The Dynamic Park-and-loop Routing Problem

Jean-François Cordeau, Nicolás Cabrera, Jorge E. Mendoza

HEC Montréal

Extended abstract submitted for presentation at the 12th Triennial Symposium on Transportation Analysis (TRISTAN XII)
June 22-27, 2025, Okinawa, Japan

February 28, 2025

Keywords: dynamic vehicle routing, park-and-loop, multi-space sampling heuristic

1 INTRODUCTION

In many countries, public utilities are in charge of providing essential services, including water, gas, electricity, and internet, to millions of customers without interruption. To achieve this, they conduct daily tasks at customer locations, such as preventive maintenance, repairs, and meter readings. Many of these tasks arise from unforeseen events, such as power outages or network disruptions, leading to on-demand requests that join pre-scheduled tasks along the day. Given the high volume of customer requests, utilities often face resource constraints, making it challenging to serve all requests within the same day. As a result, outsourcing to third-party contractors is common. Additionally, the geographic distribution of customers poses logistical challenges, particularly in areas with restricted access or limited parking. Vehicles operating in these zones encounter delays due to traffic, accidents, or adverse weather. In contrast, pedestrian mobility is generally less disrupted. Therefore, utilities often design dynamic vehicle routes using a park-and-loop structure (Parragh and Cordeau, 2017, Coindreau et al., 2019). These routes involve a main tour that is completed using a vehicle in addition to subtours that are carried out on foot after parking the vehicle. This leads to the dynamic park-and-loop routing problem (DPLRP).

The problem can be defined on a directed graph $\mathcal{G} = (\mathcal{N}, \mathcal{A})$, where $\mathcal{N}$ represents the set of nodes and $\mathcal{A}$ denotes the set of arcs. The node set $\mathcal{N}$ consists of the depot, denoted by $\overline{0}$, and the set of customer requests $\mathcal{C}$, which can be divided into two disjoint subsets: scheduled requests $\mathcal{C}^s$ and on-demand requests $\mathcal{C}^d$. The scheduled requests $\mathcal{C}^s$ are known at the start of the planning horizon and must be completed by the end of the workday, whereas the on-demand requests $\mathcal{C}^d$ appear randomly during the working hours $[e, f]$. For each on-demand request, a decision is made to either accept or reject it. If the request is accepted, it must be served within the workday. A set of homogeneous workers $\mathcal{W}$ is available to serve scheduled and on-demand requests, and arcs in $\mathcal{A}$ represent the connections between locations, each characterized by a walking distance λ_{ij}, a driving distance μ_{ij}, a walking time γ_{ij}, and a driving time δ_{ij}. Every time a worker arrives at a location using a vehicle, there is an associated parking time τ. The objective of the DPLRP is to design a route plan that maximizes the number of on-demand requests served, subject to the following constraints: (i) the total duration of all routes performed by any worker must not exceed the workday duration; (ii) the walking distance for each worker must remain within the walking distance limit Λ; and (iii) the total duration of any walking subtour must not exceed the time limit Γ.

2 SOLUTION METHOD

Solving this problem poses two big challenges. The first involves constructing an initial routing plan that accommodates all scheduled requests while maintaining sufficient flexibility to incorporate most of the on-demand requests. This initial plan must strike a delicate balance between efficiency and adaptability, ensuring that it can handle on-demand requests without substantial disruption. The second challenge lies in the online adjustment of the initial routing plan whenever a new customer request is accepted. Updating the routing plan dynamically requires an algorithm that not only re-evaluates the current routes but also integrates new requests seamlessly. In the DPLRP, updating the routing plan may involve a wide range of actions, such as extending or shortening an existing walking subtour, inserting or removing a walking subtour, reassigning requests between workers, and reordering the requests in a route, among others.

To tackle these challenges, we begin by formulating the problem as a sequential decision process. We then introduce a set of scheduling policies that include both myopic and anticipatory approaches. In the DPLRP context, a scheduling policy consists of three key components: an offline policy that generates the initial routing plan θ_0, an acceptance policy that determines whether to accept or reject incoming customer requests, and a routing policy that updates the routing plan following acceptance decisions. Our scheduling policies leverage a state-of-the-art metaheuristic, the multi-space sampling heuristic (MSH), as proposed by Mendoza and Villegas (2013). Each time a new request is revealed, this heuristic generates a set of high-quality routes by taking into account the current locations of all workers, using a variety of sampling functions. For instance, a sampling function might involve a local search procedure to refine an existing route. A new solution is then constructed by selecting from these routes through a set partitioning model, a process carried out by an assembler.

