revista principia

ISSN 2447-9187

https://doi.org/10.18265/2447-9187a2024id8297

ORIGINAL ARTICLE

SUBMITTED January 26, 2024

APPROVED June 10, 2024

PUBLISHED ONLINE June 25, 2023

FINAL FORMATTED VERSION May 28, 2025

ASSOCIATE EDITOR
Prof. Dr. Francisco Petrônio Alencar de Medeiros

A neighborhood search-based heuristic for the dynamic vehicle routing problem

Wilton Gustavo Gomes da Costa [1]

Ricardo Santos [2]

Willy Alves de Oliveira Soler [3]

Bianca de Almeida Dantas [4] *

[1] wilton.costa@platformscience.com

[2] ricardo.santos@ufms.br

[3] willy.oliveira@ufms.br

[4] bianca.dantas@ufms.br

Federal University of Mato Grosso do Sul (UFMS), Campo Grande, Mato Grosso do Sul, Brazil

* *Corresponding author.*

ABSTRACT: The Vehicle Routing Problem (VRP) is a classical optimization problem focused on determining routes for a fleet of vehicles to fulfill the demands of a set of customers. Due to its wide applicability, particularly in the logistics sector, numerous variants of the VRP have been developed. One such variant is the Dynamic Vehicle Routing Problem (DVRP), characterized by the emergence of new customer demands after the initial routing has been established. The combinatorial complexity of the DVRP poses significant challenges in finding high-quality, feasible solutions within practical runtime limits for real-world, large-scale problems, thereby driving the development of heuristic approaches. This paper introduces a novel heuristic algorithm, Dynamic Search per Neighbors Routes (DSNR), which employs neighborhood search techniques based on the 2-Opt* operator to incorporate new delivery requests into existing routes. The DSNR algorithm was tested using instances from the Loggibud benchmark, representing real-world data from three distinct Brazilian cities. In terms of distance and the number of vehicles required, the proposed approach demonstrated superior performance compared to two existing methods from the literature: the QRP Sweep (QRPS) and K-Means Greedy (KG). The DSNR achieved improvements of 15% to 17% in terms of distance and final delivery costs.

Keywords: 2-Opt*; DSNR; dynamic vehicle routing; K-means Greedy; local search heuristic; QRP Sweep.

Uma heurística baseada em busca de vizinhança para o problema de roteamento dinâmico de veículos

RESUMO: *O Problema de Roteamento de Veículos (PRV) é um problema clássico de otimização focado em determinar rotas para uma frota de veículos visando atender às demandas de um conjunto de clientes. Dada sua ampla aplicabilidade, particularmente no setor de logística, inúmeras variantes do PRV foram desenvolvidas. Uma dessas variantes é o Problema de Roteamento Dinâmico de Veículos (PRDV), caracterizado pelo surgimento de novas demandas de clientes após o estabelecimento do roteamento*

inicial. A complexidade combinatória do PRDV apresenta desafios significativos para encontrar soluções viáveis e de alta qualidade dentro de limites práticos de tempo de execução para problemas reais de grande escala, impulsionando assim o desenvolvimento de abordagens heurísticas. Este artigo apresenta um novo algoritmo heurístico, Busca Dinâmica por Rotas de Vizinhos (Dynamic Search per Neighbors Routes – DSNR), que emprega técnicas de busca de vizinhança baseadas no operador 2-Opt para incorporar novas solicitações de entrega em rotas existentes. O algoritmo DSNR foi avaliado usando instâncias do benchmark Loggibud, representando dados do mundo real de três cidades brasileiras distintas. Em termos de distância e número de veículos necessários, a abordagem proposta demonstrou desempenho superior em comparação a dois métodos existentes na literatura: o QRP Sweep (QRPS) e o K-Means Greedy (KG). O DSNR obteve melhorias de 15% a 17% em termos de distância e custos de entrega final.*

Palavras-chave: *2-Opt*; DSNR; heurísticas de busca local; K-Means Greedy; QRP Sweep; roteamento dinâmico de veículos.*

1 Introduction

The Vehicle Routing Problem (VRP) is a classical optimization problem first introduced by Dantzig and Ramser (1959). As described by Pillac *et al*. (2013), the VRP is typically formulated using a graph G = (*V*, *E*, *D*), where V = [0,1,..., *N*} represents the set of vertices. Vertex 0 denotes a depot serving as the starting point for a vehicle fleet, while nodes 1, ..., *N* represents the customers to be visited. The set of arcs is defined as $E=\{(i,j)\in V^2 \mid i \neq j\}$, and $D = (d_{if})_{(i,j)}\in E$ is the cost matrix over the arc set, typically representing costs, times, or distances between vertices. The classical VRP seeks to determine routes for identical vehicles, ensuring each customer is visited exactly once while minimizing the associated routing costs, times, or distances.

The VRP is widely applicable to various logistical challenges in real-world contexts. For instance, Ralphs *et al*. (2003) addressed a capacitated VRP, Bräysy and Gendreau (2005) incorporated delivery time windows for visiting customers, and Koç *et al*. (2016) studied a heterogeneous vehicle fleet. This paper focuses on the Dynamic Vehicle Routing Problem (DVRP), a VRP variant where some packages (customers) to be delivered are not known in advance but are dynamically revealed during the execution of routes. Specifically, this scenario involves receiving new customer orders after the initial routes have been determined, necessitating route adjustments to accommodate these new demands.

Psaraftis (1988) first characterized the DVRP as a real-time service in which vehicles must satisfy evolving demands. According to Larsen (2000), the DVRP is more challenging to solve than the traditional VRP, which is already NP-hard, as demonstrated by Lenstra and Kan (1981). This inherent complexity requires the development of specialized solution methods to obtain high-quality feasible solutions within a reasonable runtime.

In this context, the present study introduces a new neighborhood search heuristic, Dynamic Search per Neighbors Routes (DSNR), designed to address the DVRP. The DSNR examines neighboring routes to schedule dynamic vehicle packages while adhering to time, distance, and cost constraints. The proposed approach is evaluated

using the Loggibud (2021) dataset, which includes thousands of georeferenced delivery points across the capitals of three Brazilian states (Pará – PA, Distrito Federal – DF, and Rio de Janeiro – RJ). The algorithm is compared against DVRP solutions proposed by Bertsimas and Van Ryzin (1993), Gillett and Miller (1974) (QRP Sweep – QRPS), and Comert *et al*. (2018) (K-Means Greedy – KG). DSNR outperforms QRPS and KG in distance and the number of vehicles used, achieving reductions of up to 17% in distance compared to the baseline approaches. The main contributions of this work are as follows:

- A novel heuristic approach for solving the DVRP using local search strategies to schedule new dynamic packages;
- A comprehensive evaluation and statistical analysis of the proposed method using a real-world, large-scale dataset of deliveries.

The remainder of this paper is organized as follows: Section 2 introduces the DVRP and its theoretical foundations; Section 3 reviews related work; Section 4 describes the DSNR algorithm in detail. Section 5 discusses the results and findings. Finally, Section 6 concludes the study and suggests directions for future research.

2 Dynamic Vehicle Routing Problem

The DVRP, also known as real-time or online VRP, involves scenarios in which input data is only partially available at the onset of route planning. This implies that the total number of packages to be delivered is unknown beforehand and progressively revealed over time.

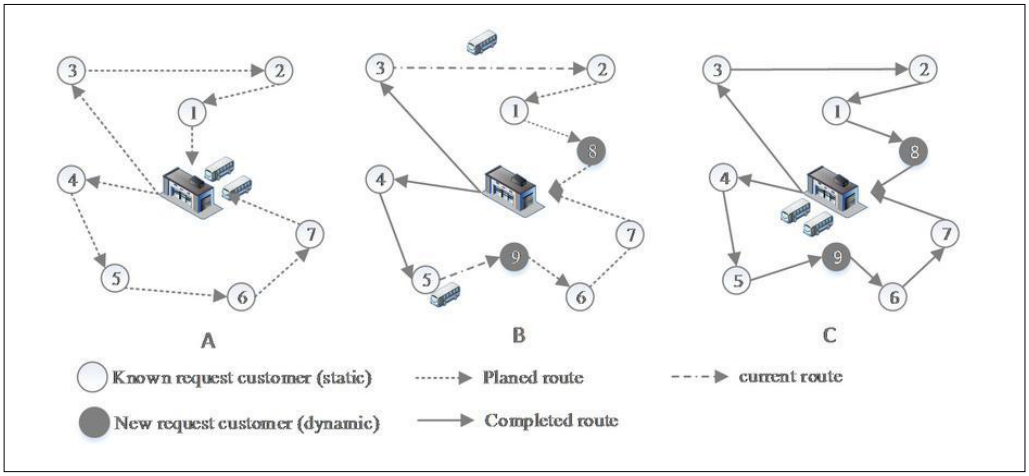
In the literature, the DVRP is classified based on the type of dynamic information considered, such as new customer requests, demands, service times, and travel times. The DVRP was first investigated by Psaraftis (1988), and since then, various methodologies have been proposed to evaluate the dynamic degree of the problem, as outlined by Larsen, Madsen, and Solomon (2002).

