






# A Real-World Multi-Depot, Multi-Period, and Multi-Trip Vehicle Routing Problem with Time Windows

Mirko Cavecchia<sup>1</sup>, Thiago Alves de Queiroz<sup>2</sup>, Riccardo Lancellotti<sup>3</sup>, Giorgio Zucchi<sup>4</sup> and Manuel Iori<sup>1</sup>

<sup>1</sup>*Department of Sciences and Methods for Engineering, University of Modena and Reggio Emilia, 42122, Reggio Emilia, Italy*

<sup>2</sup>*Institute of Mathematics and Technology, Federal University of Catalão, 75704-020, Catalão-GO, Brazil*

<sup>3</sup>*Department of Engineering “Enzo Ferrari”, University of Modena and Reggio Emilia, 41125, Modena, Italy*

<sup>4</sup>*R&D Department, Coopservice S.coop.p.a, 42122, Reggio Emilia, Italy*

**Keywords:** Vehicle Routing Problem, Multiple Depots, Multiple Periods, Multiple Trips, Variable Neighbourhood Search, Mixed Integer Linear Programming.


**Abstract:** Rich vehicle routing problems frequently arise in real-world applications. The related literature is trying to effectively address many of them, considering all their constraints and the need to solve large practical instances. In this paper, we handle a rich vehicle routing problem that emerges from an Italian service provider company operating in the delivery of pharmaceutical products. This problem is characterized by constraints that include multiple depots, periods, and trips, as well as specific time windows, customer-vehicle incompatibilities, and restrictions on vehicle working hours. To solve this problem, we propose a two-phase decomposition approach. In the first phase, we generate a set of trips that satisfy all problem constraints for a single period. Then, in the second phase, three algorithms based, respectively, on a greedy heuristic, a variable neighbourhood search metaheuristic, and a mixed integer linear programming model are tested to combine the generated trips into routes for the overall set of periods. Computational results on real-world instances show that the model hits the one-hour time limit for cases with more than 150 trips. The metaheuristic algorithm performs better in terms of objective function value. The greedy algorithm is faster but is outperformed by the other methods in terms of solution quality.


## 1 INTRODUCTION


The logistics industry has experienced rapid growth in recent years, with the COVID-19 pandemic further highlighting the need for accurate and timely decision-making. In 2020, the global logistics industry was valued at approximately USD 5.73 trillion, and this figure is projected to nearly triple by 2028 (Gosain, 2023). Today, one of the main challenges concerns the development of solution methods that closely reflect real-world scenarios, in which different factors, such as constraints related to daily operations


and unforeseen events, must be considered. The goals are to reduce operational costs, address environmental concerns, and improve labour efficiency.


This study addresses a real-world vehicle routing problem (VRP) faced by an Italian service provider company that delivers pharmaceutical products to hospitals and medical facilities. The problem involves a set of customers, each with a specific time window, who must be served by a heterogeneous fleet of vehicles. Each customer can demand products in different periods, and the products have to be delivered in the periods in which they have been demanded. Vehicles are allowed to perform multiple trips within each given period. The vehicles are associated with different depots, and each vehicle is required to begin and conclude its route at the same depot. The objective is to minimize the overall cost associated with the vehicle usage. This is computed by considering a first

<sup>a</sup> <https://orcid.org/0000-0002-4129-9876>

<sup>b</sup> <https://orcid.org/0000-0003-2674-3366>

<sup>c</sup> <https://orcid.org/0000-0002-9470-8784>

<sup>d</sup> <https://orcid.org/0000-0002-5459-7290>

<sup>e</sup> <https://orcid.org/0000-0003-2097-6572>

cost if a vehicle is used at least once in the set of periods, and a second cost incurred each time a vehicle is used. Due to geographical constraints, some customers cannot be served by larger vehicles, thus imposing incompatibilities between customers and vehicles. The resulting problem is a challenging variant of a multi-trip vehicle routing problem with time windows (MTVRPTW).

The main contributions of this work are (i) the study of a challenging MTVRPTW originating from a real-world company, (ii) the proposal of a number of solution algorithms, among which a variable neighbourhood search (VNS) based heuristic (Hansen and Mladenović, 2001), and (iii) the execution of extensive computational tests on real-world instances.

To solve the problem, we generalize the two-phase decomposition approach by (Cavecchia et al., 2024), originally developed to solve the single-period variant of the problem. In this paper, we extend it to deal with a multi-depot, multi-period, and multi-trip vehicle routing problem with time windows (MM-MVRPTW). In the first phase of the decomposition approach, we generate a set of trips that start and end at the same depot. In the second phase, we assign these trips to the set of available vehicles, creating routes with multiple trips. The aim is to minimize the costs associated with the usage of the vehicles. We use this decomposition approach in multiple ways. First of all, we present a mixed integer linear programming (MILP) model that attempts to optimally combine, in phase two, the trips created in phase one. We then propose a constructive heuristic, which we call bounded starting time greedy (BSTG) algorithm, which assigns the generated trips to the available vehicles according to increasing values of their starting times. We finally develop a novel VNS-based heuristic that attempts several assignments of the generated trips by invoking mechanisms of perturbation and local search.

The remainder of this paper is organized as follows. Section 2 reviews recent contributions related to the problem that we tackle, including surveys and seminal studies. Section 3 provides a formal problem description, outlining its objective function and constraints. Section 4 presents the details of the two-phase approach, detailing each developed algorithm. Section 5 describes the instances that we solved and the computational results that we obtained from all the algorithms, contrasting them in terms of objective function values, computing times, and numbers of vehicles used. Section 6 offers some concluding remarks and potential directions for future research.

## 2 LITERATURE REVIEW

The VRP is a well-known NP-hard problem that arises in various logistics scenarios. Researchers have progressively incorporated additional constraints to better reflect real-world conditions, such as time windows, multiple depots, heterogeneous vehicle fleets, multiple trips, and multi-period planning, among others (Toth and Vigo, 2014). When a VRP includes multiple constraints, it is usually referred to as a rich VRP (Lahyani et al., 2015). The literature has developed a large variety of algorithms to address rich VRPs, ranging from exact methods to advanced meta-heuristics (Elshaer and Awad, 2020).