Figure 1 illustrates the two phases of MSH applied to an instance of the DPLRP with ten scheduled requests and two on-demand requests. On the left side of the figure we depict the current state of the system: one worker, colored in green, is idle at the depot, while the other two workers, colored in blue and orange, are walking toward locations 2 and 10, respectively. The center of the figure shows a selection of routes sampled during the first phase of the MSH, with each route beginning from the workers' current destinations or, in the case of the idle worker, their current location. Finally, on the right side of the figure we display the updated routing plan. In this new routing plan, the worker in green is assigned to depart immediately toward customer 7.

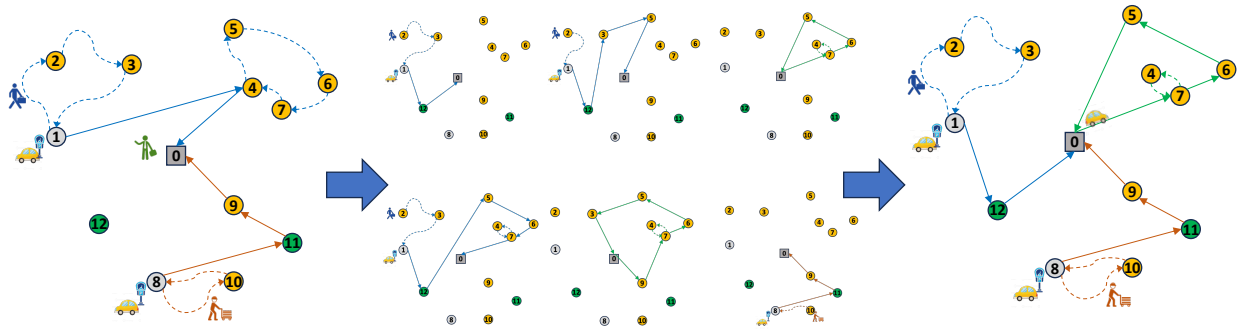

Figure 1 – *Multi-space sampling heuristic example.*

We consider two sets of sampling functions: best-insertion and split-based. The best-insertion sampling functions generate new routes by inserting the new request i at the position within route r that minimizes the insertion cost. These functions account for the possibility of adding the new request i to an existing subtour, or alternatively, creating a new subtour originating from one of the customer locations already included in route r. While these sampling functions

are highly efficient, they lack flexibility as they do not allow for reordering already accepted requests or redistributing requests among workers. In contrast, split-based sampling functions employ a two-step approach to create new routes. First, a traveling salesman problem (TSP) tour is constructed, which includes both the accepted but incomplete requests as well as the new on-demand request, using a randomized TSP heuristic. In the second step, the TSP tour is partitioned into feasible routes via a tailored split algorithm proposed by Cabrera et al. (2022). These sampling functions offer greater flexibility, enabling the redistribution of accepted requests among workers, as well as more complex operations such as extending or shortening existing walking subtours and inserting new subtours when beneficial.

The assembler can be generally defined as follows. Let Ω_w^k define a subset of feasible park-and-loop routes for each worker given its current status at decision epoch k. Let h_r be the cost of route r. Also, let a_{ir} be a binary parameter that takes the value 1 if route r serves request i, and 0 otherwise. Let y_r be a binary variable that takes the value 1 if route r is selected, and 0 otherwise. Then, the assembler at any decision epoch $k > 0$ is defined as:

$$\min \sum_{w \in \mathcal{W}} h_r y_r \tag{1}$$

subject to

$$\sum_{w \in \mathcal{W}} \sum_{r \in \Omega_w^k} a_{ir} y_r = 1 \qquad \forall i \in \mathcal{C}_k \tag{2}$$

$$\sum_{w \in \mathcal{W}} \sum_{r \in \Omega_w^k} a_{ir} y_r \leq 1 \qquad \forall i = i_k \tag{3}$$

$$\sum_{r \in \Omega_w^k} y_r = 1 \qquad \forall w \in \mathcal{W} \tag{4}$$

$$y_r \in \{0, 1\} \qquad \forall w \in \mathcal{W}, r \in \Omega_w^k. \tag{5}$$

The objective function (1) aims to minimize the total cost of the routing plan. Constraints (2) ensure that all scheduled and previously accepted requests are included in the routing plan, while constraints (3) specify that the new on-demand request can be assigned to only one route, if accepted. Constraints (4) ensure that each worker is assigned exactly one compatible route. We consider two types of assemblers: myopic and potential-based. The myopic assembler minimizes the total route duration, but this short-term focus can lead to suboptimal long-term decisions, where routes that initially seem efficient become inefficient over time. In contrast, the potential-based assembler accounts for the expected number of requests that could potentially be inserted into a given route. To compute the potential of a route, we generate a set of scenarios Σ, where each scenario σ represents a sample path of requests arrivals from the current epoch to the end of the planning horizon. We then solve a multi-knapsack approximation model as proposed by Zhang et al. (2023). This assembler leads to a more balanced distribution of requests among workers and produces routing plans that better accommodate future on-demand requests.

