UNIVERSITÀ DI SIENA 1240

USiena

A multi-depot dial-a-ride problem with heterogeneous vehicles and compatibility constraints in healthcare

This is the peer reviewed version of the following article:

Original:

Detti, P., Papalini, F., Zabalo Manrique De Lara, G. (2017). A multi-depot dial-a-ride problem with heterogeneous vehicles and compatibility constraints in healthcare. OMEGA, 70, 1-14 [10.1016/j.omega.2016.08.008].

Availability:

This version is availablehttp://hdl.handle.net/11365/998596 since 2017-05-15T19:04:23Z

Published:

DOI: http://doi.org/10.1016/j.omega.2016.08.008

Terms of use:

Open Access

The terms and conditions for the reuse of this version of the manuscript are specified in the publishing policy. Works made available under a Creative Commons license can be used according to the terms and conditions of said license.
For all terms of use and more information see the publisher's website.

(Article begins on next page)

19 November 2026

A multi-depot dial-a-ride problem with heterogeneous vehicles and compatibility constraints in healthcare

Paolo Detti [*] Francesco Papalini [†] Garazi Zabalo Manrique de Lara [‡]

Abstract

In this paper, a multi-depot dial-a-ride problem arising from a real-world healthcare application is addressed, concerning the non-emergency transportation of patients. The problem presents several constraints and features, such as heterogeneous vehicles, vehicle-patient compatibility constraints, quality of service requirements, patients' preferences, tariffs depending on the vehicles' waiting. Variable Neighborhood Search and Tabu Search algorithms are proposed able to tackle all the characteristics of the problem. A Mixed Integer Linear Programming formulation is also presented. Computational results on large real-world and random instances based on real data show the effectiveness of the proposed approaches.

keywords: Dial-a-ride problem, healthcare, Variable Neighborhood Search, Tabu Search.

[*]Dipartimento di Ingegneria dell'Informazione e Scienze Matematiche, University of Siena, Via Roma, 56, 53100 Siena, Italy, e-mail detti@dii.unisi.it

[†]Azienda Sanitaria di Firenze, Piazza Santa Maria Nuova, 1, Firenze, Italy, e-mail francescopapalini@gmail.com

[‡]Dipartimento di Ingegneria dell'Informazione e Scienze Matematiche, University of Siena, Via Roma, 56, 53100 Siena, Italy, e-mail garazizml@gmail.com

1 Introduction

In this paper, a transportation problem arising from a real-world healthcare application is addressed. Namely, the non-emergency transportation of patients in Tuscany [2, 14], an Italian region.

Non-emergency transportation is an essential service for many patients. It mainly includes the transportation of patients attending hospital clinics for medical treatments, exams or therapies (e.g., see [2, 9, 23, 33]), transferrals and discharges. In this context, a *transportation service* consists in transporting a patient from a pickup location (e.g., patient's home, hospital, etc.) to a delivery location, and, eventually, vice versa. In the addressed problem, the transportation services are coordinated by a Local Health Care Agency (ASL) and performed by non-profit organizations, by means of a heterogeneous fleet of vehicles (e.g., ambulance, bus, car, equipped vehicle), located in geographically distributed depots. Constraints concern arrival and departure times, patient-vehicle compatibility, patients' preferences and quality of service issues. In fact, (i) patients have to attend on time their healthcare services, (ii) a set of quality of service requirements on the length of the travels and the waiting times of the patients must be fulfilled, (iii) patients are heterogenous and must be only transported by vehicles that are compatible with their conditions (e.g., patients on a wheelchair or a stretcher have to use equipped vehicles or ambulances) and, eventually, belonging to given non-profit organizations (*patients' preferences*). The objective is to minimize the total transportation cost while fulfilling the quality of service requirements.

From a modeling point of view, the problem can be formulated as a Dial-a-Ride problem (DARP) [17, 23] with side constraints and features.

Although several studies about DARPs have been proposed in the literature, to the authorsnowl-

edge only the works of Braekers *et al.* [10] and Carnes *et al.* [12] address problems involving all the above features. Furthermore, in the DARP addressed in this paper, new aspects have been considered. Namely, preference constraints of the patients (limiting the set of vehicles able to transport a patient); a quality of service index, coming from the real-life application, based on the waiting of the patients at the healthcare facilities; a complex cost function depending on several factors (i.e., vehicle's type, number of transported persons by a vehicle, duration and length of the route, a stepwise function of the waiting time of the vehicles), as stated by an agreement between the (regional) government and the non-profit organizations [1]. In fact, in [10] quality of service is only limited to patients' ride times and the route costs only depend on the traveled distance. In Carnes *et al.* [12] the quality of service is only limited to patients' ride times, too, but the authors consider a more complex cost function. However, the instances handled in [12] (coming from an air-ambulance services context) contain no more than 30 requests per day, while in our case real-life problems with more than 200 requests must be tackled.

In this paper, metaheuristic algorithms, based on the Tabu Search (TS) and the Variable Neighborhood Search (VNS) techniques are presented. A Mixed Integer Linear Programming (MILP) formulation is also proposed and used to solve the problem. (A preliminary version of this work has been presented in [18].) Although TS and VNS techniques have been successfully applied to solve DARPs and more in general to the Pickup and Delivery Problems with Time Windows (PDPTW), to the authorsnowledge no study from the literature reports on the application of TS and VNS to multi-depot DARPs with heterogeneous users and vehicles.

The main contributions of this paper are threefold and listed below.

(a) A new model for a variant of DARP with heterogeneous vehicles and users and multiple depots is

introduced, coming from a real-life application. From a modeling point of view, the addressed problem has the following new features: patients' preferences (limiting the set of vehicles able to transport a patient); a quality of service index, based on the waiting of the patients at the healthcare facilities; route's cost depending on several factors (vehicle's type, number of transported persons, duration and length of the route and a stepwise function of the waiting time of the vehicle).

(b) TS and VNS algorithms able to solve the addressed DARP with multi-depots and heterogeneous users and vehicles are presented. The algorithms include adapted versions of the basic structures and neighborhoods already introduced by Cordeau and Laporte [16] and Parragh *et al.* [26], respectively, but have the following differences with the algorithms presented in the literature: new heuristics for finding the initial solution (taking into account the stringent constraints on vehicle compatibility and patients' preferences of the benchmark instances generated from the real data), a more complex evaluation function taking into account a quality index related to patients' waiting times at the healthcare facilities, and the adaptation of all the framework to the already mentioned characteristics of the studied application. A preliminary test phase (reported in Section 6.1), in which the neighborhood quality is analyzed, shows that the VNS approach outperforms the TS one.

(c) Improved versions of the VNS algorithms are proposed, in which new elements are introduced and tested. Namely, (1) two new neighborhoods (called "eliminate" and "repairing move"), (2) a new procedure, called *Adjusting*, designed to facilitate the production of feasible solutions from quasi-feasible solutions (i.e., violating only a constraint on vehicles' capacity, time windows or ride times) during the search process, (3) a procedure for minimizing the costs due to the vehicles' waiting.

Computational experiments on a set of benchmark instances based on real-life data and on a real-life instance are presented. A comparison with solutions obtained by a state-of-the-art MILP solver

shows the effectiveness of the proposed approaches.

The computational results highlight the effectiveness of the proposed algorithms and also show the benefits in terms of costs and quality of service of coordination and central dispatching versus the current system, in which the non-profit organizations route their own vehicles, to transport the patients assigned to them (by the preference constraints or by the Health Care Agency). In fact, since the preference constraints can not be omitted by any assignment scheme, the proposed algorithms are able to produce an overall routing of the vehicles taking into account all patients' preferences.

The paper is organized as follows. In Section 2, a literature review is presented. In Sections 3 and 3.2, a detailed description of the problem and a Mixed Integer Linear Programming formulation are respectively presented. Sections 4 and 5 are devoted to the presentation of the tabu search and the variable neighborhood search algorithms. A description of the real-life case and computational results on real-life and random instances are reported in Section 6.

2 Literature review

The non-emergency transportation problem addressed in this paper can be formulated as a DARP [17] with side constraints and additional features. DARP is a generalization of PDPTW [8, 21, 30], where people are transported instead of goods. Usually, in DARP, issues on the quality of the provided service and timing must be carefully taken into account (through additional constraints or by extra terms in the objective function). For recent surveys on DARP and PDPTW, we refer the reader to Cordeau and Laporte [17] and Berbeglia *et al.* [7].

