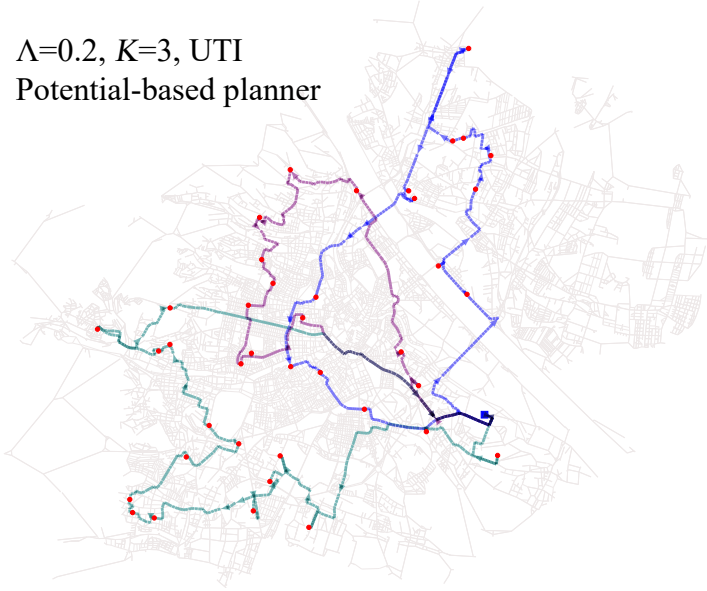
Policy	Myopic Plan	Potential-based Plan	Diff
GP_{CI}	54.1	61.2	13.0%
GP_{R}	54.9	62.7	14.1%
R_{CI}-GP_{CI}(10)	63.0	66.2	5.1%
R_{CI}-GP_{CI}(25)	65.8	69.4	5.4%
R_{CI}-GP_{CI}(50)	67.3	71.1	5.6%
R_{CI}-GP_{CI}(100)	68.2	72.1	5.8%
R_{R}-GP_{CI}(10)	63.4	67.2	6.1%
R_{R}-GP_{CI}(25)	66.2	70.7	6.7%
R_{R}-GP_{CI}(50)	67.7	72.2	6.8%
R_{R}-GP_{CI}(100)	68.5	73.3	7.0%
S-PbP(50)	72.0	75.2	4.4%
PbP(50)	**72.9**	**77.2**	**5.9%**
PFA_{CI}	61.3	71.8	17.1%
PFA_{R}	62.1	72.6	16.9%
Avg	66.4	72.4	9.0%