A solution for the DVRP comprises policies for scheduling new packages arriving at the depot to the existing routes. Figure 1 provides an illustrative example of the DVRP:

- In the initial phase (A), there are seven known customers (vertices 1-7);
- The vehicles have already commenced their deliveries in the second phase (B). One vehicle is at customer 3, while the other is at customer 5. During this phase, two new customer demands (8 and 9) are received, required route inclusion. The initial routes are recalculated to incorporate these new demands, considering the current vehicles positions;
- In the final phase (C), all customers are served, and the vehicles return to the depot.

ISSN 2447-9187

Figure 1 ▶
An example of the dynamic vehicle routing problem.
Source: Chen, Chen, and Gao (2017)



The DVRP can be characterized by its dynamic degree (Equation 1), denotes as Ω, a computed using Equation 1. For example, in Figure 1, there are two dynamic packages (two new customers) out of a total of nine packages, resulting in:

$$\Omega = \frac{Number\ of\ dynamic\ packages}{Total\ of\ packages} = \frac{2}{9} = 22.22\% \quad (1)$$

Based on Ω, the system can be classified as either dynamic (Ω > 0) or static (Ω = 0). According to Larsen, Madsen, and Solomon (2002), dynamic systems are further categorized into three levels:

- Weakly dynamic: $0\% < \Omega \leq 20\%$;
- Moderately dynamic: $20\% \leq \Omega < 80\%$;
- Strongly dynamic: $80\% \leq \Omega < 100\%$.

This study assumes that static packages may also be included in the routing process. Specifically, initial routes are created based on a set of known static customer demands. As new customer demands arrive at the depot, adjustments to the initial routes are required. The proposed heuristic addresses these scenarios by handling batches of up to 75 packages at the warehouse, necessitating route redefinition for the vehicles.

To conclude this section, the mathematical model used to define the DVRP is presented. Let N represents the total number of known customer demands, and K denotes the number of identical vehicles available to serve these customers. Furthermore, consider the graph $G = (V, E, D)$ as introduced in Section 1, in which $D = (d_{ij})$ represents the distance matrix. The following decision variables are defined:

- x_{ijk}: = 1 if the vehicle k travels from the vertex i to vertex j, and 0 otherwise;
- z_{jk}: = 1 if the vehicle k serves customer j, and 0 otherwise;
- y_{jk}: = 1 if the customer j is the last visited by vehicle k, and 0 otherwise;
- v_k: = 1 if the vehicle k is used to serve at least one customer, and 0 otherwise;
- u_{jk}: = represents the order in which the customer j is visited by vehicle k.

The mixed integer programming model minimizes the objective function (Equation 2), subject to constraints defined in Equations 3 to 14.

$$\sum_{i}^{N} \sum_{j}^{N} \sum_{k}^{K} d_{ij} x_{ijk} \tag{2}$$

subject to:

$$\sum_{k}^{K} z_{jk} = 1, \ \forall j \tag{3}$$

$$\sum_{j}^{N} z_{jk} \leq N v_k, \ \forall k \tag{4}$$

$$\sum_{i}^{N} x_{ijk} = z_{jk}, \ \forall k \tag{5}$$

$$\sum_{i}^{N} x_{ijk} = \sum_{i}^{N} x_{jik} + y_{jk}, \ \forall j,k \tag{6}$$

$$\sum_{j}^{N} x_{0jk} = v_k, \ \forall k \tag{7}$$

$$u_{jk} \geq u_{ik} + 1 - N\left(1 - x_{ijk}\right), \ \forall i,j,k \tag{8}$$

$$v_{k+1} \leq v_k, \ \forall k < K \tag{9}$$

$$v_k \in \{0,1\}, \ \forall k \tag{10}$$

$$x_{ijk} \in \{0,1\}, \ \forall i,j,k \tag{11}$$

$$z_{jk} \in \{0,1\}, \ \forall j,k \tag{12}$$

$$y_{jk} \in \{0,1\}, \ \forall j,k \tag{13}$$

$$u_{jk} \geq 0, \ \forall j,k \tag{14}$$

The objective function (2) seeks to minimize the total distance traveled to serve all customers. Constraints (3) ensure that each customer visits exactly one vehicle, while

constraints (4) limit customer assignments to utilized vehicles. Constraints (5) ensure that customers assigned to a vehicle appear on its respective route. Flow constraints are defined by (6), and constraints (7) ensure that vehicles leave the depot. Constraints (8) utilized the traditional Miller, Tucker, and Zemlin (MTZ) formulation (Miller; Tucker; Zemlin, 1960) to eliminate subroutes. Constraints (9) enforce the usage of vehicles in ascending index order, avoiding multiple equivalent solutions. Finally, constraints (10) through (14) define the domain of the decision variables.

3 Related works

As the VRP and its variants belong to the NP-hard class of problems, heuristic methods are commonly employed to address them. The literature encompasses a variety of approaches, including ant colony systems, genetic algorithms, particle swarm optimization algorithms, and combinations of data mining, classification, and ranking techniques. While this section highlights recent scientific contributions, a more detailed review can be found in Rios *et al*. (2021).

The papers discussed in this section were selected based on their proposals of heuristics and techniques for solving the DVRP. Another selection criterion was the application of these proposed solutions to instances or datasets for evaluation and comparison with other algorithms.

Numerous studies have applied Ant Colony Systems (ACS) to the VRP and its variants. Montemanni *et al*. (2005) proposed a strategy based on ACS to solve the DVRP. Their approach divided the working day into time slices of equal length, processing the arrival of new requests at its end of each time slice, effectively transforming the problem into a series of VRP instances. The quality of their algorithm was assessed using the dataset originally proposed by Kilby, Prosser, and Shaw (1998) and a real-case scenario in Lugano, Switzerland. The instances proposed by Kilby, Prosser, and Shaw (1998) feature problems involving 50 to 199 clients. The authors compared their results (minimum, maximum, and average travel times) with those obtained by the original GRASP (Greedy Randomized Adaptive Search Procedure) proposed by Resende and Ribeiro (2003). Their findings demonstrated that ACS produced higher-quality results over five runs of both algorithms. It was suitable for the proposed real-case scenario, with performance highly dependent on ACS parameter choices, particularly the number of time slices.

Gambardella, Taillard, and Agazzi (1999) proposed the Multiple Ant Colony System for Vehicle Routing Problem with Time Windows (MACS-VRPTW) to optimize the number of vehicles and total travel time. A separated colony system was proposed for each objective, with cooperation facilitated through pheromone updates. This strategy was evaluated on 56 instances from Solomon's VRPTW benchmark (Solomon, 1987), which includes six problem types with 100 nodes each. MACS-VRPTW produced competitive results, achieving the best-known solutions in most instances.

Barán and Schaerer (2003) introduced a multi-objective ant colony system to obtain Pareto optimal solutions by simultaneously: optimizing three objectives: the number of vehicles, total travel time, and total delivery time. The proposed algorithm is an extension of the MACS-VRPTW approach (Gambardella; Taillard; Agazzi, 1999) by ensuring no single objective took precedence. Evaluated using Solomon's instances, the results outperformed those of the MACS-VRPTW, although the authors only provided results for instance C101.

Van Veen *et al*. (2013) developed MACS-DVRPTW, an approach inspired by MACS, for the dynamic version of VRPTW. Their benchmark was based on Solomon's VRPTW instances, with additional dynamic elements, such as clients being revealed progressively throughout the day. Their hybrid algorithm combined ACS with construction and local search methods. They also evaluated their approach using a real-case scenario from a Dutch security company (Yang *et al*., 2017).

Necula, Breaban, and Raschip (2017) addressed the DVRPTW and introduced DVRPTW-ACS, a novel algorithm that employs a single ant colony system to optimize both the number of vehicles and the total distance traveled. Based on Van Veen *et al*. (2013) approach, they divided the working day into equal-sized time slices, treating the dynamic problem as a series of consecutive static instances. DVRPTW-ACS outperformed MACS-DVRPTW in scenarios with higher levels of dynamicity, achieving better results for dynamicity levels ranging from 50% to 100%. However, it did not exceed MACS-DVRPTW at lower dynamicity levels (0% – 10%).

Several studies, including those by Hong (2012), Pillac, Guéret, and Medaglia (2012), and Silva Júnior, Leal, and Reimann (2021), addressed DVRPTW by considering client rejection. Hong (2012) proposed decomposing the dynamic problem into static instances based on new client arrivals and solving them using an improved Large Neighborhood Search (LNS) algorithm. Pillac, Guéret, and Medaglia (2012) developed a Parallel Adaptive Large Neighborhood Search (pALNS) algorithm to compute initial solutions and update them dynamically as new clients arrive. Silva Júnior, Leal, and Reimann (2021) presented a framework comprising seven algorithmic variants, incorporating insertion heuristics, ant colony systems, and neighborhood descent methods. Evaluated using Lackner's (2004) instances – comprising 56 instances with 100 customers from Solomon's dataset – these works showed that while Pillac, Guéret, and Medaglia (2012) approach was more effective in less complex rejection scenarios. Silva Júnior, Leal, and Reimann (2021) algorithm minimized unserved clients more effectively under varying heuristic parameters.