One of the most promising exact technique used in the literature for solving DARP, PDPTW and

their variants is column generation (e.g., see [5, 8, 26, 30, 32]), embedded in a Branch-and-Price scheme. However, due to the difficulty of modeling all the constraints of the real-world problems and of solving large dimension instances, many studies have been also focused on the development of heuristic and metaheuristic approaches [22]. The tabu search technique has been widely applied to DARPs. Cordeau and Laporte [16] first applied TS to a case of DARP with a single depot, where constraints related to vehicles' capacity, route duration and maximum ride time of any user on a vehicle are considered. One of the first applications of TS to the DARP with heterogeneous vehicles was proposed by Toth and Vigo [31], for solving a problem concerning the transportation of handicapped persons in an urban area. The authors present an insertion algorithm and a tabu thresholding procedure to improve an initial solution. Paquette *et al.* [25] developed a multicriteria heuristic embedding a tabu search process for solving a DARP, with a heterogeneous fleet of vehicles and two types of users: ambulatory and wheelchair-bound. In addition to travel costs, the objective function includes three quality terms, in order to reduce user's inconveniences: the waiting time at the pickup node, the waiting time at the delivery node and user's ride time. The proposed solution procedure combines some features of the tabu search heuristic of Cordeau and Laporte [16] and of the multicriteria reference point method of Climaco *et al.* [13]. Melachrinoudis *et al.* [23] proposed a TS heuristic to address a single-depot DARP with soft time windows that arises in a non-profit organization system operating in the Boston metropolitan area. To minimize user's inconveniences, they employ an objective function that includes excess riding time, early/late delivery time before service and late pickup time after service. Attanasio *et al.* [4] propose a TS algorithm for a dynamic variant of DARP, in which only a given set of requests are known during the planning phase, while other requests arrive in real time. In this work, several parallel implementations of the TS procedure that Cordeau and

Laporte proposed in [16] for the static version are developed and compared.

Beaudry *et al.* [6] implemented a two-phase procedure for the dynamic DARP arising in several large hospitals. Different modes of transportation (e.g. a wheelchair or a stretcher) are considered, and a maximum ride time is given to limit patients' inconveniences. The first phase consists in generating an initial feasible solution through a simple and fast insertion scheme. In the second phase, this solution is improved with a TS scheme.

In recent years, also variable neighborhood search based heuristics have been successfully proposed for solving transportation problems. Parragh *et al.* [26] proposed a VNS heuristic with three neighborhood types for a single-depot DARP with homogeneous vehicles, constraints on route duration and time windows and maximum users' ride times. In [27], a collaborative scheme has been proposed, integrating the VNS heuristic into a column generation framework, for a variant of DARP, arising in the Austrian Red Cross context, with driver-related constraints, heterogeneous vehicles and patients. Muelas *et al.* [24] propose a VNS-based algorithm with seven different neighborhood classes (or shakers) for a DARP that arises in the area of San Francisco. In their work patients and visitors must be transported from a set of hospitals located around the city to their home addresses (and vice versa), using a fleet of homogeneous vehicles.

Recently, new exact and heuristic approaches have been proposed for DARP in [10] and [28]. In [10], a multi-depot DARP with heterogeneous vehicles and users is addressed, and a branch-and-cut exact algorithm and a deterministic annealing metaheuristic are proposed. In [28], a hybrid column generation and large neighborhood search algorithm is presented for solving a single-depot DARP with homogeneous vehicles, maximum user ride times, route duration limits, and vehicle capacity constraints.

In healthcare, the studies on transportation problems of patients are usually focused on emergency transportation (e.g., see [3, 11, 19]). Studies on non-emergency transportation problems of patients within hospitals are presented in [6, 20]. Recently, Bowers *et al.* [9] present a decision support tool for patient transportation between homes and healthcare facilities, and Zhang *et al.* [33] address a multi-trip dial-a-ride problem for transporting disabled and elderly patients.

3 Problem description and formulation

In the non-emergency transportation problem we address, a set of transportation services must be performed for carrying patients to healthcare facilities, to attend medical treatments, exams or therapies (e.g., see [2, 9]), or for transferrals and discharges.

A fleet of heterogeneous vehicles (e.g., ambulance, bus, car, etc.), located in geographically distributed depots, is available for performing the transportation services. The vehicles belong to different non-profit organizations, and patients may ask to be transported by a vehicle of a given organization (*patients' preferences*).

The problem consists in assigning the transportation services to the vehicles and in finding a routing of each vehicle, in such a way the total transportation cost is minimized and quality of service requirements are respected. Many constraints must be taken into account, regarding the vehicles' capacities, the patient-vehicle compatibility, the fulfillment of pickup and delivery time windows, patients' preferences, precedence relationships among pickup and delivery locations, and the quality and the timing of the service provided. The capacity constraints are related to the number of available seats in the vehicle and to the number of seats occupied by each patient (e.g., patients may need a

stretcher or a wheelchair to be transported, causing an occupation of more than one seat on a given vehicle). The compatibility constraints take into account the specific set up of the vehicle that can be used to serve a given patient typology (for example, a patient on a stretcher can only be transported by an ambulance) and, eventually, the fact that two patient typologies can not be transported at the same time. The time windows constraints are related to the patients, requiring that each pickup and delivery location need to be reached within a time interval, and may depend on the schedule of the healthcare services the patients have to attend. In fact, patients that must be transported to healthcare facilities to receive a given healthcare service must arrive on time, according to the schedule of the treatments or visits. Patient preferences take into account the fact that a patient may require to be transported by vehicles of a given non-profit organization (and, hence of a given depot). Finally, quality of service constraints require that the ride times of each patient do not exceed pre-specified limits and that the waiting times of the patients at the healthcare facilities are minimum.

In summary, we can describe the problem characterizing its two main elements: the *transportation services* and the *vehicle types*. In the following, the main features of these elements are listed.
A transportation service is defined by:

- pickup and delivery locations;
- healthcare service associated with the transportation service;
- patient typology (e.g., patient on a stretcher or a wheelchair);
- number of seats occupied on a vehicle;
- pickup and delivery time windows;

- patients' preferences.

A vehicle type is defined by:

- depot location and non-profit organization;
- capacity (i.e., the number of seats in the vehicle);
- typology; (i.e., ambulance, equipped vehicle, car or bus);
- number of available vehicles.

Regarding the monetary transportation cost, several components must be considered, depending on agreements between the (regional) government and the non-profit organizations [1]. More precisely, the cost of a route, formally defined later, depends on the length and the duration of the route, the number of passengers transported, the vehicle typology and the waiting times of the vehicles at the healthcare facilities.

We point out that, the cost related to the waiting times of a vehicle (regulated by a stepwise function) occurs in two cases: (1) when a vehicle transports a patient on a wheelchair or on a stretcher and arrives at a healthcare facility before the visit time of the patient; (2) when a vehicle waits for the patients to be loaded at the pickup location. In the first case, the vehicle has to wait until the beginning of the visit before resuming the travel (i.e., as explained later the end of the time window associated to the delivery node), in the latter the vehicle has, obviously, to wait until the patient is available for loading.

Observe that, the waiting times of the vehicles and the waiting times of the patients are two different terms that must be taken into account in different ways. In fact, the waiting times of the

vehicles are included in the (monetary) cost of the route, while the waiting times of the patients are a quality of service indicator.

In the reminder of this section, notation is introduced and the problem is formally defined.

Let $G = (V, A)$ be a complete directed graph, where $V = \{1, \ldots, n, n+1, \ldots, 2n, 2n+1, \ldots, 2n+p\}$ is the node set and $A = \{(i, j) : i, j \in V\}$ is the arc set. The nodes $P = \{1, \ldots, n\}$ are pickup nodes, the nodes $D = \{n+1, \ldots, 2n\}$ are delivery nodes and the nodes $DEP = \{2n+1, \ldots, 2n+p\}$ represent the depots. Hence, each transportation service i can be denoted by the nodes $\{i, i+n\}$ in V. The pickup and the delivery nodes are further classified either as home pickup nodes (home delivery nodes) or as hospital pickup nodes (hospital delivery nodes). In what follows, we denote by P_{HOME}, D_{HOME}, P_{HOSP} and D_{HOSP} the sets of the home pickup nodes, home delivery nodes, hospital pickup nodes and hospital delivery nodes, respectively.

The following notation and parameters will be used throughout the paper.