Potential-based vs Myopic Plans

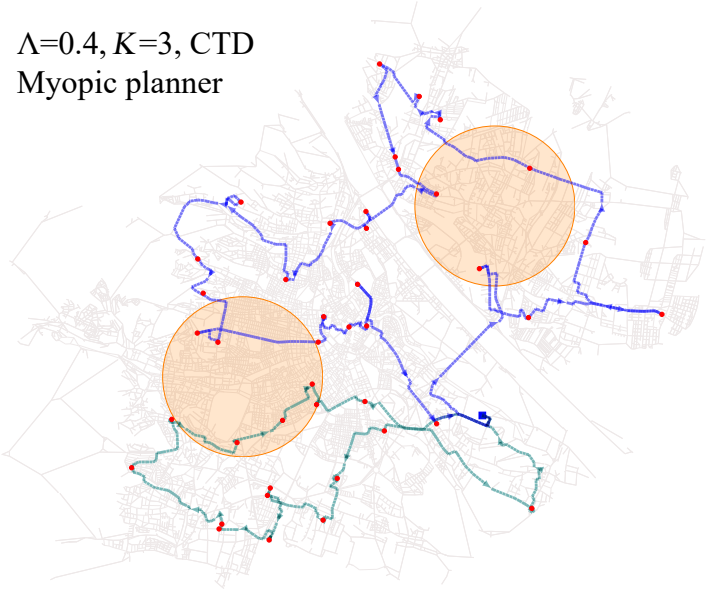
Λ=0.2, K=3, UTI
Myopic planner



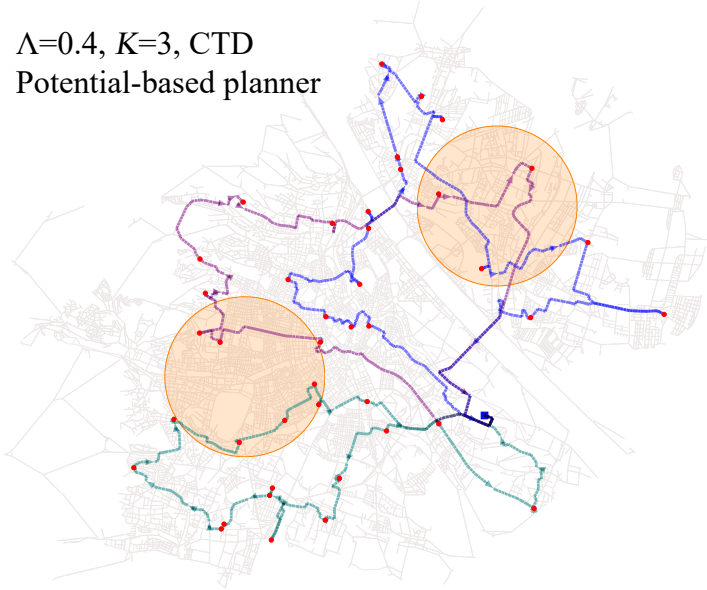