Marinakis and Marinaki (2010) proposed a three-phase solution for DVRP, integrating a genetic algorithm with their Multiple Phase Neighborhood Search-Greedy Randomized Adaptive Search Procedure (MPNS-GRASP) (Marinakis; Migdalas; Pardalos, 2009) and a Particle Swarm Optimization (PSO) algorithm. This solution was evaluated using Solomon (1987) dataset.

Fonseca-Galindo *et al*. (2022) introduced a heuristic for DVRP involving limited-capacity vehicles and stochastic clients. Their approach combined a multi-agent system with trajectory data mining techniques to identify territorial patterns. Using the FP-Growth algorithm for distributed computing, the method reduced data granularity by mapping client locations into of geographic areas, referred to as cells, using the Google S2 geometry library. The heuristic was applied to a real-world dataset of 136,000 packages delivered in Belo Horizonte, Brazil, in 2019.

This section has presented a range of approaches to addressing DVRP. Most solutions were tested on small instances, highlighting a gap in scalability. Fonseca-Galindo *et al*. (2022) applied their heuristic to a large dataset but did not compare the results to other algorithms. Consequently, there remains a need for scalable solutions capable of handling large instances and long distances while facilitating comparisons with other DVRP algorithms. The present work addresses this gap, demonstrating that the proposed heuristic surpasses classical algorithms regarding traveled distance and the number of vehicles allocated for delivery.

ISSN 2447-9187

4 Dynamic Search per Neighbor Route

The proposed algorithm processes package batches during specific periods, as illustrated in Figure 2. The first batch, received at the depot at time T_0, can be considered a static problem (CVRP) as all the packages in the batch are known a priori. From the second batch (time T_1), the problem becomes dynamic (dynamic degree ≤ 50%), as half of the packages already have routes from the first batch. At this time T_{N-1}, the N_{th} batch is incorporated into the problem.

Figure 2 ▶
Batch packages routing in T_N periods.
Source: elaborated by the authors

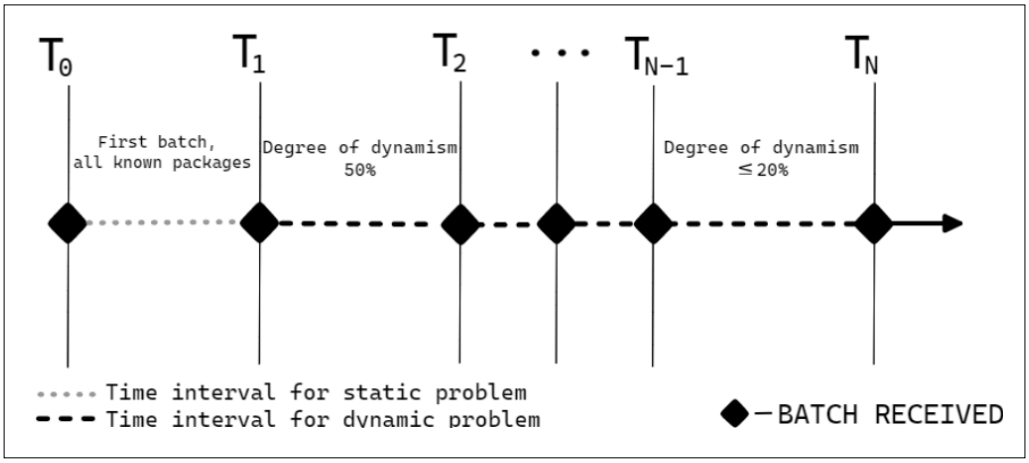
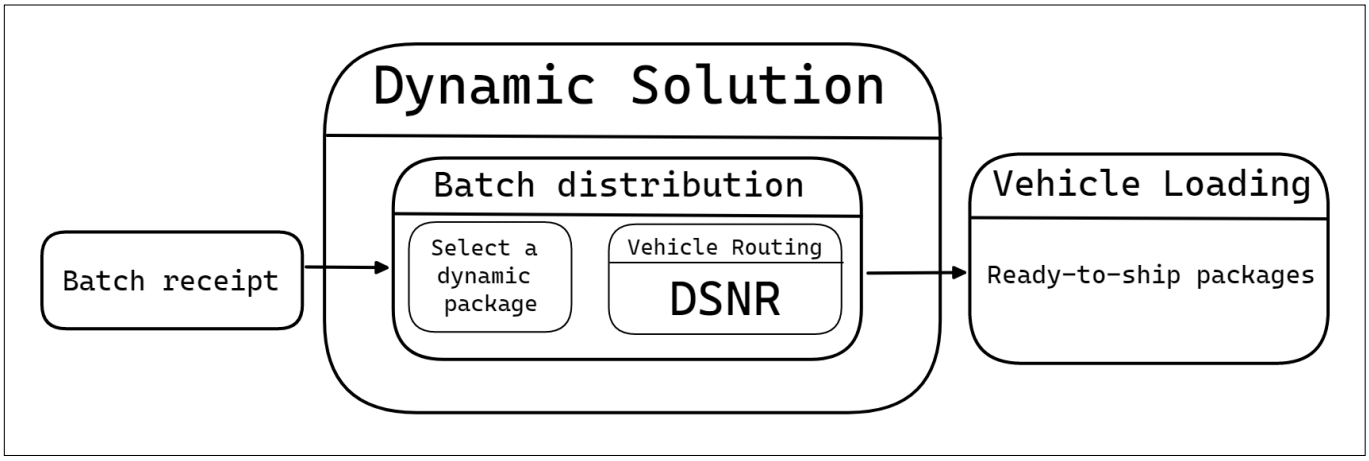


Figure 3 outlines the steps involved in applying the algorithm to a new package batch. Initially, a dynamic package is selected from the batch, and Algorithm 1 is executed. This algorithm determines whether to create a new route or insert the package into an existing neighboring route. If a new route is created, it must be balanced with the nearest existing one, resulting in an updated route solution. Delivery may begin once the vehicle assigned to the route is fully loaded.

Figure 3 ▼
Steps for routing dynamic packages.
Source: elaborated by the authors



The "Dynamic Solution" step in Figure 3 consists of two main components:

a. Select a dynamic package: a batch comprises a set of packages that can be sorted before dispatching them to the final distribution center. Equation 15 calculates the weight of each package. The weight w_i represents the average distance (d_{ij}) between a given

packet i and the already routed packages j, including the depot. The set S includes all packages (including the depot):

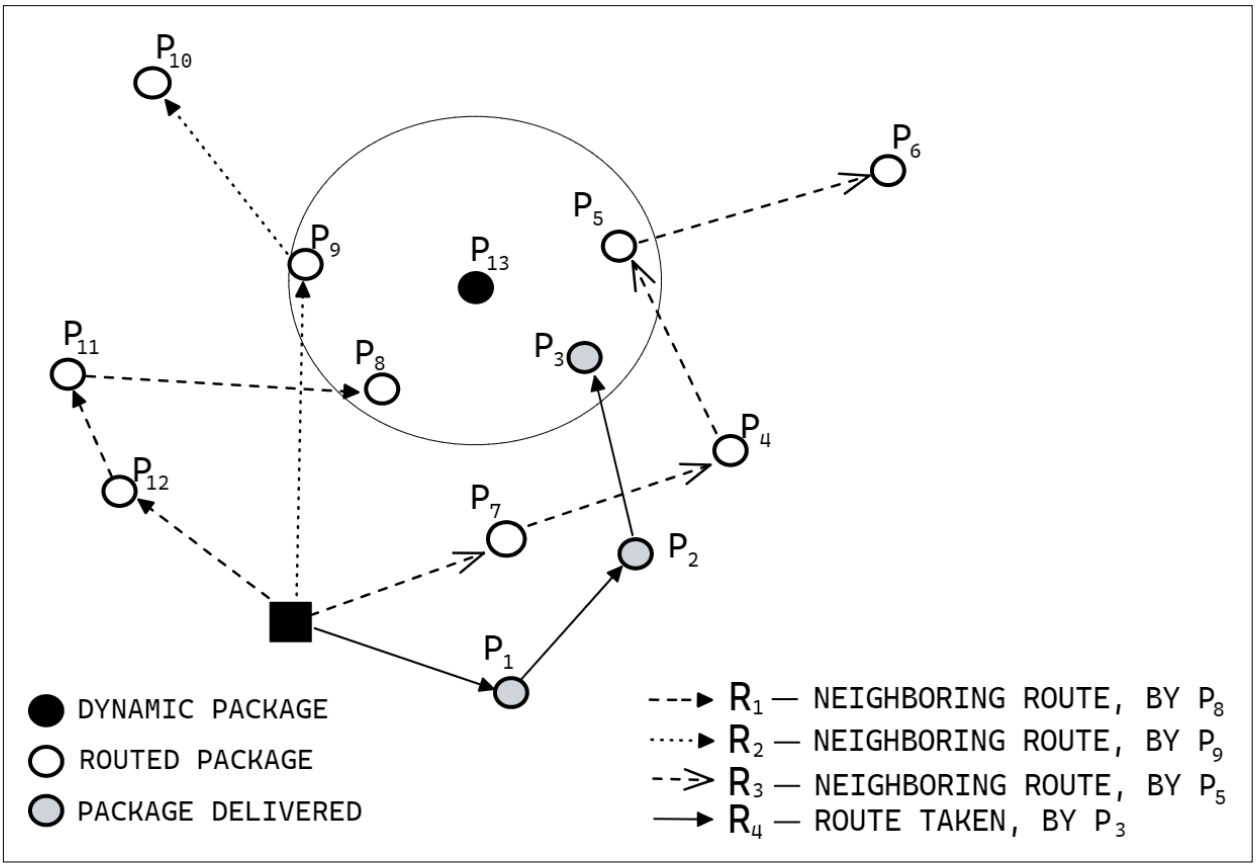
$$w_i = \frac{\sum_{j}^{N} d_{ij}}{|S|} \tag{15}$$

The calculated weight can be utilized to sort the packages in either ascending or descending order.