- $TR = \{1, \ldots, n\}$ set of transportation services;
- $M = \{1, \ldots, m\}$ set of vehicles;
- K number of vehicle types (defined by the pair vehicle typology and depot in which the vehicle is located);
- M_k, with $|M_k| = m_k$, set of vehicles of type k, $k = 1, \ldots, K$. Hence, $M = \bigcup_{k=1}^{K} M_k$;
- $dep(p)$ depot in which vehicle $p \in M$ is located;
- c_k unit length cost for a vehicle of type k;
- λ_k distance fixed cost of a vehicle of type k;

- L maximum travel distance under which the cost λ_k only applies;
- q^p capacity of a vehicle $p \in M$;
- ct_k unit time (per hour) cost for a vehicle of type k;
- cp cost per patient transported;
- $d_{i,j} \geq 0$ travel distance for arc $(i,j) \in A$;
- $t_{i,j} \geq 0$ travel time for arc $(i,j) \in A$;
- $u_i = u_{i+n} \in \{a,b\}$ waiting tariff that applies if a vehicle waits for a patient related to the transportation service $\{i, i+n\}$;
- $\pi_i = \pi_{i+n}$ typology of the patient related to the transportation service $\{i, i+n\}$ ($\pi_i = 0$ seated patient, $\pi_i = 1$ patient on a stretcher or a wheelchair);
- st_i, *service time*, time to load or unload a patient at a node $i \in P \cup D$;
- ca_t time dependent vehicle waiting cost for a patient with waiting tariff of type a;
- cb_t time dependent vehicle waiting cost for a patient with waiting tariff of type b;
- T_i maximum ride time (including the waiting time at the delivery location, if any) of the patient related to the transportation service $\{i, i+n\}$;
- K_i set of vehicle types that are able (or that are preferred by the patient) to serve the transportation service $\{i, i+n\}$;

- s_i number of seats that are occupied (if i is a pickup node) or freed (if i is a delivery node) when node $i \in P \cup D$ is visited;
- e_i beginning of the time window at node i;
- l_i end of the time window at node i.

The time window $[e_i, l_i]$ of each node $i \in P \cup D$ indicates that a service at node i can only take place between time e_i and l_i. A vehicle is allowed to arrive to the location of i before the start of the time window, but it has to wait e_i to begin the load or the unload of the patient. A vehicle arriving at node $i \in D_{HOSP}$ has to wait until the end of the time window l_i, if the patient related to i is on a stretcher or a wheelchair (i.e., $\pi_i = 1$). A cost for the vehicle waiting applies if a vehicle waits at node $i \in P_{HOME} \cup P_{HOSP} \cup D_{HOSP}$. This cost is formally defined in Section 3.1.

The problem consists in finding a set of routes on G such that:

1. every route starts and ends at the same depot;
2. the number of routes assigned to vehicles of type k does not exceed m_k;
3. for each transportation service $\{i, i+n\}$, nodes i and $i+n$ belong to the same route, assigned to a vehicle in K_i, and node $i+n$ is visited later than node i;
4. the ride time of a patient related to the transportation service $\{i, i+n\}$ does not exceed T_i;
5. the service at node i begins in the interval $[e_i; l_i]$;
6. a quality index defined by the sum of the waiting times of the patients at the nodes $P_{HOSP} \cup D_{HOSP}$ is minimum;

7. the total cost of the routes is minimized.

3.1 Routing costs and quality of service requirements

The (monetary) cost of a route assigned to a vehicle $p \in M_k$ has the following components. Travel distance cost: a route of length Λ costs λ_k if $\Lambda \leq L$ and $(\lambda_k + c_k(\Lambda - L))$ otherwise. Travel time cost: a route of duration D costs $ct_k D$. Cost depending on the transported persons: $cp(n_p - 1)$ where n_p is number of patients transported in the route. Vehicle waiting cost: a vehicle waiting at a node $i \in P_{HOME} \cup P_{HOSP} \cup D_{HOSP}$ has a cost depending on the tariff $u_i \in \{a, b\}$ and the waiting time, W_i^p. More precisely, there are $l-1$ time intervals (Δ_t, Δ_{t+1}), for $t = 1, \ldots, l-1$, where $\Delta_1 = 0$ and $\Delta_l = \Delta_{max}$. The cost depends on the interval W_i^p belongs to. For example, if $W_i^p \in (\Delta_t, \Delta_{t+1})$ and $u_i = a$, the waiting cost at node i is given by $\sum_{p=1}^{t} ca_p$ (In the real case, as reported in Table 2, we have that $ca_1 = 0$ and $ca_p > 0$, for $p = 1, \ldots, l-1$.)

The quality of service index on the waiting times of the patients is given by the sum of the waiting times of the patients at the nodes $i \in P_{HOSP} \cup D_{HOSP}$. The waiting time of a patient at node i, WP_i, is defined as follows

$$WP_i = \left\{ \begin{array}{cc} A_i - e_i & \text{if } e_i < A_i \text{ and } i \in P_{HOSP} \\ l_i - A_i & \text{if } l_i > A_i \text{ and } i \in D_{HOSP} \\ 0 & \text{otherwise} \end{array} \right\} \tag{1}$$

where A_i is the arrival time at node i of the vehicle serving it. Then, the quality of service index to

minimize is

$$\psi \sum_{i \in P_{HOSP} \cup D_{HOSP}} WP_i, \tag{2}$$

where $\psi \geq 0$ is a suitable parameter.

The quality of service index on the ride times imposes that the ride time of the patient corresponding to the transportation service $\{i, i+n\}$ is not bigger than T_i.

3.2 A MILP formulation

In this section, a MILP formulation for the addressed problem is presented, in which both integer and real variables are used. The variables are listed below.

- A_i^p arrival time of vehicle p at node i;
- B_i^p beginning of the service of vehicle p at node i;
- L_i^p ride time of the user related to node i on vehicle p;
- Q_i^p load of vehicle p leaving node i;
- W_i^p waiting time of vehicle p at node i, with $i \in P_{HOME} \cup P_{HOSP} \cup D_{HOSP}$;
- $Dist^p$ is equal to $\max\{\Lambda - L; 0\}$, where Λ is the total distance (in kms) traveled by vehicle p and L is the parameter introduced in the above section;
- WP_i waiting time of the patient at node i, with $i \in P_{HOSP} \cup D_{HOSP}$;
- $x_{ij}^p \in \{0,1\}$ be equal to 1 if vehicle p traversed arc (i, j) and 0 otherwise;

- $y^p \in \{0,1\}$ be 1 if vehicle p is used and 0 otherwise;

- $z_{i,t} \in \{0,1\}$ be 1 if waiting time at node i is bigger than Δ_t, for $t = 1,\ldots,l-1$, and 0 otherwise.

The objective function reads as follows

$$\min \sum_{k=1}^{K} \sum_{p \in M_k} \lambda_k y^p + \sum_{k=1}^{K} \sum_{p \in M_k} c_k Dist^p + \sum_{k=1}^{K} \sum_{p \in M_k} ct_k (A^p_{dep(p)} - B^p_{dep(p)}) + \sum_{p \in M} cp\Big(\sum_{(i,j) \in A : i \in P} x^p_{ij} - y^p \Big) + \sum_{i \in P_{HOME} \cup P_{HOSP} \cup D_{HOSP} : u(i) = a} \sum_{t=1}^{l-1} ca_t z_{it} + \sum_{i \in P_{HOME} \cup P_{HOSP} \cup D_{HOSP} : u(i) = b} \sum_{t=1}^{l-1} cb_t z_{it} + \psi \sum_{i \in P_{HOSP} \cup D_{HOSP}} WP_i \tag{3}$$

In (3), the first and the second term are related to the fixed and variable distance costs, respectively. The third term is the cost depending on the travel time of each vehicle p, equal to the difference between the arrival time at the depot ($A^p_{dep(p)}$) and the departure time from the depot ($B^p_{dep(p)}$). The forth term is the cost related to the number of patients (except the first) transported in each route. In fact, if $y^p = 1$ (i.e., the vehicle is used), the sum $\sum_{(i,j) \in A : i \in P} x^p_{ij}$ gives the number of pickup nodes (and hence the number of loaded patients) visited by the vehicle. Observe that, if $y^p = 0$, by constraints (8), all variables x^p_{ij} are forced to be 0. The fifth and the sixth terms are the costs depending on the vehicles' waiting times that apply to the transportation service i, as explained in Section 3.1. Finally, the last term is the quality index on the waiting times, i.e., the sum of the patients waiting times (see relation (2)) multiplied by a suitable parameter $\psi \geq 0$.