Time windows are a common constraint in VRPs, restricting vehicles to serve each customer within a specified time period. Time windows can be either soft or hard. The former ones are flexible and may be violated, although this causes a penalty cost. The latter ones must be strictly observed and a violation makes the problem solution infeasible. Surveys on the VRP with time windows (VRPTW) can be found in (Bräysy and Gendreau, 2005a) and (Bräysy and Gendreau, 2005b). These authors reviewed key problem components and various heuristic and meta-heuristic solution methods developed in the literature. Recently, (Pan et al., 2021) addressed a variant of the VRPTW involving time-dependent travel times. They proposed a hybrid heuristic using an adaptive large neighbourhood search (ALNS) and a tabu search, successfully improving results from previous studies. Similarly, (Zhou et al., 2022) developed a MILP formulation and a hybrid heuristic combining VNS with tabu search to solve a variant of the VRPTW, which incorporates two-echelon logistics and pickup and delivery constraints. Other studies involving the VRPTW were recently surveyed by (Zhang et al., 2022). Besides defining and presenting the problem mathematical formulation, the authors discussed the recent progress made by the literature on this problem.

The VRPTW has also been investigated with additional features, such as the presence of multiple depots and periods, besides the opportunity to use multiple trips per period. Concerning the variant with multiple trips, (Brandão and Mercer, 1998) handled a real-world problem related to the food industry. They proposed a tabu search algorithm where the initial solution is generated by combining nearest-neighbour and insertion movements. The tabu search was compared with prior literature and produced solutions that balanced daily tours more effectively. (Despaux and Basterrech, 2016) introduced a hybrid algorithm merging simulated annealing and local search tech-

niques, generating new solutions by moving customers from their routes as well as assigning vehicles to different routes. (Cattaruzza et al., 2016) introduced a new VRPTW variant arising from a real-case scenario concerning a two-level mutualized distribution in cities. This variant requires that the vehicle routes satisfy multiple trips and release dates. The authors proposed a hybrid genetic algorithm incorporating a local search procedure to refine solutions after the crossover operation. (Huang et al., 2024) tackled a problem originating from waste collection operations. The authors proposed an arc flow model and a branch-price-and-cut algorithm based on a set partitioning model. A column generation algorithm and a surrogate relaxation strategy were combined together to generate trips. Their algorithm proved to be efficient in comparison with the previous literature.

Regarding the multiple depot VRPTW variants, (Li et al., 2016) studied the case in which vehicles can start and end their routes in different depots. They proposed a hybrid genetic algorithm with an adaptive local search mechanism. The computational results they obtained showed that changing depot may lead to an interesting 2% reduction in the traveling costs. (Bae and Moon, 2016) examined the multiple depot VRPTW with additional constraints involving the delivery and installation of electronic components. They developed a MILP model and two heuristics. The first heuristic assigns customers to their nearest depot and builds routes using a nearest-neighbor-based algorithm. The second heuristic is a genetic algorithm making use of penalty and fitness functions, as well as crossover operators. Recently, (Su et al., 2024) combined a genetic algorithm with a VNS to address a multiple depot VRPTW variant involving green requirements and customer satisfaction, with the genetic algorithm serving as the upper layer and the VNS as the local search mechanism. Computational results on instances of different sizes demonstrated the good performance of their approach.

With respect to VRP variants with multiple periods, (Wen et al., 2010) solved a dynamic, real-world problem for a large Swedish distributor. They proposed a MILP model and a three-phase rolling horizon heuristic. In the first phase, customers are selected to be visited. In the next phase, the selected customers are arranged in routes by observing a planning problem, and, in the last phase, a tabu search is used to improve the routes. (Mancini, 2016) introduced a MILP model and an ALNS to solve a rich VRP with multiple periods. Their ALNS considered four destroy operators based on randomly selecting customers and reallocating them to different routes. More recently, (Zhang et al., 2024) tackled a real-

world application in the drug distribution system for epidemic situations. To solve large instances, the authors proposed a hybrid tabu search algorithm. It includes intra- and inter-route operators, as well as destroy and repair mechanisms to improve efficiency.

In this work, we address a very general VRPTW variant that combines multiple periods, multiple depots, and multiple trips. To the best of our knowledge, the literature on this rich problem is scarce. As cited above, most of the works considered these constraints separately, but none with all of them together. When solving similar problems, the literature has mostly focused on developing metaheuristic algorithms, especially when dealing with large and real-world instances. Similar to the literature, we also aim to propose an effective metaheuristic. Our aim is indeed to tackle a real-world problem and solve real-world instances using such an approach.

### 3 PROBLEM DESCRIPTION

The optimization problem that we face is defined on a directed graph  $G = (N, A)$ , where  $N$  represents the set of nodes, consisting of a set  $D$  of depots and a set  $C$  of customers. The set  $A$  of arcs is defined as  $\{(i, j) : i, j \in N, i \neq j\}$ , where each arc  $(i, j)$  is associated with a non-negative travel time  $t_{ij}$ . The problem spans over a time horizon divided into a set  $P$  of periods. In each period, each vehicle is constrained by a limit  $T$  on the maximum number of hours it can work. For each customer  $i \in C$  and for each period  $p \in P$ , we are given a demand  $q_{ip}$ , a service time  $s_{ip}$ , and a hard time window  $[e_{ip}, l_{ip}]$ , where  $e_{ip}$  is the earliest possible arrival time and  $l_{ip}$  is the latest possible arrival time. A vehicle is allowed to arrive in  $i$  before  $e_{ip}$ , but in such a case it will have to wait until  $e_{ip}$ . Arriving after  $l_{ip}$  is, instead, not allowed. A heterogeneous vehicle fleet, divided into a set  $V$  of vehicle types, is available at each single depot. Each vehicle type  $v \in V$  consists of a set  $K_v$  of different vehicles. Each vehicle  $k \in K_v$ , for  $v \in V$ , has a loading capacity  $Q_k$ . The sum of the loaded demands cannot exceed the vehicle capacity at any moment during the execution of a route.