The proposed algorithm is designed to handle new package batches and reusable routes. As depicted in Figure 4, the scenario includes dynamic packages (black circles), routed packages (white circles), and delivered packages (gray circles). Dynamic packages are those in those in the current batch. Routed packages are assigned to vehicles that have not yet departed the depot. Delivered packages, already allocated to vehicles that have left the depot, cannot have their routes modified.

b. Vehicle routing (DSNR): once a dynamic package is selected, the next task is to identify a neighboring route for its insertion. The routed packages are listed in ascending order of their distance to the dynamic package. After sorting, these routes are defined as neighboring routes of the dynamic package. There may be R neighboring routes, where $1 \leq R \leq N$, and N represents the number of routed packages neighboring the dynamic package. Figure 4 provides an example with $N = 3$, where packages P_5, P_8 and P_9 are selected. P_3 is part of a closed route (the vehicle has already commenced delivery).

Figure 4 ▼
A scenario with one new dynamic package (P_{13}) to be allocated to a new or an existing route.
Source: elaborated by the authors



ISSN 2447-9187

Algorithm 1 ▼
Insert dynamic package into a neighboring route.
Source: elaborated by the authors

Given the dynamic packages' neighboring routes, Algorithm 1 allows us to select the first route, R_3.

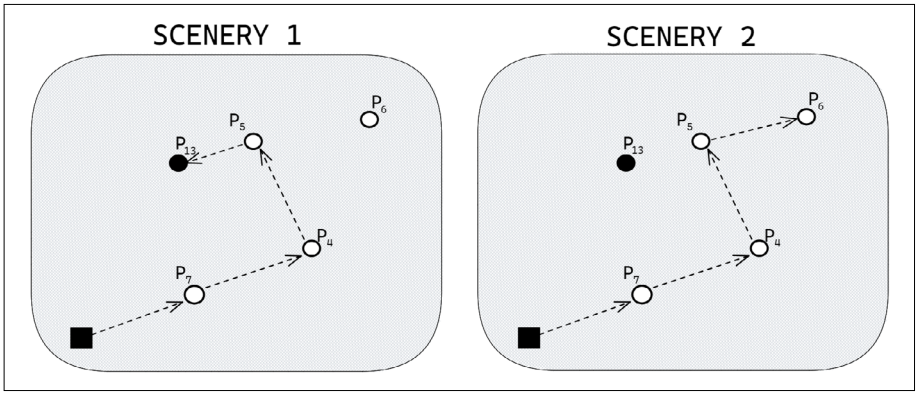
```
function InsertPackage(p, route, routing_packages) {package p insert to route, routing packages is a list of packages that are in a route.}

auxroute = PackageInserted(p, route)
if CapacityExceeded(auxroute) then
    scenery1 = RouteWithDynamicNoWorst(route)
    scenery2 = RouteNoDynamicWithWorst(route)
    if dist(scenery1) < dist(scenery2) then
        EjectsTheWorstPackage(route, routing_packages)
        InsertPackage(p, route, routing_packages)
        return scenery1
    else
        return scenery2 {Scenery 2 = Route without dynamic route}
    end if
else
    return auxroute
end if
end function
```

The algorithm generates an auxiliary route, where the dynamic package is inserted at the optimal position. After insertion, the vehicle capacity of the auxiliary route is evaluated:

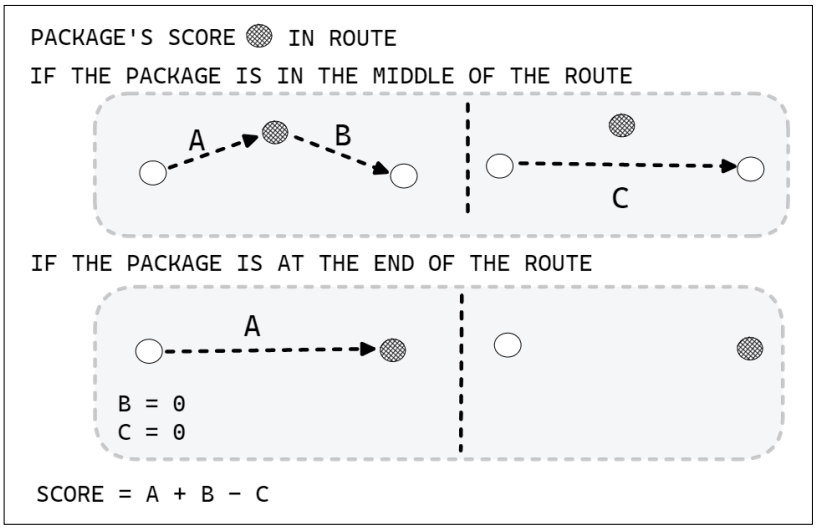
- If the capacity is exceeded, two scenarios are considered, as illustrated in Figure 5. In the first scenario, the dynamic package replaces the worst package in the route. The worst package is the one with the highest score (based on distance). Removing this package enhances the route, as demonstrated in Figure 6. In the second scenario, the dynamic package is not inserted into the route.

Figure 5 ▶
Scenarios for inserting a new package.
Source: elaborated by the authors



ISSN 2447-9187

Figure 6 ▶
Score of a package on a route.
Source: elaborated by the authors



If scenario 1 results in a smaller total distance, the worst package is removed from the route and reclassified as dynamic package, to be inserted at the end of the batch. Otherwise, the dynamic package is evaluated for insertion into the next neighbor route. If no additional neighboring routes are available, a new route is created for the dynamic package.

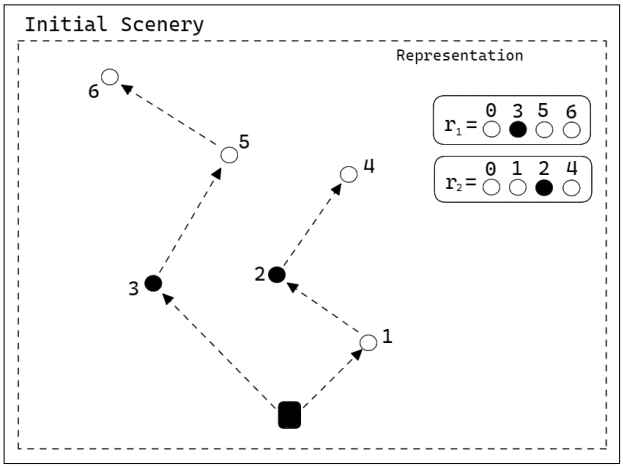
If the route can accommodate the new dynamic package, the auxiliary route is accepted, allowing for the selection of another package from the batch.

After the dynamic package is inserted, any newly created route undergoes optimization with the nearest existing route. This optimization employs a novel exchange operator based on the 2-Opt* method. The operator is applied iteratively until no further improvements in the total distance of the routes can be achieved.

4.1 4C 2-Opt** Optimization

After package routing, two local search operators are applied to refine the solution. The first operator, 2-Opt* (Fahrion; Wrede, 1990), evaluates whether exchanging two packages from different routes improves the total travelled distance. The 2-Opt** operator identifies the optimal placement for a new package within a route. Figure 7 illustrates an example in which r_1 represents the first route and r_1 the second route, with packages 2 and 3 selected to explore new insertion possibilities for dynamic package.