In what follows, the constraints of the formulation are reported.

$$\sum_{p\in M}\sum_{j\in V} x_{ij}^{p} = 1 \ \ \forall i \in P \tag{4}$$

$$\sum_{j\in V} x_{ij}^{p} - \sum_{j\in V} x_{n+i,j}^{p} = 0 \ \ \forall i \in P,\ p \in M \tag{5}$$

$$\sum_{i\in V} x_{ij}^{p} - \sum_{i\in V} x_{ji}^{p} = 0 \ \ \forall j \in V,\ p \in M \tag{6}$$

$$\sum_{j\in P} x_{dep(p),j}^{p} = y^{p} \ \ \forall p \in M \tag{7}$$

$$x_{ij}^{p} \leq y^{p} \ \ \forall i,j \in A \ \ p \in M \tag{8}$$

Constraints (4) ensure that each request is served exactly once and constraints (5) ensure that each origin–destination pair is visited by the same vehicle. Flow conservation is imposed by equalities (6). Constraints (7) state that if a vehicle $p \in M$ is used, then exactly one arc $(dep(p), j)$, with $j \in P$, must be used by p. Constraints (8) set to 0 variables x_{ij}^{p} if vehicle p is not used.

$$Dist^{p} \geq \sum_{(i,j)\in A} d_{ij} x_{ij}^{p} - L \ \ \forall p \in M \tag{9}$$

$$Dist^{p} \geq 0 \tag{10}$$

Constraints (9)–(10) set $Dist^{p}$, used to compute the variable length cost occurring for routes longer than L.

$$Q_{j}^{p} \geq s_{j} + Q_{i}^{p} - q^{p}(1 - x_{ij}^{p}) \ \ \forall i,j \in V \ \ p \in M \tag{11}$$

$$A_{j}^{p} \geq B_{i}^{p} + st_{i} + t_{ij} - (l_{i} + st_{i} + t_{ij})(1 - x_{ij}^{p}) \ \ \forall i,j \in V \ \ p \in M_{k} \ \ k \in K \tag{12}$$

$$A_{j}^{p} \leq B_{i}^{p} + st_{i} + t_{ij} + l_{dep(p)}(1 - x_{ij}^{p}) \ \ \forall i,j \in V \ \ p \in M \tag{13}$$

Constraints (11) are load propagation inequalities. Constraints (12) and (13) state that the arrival time at node j must be equal to $B_{i}^{p} + st_{i} + t_{ij}$ if $x_{ij}^{p} = 1$. More precisely, constraints (13) impose that A_{j}^{p} can

not be bigger than $B_i^p + st_i + t_{ij}$, if $x_{ij}^p = 1$. If $x_{ij}^p = 0$, they become $A_j^p \leq B_i^p + st_i + t_{ij} + l_{dep(p)}$, where $l_{dep(p)}$ is the end of the time window at the depot of vehicle p.

$$W_i^p \geq B_i^p - A_i^p - l_i(1 - \sum_{(j,i)\in\delta^-(i)} x_{ji}^p) \ \forall i \in P \cup D \ \ p \in M_k \ \ k \in K \tag{14}$$

$$W_i^p = 0 \ \forall i \in D_{HOME} \ \ p \in M_k \ \ k \in K \tag{15}$$

$$\sum_{p\in M} W_i^p \leq \sum_{t=1}^{l-1} (\Delta_{t+1} - \Delta_t) z_{i,t} \ \ \forall i \in P \cup D \tag{16}$$

$$z_{i,t-1} \geq z_{i,t} \ \ \forall i \in P \cup D \ \ t = 2, \ldots, l-1 \tag{17}$$

Constraints (14) set the waiting time of vehicle p at node i, W_i^p, where $\delta^-(i)$ denotes the set of arcs entering node i. Constraints (15) state that no waiting is possible at the nodes in D_{HOME}. Constraints (16) and (17) link the variables W_i^p and $z_{i,t}$, that are used to define the waiting cost of the vehicle, as explained in Section 3.1.

$$L_i^p = B_{n+i}^p + st_{n+i} - (B_i^p + st_i) \ \ \forall i \in P \ \ p \in M_k \ \ k \in K \tag{18}$$

$$t_{i,n+i} + st_{n+i} \leq L_i^p \leq T_i \ \ \forall i \in P \ \ p \in M_k \ \ k \in K \tag{19}$$

$$e_i \leq B_i^p \leq l_i \ \ \forall i \in P \cup D \ \ p \in M_k \ \ k \in K \tag{20}$$

$$B_i^p \geq l_i \ \ \forall i \in D_{HOSP} : \pi_i = 1, \ \ p \in M_k \ \ k \in K \tag{21}$$

$$A_i^p - e_i \leq WP_i \ \ \forall i \in P_{HOSP} \ \ p \in M \tag{22}$$

$$l_i - A_i^p \leq WP_i \ \ \forall i \in D_{HOSP} \ \ p \in M \tag{23}$$

$$WP_i \geq 0 \ \ \forall i \in P_{HOSP} \cup D_{HOSP} \tag{24}$$

$$A_{dep(p)}^p - B_{dep(p)}^p \geq 0 \ \ \forall p \in M \tag{25}$$

Equalities (18) define the ride time of each user. Time window and maximum ride time compliance are ensured by (19) and (20). Constraints (20) and (21) set $B_i^p = l_i$ for each node $i \in P_{HOSP} \cup D_{HOSP}$

related to a patient on a stretcher or on a wheelchair (i.e., such that $\pi_i = 1$).

According to (1), constraints (22)–(24) set the waiting times of the patients at pickup and delivery hospital nodes. Constraints (25) state that the arrival of a vehicle to the depot must be after the beginning of the service, corresponding to the departure time of the vehicle. Finally, the compatibility constraints between vehicles and patients and the patient preferences (constraints (26)), and the variable domains read as

$$x_{i,j}^p = 0 \;\; i \in P \cup D \;\; \forall p \in M_k : \; k \in K \setminus K_i \tag{26}$$

$$A_i^p, B_i^p, L_i^p, Q_i^p \geq 0 \;\; \forall i \in V \;\; p \in M_k \;\; k \in K \tag{27}$$

$$x_{ij}^p \in \{0,1\} \;\; \forall i, j \in V \;\; p \in M_k \;\; k \in K \tag{28}$$

$$y^p \in \{0,1\} \;\; \forall p \in M_k \;\; k \in K \tag{29}$$

$$z_{i,t} \in \{0,1\} \;\; \forall i \in P \cup D \;\; t = 1, \ldots, l-1. \tag{30}$$

In the MILP model presented above, following the lines of Cordeau [15], the number variables can be reduced by using aggregate variables B_i, A_i, L_i and W_i for each node $i \in P \cup D$. Hence, constraints (12) and (13) can be rewritten as

$$A_j \geq B_i + st_i + t_{ij} - (l_i + st_i + t_{ij})(1 - x_{ij}^p) \;\; \forall i, j \in P \cup D \;\; p \in M_k \;\; k \in K \tag{31}$$

$$A_j \geq B_i^p + st_i + t_{ij} - (l_i + st_i + t_{ij})(1 - x_{ij}^p) \;\; \forall i \in DEP, j \in P \;\; p \in M_k \;\; k \in K \tag{32}$$

$$A_j^p \geq B_i + st_i + t_{ij} - (l_i + st_i + t_{ij})(1 - x_{ij}^p) \;\; \forall i \in D, j \in DEP \;\; p \in M_k \;\; k \in K \tag{33}$$

$$A_j \leq B_i + st_i + t_{ij} + l_{dep(p)}(1 - x_{ij}^p) \;\; \forall i, j \in P \cup D \;\; p \in M \tag{34}$$

$$A_j \leq B_i^p + st_i + t_{ij} + l_{dep(p)}(1 - x_{ij}^p) \;\; \forall i \in DEP, j \in P \;\; p \in M \tag{35}$$

$$A_j^p \leq B_i + st_i + t_{ij} + l_{dep(p)}(1 - x_{ij}^p) \;\; \forall i \in D, j \in DEP \;\; p \in M \tag{36}$$

and constraints (14)–(16) and (18)–(23) become

$$W_i \geq B_i - A_i - l_i(1 - \sum_{(j,i)\in\delta^-(i)} x_{ji}^p) \ \forall i \in P \cup D \ \ p \in M_k \ \ k \in K \tag{37}$$

$$W_i = 0 \ \forall i \in D_{HOME} \tag{38}$$

$$W_i \leq \sum_{t=1}^{l-1} (\Delta_{t+1} - \Delta_t) z_{i,t} \ \forall i \in P \cup D \tag{39}$$

$$L_i = B_{n+i} + st_{n+i} - (B_i + st_i) \ \forall i \in P \tag{40}$$

$$t_{i,n+i} + st_{n+i} \leq L_i \leq T_i \ \forall i \in P \tag{41}$$

$$e_i \leq B_i \leq l_i \ \forall i \in P \cup D \tag{42}$$

$$B_i \geq l_i \ \forall i \in D_{HOSP} : \pi_i = 1 \tag{43}$$

$$A_i - e_i \leq WP_i \ \forall i \in P_{HOSP} \tag{44}$$

$$l_i - A_i \leq WP_i \ \forall i \in D_{HOSP}. \tag{45}$$

4 A tabu search algorithm

In this section, we present a Tabu Search (TS) algorithm designed for solving the addressed problem. The algorithm is based on the one presented in [16] and has two main phases. In the first phase, an initial solution (not necessarily a feasible solution) is generated through a fast insertion heuristic. Then, a TS scheme iteratively transforms and tries to improve this solution. At each iteration t, starting from an initial solution $s_{curr} = s_0$, a neighborhood $N(s_{curr})$ is generated by perturbing the current solution. Then, the best solution in $N(s_{curr})$, according to an evaluation function (introduced later), is selected and chosen as the new current solution s_{curr}, even if it is worse than the old s_{curr}. The whole process is repeated until a stopping criterion is satisfied. To avoid cycling, recently visited

solutions are forbidden (tabu) for a number of iterations. A tabu list is used to keep track of recent moves or visited solutions. Since the problem described in Section 3 is strongly constrained, our procedure allows the exploration of infeasible solutions during the search. Constraints' violations are weighted in the evaluation function with self-adjusting parameters (penalties) in order to drive the search to feasible regions.