The MMMVRPTW is subject to the following constraints: each route can consist of one or more trips, and each trip must start and end at the same depot; the load on each trip must not exceed the vehicle capacity; and, each route may not last longer than the maximum number  $T$  of working hours. When computing the total time of a route, a fixed time  $\Delta$  for loading/unloading operations is imposed between two consecutive trips performed by the same vehi-

cle. Each customer is assigned to a unique trip and must be visited exactly once. Each visit should be performed during the customer time window, and the customer’s demand must be entirely fulfilled during that visit. Furthermore, the problem incorporates incompatibilities between vehicles and customers. In other words, only specific vehicle types can serve certain customers due to geographical location and demand (e.g. on a mountainous territory only a small vehicle can perform the delivery).

The objective of the problem is to find an optimal solution that satisfies all the problem constraints while minimizing the total cost of vehicle usage. Specifically, the overall cost includes two components. The first is a fixed cost,  $A_v$ , which depends on the vehicle type  $v \in V$  and is incurred if a vehicle of this type is used at least once across the set of periods. The second is a variable cost,  $B_v$ , which applies each time a vehicle of type  $v$  is used for a route in a period. The resulting problem is not only NP-hard (as it generalizes the VRP) but also very difficult in practice.

## 4 PROPOSED APPROACH

As anticipated, in this paper, we propose a two-phase approach. In the first phase, we generate a set of trips that respect all the problem constraints. However, since a vehicle can perform multiple trips within a single period (multi-trip) and be reused across different periods (multi-period), the second phase focuses on combining these trips to minimize the cost associated with the usage of the vehicles for all periods.

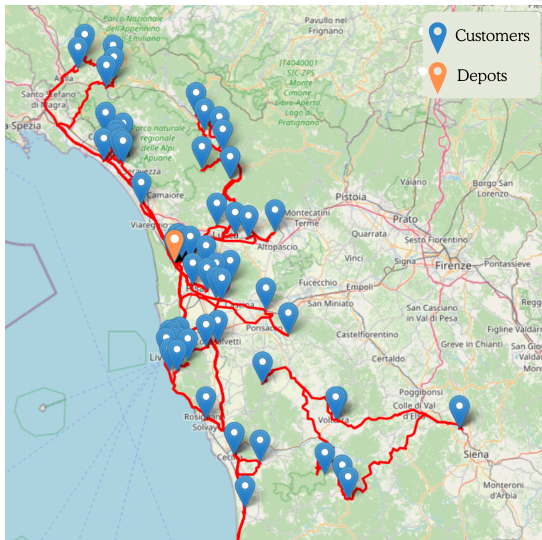


Figure 1: Trips calculated with the ILS metaheuristic in (Kramer et al., 2019).

In the first phase, we apply the metaheuristic proposed by (Kramer et al., 2019). The goal is to generate a set of trips while respecting the problem constraints outlined in Section 3. Specifically, the authors developed an iterated local search (ILS) metaheuristic, which includes a restart mechanism after a specified number of iterations. The ILS consists of a constructive heuristic, a local search, and a perturbation procedure. The local search uses seven neighbourhood structures based on simple movements (e.g., swaps, relocations, and 2-opt exchanges). The perturbation step introduces randomness by performing swaps and relocations. We refer the reader to the original publication for further details. Figure 1 illustrates an example of the trips obtained with the ILS, in which the blue points are customers, the orange points are depots, and trips are depicted in red.

The trips obtained in the first phase are then used and combined in the second phase to create the final routes. In order to do that, we propose a VNS-based heuristic, where the initial solution is generated using a constructive heuristic. This solution is subsequently refined through a shaking procedure and a local search. In the proposed algorithm, we define a stopping criterion based on two conditions: the number of consecutive iterations without improvement and a maximum time limit. Further details are provided in the following section. Additionally, we adapt the MILP model and the BSTG heuristic proposed by (Cavecchia et al., 2024) in order to accommodate multiple periods and the more general cost function. All these algorithms are applied to solve a number of real-world instances, and computational results demonstrate their competitiveness.

### 4.1 Adapted Methods

In (Cavecchia et al., 2024), a single period VRPTW variant with multiple depots and multiple trips was solved by means of a MILP model, two greedy heuristics and an ILS. The authors employed the forward-time slack procedure proposed by (Savelsbergh, 1992) in order to define the earliest and latest starting times of each trip. The idea behind this procedure is to allow a vehicle to depart as early as possible without generating infeasible trips. It calculates the forward time slack, which indicates how far the departure time of each customer can be shifted forward.

As in (Cavecchia et al., 2024), we first generate the trips  $R_p$  for each period  $p \in P$  using the metaheuristic proposed by (Kramer et al., 2019). Next, we apply the forward-time slack procedure to determine the earliest starting time  $\bar{e}_r$  and the latest starting time

$\bar{l}_r$  for each trip  $r \in R_p$ . These parameters are then used by the MILP model described below. The model also receives in input the duration  $T_r$  of each generated trip  $r \in R_p$ .

As in (Cavecchia et al., 2024), the model includes four sets of binary decision variables and three sets of continuous decision variables. However, now the decision variables have an additional index  $p$  representing the period they refer to. To correctly assess the new cost function, we introduce an additional binary decision variable  $w_{kv}^p$ .