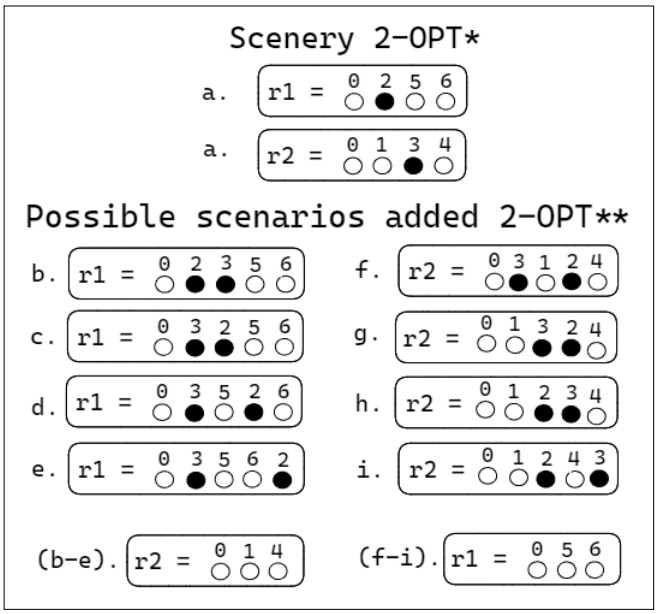
Figure 7 ▶
Initial route for applying 2-Opt*.
Source: elaborated by the authors



ISSN 2447-9187

In the 2-Opt* operator, a single scenario is possible: the direct exchange between packages 2 and 3 (a. in Figure 8). The 2-Opt** operator extends the allocation possibilities by allowing both selected packages to be routed within the same route. The heuristic determines the optimal position for one of the selected packages in the alternative route.

Figure 8 ▶
Figure 8 – 2-Opt** example.
Source: elaborated by the authors



5 Dataset and heuristics evaluation

The proposed routing solutions were tested and evaluated using the Loggibud urban routes benchmark (Loggibud, 2021). This repository provides historical delivery data from three Brazilian cities: Rio de Janeiro, Belém, and Brasília. Each package weighs between 1 and 10 load units, while each vehicle has a maximum capacity of 180 load units. Furthermore, the repository includes georeferenced latitude and longitude data for each package, with distances calculated using the Open Search Routing Machine (OSRM) server.

Deliveries are organized into instances that take place within a single business day, with packages arriving at the warehouse unsorted. However, the system permits sorting into batches, with a maximum of 75 packages per batch. Each batch is assumed to arrive at the warehouse at full capacity.

The repository includes two algorithms designed to solve the dynamic routing problem. Both require parameter *I*, which defines the number of initial regions for training.

Two types of files are available in the Loggibud repository:

- Input file (CVRPInstance): contains data on the warehouse (depot) location and the georeferenced coordinates of all packages schedule for delivery within one business day;
- Output file (CVRPSolution): lists the sequence of packages deliveries for each vehicle departing from the depot.

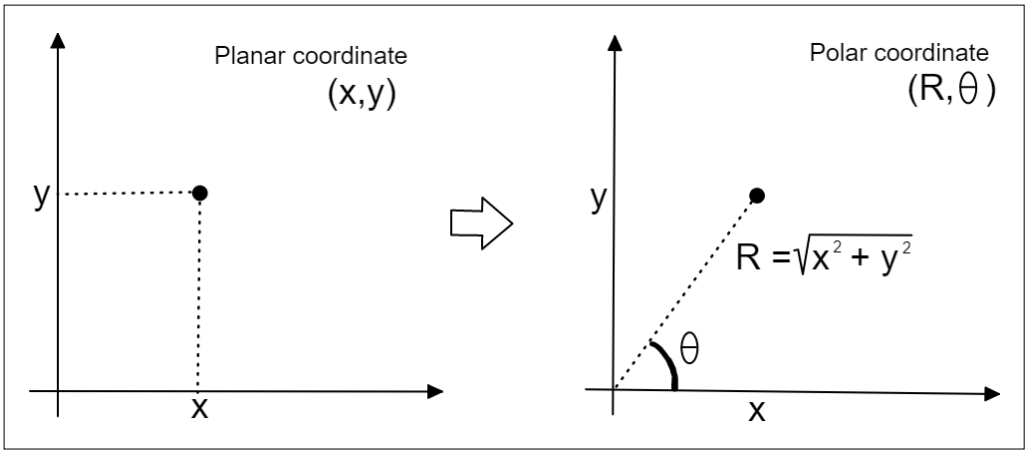
The Loggibud repository implements two dynamic routing algorithms: QRP Sweep (QRPS) and K-Means Greedy (KG).

5.1 QRP Sweep

This algorithm segments the delivery regions using a scanning method, as described by Bertsimas and Van Ryzin (1993) and Gillett and Miller (1974). Initially, a training phase based on historical data is carried out, during which delivery demands points are transformed from planar coordinates (x, y) to polar coordinates (R, Θ), where Θ varies within the range $[-180^{o}, 180^{o}]$ and $R=\sqrt{x^2+y^2}$, as depicted in Figure 9.

A K-Means clustering algorithm is applied to the training dataset in the polar coordinate system, with the depot serving as the central reference point. Once the regions are established, each package is assigned to its corresponding region. When a vehicle reaches its maximum package capacity or when no additional packages remain in the region, it is dispatched. Before dispatch, a Traveling Salesman Problem (TSP) is solved using the Branch-and-Cut algorithm implemented in a commercial high-performance Mixed-Integer Programming (MIP) solver, as detailed in Section 6.

Figure 9 ▶
Converting planar to polar coordinates.
Source: elaborated by the authors



5.2 K-Means Greedy (KG)

The KG algorithm also incorporates a training phase that utilizes historical data. Following the training, each package is allocated to the region with the nearest centroid, with each region linked to a specific vehicle. The corresponding vehicle is dispatched when a region reaches its predetermined limit (i.e., the vehicle's capacity). Similar to the QRPS algorithm, a Branch-and-Cut algorithm from a commercial MIP solver is employed to determine the optimal routing sequence for the dispatched vehicle within the designated region.

6 Results and discussions

This section describes the experiments conducted to evaluate the DSNR algorithm. The proposed algorithm was analyzed and compared with the QRPS and KG implementations available in the Loggibud repository. All algorithms were run on a computer equipped with an AMD Ryzen 5600x processor (6 cores, 12 threads), a RTX 2080 Super, and 16 GB of RAM at 3200 MHz. The Branch-and-Cut algorithm from the IBM ILOG Cplex 20.1 commercial solver was utilized with default settings to solve the necessary mixed integer programming (MIP) subproblems.

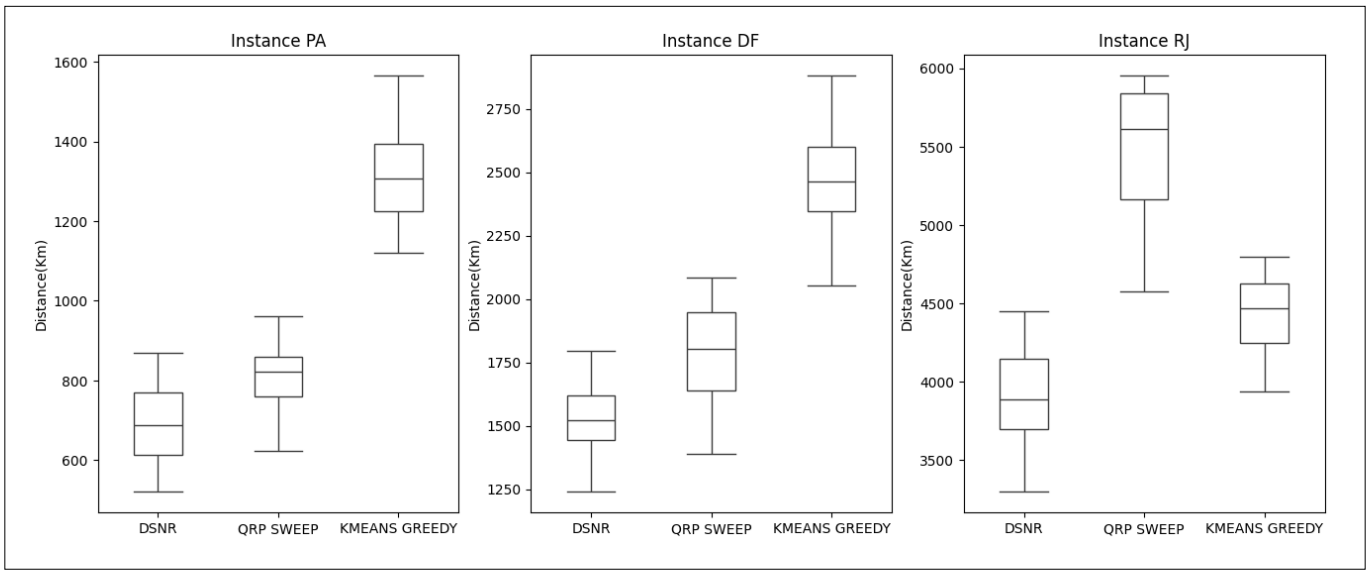
The DSNR to QRPS and KG comparison was motivated by the fact that all three algorithms utilize the same datasets (from the Loggibud benchmark) as inputs to solve the DVRP. Additionally, their solutions provide two key outputs: the number of vehicles required and the total distance traveled (in kilometers).