A scheme of the algorithm in reported in Algorithm 1. In the remaining part of this section, the overall framework and the building blocks of our two-step approach are described into detail.

Algorithm 1 Scheme of the Tabu Search algorithm.

Step 1

Generate an initial solution s_0, Set the current solution $s_{curr} := s_0$.

If s_{curr} is feasible

Set the best solution $s_{best} := s_{curr}$. Otherwise $s_{best} := +\infty$.

Step 2

While A maximum number of iterations It_{max} is not reached **do**

begin

Generate the neighborhood of the current solution $N(s_{curr})$.

Select s^*, the not tabu solution in $N(s_{curr})$ with the lowest value of the evaluation function, and set $s_{curr} := s^*$.

If the s_{curr} is feasible and better than s_{best}

Set $s_{best} := s_{curr}$.

If s^* is a tabu solution and satisfies the aspiration criterion

Set s_{curr} and s_{best} to s^*.

Update the tabu list.

Update the penalty coefficients.

end

Return the best solution s_{best}.

4.1 Evaluation function

The TS algorithm allows the exploration of infeasible solutions in order to facilitate the search in the solution space.

Following Cordeau and Laporte [16] and Parragh *et al.* [26], we allow the violation of three main constraints of the problem, i.e., the constraints on patients' ride times, on the time windows and on the vehicles' capacity. The violations of these constraints are weighted in the evaluation function by penalty positive coefficients.

Given a solution s of the problem, the evaluation function tackled by TS has three terms and is equal to $f(s) = f_1(s) + f_2(s) + f_3(s)$. The three terms are described in the following.

Functions $f_1(s)$ and $f_2(s)$ correspond to the monetary cost (i.e., the first six terms of the objective function (3)) and the quality index (2), respectively. The term $f_3(s)$ is a penalization component of the form: $f_3(s) = \alpha t(s) + \beta w(s) + \gamma q(s)$, where $t(s)$, $w(s)$ and $q(s)$ represent the total violation of solution s with respect to the constraints on patient ride times, time windows and vehicle capacities, respectively, and α, β and γ are positive penalty coefficients. In more detail, violations are calculated as follows, where, given a node $i \in P \cup D$, we denote by $p(i)$ the vehicle serving node i.

- $t(s) = \sum_{i \in P} (L_i^{p(i)} - T_i)^+$;
- $w(s) = \sum_{i \in P \cup D} (A_i^{p(i)} - l_i)^+ + (e_i - A_i^{p(i)})^+$. Recall that early arrival is allowed, but the vehicle has to wait until the start of the time window to begin the load or the unload service.
- $q(s) = \sum_{i \in P \cup D} (Q_i^{p(i)} - q^{p(i)})^+$.

Initially, coefficients α, β and γ are set to given values (i.e., α_0, β_0 and γ_0), and, at each iteration, are modified by a factor $1 + \delta$, where $\delta > 0$, as follows. If the ride time constraints are violated by the

current solution s_{curr}, then $\alpha = \alpha(1+\delta)$ is set and $\alpha = \alpha/(1+\delta)$ otherwise. The same rule is applied to β and γ with respect to time windows and vehicles' capacity constraints violations, respectively.

4.2 Initial solution

In the TS algorithm, a deterministic insertion heuristic is used for the initial solution. The heuristic is a two-step procedure. In the first step, all the transportation services $i \in TR$ (corresponding to the nodes i and $n+i$ in G) are sorted according to the end of the time window of the delivery node, l_{n+i}. Then, the transportation services are assigned one by one to the routes according to the ordering. During this assignment procedure the algorithm first attempts to assign the service to an empty route, if any, and then considers not empty routes. Each service i is assigned to the compatible route corresponding to a vehicle in K_i, having the lower distance value between the last scheduled node in the route and the pickup node i of the service. The pickup node i and the delivery node $n+i$ are inserted sequentially at the end of the selected route. During the insertion, a service is assigned only to routes that respect the patient-vehicle compatibility constraints. The overall procedure ensures the satisfaction of all the constraints of the problem, with the exception of the time windows constraints. These constraints are surely satisfied only if the number of services is smaller than the number of available vehicles and if every service can be served by any type of vehicle. A scheme of the heuristic is reported in Algorithm 2.

Algorithm 2 Scheme of the procedure for finding an initial solution in TS.

Order all the services $i \in TR$ in non-decreasing order of values of l_{i+n}.

Assign an empty route to each vehicle.

For each service i

If there are empty routes in K_i

Insert i and $n+i$ sequentially at the end of the empty route in K_i such that the distance between i and the depot is minimum.

If no empty route exists in K_i

Insert i and $n+i$ sequentially at the end of route in K_i such that the distance between i and the last scheduled node is minimum.

4.3 Neighborhood structure and generation

In the tabu search algorithm, a move from a solution s to a neighbor solution s' is performed by removing a transportation service from a route r and inserting it into another route r'. More precisely, a solution in the neighborhood of s is generated by a local move that attempts to remove the pickup-delivery pair, i and $n+i$, of a given transportation service i from its current route r and to insert it into another route r' of s. Nodes i and $n+i$ are inserted into route r' as to minimize the total increase of the evaluation function $f(s)$ without changing the ordering of the nodes already in r'.

The neighborhood is generated by a procedure that randomly selects a subset of cardinality w of the pickup nodes in the current solution. Once one of the w pickup nodes, say i, is identified in a route r, the procedure attempts to place it into a randomly chosen route $r' \neq r$ in K_i. If r' is an empty route, i and $n+i$ are simply removed from r and sequentially inserted in r'. If r' is not empty, a position in r' is considered admissible for inserting i if the travel time after serving the predecessor node in r' does not exceed l_i. If, at least, an admissible position is found for the pickup node i, it is inserted in the admissible position with minimum insertion cost. Then, also the associated delivery node $n+i$

is inserted in r', after i, in the position that minimizes its insertion cost. If no admissible position exists for the pickup node i, no move is performed. Note that, according to this procedure, neighbor solutions may violate the time window, the ride time and the capacity constraints.

4.4 Tabu list and current and best solution updating

The tabu list has a fixed length θ and new insertions and removals follow a FIFO strategy. The parameter θ (the *tabu tenure*) is fixed during the search. The tabu status of an attribute can be revoked if that would allow the search process to explore a feasible solution with a value of the evaluation function lower than the value of the best known solution (aspiration criterion).

At each iteration, the tabu search algorithm generates the neighborhood of the current solution $N(s_{curr})$ and chooses the best solution in $N(s_{curr})$, called s^*, according to the evaluation function $f(s) = f_1(s) + f_2(s) + f_3(s)$. If s^* is not a solution in the tabu list (or satisfies the aspiration criterion), it is chosen as the new current solution. Hence, $s_{curr} = s^*$. If $f_3(s^*) = 0$, the new current solution is feasible and if $f(s^*) < f(s_{best})$, then s^* is also set as the new best solution s_{best}. The tabu search algorithm stops if a given number of iterations IT_{max} is reached returning the best solution s_{best}.