- $x_{rkv}^p$ : binary variable that takes the value 1 if trip  $r \in R_p$  is assigned to vehicle  $k \in K_v$  of type  $v \in V$  on period  $p \in P$ ; otherwise it takes the value 0;
- $y_{kv}$ : binary variable that takes the value 1 if vehicle  $k \in K_v$  of type  $v \in V$  is used; otherwise it takes the value 0;
- $w_{kv}^p$ : binary variable that takes the value 1 if vehicle  $k \in K_v$  of type  $v \in V$  is used on period  $p \in P$ ; otherwise it takes the value 0;
- $z_{rskv}^p$ : binary variable that takes the value 1 if trip  $r \in R_p$  precedes trip  $s \in R_p$ , both assigned to vehicle  $k \in K_v$  of type  $v \in V$  on period  $p \in P$ ; otherwise it takes the value 0;
- $\alpha_{kv}^p$ : continuous variable that represents the starting time of vehicle  $k \in K_v$  of type  $v \in V$  on period  $p \in P$ ;
- $\beta_{kv}^p$ : continuous variable that represents the ending time of vehicle  $k \in K_v$  of type  $v \in V$  on period  $p \in P$ ;
- $t_{rkv}^p$ : continuous variable that represents the starting time of trip  $r \in R_p$  assigned to vehicle  $k \in K_v$  of type  $v \in V$  on period  $p \in P$ .

The MILP model has the objective function (1) that accounts for minimizing the total fixed cost associated with the different vehicles used across all periods ( $A_v$ ), as well as the daily cost encountered by considering the vehicles used in every single period ( $B_v$ ). The model constraints are given by equations (2)-(21) and are direct extensions of those in (Cavecchia et al., 2024). The model is solved independently for each depot  $d \in D$ , as the trips in  $R_p$  have already been associated with a depot  $d$  and are independent of one another. In other words, each depot corresponds to an instance of the problem.

Constraints (2)-(4) ensure that if a vehicle  $k$  is used during period  $p$ , then it must be considered in the solution cost through variables  $y$  and  $w$ . Constraints (5) impose that each trip  $r$  on period  $p$  must be assigned to exactly one vehicle  $k$ . Precedence constraints are ensured in (6) and (7) for any two trips assigned to the same vehicle. Constraints (8) ensure

that trips assigned to the same vehicle have different starting times, i.e., one must start after the other finishes, considering the total duration  $T_r$  and the fixed loading time  $\Delta$  between the consecutive trips  $r$  and  $s$ . Constraints (9) require each trip to respect its earliest  $\bar{e}_r$  and latest  $\bar{l}_r$  starting times. Constraints (10), (11), and (12) guarantee that each vehicle performs a set of trips that respects the working hours limit  $T$ . In (13), valid inequalities are applied to ensure that the working hours of each vehicle are respected. Constraints (14) stipulate that vehicles can only perform trips according to the customer's geographical location and demand, thus taking care of the vehicle-customer incompatibilities. Formally, set  $I$  contains those generated trips  $r \in R_p$  that a vehicle of type  $v$  cannot perform due to geographical restrictions. Finally, constraints (15)-(21) define the domain of the variables.

In addition to the MILP model, we adapt the bounded starting time greedy (i.e., BSTG) algorithm developed by (Cavecchia et al., 2024). This heuristic considers a variable  $\sigma$  to track the vehicles from each depot and type used in each period. First, the trips  $r \in R_p$  are sorted in a non-decreasing order of the earliest starting time  $\bar{e}_r$ . In case of ties, trips with longer duration are given priority. Trips are then assigned to vehicles, considering their working hours and type. Algorithm 1 in (Cavecchia et al., 2024) provides the pseudocode for the BSTG algorithm. Differently from the original algorithm, we give preference to vehicles in  $\sigma$  that have already been used in previous periods. If no such vehicle is available, a new one is created and added to  $\sigma$ . Priority is given to vehicle types with the lowest associated costs. We take advantage of previously used vehicles and try to first assign trips to used vehicles of minimum cost, following their costs.

## 4.2 VNS-Based Heuristic

The VNS is a metaheuristic that has been applied to solve many hard combinatorial optimization problems. We can find relevant VNS applications in problems like vehicle routing, cutting, packing, facility location, scheduling, lot sizing, and many others (Hansen and Mladenović, 2001). Recently, (Lan et al., 2021) surveyed healthcare problems that were solved by VNS-based heuristics. Besides classifying the works in the literature into five classes, the authors detailed the main VNS phases, e.g., shaking and local search, developed by the literature to obtain high-quality solutions. They also commented on the efficiency of classical operators based on swap, exchange, insertion, and move, among others, to define neighbourhood structures.