Λ=0.2, K=3, UTI
Potential-based planner



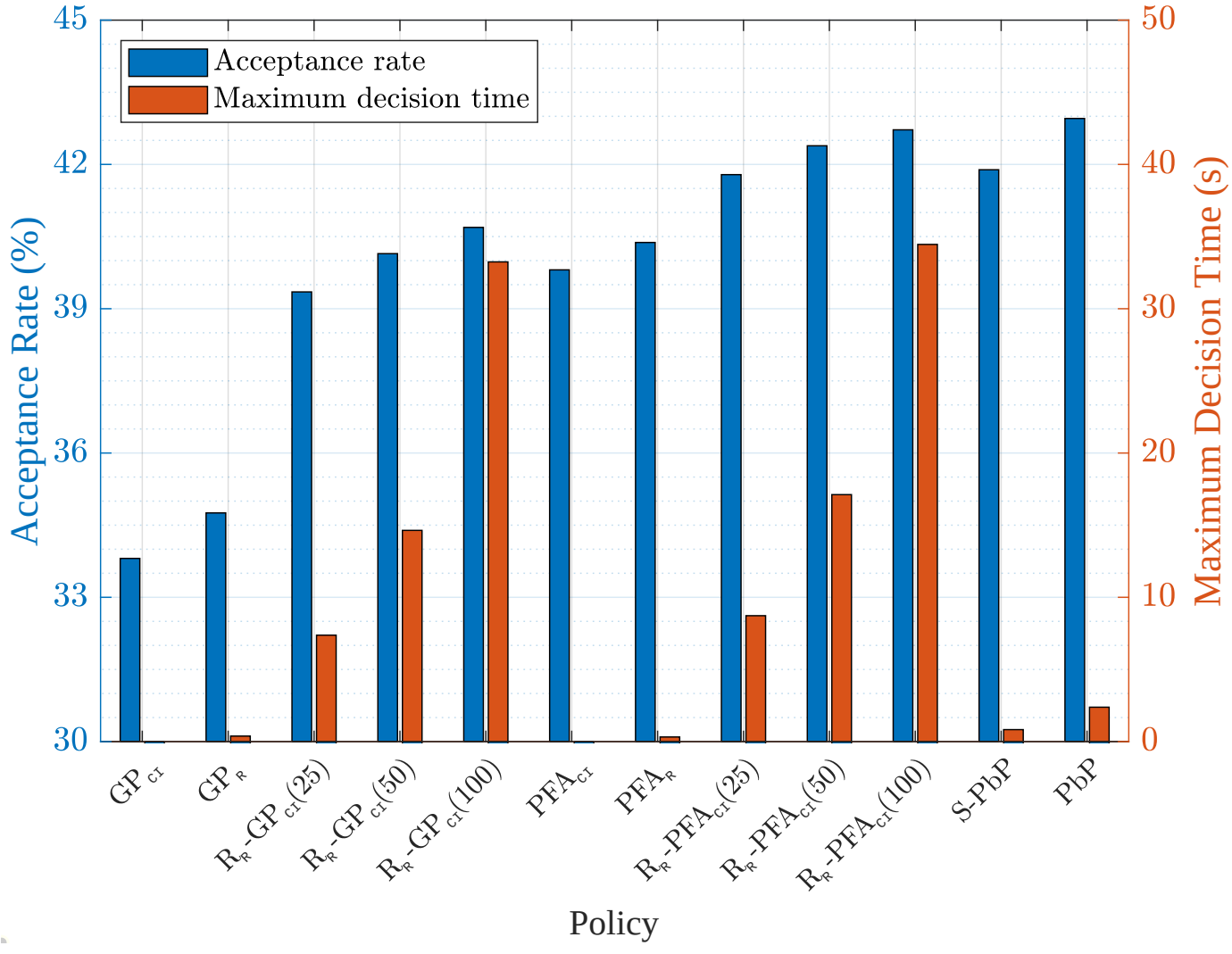
Λ=0.4, K=3, CTD
Myopic planner



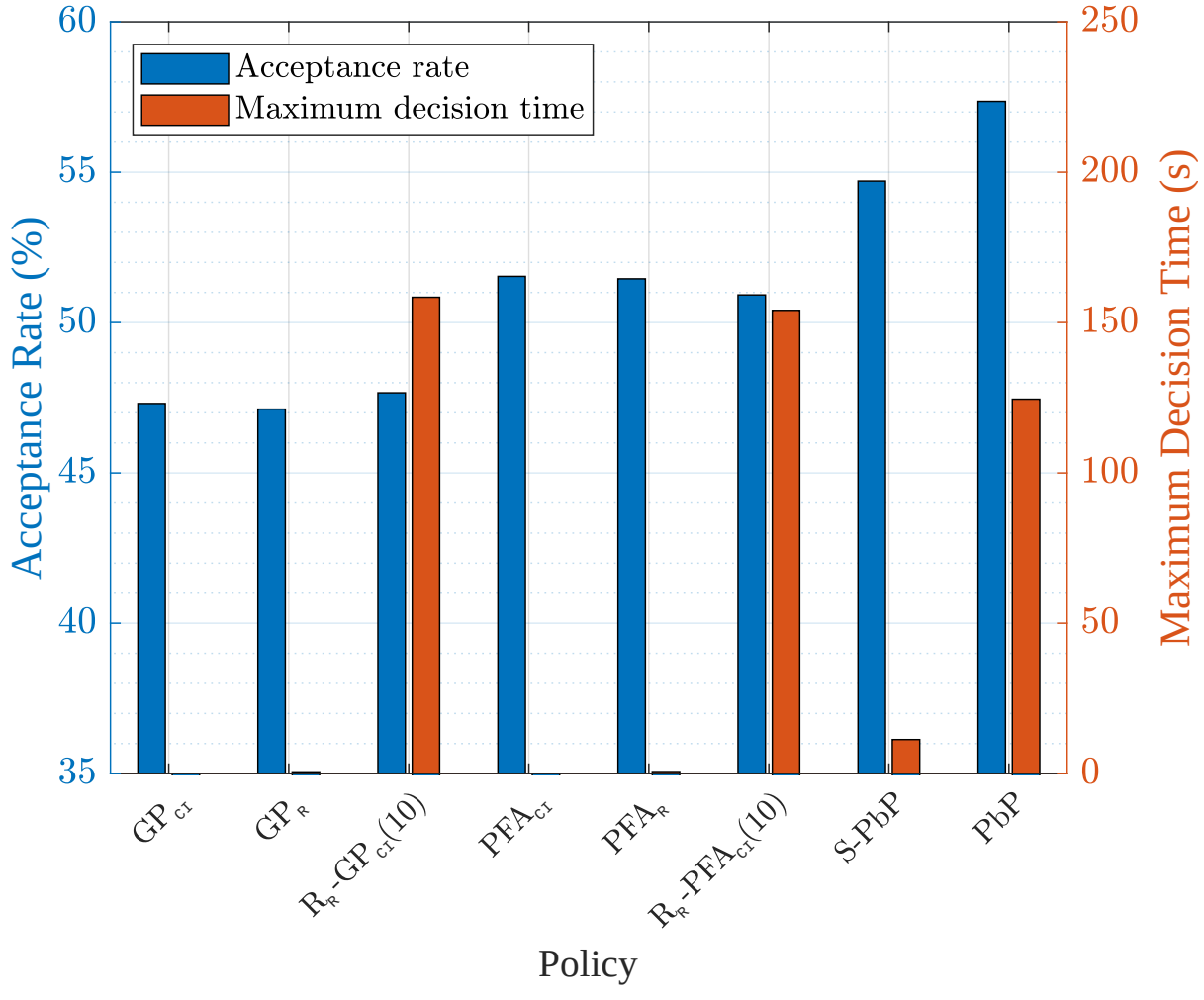