Subsection 6.1 characterizes each dataset instance (DF, PA, and RJ) based on the distances calculated by each algorithm. Subsection 6.2 describes how the DSNR algorithm functions regarding package ordering. Subsection 6.3 compares the three algorithms across the same instances, considering the number of vehicles required and the total traveled distance. The final delivery costs, computed based on vehicle usage and travel distances reported by each algorithm, are also presented.

6.1 Characterization of dataset instances

Figure 10 ▼
Boxplot of distances calculated by each algorithm for the PA, DF, and RJ datasets.
Source: https://www.kaggle.com/datasets/wiltoncosta7/pesquisa-por-rotas-vizinhas-dinamicas

Figure 10 presents a boxplot of the distances calculated by each algorithm using the Loggibud benchmark. The results indicate that the distances computed by DSNR are consistently lower than those obtained by KG across all scenarios. In the DF dataset, the median of DSNR closely aligns with that of QRPS. However, in the PA and RJ datasets, DSNR exhibit a clear advantage, producing lower distances than both QRPS and KG. These findings suggest that DSNR performs particularly well in instances involving longer distances.



6.2 Ordering of the dynamic search per neighbor route

The DSNR approach accepts a package that is to be routed and delivered. Each package is part of a set, and the packages within a set can be ordered according to their weights. The weight of a package is determined by the average distance between that package and all others in the set (Equation 2). Once the package weights are assigned, they can be input into the algorithm based on the following approaches:

- Non-ordering (*N*): packages are processed in the original sequence as they appear in the set;

- Increasing ordering (*I*): packages are processed in ascending order of weight;
- Decreasing ordering (*D*): packages are processed in descending order of weight.

These three ordering approaches were compared to evaluate their effect on algorithm performance. Each set comprises 30 packages. Tables 1, 2, and 3 present the total distance travelled and the number of vehicles required for each ordering technique (*N*, *I*, *D*) across 10 dataset instances from three cities in Brazil: Belém (PA), Brasília (DF), and Rio de Janeiro (RJ). The results for all 30 instances are available at https://www.kaggle.com/datasets/wiltoncosta7/pesquisa-por-rotas-vizinhas-dinamicas. For each instance, the optimal (lowest) distance and vehicles count are highlighted in bold.

Table 1 ▶
Traveled distance and number of vehicles for the three ordering approaches (Belém – PA dataset).
Source: https://www.kaggle.com/datasets/wiltoncosta7/pesquisa-por-rotas-vizinhas-dinamicas

Instance	Distance			Vehicles		
	DSNR-N	DSNR-I	DSNR-D	DSNR-N	DSNR-I	DSNR-D
pa-90	608	**594**	669	10	**10**	10
pa-91	706	**699**	769	10	**10**	10
pa-92	**520**	700	533	**10**	10	10
pa-93	552	593	**513**	10	10	**10**
pa-94	806	**733**	806	10	**10**	10
pa-95	**675**	854	804	**10**	10	10
pa-96	836	**784**	843	10	**10**	10
pa-97	**649**	721	698	**10**	10	10
pa-98	780	**696**	765	10	**10**	10
pa-99	**761**	908	806	**10**	10	10
Average	**689.3**	728.2	720.6	**10**	10	10

Table 2 ▶
Traveled distance and number of vehicles for the three ordering approaches (Brasília – DF dataset).
Source: https://www.kaggle.com/datasets/wiltoncosta7/pesquisa-por-rotas-vizinhas-dinamicas

Instance	Distance			Vehicles		
	DSNR-N	DSNR-I	DSNR-D	DSNR-N	DSNR-I	DSNR-D
df-90	1,465	1,466	**1,448**	46	**44**	45
df-91	**1,293**	1,344	1,331	39	**37**	41
df-92	**1,394**	1,402	1,409	43	42	**40**
df-93	1,356	**1,255**	1,269	40	39	**38**
df-94	1,621	**1,442**	1,693	34	**31**	35
df-95	1,582	1,602	**1,540**	50	**48**	51
df-96	**1,508**	1,647	1,795	41	**39**	39
df-97	**1,303**	1,396	1,344	39	**34**	39
df-98	1,321	**1,261**	1,355	**37**	38	41
df-99	1,519	1,455	**1,441**	31	**30**	36
Average	1,436.2	**1,427**	1,462.5	40	**38.2**	40.5

Table 3 ▶
Traveled distance and number of vehicles for the three ordering approaches (Rio de Janeiro – RJ dataset). *Source: https://www.kaggle.com/datasets/wiltoncosta7/pesquisa-por-rotas-vizinhas-dinamicas*

Instance	Distance			Vehicles		
	DSNR-N	DSNR-I	DSNR-D	DSNR-N	DSNR-I	DSNR-D
rj-90	**3,896**	3,919	4,007	**188**	193	207
rj-91	4,192	**3,978**	4,134	220	**205**	213
rj-92	4,077	**4,019**	4,185	**203**	207	213
rj-93	4,017	**3,963**	4,118	172	**158**	162
rj-94	**3,881**	3,887	3,998	**161**	164	171
rj-95	3,455	3,451	**3,432**	**113**	117	121
rj-96	3,980	**3,801**	3,994	157	147	**146**
rj-97	3,371	**3,227**	3,498	171	**156**	170
rj-98	3,985	**3,979**	4,096	**169**	172	183
rj-99	**4,221**	4,241	4,385	**215**	217	230
Average	3,907.5	**3,846.5**	3,984.7	176.9	**173.6**	181.6

To estimate the cost of trips for each instance, the following constraints were applied:

- Distances are measured in kilometers (km);
- Each instance must be completed within a single workday;
- Each vehicle incurs an average cost of R$ 1.92 per km;
- The minimum cost for allocating a vehicle to a delivery is R$ 13.90.

All vehicles are assumed to be homogeneous, with identical capacity and operational costs. The total delivery cost is computed using Equation 16:

$$Cost = R\$\,1.92 \times Distance + R\$\,13.90 \times Vehicles \tag{16}$$

Table 4 presents the total costs (in thousands of reais, R$) for all instances in Belém, Brasília, and Rio de Janeiro, based on the distance and vehicle data from Tables 1, 2, and 3.

Table 4 ▶
Total delivery costs (R$ × 1,000) for the three ordering approaches. *Source: https://www.kaggle.com/datasets/wiltoncosta7/pesquisa-por-rotas-vizinhas-dinamicas*

City - Nº samples	DSNR-N (COST)	DSNR-I (COST)	DSNR-D(COST)
PA-300	**R$ 43,954**	R$ 45,115	R$ 44,080
DF-1000	R$ 105,181	R$ 105,335	**R$ 104,850**
RJ-4000	R$ 295,354	**R$ 294,580**	R$ 299,676

Depending on the ordering approach, costs differed. To determine whether this difference was statistically significant, tests were conducted. The Shapiro-Wilk test for normality was applied to the cost values for each dataset instance across the three cities. Assuming a confidence level of a $\alpha = 0.05$, the test results showed p-values greater than α, indicating no grounds to reject the normality hypothesis.

Given that the normality was not rejected, an analysis of variance (ANOVA) test was performed to assess potential differences among the three ordering approaches. The

ISSN 2447-9187

lowest p-value obtained from ANOVA was 0.41, greater than α, suggesting no statistically significant effect from the ordering strategy. Consequently, the null hypothesis of no effects among the ordering approaches is accepted.

Based on this analysis, the non-ordering approach (N) was adopted for further evaluation and comparison with other dynamic routing techniques.

6.3 Comparison and data analysis

Tables 5, 6, and 7 show the results concerning distance and vehicle counts for the DSNR (non-ordering), QRPS, and KG algorithms. QRPS and KG were executed following *task2* in the Loggibud benchmark repository[1]. All algorithms were applied to instances from PA, DF, and RJ, with each batch consisting of 75 packages.

[1] Available at: https://github.com/loggi/loggibud. Accessed on: 12 Feb. 2025.