$$\begin{aligned}
\min \quad & Z = \sum_{v \in V} \sum_{k \in K_v} A_v y_{kv} + \sum_{p \in P} \sum_{v \in V} \sum_{k \in K_v} B_v w_{kv}^p & (1) \\
\text{s.t.} \quad & x_{rkv}^p \leq w_{kv}^p, & \forall p \in P, \forall r \in R_p, \forall v \in V, \forall k \in K_v & (2) \\
& y_{kv} \geq w_{kv}^p, & \forall p \in P, \forall v \in V, \forall k \in K_v & (3) \\
& x_{rkv}^p \leq y_{kv}, & \forall p \in P, \forall r \in R_p, \forall v \in V, \forall k \in K_v & (4) \\
& \sum_{v \in V} \sum_{k \in K_v} x_{rkv}^p = 1, & \forall p \in P, \forall r \in R_p & (5) \\
& z_{rskv}^p + z_{srkv}^p \geq x_{rkv}^p + x_{skv}^p - 1, & \forall p \in P, \forall v \in V, \forall k \in K_v, \forall r, s \in R_p : r \neq s & (6) \\
& z_{rskv}^p + z_{srkv}^p \leq 1, & \forall p \in P, \forall v \in V, \forall k \in K_v, \forall r, s \in R_p : r \neq s & (7) \\
& t_{rkv}^p + (T_r + \Delta) \leq t_{skv}^p + M(1 - z_{rskv}^p), & \forall p \in P, \forall v \in V, \forall k \in K_v, \forall r, s \in R_p : r \neq s & (8) \\
& \bar{e}_r \leq t_{rkv}^p \leq \bar{l}_r, & \forall p \in P, \forall r \in R_p, \forall v \in V, \forall k \in K_v & (9) \\
& \alpha_{kv}^p \leq t_{rkv}^p + M(1 - x_{rkv}^p), & \forall p \in P, \forall r \in R_p, \forall v \in V, \forall k \in K_v & (10) \\
& \beta_{kv}^p \geq t_{rkv}^p + (T_r + \Delta) - M(1 - x_{rkv}^p), & \forall p \in P, \forall r \in R_p, \forall v \in V, \forall k \in K_v & (11) \\
& \beta_{kv}^p - \alpha_{kv}^p \leq (T + \Delta)y_{kv}, & \forall p \in P, \forall v \in V, \forall k \in K_v & (12) \\
& \sum_{r \in R_p} (T_r + \Delta)x_{rkv}^p \leq (T + \Delta)y_{kv}, & \forall p \in P, \forall r \in R_p, \forall v \in V, \forall k \in K_v & (13) \\
& x_{rkv}^p = 0, & \forall p \in P, \forall (r, v) \in I, \forall k \in K_v : q_r > Q_k & (14) \\
& x_{rkv}^p \in \{0, 1\}, & \forall p \in P, \forall r \in R_p, \forall v \in V, \forall k \in K_v & (15) \\
& y_{kv} \in \{0, 1\}, & \forall v \in V, \forall k \in K_v & (16) \\
& w_{kv}^p \in \{0, 1\}, & \forall p \in P, \forall v \in V, \forall k \in K_v & (17) \\
& z_{rskv}^p \in \{0, 1\}, & \forall p \in P, \forall v \in V, \forall k \in K_v, \forall r, s \in R_p : r \neq s & (18) \\
& t_{rkv}^p \geq 0, & \forall p \in P, \forall r \in R_p, \forall v \in V, \forall k \in K_v & (19) \\
& \alpha_{kv}^p \geq 0, & \forall p \in P, \forall v \in V, \forall k \in K_v & (20) \\
& \beta_{kv}^p \geq 0, & \forall p \in P, \forall v \in V, \forall k \in K_v & (21)
\end{aligned}$$

According to (Hansen et al., 2019), the VNS starts with an initial (random) solution and iterates over a given set of neighbourhood structures in order to obtain a solution that is globally optimal concerning all structures. At each iteration, a new neighbour solution is generated in the shaking phase. Depending on the VNS version, this neighbour solution is improved through a local search phase. The resulting solution is then evaluated and accepted if it is better than the previous one. In such a case, the search restarts from the first neighbourhood structure. Otherwise, the search continues over the next neighbourhood.

In this work, we propose a heuristic based on the VNS framework, as outlined in Algorithm 1. The algorithm receives in input the routes  $R_p$  generated in the first phase and the set of periods  $P$ . As output, it returns the best solution found,  $x$ , which consists of the routes performed by each vehicle over  $P$ . In line 1, we apply the adapted BSTG algorithm to obtain an initial solution. The BSTG algorithm assigns vehicles

to trips following the order of the periods given in  $P$ . This means that a different order of the periods may imply a different (and hopefully better) solution. In the first while loop (line 2), we use as stopping criteria the maximum number of iterations without an improvement or a maximum time limit. This first loop starts by imposing the search over the first neighbourhood. Next, we enter a second inner loop that iterates through all the neighbourhood structures (line 4). We develop  $K_{\max} = 3$  structures that all work on the set of periods, namely: (i) swap two periods in the order, (ii) insert one period before another one in the order, and (iii) swap three periods. The idea is to generate different orders for the periods, thus leading to good diversification of the search.

The shake phase in line 5 uses the neighbourhood structures to generate a new order  $P'$  to the periods in  $P$ . Next, we modified the adapted BSTG algorithm in order to receive the current solution  $x$  and  $P'$ . This modification consists of using the trips as they have

Algorithm 1: VNS-based heuristic.

---

```

Data:  $R_p, P$ 
Result:  $x$ 
1  $x \leftarrow \text{BSTG}(R_p, P)$ ;
2 while not reaching one of the stopping
   criteria do
3    $k \leftarrow 1$ ;
4   while  $k \leq K_{\max}$  do
5      $P' \leftarrow \text{SHAKE}(x, P, k)$ ;
6      $x' \leftarrow \text{BSTG}(x, P')$ ;
7      $x'' \leftarrow \text{LOCALSEARCH}(x')$ ;
8     if  $\text{COST}(x'') < \text{COST}(x)$  then
9        $x \leftarrow x''$ ;
10       $k \leftarrow 1$ ;
11     else
12       $k \leftarrow k + 1$ ;
13     end
14   end
15 end

```

---

been assigned in the solution. Then, these trips are reassigned in accordance with the order of the periods in  $P'$ . After that, we try to improve the neighbour solution  $x'$  with the local search phase, resulting in a solution  $x''$ . In line 8, we compare the costs of  $x'$  and  $x''$ , updating  $x$  in case  $x''$  is better and making the search restart from the first neighbourhood. In case no improvement is obtained, we continue the search with the next neighbourhood (line 12).

Our implementation of the local search consists of three steps. In the first step, we perform all possible swaps of two routes, stopping when a movement can improve the solution (i.e., we use a truncated local search, with the aim of keeping a low computational effort). This improved solution is then used in the second step, where all possible insertions between two routes are performed. Similarly, we stop in the first insertion movement able to improve the input solution. Finally, in the third step, we select one vehicle from a period and a type in order to reassign all its trips to other vehicles in that period that are able to accommodate them. The vehicle we attempt to empty is taken from the period and vehicle type that is most used among all ones. In fact, we try to empty all vehicles of that type, until reaching one in which this is possible. If no vehicle can be emptied, we return the solution given as output by the previous local search operator.