Λ=0.4, K=3, CTD
Potential-based planner



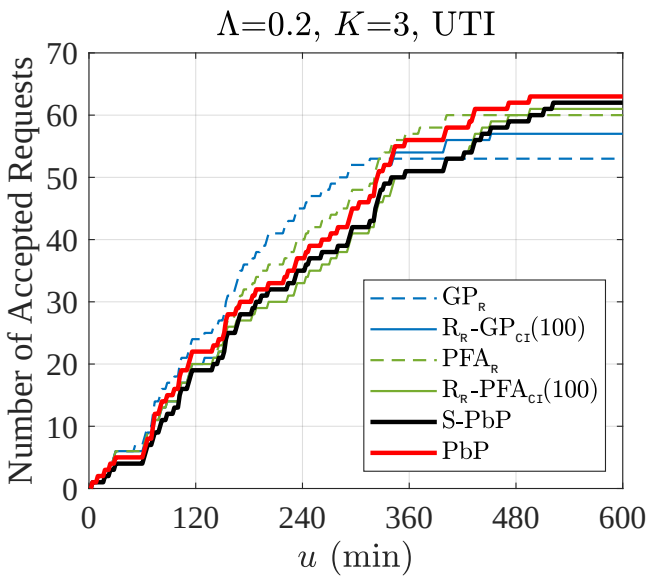
Policy Comparison ($\Lambda \in \{0.2, 0.4\}$)