Table 5 ▶
Distance (km) and number of vehicles for the dynamic algorithms on instances from Belém (PA).
Source: https://www.kaggle.com/datasets/wiltoncosta7/pesquisa-por-rotas-vizinhas-dinamicas

Instance	Distance			Vehicles		
	DSNR	QRPS	KG	DSNR	QRPS	KG
pa-90	**608**	718	1,198	**10**	12	82
pa-91	**706**	876	1,255	**10**	12	81
pa-92	**520**	816	1,322	**10**	12	86
pa-93	**552**	622	1,182	**10**	12	81
pa-94	**806**	807	1,295	**10**	11	85
pa-95	**675**	917	1,548	**10**	13	88
pa-96	**836**	952	1,524	**10**	12	90
pa-97	**649**	754	1,135	**10**	13	76
pa-98	780	**772**	1,215	**10**	12	78
pa-99	**761**	912	1,564	**10**	12	88
Average	**689.3**	814.6	1,323.8	**10**	12.1	83.5

Table 6 ▶
Distance (km) and number of vehicles for the dynamic algorithms on instances from Brasília (DF).
Source: https://www.kaggle.com/datasets/wiltoncosta7/pesquisa-por-rotas-vizinhas-dinamicas

Instance	Distance			Vehicles		
	DSNR	QRPS	KG	DSNR	QRPS	KG
df-90	**1,465**	1,748	2,549	**46**	46	107
df-91	**1,293**	1,540	2,231	39	**37**	100
df-92	**1,394**	1,681	2,472	43	**41**	108
df-93	**1,356**	1,388	2,053	40	**36**	92
df-94	**1,621**	1,821	2,729	**34**	42	110
df-95	**1,582**	1,986	2,775	50	**45**	114
df-96	**1,508**	1,932	2,419	**41**	46	106
df-97	**1,303**	1,535	2,449	39	**36**	106
df-98	**1,321**	1,594	2,362	**37**	38	103
df-99	**1,519**	1,626	2,180	**31**	39	98
Average	**1,436.2**	1,685.1	2,421.9	**40**	40.6	104.4

Table 7 ▶
Distance (km) and number of vehicles for the dynamic algorithms on instances from Rio de Janeiro (RJ).
Source: https://www.kaggle.com/datasets/wiltoncosta7/pesquisa-por-rotas-vizinhas-dinamicas

Instance	Distance			Vehicles		
	DSNR	QRPS	KG	DSNR	QRPS	KG
rj-90	**3,896**	3,919	4,007	**188**	193	207
rj-91	4,192	**3,978**	4,134	220	**205**	213
rj-92	4,077	**4,019**	4,185	**203**	207	213
rj-93	4,017	**3,963**	4,118	172	**158**	162
rj-94	**3,881**	3,887	3,998	**161**	164	171
rj-95	3,455	3,451	**3,432**	**113**	117	121
rj-96	3,980	**3,801**	3,994	157	147	**146**
rj-97	3,371	**3,227**	3,498	171	**156**	170
rj-98	3,985	**3,979**	4,096	**169**	172	183
rj-99	**4,221**	4,241	4,385	**215**	217	230
Average	3,907.5	**3,846.5**	3,984.7	176.9	**173.6**	181.6

Table 8 shows the total delivery costs for each set of instances (DF, PA, RJ) using various dynamic algorithms. The parameters for QRPS and KG were optimized to find the best configuration for each algorithm. Both algorithms require an initial number of clusters. Furthermore, a case study was performed for the KG algorithm to identify the optimal I parameter. The empirically obtained values that provided the best results for each algorithm are presented below:

- For the PA instance, QRPS achieved optimal performance with 10 clusters, whereas KG performed best with 8 clusters;
- For the DF instance, QRPS excelled with 40 clusters, while KG obtained the best results with 10 clusters;
- For the RJ instance, the most effective performance was realized when the number of clusters equaled the number of vehicles.

Even with the optimal cluster configurations, DSNR achieved the lowest cost for DF and RJ instances. For the PA instance, there was a slight difference between DSNR and KG, which was not statistically significant.

Table 8 ▶
Total cost of the dynamic approaches.
Source: https://www.kaggle.com/datasets/wiltoncosta7/pesquisa-por-rotas-vizinhas-dinamicas

City - Nº samples	DSNR	QRPS	KG
PA-300	R$ 43,954	R$ 48,604	**R$ 43,896**
DF-1000	**R$ 105,181**	R$ 120,947	R$ 107,731
RJ-4000	**R$ 295,354**	R$ 395,411	R$ 342,507

The maximum cost reduction achieved by the DSNR algorithm, compared to the best cost obtained using the KG algorithm on the RJ instance, was 13.76%. The Shapiro-Wilk test was applied to analyze the normality of the cost data for each sample (Tables 9, 10, and 11). Most p-values were higher than $\alpha = 0.05$, indicating no significant effect to reject the null hypothesis (normality).

Table 9 ▶
Shapiro-Wilk normality test for distance data from Belém (PA). *Source: https://www.kaggle.com/datasets/wiltoncosta7/pesquisa-por-rotas-vizinhas-dinamicas*

Shapiro-Wilk	DSNR	QRPS	KG
W	0.9889	0.9770	0.9768
***p*-value**	0.7951	0.7425	0.7351

Table 10 ▶
Shapiro-Wilk normality test for distance data from Brasília (DF). *Source: https://www.kaggle.com/datasets/wiltoncosta7/pesquisa-por-rotas-vizinhas-dinamicas*

Shapiro-Wilk	DSNR	QRPS	KG
W	0.9771	0.9610	0.9869
***p*-value**	0.7448	0.3278	0.9644

Table 11 ▶
Shapiro-Wilk normality test for distance data from Rio de Janeiro (RJ). *Source: https://www.kaggle.com/datasets/wiltoncosta7/pesquisa-por-rotas-vizinhas-dinamicas*

Shapiro-Wilk	DSNR	QRPS	KG
W	0.9572	0.8805	0.9768
***p*-value**	0.2622	**0.029**	0.1284

The ANOVA test was applied to compare the differences in distance among the algorithms. All p-values were less than α (p-value $< 2.2e^{-16}$), indicating significant differences in distance among DSNR, QRPS, and KG. The results presented in this section are available online[2].

[2] Available at: https://www.kaggle.com/datasets/wiltoncosta7/pesquisa-por-rotas-vizinhas-dinamicas. Accessed on: 12 Feb. 2025.

7 Conclusions

This study presents the design and development of a new heuristic for the Dynamic Vehicle Routing Problem (DVRP). The proposed algorithm, Dynamic Search per Neighbors Routes (DSNR), generates routes based on delivery regions, enabling new package batches to reuse previously established routes.

The DSNR approach was assessed using a set of routing instances from the Loggibud benchmark, which represents delivery points from three different cities in Brazil. The proposed method surpassed two other DVRP techniques, QRPS and KG, regarding distance and the number of vehicles required. Specifically, DSNR achieved distance savings ranging from 15% to 17%.

The advantages of the DSNR heuristic are also evident in cost reductions, as shorter distances lead to lower total operational costs. Furthermore, the proposed approach contributes to the sustainability of delivery operations, as reduced travel distances and fewer required vehicles directly affect greenhouse gas emissions.

One limitation of the DSNR approach lies in the pre-processing stage for package ordering. The data in Table 4 suggest that ordering may influence the final routing results and associated costs. This poses challenges for implementing DSNR in real-world systems with high dynamicity, where delivery distances frequently fluctuate.

Future research could explore integrating the K-Means clustering technique with DSNR to define delivery regions based on historical data. Another potential direction would involve initiating the dynamic algorithm with an optimized static approach for the

initial package set before applying DSNR to subsequent packages. These enhancements could contribute to further advancements in the field of dynamic vehicle routing.

Financial Support

The authors acknowledge the financial support provided by the Brazilian Research Agency – CNPq (Process nº 133954/2020-0) and the Federal University of Mato Grosso do Sul.

Conflict of Interest

The authors declare that they have no conflict of interest.

Note

This article is derived from a Master's dissertation in Computer Science at the Federal University of Mato Grosso do Sul (UFMS), available at: https://repositorio.ufms.br/handle/123456789/5173.

Contributions to the article

COSTA, W. G. G.; SANTOS, R.: conception or design of the study/research; data analysis and/or interpretation; final review with critical and intellectual contributions to the manuscript. **SOLER, W. A. O.; DANTAS, B. A.:** data analysis and/or interpretation; final review with critical and intellectual contributions to the manuscript. All authors participated in the writing, discussion, reading, and approval of the final version of the article.

References

BARÁN, B.; SCHAERER, M. A multiobjective ant colony system for vehicle routing problem with time windows. *In*: IASTED INTERNATIONAL CONFERENCE APPLIED INFORMATICS, 21., 2003, Innsbruck. **Proceedings** […]. Innsbruck: International Association of Science and Technology for Development, 2003. p. 97-102. Available at: https://www.cnc.una.py/publicaciones/1_75.pdf. Accessed on: 11 June 2024.

BERTSIMAS, D. J.; VAN RYZIN, G. Stochastic and dynamic vehicle routing with general demand and interarrival time distributions. **Advances in Applied Probability**, v. 25, n. 4, p. 947-978, 1993. DOI: https://doi.org/10.2307/1427801.

BRÄYSY, O.; GENDREAU, M. Vehicle routing problem with time windows, Part II: Metaheuristics. **Transportation Science**, v. 39, n. 1, p. 119-139, 2005. DOI: https://doi.org/10.1287/trsc.1030.0057.

CHEN, S.; CHEN, R.; GAO, J. A monarch butterfly optimization for the dynamic vehicle routing problem. **Algorithms**, v. 10, n. 3, 107, 2017. DOI: https://doi.org/10.3390/a10030107.

COMERT, S. E.; YAZGAN, H. R.; KIR, S.; YENER, F. A cluster first-route second approach for a capacitated vehicle routing problem: a case study. **International Journal of Procurement Management (IJPM)**, v. 11, n. 4, p. 399-419, 2018. DOI: https://dx.doi.org/10.1504/IJPM.2018.092766.