5 A variable neighborhood search algorithm

In this section, the general framework of the VNS-based algorithm developed for the addressed DARP is presented. The algorithm is based on the approach proposed in [26] and has two main steps. First an initial solution, s_0, is generated not necessarily feasible, through a fast insertion heuristic. Then, s_0 undertakes a local search step, yielding s, being this the first incumbent solution. The second step

of the algorithm is an iterative procedute that starts with the shaking procedure. At each iteration of the algorithm, a new randomly generated solution, s', is created in the current neighborhood $N_h(s)$. A local search step is applied to s' yielding s''. If s'' is feasible and better than s_{best}, s is replaced by s'' and the search starts with the first neighborhood, $h = 1$. If s'' is not feasible or if it is worse than s_{best} and better than s, s is replaced by s'' and the search starts with the first neighborhood. If s'' is worse than s, s is not replaced and the next neighborhood is used for the following iteration. If only a violation on a single type of constraint, i.e., capacity, ride time or time windows, is found in s'', an *adjusting procedure* is applied, generating s_{adj}. The adjusting procedure aims to make feasible a slightly infeasible solution. Whenever the maximum number of neighborhoods h_{max} is reached the search continues with the first neighborhood, until the maximum number of iterations It_{max} is reached.

A sketch of the VNS algorithm is given in Algorithm 3. In the remaining part of this section, the main building blocks of the algorithm are described into detail.

5.1 Evaluation function, forward time slack and minimization of the vehicle waiting cost

The evaluation function used by the VNS algorithm is similar to the one employed by the TS algorithm and it is based on the one used in Cordeau and Laporte [16] and Parragh *et al.* [26]. The only difference concerns the function $f_3(s)$. In particular, in the VNS, $f_3(s)$ contains an extra term related to the length of the routes, i.e., $f_3(s) = \alpha t(s) + \beta w(s) + \gamma q(s) + \tau d(s)$, where $d(s) = \sum_{p \in R(s)} ((A^p_{dep(p)} - B^p_{dep(p)}) - RD)^+$, being RD the maximum route duration. Although such a constraint is not required by the addressed application, VNS has been designed to cope even with limits

Algorithm 3 Scheme of the Variable Neighborhood Search algorithm

Generate the initial solution s_0;

Apply local search to s_0 yielding s;

If s is feasible. Set $s_{best} := s'_{best} := s$. **Else** set $s_{best} := s'_{best} := +\infty$;

Set $h := 1$;

Repeat

{ Randomly choose s' in $N_h(s)$;

If $f_1(s) < 1.02 f_1(s')$

Apply local search to the routes in s' that have been changed yielding s'';

else Set $s'' := s'$;

If s'' is feasible and $f(s'') < f(s_{best})$

Set $s_{best} := s''$ and $s := s''$ and $h := 0$;

Else

If $f(s'') < f(s)$

{ Set $s := s''$ and set $h := 0$;

If s'' only violates capacity, (ride time, time windows) constraint and $q(s) < q_{max}$ $(t(s) < t_{max}, w(s) < w_{max})$

{ Apply the adjusting procedure to s'' yielding s_{adj};

If s_{adj} is feasible and $f(s_{adj}) < f(s'_{best})$

Set $s'_{best} := s_{adj}$;

}

}

Set $h := (h \bmod h_{max}) + 1$;

} **Until** It_{max} iterations are reached

If $f(s_{best}) < f(s'_{best})$ return s_{best} **Else** return s'_{best}.

on maximum route duration, and tested in Section 6 on random instances. Initially, coefficients α, β, γ and τ are set to given values α_0, β_0, γ_0 and τ_0, respectively, and each time a new incumbent solution is found are modified, as in the TS algorithm by a factor of $1+\delta$. In the VNS, δ is randomly chosen between 0.05 and 0.1, and changes every time a new incumbent solution is found.

In order to minimize the timing and the waiting costs of the routes, two procedures are applied to each solution s found during the algorithm. One in order to set the beginning of a route in the best possible way, and the other one to reduce the costs related to the waiting of the vehicles (introduced in Section 3.1).

In order to set the beginning of a service in a route in the best possible way and to minimize the total duration of the route, the evaluation scheme introduced in [16] and used in [26], too, is applied. Given a route, it employs the idea of forward time slack, F_i, where F_i is defined as the largest increase in the beginning of the service at node i that does not cause any violation, i.e.,

$$F_i = \min_{i \le j \le q} \left\{ \sum_{i<r<j} W_r^p + \min\{(l_j - B_j)^+\} \right\} \qquad \forall i \in DEP \cup P$$

$$F_i = \min_{i \le j \le q} \left\{ \sum_{i<r<j} W_r^p + \min\{(l_j - B_j)^+, (T_j - L_j^p)^+\} \right\} \qquad \forall i \in D$$

where q is the last node in the route.

The scheme is described in Algorithm 4.

Furthermore, in order to reduce the waiting costs of the vehicles, a heuristic procedure has been developed to spread the waiting times along the route, taking into account the waiting intervals and tariffs (see Section 3.1). Given a route, the procedure tries to reduce the waitings of the vehicle in the nodes where waiting costs apply, and to increase the waiting in the nodes with no costs, respecting time windows and ride time constraints (the route duration is not affected by the procedure). More

Algorithm 4 Evaluation scheme

1: Set $A_1^p = e_1$ and $B_0^p = A_1^p - t_{0,1}$

2: Compute A_i^p , W_i^p, B_i^p and Q_i^p for each node

3: **If** $B_i^p > l_i$ or $Q_i^p > q^p$ **then go to** step 13

4: Compute F_1

5: Set $A_1 = B_1 + \min\{\sum_{1<r<q} W_r^p, F_1\}$ and $B_0^p = A_1^p - t_{0,1}$

6: Update A_i^p , W_i^p and B_i^p for each vertex

7: Compute L_i^p for each request in p. If all $L_i^p \leq T_i$ **go to** step 13

8: **For** every pickup

9: Calculate F_j

10: $B_j^p = B_j^p + min\{F_j, \sum_{j<r<q} W_r^p\}$ and $W_j^p = B_j - A_j$

11: **For** every vertex after j update A_i^p, W_i^p and B_i^p

12: Update L_i^p for each request in p.

13: Compute changes in capacity, ride time and time window violations.

precisely, at each node i of the route, the maximum increase of the waiting of the vehicle that does not produce an increase of the waiting cost is calculated by a function J_i, defined below. Given a route p with waiting costs, let i_1 be the second pickup node of the route and let i_2 be the last node in which a cost for the vehicle waiting arises. Let $U(i)$ be the set of patients on the vehicle at node i, recalling that the smallest waiting tariff occurs in the interval $(\Delta_1 = 0, \Delta_2)$ (as reported in Section 3.1), we set

$$J_i = \min \left\{ \min_{i \leq j < i_2} \{l_j - B_j^p\}; \min_{b \in U(i)} \{T_b - L_b^p\}; (\Delta_2 - W_i^p)^+; (W_{i_2}^p - \Delta_2) \right\}.$$

In the above formula, the first three terms define the maximum delay of the departure at node i in such a way that: no violation of the time window constraints at the nodes from i on occurs (first term), no violation of the ride times of the patients on the vehicle arises (second term), the waiting time belongs to the first time interval (Δ_1, Δ_2), i.e., the first waiting tariff applies (third term). Finally, the fourth term indicates the minimum reduction of the waiting at node i_2 in order to belong to the first time interval (Δ_1, Δ_2). A scheme of the procedure used to reduce the waiting costs of a route is presented

in Algorithm 5.

Algorithm 5 Minimizing the waiting costs

Let i_1 be the second pickup node of the route and let i_2 be the last node in which a cost for the vehicle waiting arises;

For every node i, with $i_1 \leq i < i_2$;

begin

Compute J_i;

Increase the waiting time on the node i by J_i;

Update A_j^p, W_j^p, B_j^p for each node $j > i$;

end

As an example, Figure 1 shows the application of the evaluation scheme (Algorithm 4) and of the procedure for minimizing the waiting costs (Algorithm 5) to the route depicted in Figure 1.a. Let us suppose that a cost occurs only if the vehicle waits more than 60 minutes (as in the case study), i.e., $ca_t = cb_t = 0$ if $t \in [0,60]$. In Figure 1.a, a feasible route with 3 transportation services is reported, each service denoted by the pickup nodes p_i and the delivery nodes d_i, $i = 1,2,3$. In the upper part of Figure 1.a, the time window (e_i, l_i) of each node are reported, while in the lower part the arrival time (A_i), the beginning of service (B_i) and the waiting of the vehicle (W_i) at each node in the current routing are given. Observe that, the route is 268 minutes long and that the vehicle waits at nodes p_2 and p_3, 57 and 70 minutes, respectively. Figure 1.b shows the route after the application of the evaluation scheme. Now, the length of the route is smaller than 20 minutes and the waiting at node p_2 has been reduced of 20 minutes. Finally, by applying the procedure for minimizing the waiting costs to route in Figure 1.b, we obtain the route in Figure 1.c. Observe that, no change in the route length occurs, but now the waiting at nodes p_2 and p_3 is 47 and 60 minutes, respectively, generating a smaller cost (i.e., equal to 0) for the waiting of the vehicle.