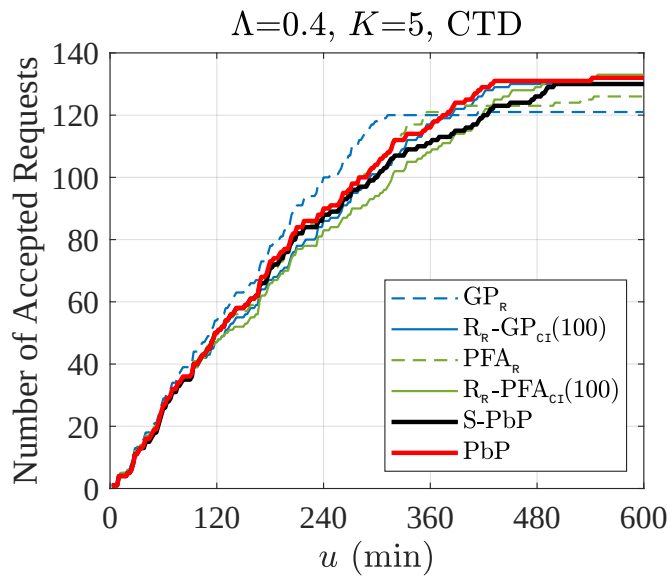


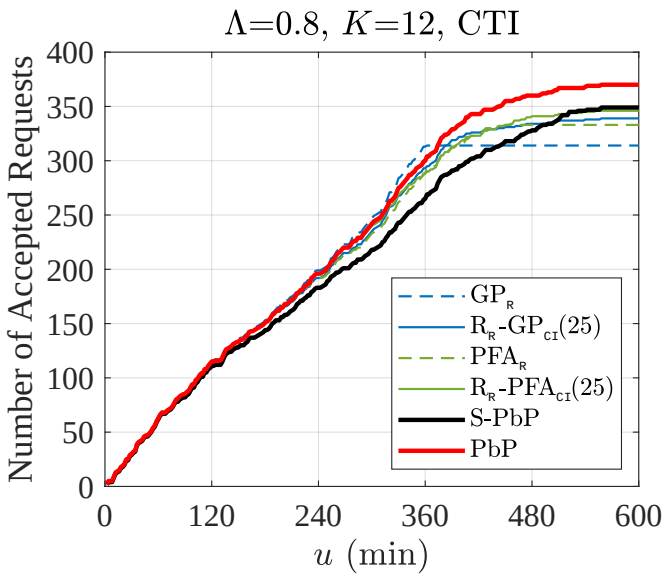
Policy Comparison ($\Lambda \in \{0.8, 1.5\}$)