3 RESULTS

We evaluate several scheduling policies. Each scheduling policy is a combination of three components: (i) an offline policy, namely myopic (ϑ_M) or potential-based (ϑ_P); (ii) a routing policy that uses the best-insertion sampling functions and can be myopic (ρ_{MI}) or potential-based (ρ_{PI}), or that uses the split-based sampling functions and can be myopic (ρ_{PS}) or potential-based (ρ_{PS}); and (iii) an acceptance policy that can be either greedy (ϱ_G), multi-knapsack based (ϱ_{MK-50}), or rollout-based (ϱ_{R-50}).

To evaluate the efficiency and effectiveness of our scheduling policies, we conducted computational experiments on a set of 180 instances, with each instance containing between 50 and 100 customer requests. Figure 2 compares the performance of the proposed scheduling policies based on the average acceptance rate across these instances. The results indicate that the scheduling policy $\vartheta_P - \rho_{PS} - \varrho_G$ achieves the highest acceptance rate, at 79.8%. Our findings suggest that accounting for the uncertainty of new request arrivals, both when designing the initial routing plan and when updating it, has a positive effect on the acceptance rate. In contrast, the scheduling policy $\vartheta_M - \rho_{MI} - \varrho_{MK-50}$, which neither anticipates new request arrivals in the initial routing plan nor permits request reassignment among workers during updates, exhibits the lowest acceptance rate. This lack of flexibility makes it more challenging to adapt to new requests.

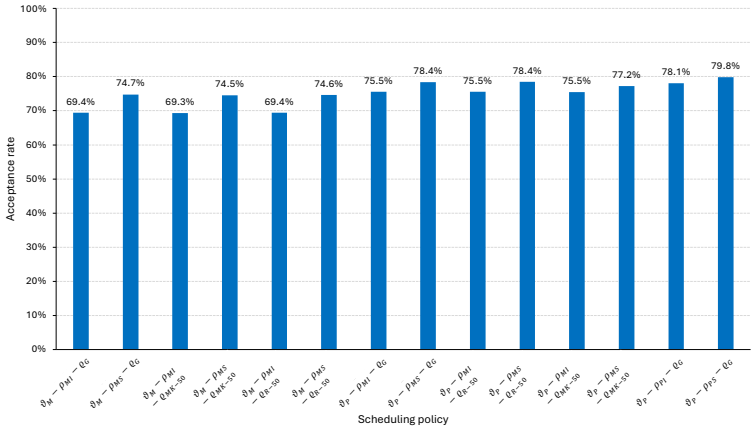

Figure 2 – *Comparison of scheduling policies acceptance rate.*

The contributions of this research are fourfold. First, we introduce and solve the dynamic park-and-loop routing problem. This problem extends the park-and-loop routing problem by considering both scheduled and on-demand requests. Second, we develop a set of scheduling policies that leverage the state-of-the-art metaheuristic known as the multi-space sampling heuristic. In contrast to other scheduling policies in the literature, our best policy allows for inter-route operations. Third, we describe a method to compute a bound on the number of requests that can be served under the assumption of complete information. Fourth, we propose a new set of instances using the street network of Vienna, Austria. We make the instances, our results, and a solution checker publicly available.

References

N. Cabrera, J.-F. Cordeau, and J. E. Mendoza. The doubly open park-and-loop routing problem. *Computers & Operations Research*, 143:105761, 2022.

M.-A. Coindreau, O. Gallay, and N. Zufferey. Vehicle routing with transportable resources: Using carpooling and walking for on-site services. *European Journal of Operational Research*, 279:996–1010, 2019.

J. E. Mendoza and J. G. Villegas. A multi-space sampling heuristic for the vehicle routing problem with stochastic demands. *Optimization Letters*, 7:1503–1516, 2013.

S. N. Parragh and J.-F. Cordeau. Branch-and-price and adaptive large neighborhood search for the truck and trailer routing problem with time windows. *Computers and Operations Research*, 83:28–44, 2017.

J. Zhang, K. Luo, A. M. Florio, and T. Van Woensel. Solving large-scale dynamic vehicle routing problems with stochastic requests. *European Journal of Operational Research*, 306(2):596–614, 2023.