## 5 COMPUTATIONAL RESULTS

The two-phase approach was implemented in the following way: for the first phase we invoked the algorithm by (Kramer et al., 2019), which was coded in C++; we then coded all the algorithms used in the second phase in Python. The computer used to run the experiments is an Intel(R) Xeon(R) E3-1245 3.50 GHz CPU, 32 GB of RAM, running under Ubuntu 22.04 LTS.

The MILP model (1)-(21) was solved with Gurobi Optimizer version 11. We set the maximum number of consecutive iterations without improvement of the VNS-based heuristic equal to the number of neighbourhood structures (i.e., equal to three). In compliance with the real-world case study, we set the fixed loading/unloading time to  $\Delta = 30$  minutes, and the limit on the maximum number of working hours to  $T = 480$  minutes. Each algorithm of the second phase was executed five times. Each execution was limited to 3600 seconds. In the objective function, the values of  $A_v = \{160, 140, 120, 100, 80, 60\}$  are assigned to vehicle types  $v \in V$ , ordered in descending capacity. Additionally,  $B_v$  is set to 1 uniformly across all vehicle types, regardless of their specific characteristics. For each algorithm, the tables below report the average results regarding objective function value and computing time considering the five executions per instance. The term TL stands for time limit and is used when all executions reach 3600 seconds.

The two-phase approach was applied to solve realistic instances from an Italian service company. They are based on three regions in Italy: Emilia Romagna, Sardinia, and Tuscany. We have been provided with three instances from Emilia Romagna, two from Sardinia, and one from Tuscany. Table 1 contains some details about these instances, including the name, the number of depots, the number of vehicle types, the number of customers, the number of trips obtained with the algorithm of (Kramer et al., 2019) in the first phase of our decomposition approach, the average number of customers that a vehicle could service in a single trip ( $\#CT$ ), and the average number of periods in which a customer requires products ( $\#PC$ ). Moreover, the maximum number of periods is six.

Table 2 presents the results obtained with all the algorithms of the second phase, applied for each depot. We present the objective function  $Z$ , computed according to (1), and the computing time  $t(s)$  in seconds, for each algorithm. For the MILP model, we also include the lower bound  $Z_{LB}$  and the percentage gap,  $GAP(\%)$ , whose values are both directly returned by the solver. For each instance, the best objective values are highlighted in bold. In general, we

Table 1: Main characteristics of the instances.

Instance	$ D $	$ V $	$ C $	$ R_p $	$\#CT$	$\#PC$
West Emilia Romagna	1	1	206	119	2.6	1.5
Central Emilia Romagna	2	1	210	194	3.0	3.2
East Emilia Romagna	1	1	201	177	3.0	3.1
North Sardinia	3	2	152	48	8.4	1.5
South Sardinia	1	2	154	58	7.3	1.6
Tuscany	2	6	155	101	5.1	2.3

observe that the VNS-based heuristic is able to produce better objective function values, although it may spend large computing times. The MILP model also presents competitive results. However, it reaches the time limit of one hour for those instances having the largest number of trips. The adapted BSTG algorithm is fast but does not perform so well compared to the other approaches.

Observing the results in Table 2, the VNS-based heuristic has the best objective value for the Central Emilia Romagna (depot 1) instance. Moreover, in comparison with the adapted BSTG algorithm, it is still superior for the instances West Emilia Romagna and Tuscany (depot 1). Similarly, it performs better than the MILP model also for the East Emilia Romagna instance. The best improvement obtained with the VNS-based heuristic over the adapted BSTG algorithm is for the Tuscany (depot 1) instance, reaching a percentage decrease of 10.66%. With respect to the MILP model, it reaches an improvement of 4.02% for the East Emilia Romagna instance. Concerning the comparison between the MILP model and the adapted BSTG algorithm, the former is better for the instances West Emilia Romagna and Tuscany (depot 1), while the latter is superior for the instances Central Emilia Romagna (depot 1) and East Emilia Romagna.

In terms of computing time, the results in Table 2 indicate the best performance of the adapted BSTG algorithm. It presents an average computing time of 0.14 seconds. On the other hand, the VNS-based heuristic has an average computing time of 1281.57 seconds, reaching the imposed time limit for the instances Central Emilia Romagna (depot 1) and East Emilia Romagna. The large computing times reported by the VNS-based heuristic are justified by its local search phase. Similarly, the MILP model reaches the

time limit for the same instances, with an overall average computing time of 725.87 seconds. For these instances, the model GAPs are 7.63% and 15.58%, respectively. From a scalability point of view, the model struggles as the size of the instances grows, reaching the time limit and increasing the GAP. In this way, the BSTG algorithm and the VNS-based heuristic are effective alternatives to the model.

In Table 3 we present the number of vehicles used over the set of periods, considering the final solution obtained with the algorithms for the second phase. The results are shown for each depot and vehicle type that uses at least one vehicle. All the algorithms return the same amount of vehicles, except for the instances East Emilia Romagna and Tuscany (depot 1, vehicle type 2). In these specific cases, the MILP model requires one more vehicle and the adapted BSTG algorithm requires one more vehicle with respect to the other algorithms, respectively. Despite the same number of vehicles required in the final solution by the algorithms, the objective function values may be different as previously shown in Table 2, because it refers to the fixed cost of the vehicles plus the number of times that each vehicle is used over the periods.

Figure 2 presents the graphical output generated by the VNS-based heuristic for the Tuscany (depot 1) instance. The chart displays the assignment of trips to vehicles over the considered set of periods. Each colored bar represents a trip, with bars of the same color corresponding to trips assigned to the same vehicle. The horizontal axis indicates the time, while the vertical axis shows the vehicles and the created routes, which consist of multiple trips.



Table 2: Computational results on the ten real-world instances.