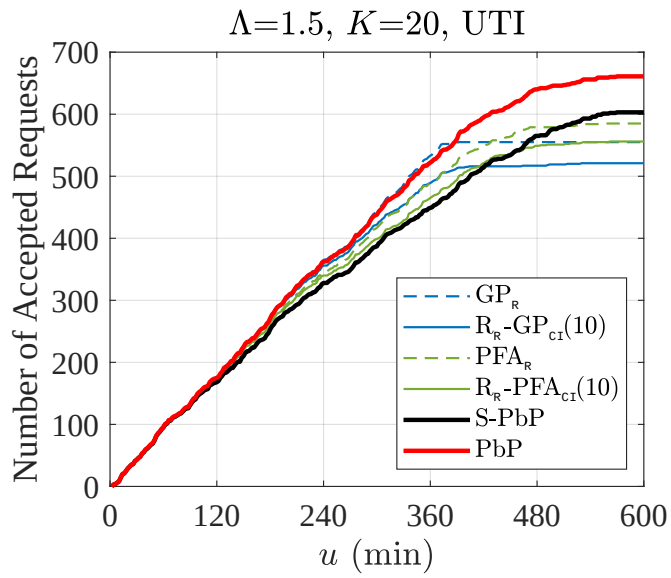


Acceptance Profiles

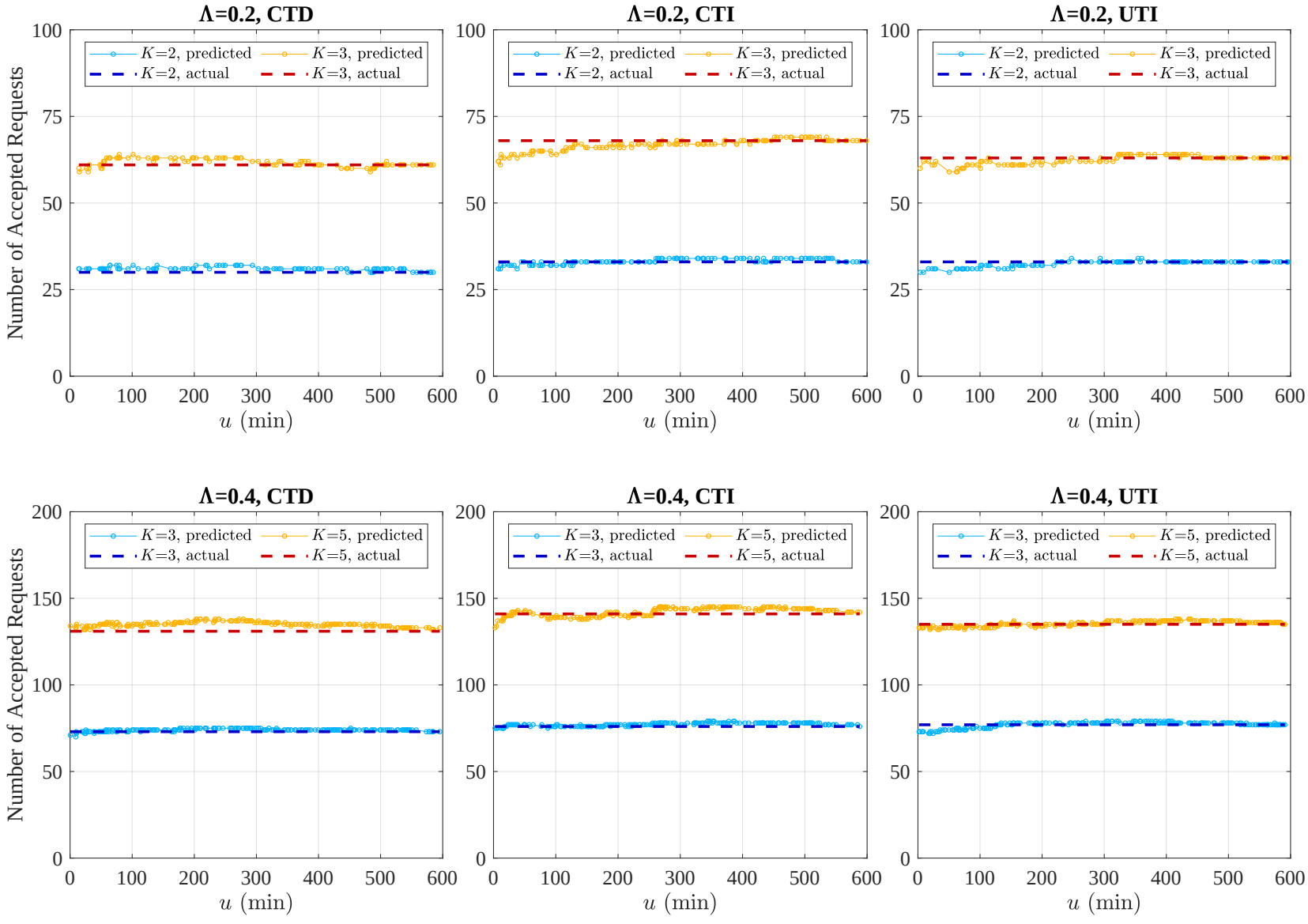




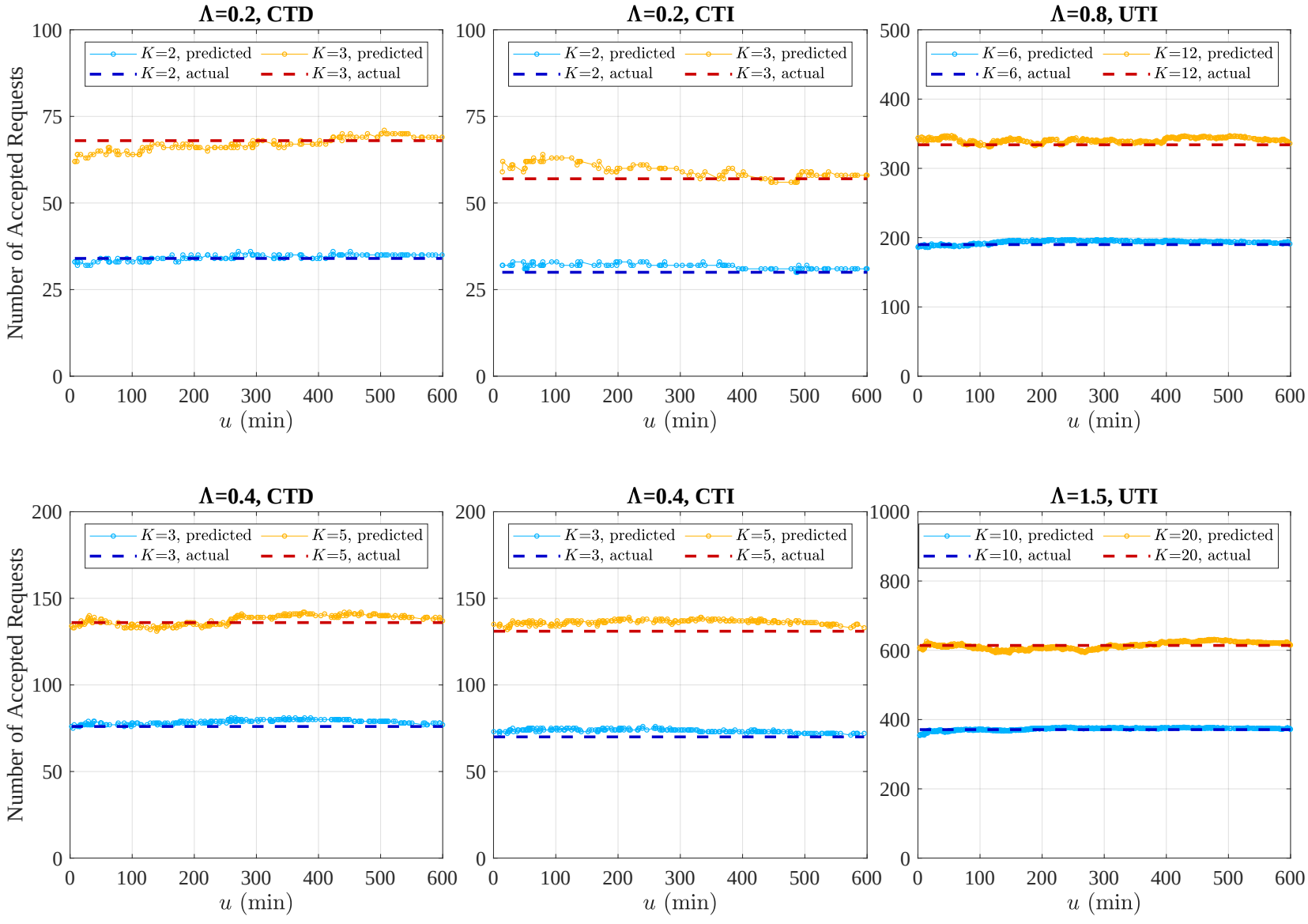




Multiple-Knapsack Potential Approximations



Single-Knapsack Potential Approximations



Conclusions

- Main contributions and takeaways:
  - Expected reward-to-go can be accurately and efficiently approximated by knapsack models
  - Accurate potential approximations enable high-performing scheduling policies, which outperform classical ADP methods traditionally used for DVRPs such as rollout algorithms and PFA
  - Potential is key for both offline and online decisions
  - Coverage of the service area is more important than budget alone

- Possible extensions and future research:
  - Vehicle capacity constraints
  - Time windows
  - Self-imposed time windows (i.e., time window assignment)
  - Pickup and delivery
  - Reassignment of requests among vehicles (more complex)
  - Comparison against 'batch and optimize' approach

Acknowledgements

This project has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No 754462

Computational experiments were performed on the Dutch national e-infrastructure with the support of SURF Cooperative

Geographical data for Vienna are copyrighted to OpenStreetMap contributors and available at http://openstreetmap.org

Questions & Discussions