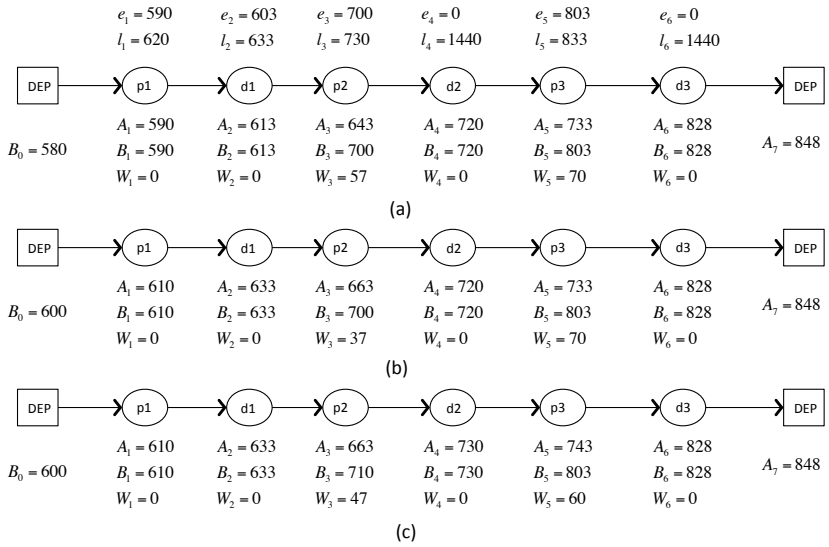

Figure 1: Application of the evaluation scheme and of the procedure for minimizing the waiting costs: (a) Initial route; (b) the route after the evaluation scheme; (c) the route after the procedure for minimizing the waiting costs.

5.2 Initial Solution

A heuristic has been designed to generate the initial solution s_0, working as follows. The transportation services are first sorted according to the pickup earliest time, e_i. According to this ordering, the services are assigned and inserted in the routes: first the pickup node is inserted and right after the delivery node. A service can be assigned to a route only if the compatibility constraints and the preferences of the patients are respected. Capacity or ride time violations are not considered during this insertion phase. The assignment of the services to the routes is performed according to a minimum distance criterium. More precisely, a service is assigned to the route minimizing the average between the distance from the destination of the last service on the route (if the route is still empty the depot destination is taken instead) to the origin of the service to assign, and the distance from the destination of the service to assign to the depot. If an assignment does not satisfy the time window constraints or pre-specified limits on the length of the route, the insertion is not performed and the next route that minimizes the distance criterium, as described above, is considered. If, at the end, the service is not assigned to any route, the service is assigned to the route that produces the smallest increase on the route length (without regarding the time window constraints).

5.3 Shaking

Five different types of neighborhoods are used, called *swap*, *move*, *repairing move*, *chain* and *eliminate*. The neighborhoods swap and chain are similar to those used in [26], but they have been adapted to manage the patient-vehicle compatibility constraints, patients' preferences and the multi-depot case. While neighborhoods *repairing move* and *eliminate* are completely new and designed for the

problem under study.

5.3.1 Swap

In the neighborhood swap, two sequences of services belonging to two different routes are exchanged. The two sequences have a length not bigger than h, where h is the size of the current neighborhood. Hence, a swap of size $h = 2$ consists in exchanging two sequences not longer than two. More precisely, the two sequences are exchanged, as follows. First two routes are randomly selected. Then, the lengths of the sequences in the two routes to be swapped are randomly chosen. (These lengths must be at most the size of the neighborhood h). Then, the first nodes of the two sequences are randomly chosen. Due to the compatibility and preference constraints, all services of the first (second) sequence must be compatible with the vehicle of the second (first) route. Otherwise, the whole process will start again. After the selection of the routes and of the sequences, the services of each sequence are removed from their original route and inserted in the best possible position in the destination route (respecting the order of the sequence). The allocation of the services of each sequence is done one by one, first inserting the pickup and then inserting the delivery. The insertion is performed in the location that minimizes the evaluation function.

5.3.2 Move

In the neighborhood move, a number of transportation services are randomly selected and inserted in randomly chosen routes, respecting compatibility and preference constraints. The number of transportation services to move is randomly chosen between 1 and the size of the current neighborhood, h. The insertion of the services is performed as in swap. Hence, a move of size $h = 3$ consists in

inserting at most three transportation services each of them on a different randomly selected route.

5.3.3 Repairing move

Repairing move basically works as move, but it applies to infeasible solutions only. In repairing move, the transportation services to move are randomly selected only from infeasible routes and inserted, as in move, in randomly chosen routes. The maximum number of transportation services to move is randomly chosen between 1 and the size of the current neighborhood, h. The insertion of the services in the randomly selected routes is performed as in move and swap.

5.3.4 Chain

In chain, a sequence of services is first randomly selected as in swap from an *initial* route, say r_0. Then, the sequence is inserted, as in swap, in a randomly selected *destination* route, compatible with the services of the initial sequence. The above procedure is repeated on the destination route (that becomes the new initial route). The procedure is repeated at most h times, where h is the neighborhood size. Hence, a chain of size $h = 3$ consists in moving a random sequence of length at most three from its initial route to a destination route. Then, from this destination route, another sequence of length at most three is moved to another destination route. Finally, from this third route again a sequence of length at most three is moved to a fourth route.

5.3.5 Eliminate

Eliminate generates new solutions by eliminating some existing routes. The routes to eliminate are those with the biggest ratio between the total cost of the route and the number of services in the route.

Eliminate attempts to assign each service of the eliminated routes at the beginning or at the end of other already used compatible routes or to empty compatible routes. The assignment is performed at the route causing the smaller increase of the solution cost.

5.3.6 Neighborhood Order

In the VNS algorithm, first the simplest neighborhoods are applied and afterwards the more complex and time consuming ones. More precisely, in the shaking phase, $h_{max} = 6$ is set and the following neighborhood order is applied: $S1-RM1-M1-C1-E1-S2-RM2-M2-C2-E2-S3-RM3-M3-C3-E3-S4-RM4-M4-C4-E4-S5-RM5-M5-C5-E5-S6-RM6-M6-C6-E6$, where S stands for swap, RM for repairing move, M for move, C for chain and E for eliminate, and the number is the size of the neighborhood. Hence, $N_1(s) = S1$, $N_2(s) = RM1$ and so on.

5.4 Local Search

VNS employs a local search procedure in which only intra-route changes are made. The local search procedure is very similar to the one proposed in [26], but it performs a smaller number of changes, as explained in the following. In the local search, the nodes corresponding to each service of a route are iteratively removed and inserted in new positions on the route, if the evaluation function is improved. More precisely, given a route, the first pickup node and its delivery node are removed from the route. Then, the pickup node, if the service is an inbound service, or the delivery node, if it is an outbound service are inserted in the best possible position according to the time window constraints. Observe that, while in the procedure proposed in [26] such node can be further moved during the search process, in our procedure the position of this node is not changed anymore. Afterwards, the

remaining node is inserted right after (if it is a pickup node) or right before (if it is a delivery node) the first assigned node. If this change improves the evaluation function, then the nodes are fixed in the new positions, otherwise, the procedure attempts to move the remaining node to the next possible position, increasing the distance from the first assigned node. This is repeated until the evaluation function is improved or no other possible positions exist for the remaining node. If no improvement is found the service is reinserted in the original position. Then, the procedure continues considering the nodes of the next service on the route. The local search ends when the procedure described above is applied to all the routes. During the VNS algorithm, in order to avoid unnecessary operations, only the routes that have been changed by the shaking procedure are involved in the local search.

5.5 Move or Not

At each iteration of the VNS algorithm, the new solution s'' created after the shaking and the local search procedures is compared with the best solution s. If $f(s'') < f(s)$ then s is replaced by s''. Furthermore, if s'' is feasible and better than s_{best}, s_{best} is replaced by s'', too.

5.6 Adjustment Procedure

This procedure tries to produce feasible solutions by perturbing quasi-feasible solutions. The procedure involves incumbent solutions s, found during the shaking procedure violating only a single type of constraint, i.e., capacity, ride time or time window, and in which the violation is below a given threshold. In the rest of the paper, the thresholds for the violations of the capacity, ride time and time window constraints are denoted as q_{max}, t_{max} and w_{max}, respectively. The adjustment procedure

chooses, one by one, services violating the constraint and assigns them either (i) at the beginning or at the end of other already used compatible routes, or (ii) to empty compatible routes, if any. The assignment is performed to the route minimizing the evaluation function. The possibly feasible solution obtained so far is then compared with s'_{best}.