Instance	Depot	BSTG		VNS		MILP Model			
		Z	t(s)	Z	t(s)	Z	Z <sub>LB</sub>	GAP(%)	t(s)
West Emilia Romagna	1	3949	0.28	<b>3947</b>	3393.38	<b>3947</b>	3947	0.00%	7.94
Central Emilia Romagna	1	2450	0.25	<b>2449</b>	TL	2451	2264	7.63%	TL
	2	<b>980</b>	0.05	<b>980</b>	113.57	<b>980</b>	980	0.00%	34.56
East Emilia Romagna	1	<b>3919</b>	0.48	<b>3919</b>	TL	4083	3447	15.58%	TL
North Sardinia	1	<b>742</b>	0.03	<b>742</b>	8.16	<b>742</b>	742	0.00%	0.15
	2	<b>430</b>	0.01	<b>430</b>	0.22	<b>430</b>	430	0.00%	0.02
	3	<b>433</b>	0.01	<b>433</b>	0.47	<b>433</b>	433	0.00%	0.03
South Sardinia	1	<b>1632</b>	0.08	<b>1632</b>	172.60	<b>1632</b>	1632	0.00%	1.29
Tuscany	1	1342	0.20	<b>1199</b>	1926.93	<b>1199</b>	1199	0.00%	14.65
	2	<b>208</b>	0.01	<b>208</b>	0.39	<b>208</b>	208	0.00%	0.05

Table 3: Number of vehicles used in the solutions found.

Instance	Depot	Vehicle Type	BSTG	VNS	MILP Model
West Emilia Romagna	1	1	24	24	24
Central Emilia Romagna	1	1	15	15	15
	2	1	6	6	6
East Emilia Romagna	1	1	24	24	25
North Sardinia	1	1	1	1	1
		2	4	4	4
	2	2	3	3	3
South Sardinia	1	1	2	2	2
		2	9	9	9
		1	1	1	1
Tuscany		2	2	1	1
		3	2	2	2
	1	4	1	1	1
		5	1	1	1
		6	7	7	7
	2	2	1	1	1
		6	1	1	1

## 6 CONCLUDING REMARKS

The rich VRP handled in this work originates from a real-world case study. It contains many different constraints and thus is solved by a two-phase approach. The first phase is concerned with obtaining a set of trips that respect the problem’s constraints. The second phase aggregates these trips into routes in order

to minimize the costs associated with the usage of the vehicles and the number of times each vehicle is used over a set of periods. The trips are calculated with a metaheuristic from the literature and the routes may be defined by three different algorithms: an adapted BSTG algorithm, a VNS-based one, and a MILP model.

Considering experiments on realistic instances,

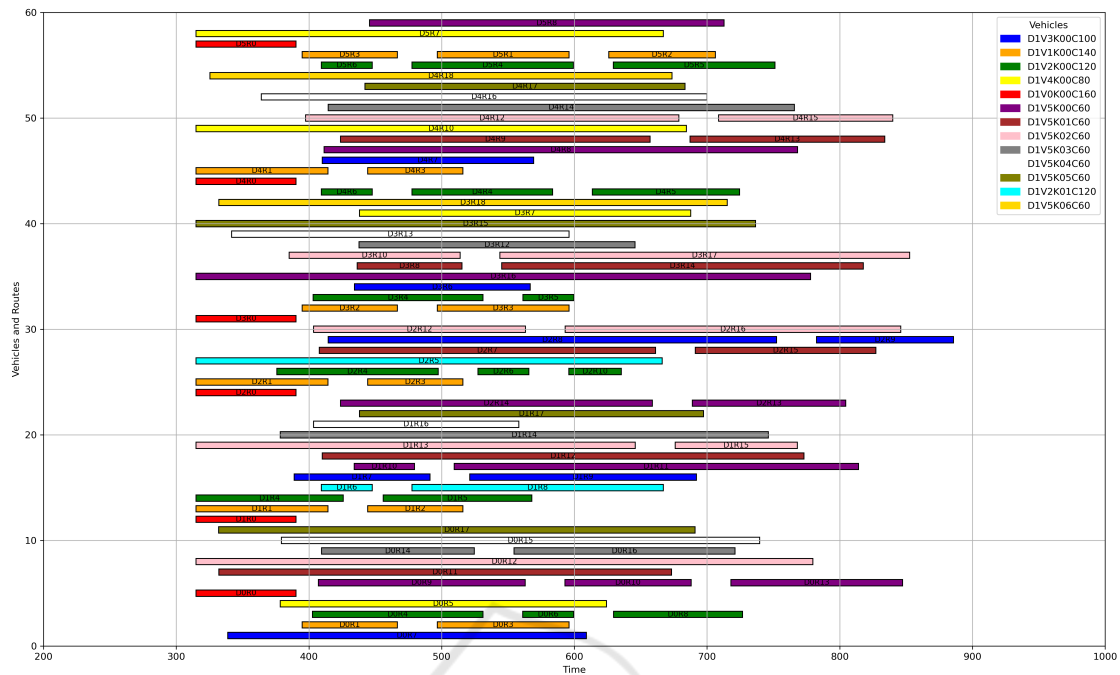


Figure 2: Final solution for Tuscany instance generated by the VNS-based heuristic.

the proposed algorithms are effective in returning solutions close to each other. Differences are observed with the VNS-based heuristic, which can improve some of the solutions computed by the others, e.g., reaching improvements superior to 10% compared to the adapted BSTG algorithm. On the other hand, it has the largest computing times. The MILP model works well for small-size instances, but it is not able to achieve optimal solutions for the larger instances within 3600 seconds.

After all, we note there is room for improvement. A sensitivity analysis (e.g., varying the vehicle capacity and the customers’ time windows) could help the company in deciding about the fleet of vehicles. On the other hand, the VNS-based heuristic is time-consuming due to its local search phase. One direction could be to explore other VNS variants that do not have the local search phase. Another interesting research avenue lies in the proposal of different heuristics and metaheuristics (e.g., column generation-based heuristics, genetic algorithms, and tabu search) to compare with the methods proposed in this paper. One could extend the proposed algorithms to handle the entire problem instead of first calculating the trips and then aggregating them, as well as other rich VRP variants. We are also interested in studying robust/stochastic problems, where the total duration of the trips is not deterministically determined but may assume uncertain values taken from

intervals (in case of robust optimization) or scenarios (in case of stochastic programming).