DANTZIG, G. B.; RAMSER, J. H. The truck dispatching problem. **Management Science**, v. 6, n. 1, p. 80-91, 1959. Available at: https://www.jstor.org/stable/2627477. Accessed on: 18 May 2025.

FAHRION, R.; WREDE, M. On a principle of chain-exchange for vehicle-routing problems (1-VRP). **Journal of the Operational Research Society**, v. 41, n. 9, p. 821-827, 1990. DOI: https://doi.org/10.2307/2583497.

FONSECA-GALINDO, J. C.; SURITA, G. C.; MAIA NETO, J.; CASTRO, C. L.; LEMOS, A. P. A multi-agent system for solving the Dynamic Capacitated Vehicle Routing Problem with stochastic customers using trajectory data mining. **Expert Systems with Applications**, v. 195, 116602, 2022. DOI: https://doi.org/10.1016/j.eswa.2022.116602.

GAMBARDELLA, L. M.; TAILLARD, É.; AGAZZI, G. MACS-VRPTW: A multiple ant colony system for vehicle routing problems with time windows. *In*: CORNE, D.; DORIGO, M.; GLOVER, F. (ed.). **New ideas in optimization**. London: McGraw-Hill, 1999. p. 63-76.

GILLETT, B. E.; MILLER, L. R. A heuristic algorithm for the vehicle-dispatch problem. **Operations Research**, v. 22, n. 2, p. 340-349, 1974. Available at: https://www.jstor.org/stable/169591. Accessed on: 18 May 2025.

HONG, L. An improved LNS algorithm for real-time vehicle routing problem with time windows. **Computers & Operations Research**, v. 39, n. 2, p. 151-163, 2012. DOI: https://doi.org/10.1016/j.cor.2011.03.006.

KILBY, P.; PROSSER, P.; SHAW, P. Dynamic VRPs: a study of scenarios. Report APES-06-1998. **University of Strathclyde Technical Report**, v. 1, n. 11, 1998. Available at: https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=19df8728e4a9b273b6e0a6f84b7f39ff4543e565. Accessed on: 18 May 2025.

KOÇ, Ç.; BEKTAŞ, T.; JABALI, O.; LAPORTE, G. Thirty years of heterogeneous vehicle routing. **European Journal of Operational Research**, v. 249, n. 1, p. 1-21, 2016. DOI: https://doi.org/10.1016/j.ejor.2015.07.020.

LACKNER, A. **Dynamische Tourenplanung mit ausgewählten metaheuristiken**: eine untersuchung am beispiel des kapazitätsrestriktiven dynamischen tourenplanungsproblems mit zeitfenstern. Göttingen: Cuvillier Verlag, 2004. (Göttinger Wirtschaftsinformatik, v. 47). Available at: https://cuvillier.de/de/shop/publications/2964-dynamische-tourenplanung-mit-ausgewahlten-metaheuristiken. Accessed on: 18 May 2025. In German.

LARSEN, A. **The dynamic vehicle routing problem**. 2000. Ph.D Thesis (Doctorate in Mathematical Modelling) – Technical University of Denmark, Lyngby, 2000. Available at: https://backend.orbit.dtu.dk/ws/portalfiles/portal/5261816/imm143.pdf. Accessed on: 18 May 2025.

LARSEN, A.; MADSEN, O.; SOLOMON, M. Partially dynamic vehicle routing: models and algorithms. **Journal of the Operational Research Society**, v. 53, n. 6, p. 637-646, 2002. DOI: https://doi.org/10.1057/palgrave.jors.2601352.

LENSTRA, J. K.; KAN, A. H. G. R. Complexity of vehicle routing and scheduling problems. **Networks**, v. 11, n. 2, p. 221-227, 1981. DOI: https://doi.org/10.1002/net.3230110211.

LOGGIBUD. LoggiBUD: Loggi Benchmark for Urban Deliveries. **GitHub Repository**, 2021. Available at: https://github.com/loggi/loggibud. Accessed on: 11 June 2024.

MARINAKIS, Y.; MARINAKI, M. A hybrid genetic: Particle Swarm Optimization Algorithm for the vehicle routing problem. **Expert Systems with Applications**, v. 37, n. 2, p. 1446-1455, 2010. DOI: https://doi.org/10.1016/j.eswa.2009.06.085.

MARINAKIS, Y.; MIGDALAS, A.; PARDALOS, P. M. Multiple phase neighborhood search: GRASP based on Lagrangean relaxation, random backtracking Lin-Kernighan and path relinking for the TSP. **Journal of Combinatorial Optimization**, v. 17, n. 2, p. 134-156, 2009. DOI: https://doi.org/10.1007/s10878-007-9104-2.

MILLER, C. E.; TUCKER, A. W.; ZEMLIN, R. A. Integer programming formulation of traveling salesman problems. **Journal of the ACM (JACM)**, v. 7, n. 4, p. 326-329, 1960. DOI: https://doi.org/10.1145/321043.321046.

MONTEMANNI, R.; GAMBARDELLA, L. M.; RIZZOLI, A. E.; DONATI, A. V. Ant colony system for a dynamic vehicle routing problem. **Journal of Combinatorial Optimization**, v. 10, n. 4, p. 327-343, 2005. DOI: https://doi.org/10.1007/s10878-005-4922-6.

NECULA, R.; BREABAN, M.; RASCHIP, M. Tackling dynamic vehicle routing problem with time windows by means of ant colony system. *In*: 2017 IEEE CONGRESS ON EVOLUTIONARY COMPUTATION (CEC), 2017, Donostia. **Proceedings** […]. Donostia: IEEE, 2017. p. 2480-2487. DOI: https://doi.org/10.1109/CEC.2017.7969606.

PILLAC, V.; GENDREAU, M.; GUÉRET, C.; MEDAGLIA, A. L. A review of dynamic vehicle routing problems. **European Journal of Operational Research**, v. 225, n. 1, p. 1-11, 2013. DOI: https://doi.org/10.1016/j.ejor.2012.08.015.

PILLAC, V.; GUÉRET, C.; MEDAGLIA, A. **A fast re-optimization approach for dynamic vehicle routing**: Research Report. Nantes: École des Mines de Nantes, 2012. Available at: https://hal.science/hal-00739782. Accessed on: 11 June 2024.

PSARAFTIS, H. N. Dynamic vehicle routing problems. *In*: GOLDEN, B. L.; ASSAD, A. A. (ed.). **Vehicle routing**: methods and studies. Amsterdam: North Holland: Elsevier, 1988. v. 16, p. 223-248.

RALPHS, T. K.; KOPMAN, L.; PULLEYBLANK, W. R.; TROTTER, L. E. On the capacitated vehicle routing problem. **Mathematical Programming**, v. 94, n. 2, p. 343-359, 2003. DOI: https://doi.org/10.1007/s10107-002-0323-0.

RESENDE, M. G. C.; RIBEIRO, C. C. Greedy randomized adaptive search procedures. *In*: GLOVER, F.; KOCHENBERGER, G. A. (ed.). **Handbook of metaheuristics**. Boston: Springer, 2003. p. 219-249. DOI: https://doi.org/10.1007/0-306-48056-5_8.

RIOS, B. H. O.; XAVIER, E. C.; MIYAZAWA, F. K.; AMORIM, P.; CURCIO, E.; SANTOS, M. J. Recent dynamic vehicle routing problems: a survey. **Computers & Industrial Engineering**, v. 160, 107604, 2021. DOI: https://doi.org/10.1016/j.cie.2021.107604.

SILVA JÚNIOR, O. S.; LEAL, J. E.; REIMANN, M. A multiple ant colony system with random variable neighborhood descent for the dynamic vehicle routing problem with time windows. **Soft Computing,** v. 25, n. 4, p. 2935-2948, 2021. DOI: https://doi.org/10.1007/s00500-020-05350-4.

SOLOMON, M. M. Algorithms for the vehicle routing and scheduling problems with time window constraints. **Operations Research**, v. 35, n. 2, p. 254-265, 1987. Available at: https://www.jstor.org/stable/170697. Accessed on: 18 May 2025.

VAN VEEN, B.; EMMERICH, M.; YANG, Z.; BÄCK, T.; KOK, J. Ant colony algorithms for the dynamic vehicle routing problem with time windows. *In*: VICENTE, J. M. F.; SÁNCHEZ, J. R. Á.; LÓPEZ, F. P.; MOREO, F. J. T. (ed.). **Natural and Artificial Computation in Engineering and Medical Applications**. Berlin: Springer, 2013. p. 1-10. DOI: https://doi.org/10.1007/978-3-642-38622-0_1.

YANG, Z.; VAN OSTA, J.-P.; VAN VEEN, B.; VAN KREVELEN, R.; VAN KLAVEREN, R.; STAM, A.; KOK, J.; BÄCK, T.; EMMERICH, M. Dynamic vehicle routing with time windows in theory and practice. **Natural Computing**, v. 16, n. 1, p. 119-134, 2017. DOI: https://doi.org/10.1007/s11047-016-9550-9.