## ACKNOWLEDGEMENTS

This paper was partially financed by the National Council for Scientific and Technological Development (CNPq) [grant numbers 405369/2021-2, 408722/2023-1, and 315555/2023-8], the State of Goiás Research Foundation (FAPEG), the National Recovery and Resilience Plan (NRRP), Mission 04 Component 2 Investment 1.5–NextGenerationEU, Call for tender n. 3277 dated 30/12/2021, Award Number: 0001052 dated 23/06/2022, and Coopservice Soc.coop.p.A. Their support is gratefully acknowledged.

## REFERENCES

Bae, H. and Moon, I. (2016). Multi-depot vehicle routing problem with time windows considering delivery and installation vehicles. *Applied Mathematical Modelling*, 40(13):6536–6549.

Brandão, J. C. S. and Mercer, A. (1998). The multi-trip vehicle routing problem. *Journal of the Operational Research Society*, 49(8):799–805.

Bräysy, O. and Gendreau, M. (2005a). Vehicle routing problem with time windows, part I: Route construction

- and local search algorithms. *Transportation Science*, 39(1):104–118.
- Bräysy, O. and Gendreau, M. (2005b). Vehicle routing problem with time windows, part II: Metaheuristics. *Transportation Science*, 39(1):119–139.
- Cattaruzza, D., Absi, N., and Feillet, D. (2016). The Multi-Trip Vehicle Routing Problem with Time Windows and Release Dates. *Transportation Science*, 50(2):676–693. Publisher: INFORMS.
- Cavecchia, M., Alves de Queiroz, T., Iori, M., Lancellotti, R., and Zucchi, G. (2024). An optimization-based decision support system for multi-trip vehicle routing problems. *SN Computer Science*, 5(2):225.
- Despoux, F. and Basterrech, S. (2016). Multi-trip vehicle routing problem with time windows and heterogeneous fleet. *International Journal of Computer Information Systems and Industrial Management Applications*, 8:9.
- Elshaer, R. and Awad, H. (2020). A taxonomic review of metaheuristic algorithms for solving the vehicle routing problem and its variants. *Computers & Industrial Engineering*, 140:106242.
- Gosain, S. (2023). The vital role of logistics in business to business (B2B) success. <https://insights.worldref.co/logistics-in-business-to-business-b2b/> [Accessed: 2024-08-30].
- Hansen, P. and Mladenović, N. (2001). Variable neighborhood search: Principles and applications. *European Journal of Operational Research*, 130(3):449–467.
- Hansen, P., Mladenović, N., Brimberg, J., and Pérez, J. A. M. (2019). Variable Neighborhood Search. In Gendreau, M. and Potvin, J.-Y., editors, *Handbook of Metaheuristics*, pages 57–97. Springer International Publishing, Cham.
- Huang, N., Qin, H., Xu, G., and Wan, F. (2024). An enhanced exact algorithm for the multi-trip vehicle routing problem with time windows and capacitated unloading station. *Computers & Operations Research*, 168:106688.
- Kramer, R., Cordeau, J.-F., and Iori, M. (2019). Rich vehicle routing with auxiliary depots and anticipated deliveries: An application to pharmaceutical distribution. *Transportation Research Part E: Logistics and Transportation Review*, 129:162–174.
- Lahyani, R., Khemakhem, M., and Semet, F. (2015). Rich vehicle routing problems: From a taxonomy to a definition. *European Journal of Operational Research*, 241(1):1–14.
- Lan, S., Fan, W., Yang, S., Pardalos, P. M., and Mladenovic, N. (2021). A survey on the applications of variable neighborhood search algorithm in healthcare management. *Annals of Mathematics and Artificial Intelligence*, 89(8):741–775.
- Li, J., Li, Y., and Pardalos, P. M. (2016). Multi-depot vehicle routing problem with time windows under shared depot resources. *Journal of Combinatorial Optimization*, 31(2):515–532.
- Mancini, S. (2016). A real-life Multi Depot Multi Period Vehicle Routing Problem with a Heterogeneous Fleet: Formulation and Adaptive Large Neighborhood Search based Matheuristic. *Transportation Research Part C: Emerging Technologies*, 70:100–112.
- Pan, B., Zhang, Z., and Lim, A. (2021). A hybrid algorithm for time-dependent vehicle routing problem with time windows. *Computers & Operations Research*, 128:105193.
- Savelsbergh, M. W. P. (1992). The Vehicle Routing Problem with Time Windows: Minimizing Route Duration. *ORSA Journal on Computing*, 4(2):146–154.
- Su, Y., Zhang, S., and Zhang, C. (2024). A lightweight genetic algorithm with variable neighborhood search for multi-depot vehicle routing problem with time windows. *Applied Soft Computing*, 161:111789.
- Toth, P. and Vigo, D. (2014). *Vehicle routing: Problems, methods, and applications*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA.
- Wen, M., Cordeau, J.-F., Laporte, G., and Larsen, J. (2010). The dynamic multi-period vehicle routing problem. *Computers & Operations Research*, 37(9):1615–1623.
- Zhang, H., Ge, H., Yang, J., and Tong, Y. (2022). Review of vehicle routing problems: Models, classification and solving algorithms. *Archives of Computational Methods in Engineering*, 29(1):195–221.
- Zhang, J., Li, Y., and Lu, Z. (2024). Multi-period vehicle routing problem with time windows for drug distribution in the epidemic situation. *Transportation Research Part C: Emerging Technologies*, 160:104484.
- Zhou, H., Qin, H., Zhang, Z., and Li, J. (2022). Two-echelon vehicle routing problem with time windows and simultaneous pickup and delivery. *Soft Computing*, 26(7):3345–3360.