6 Experimental results

In this section, experimental results on a large real-life instance and random instances, extracted by real data are presented. All tests have been performed on a PC equipped with Intel i7 processor and 64 Gb of RAM.

The real-life instance arises from data collected by the Health Care System of Tuscany, an Italian region, and includes all the transportation services performed in a day (of June 2013) by a Local Health Care Agency (ASL 11) and all the vehicles (distributed in several depots) available on that day. In the instance, there are 246 transportation services and 313 vehicles of different typologies, distributed over 17 depots and belonging to 29 non-profit organizations. Patients' preferences prescribe that 188 of the 246 transportation services must be transported only by vehicles of a given organization. In Table 1, the number of ambulances (AMB), buses (BUS), cars (CAR) and equipped vehicles (E-V) of each depot are reported. The service time to load or unload a patient is $st_i = 10$ minutes, and the maximum ride time for a transportation service $(i, n+i)$ is $T_i = 2.5 \times t_{i,n+i}$ (recall that $t_{i,n+i}$ is the traveling time from the pickup and delivery location of the transportation service).

From the real data described above, two sets of random instances have been extracted, called *large* and *small* instances, in which the features of the selected vehicles and requests have been maintained

(including patients' preferences). The large instance set contains 15 instances with the following characteristics: Instances 1–5 have 80 transportation services and 60 vehicles, instances 6–10 have 100 transportation services and 75 vehicles, and instances 11–15 have 100 transportation services and 80 vehicles. The small instance set contains 11 instances, in which the number of requests and vehicles are in the range 10–35 and 4–20, respectively. A route duration limit has been set in the small instances, in order to evaluate the VNS algorithm even with such a constraint. In large and small instances, more stringent conditions has been considered on service times and maximum ride times, i.e., $st_i = 15$ and $T_i = 2 \times t_{i,n+i}$ for all $i \in TR$. In all the instances, the width of each time window $(l_i - e_i)$ has been set to 40 minutes for nodes i in $P_{HOSP} \cup D_{HOSP} \cup P_{HOME}$, while $(e_i, l_i) = (0, 1440)$ for i in D_{HOME}.

Table 2 reports on the tariffs used to compute the waiting costs of the vehicles (see Section 3.1), i.e., $\sum_{p=1}^{t} ca_p$ and $\sum_{p=1}^{t} cb_p$, for each time interval (Δ_t, Δ_{t+1}) (in minutes). Observe that, no cost occurs if the waiting time of a vehicle is smaller than or equal to 60 minutes at a node $i \in P_{HOSP} \cup D_{HOSP}$.

This section is organized as follows. In Section 6.1, preliminary results are presented to allow a comparison between the TS and VNS approaches and the different versions of the VNS algorithm. In fact, in order to assess the effectiveness of the new neighborhoods, the adjusting procedure and the procedure for minimizing the costs of the vehicle waiting times (Algorithm 5), five VNS algorithms have been developed whose features will be described in Section 6.1.

Section 6.2 reports the results of the best algorithms detected in Section 6.1, both on the real-life instance and on the small instances. On these last instances, a comparison with the solutions provided by CPLEX is given, running on the formulation presented in Section 3.2.

depot	AMB	BUS	CAR	E-V
1	16	2	22	10
2	5	1	3	1
3	9	2	14	5
4	2	0	3	0
5	9	1	9	4
6	3	2	3	2
7	5	0	4	2
8	5	1	3	2
9	3	0	2	1
10	3	1	2	1
11	1	0	3	3
12	7	2	10	2
13	12	4	18	4
14	3	0	4	0
15	4	2	10	1
16	14	1	12	3
17	3	0	3	0
Total	113	21	137	42

Table 1: Vehicles' distribution over the depots.

t	(Δ_t, Δ_{t+1})	**Tariff of type** a: $\sum_{p=1}^{t} ca_p$	**Tariff of type** b: $\sum_{p=1}^{t} cb_p$
1	$(0,60)$	0	0
2	$(61,120)$	13.04	9.77
3	$(121,180)$	39.05	19.54
4	$(181,240)$	78.11	29.29
5	$(241,1000)$	130.20	39.50

Table 2: Tariffs for the waiting of the vehicles (in Euros).

6.1 Preliminary results

A preliminary test phase has been performed in order to compare the performances of TS and VNS and the effectiveness of the new neighborhoods and procedures designed for VNS. In this phase, the algorithms have been tested on the large instances. In the preliminary test phase, a first analysis has been performed to compare the neighborhood strength of TS and VNS algorithms. Then, different VNS algorithms are compared to assess the effectiveness of the new neighborhoods and procedures.

Table 3 describes the main features of the VNS algorithms that have been developed and tested. Observe that, algorithms VNS1 and VNS2 only contain the shaking phase, but with different neighborhoods. In VNS1 (in VNS2), the neighborhoods swap, move and chain (swap, repairing move, move, chain and eliminate) have been considered. In algorithm VNS3, the local search and the evaluation scheme introduced in [16] have been enabled, and the shaking phase contains the neighborhoods swap, move and chain. Finally, algorithms VNS4 and VNS5 contain all the neighborhoods and the new proposed procedures (observe that, in VNS4, the adjusting procedure is disabled).

First of all, a tuning phase has been performed for TS and VNS to set the parameters of the algorithms. At this aim, instances of the large instance set have been used. In the tuning phase of TS, $IT_{max} = 250$ has been set and the following parameters have been varied: α_0, β_0 and γ_0, tabu list length θ and size of the neighborhood $|N(s_{curr})|$. For VNS, we have set $IT_{max} = 100000$ and α_0, β_0 and γ_0 have been varied. The output of the tuning phase gives the following parameter configurations: $|N(s_{curr})| = 150$, $\theta = 100$, $\alpha_0 = 1$, $\beta_0 = 1000$ and $\gamma_0 = 10$, for TS, and $\alpha_0 = 100$, $\beta_0 = 1$, $\gamma_0 = 10000$ for VNS.

Table 4 presents results for TS, VNS1 and VNS2 where the strength of the neighborhoods has

Algorithm	Neighborhoods	Local Search	Eval. scheme (Algorithm 4)	Wait. min. (Algorithm 5)	Adjusting
VNS1	S, M, C	-	-	-	-
VNS2	S, RM, M, C, E	-	-	-	-
VNS3	S, M, C	X	X	-	-
VNS4	S, RM, M, C, E	X	X	X	-
VNS5	S, RM, M, C, E	X	X	X	X

Table 3: VNS algorithms.

been evaluated. Table 5 presents a comparison among the algorithms VNS3, VNS4 and VNS5. On all these experiments, the TS algorithm has been executed with $IT_{max} = 3500$. In all the versions of VNS, $IT_{max} = 250000$ has been set, and, in VNS5, $q_{max} = 3$, $t_{max} = 40$ and $w_{max} = 100$ have been used in the adjusting procedure. In Tables 4 and 5, for each instance, the average results over five runs of the algorithms are reported. The last row of the two tables reports on the average values computed over all the instances. In the tables, the second and third column report on the number of services and vehicles in each instance, respectively. For each algorithm, **obj** is the objective function value, **cost** is the monetary cost of the solution, **wt** is the average patient waiting times (minutes), **v** is the number of used vehicles, **t** is the computational time in seconds and **best** is the best value of the solution found over the five runs.

Observe that, in Table 4, VNS1 always finds better solutions than TS in a smaller amount of time. When comparing VNS1 and VNS2, VNS2 mainly attains the best results (except on instances 2, 4 and 6) in terms of objective function value (**obj**) and patient waiting times (**wt**). On the other hand, VNS2 requires more computational time (about 176 and 130 seconds on average for VNS2 and VNS1, respectively) and a slightly bigger number of vehicles (about 59 and 58 on average for VNS2 and VNS1, respectively), mainly due to the eliminate neighborhood, computationally expensive, that

			TS						VNS1						VNS2					
Inst.	*n*	*m*	**obj**	**cost**	**wt**	**v**	**t**	**best**	**obj**	**cost**	**wt**	**v**	**t**	**best**	**obj**	**cost**	**wt**	**v**	**t**	**best**
1	80	60	3963.85	3097.05	10.84	49.00	214.49	3889.91	3745.71	3026.51	8.60	48.60	99.60	3670.41	3660.46	3049.46	7.64	49.60	134.13	3636.55
2	80	60	4694.33	3811.33	11.04	48.20	193.63	4504.50	4231.11	3430.91	9.60	48.40	98.00	4187.88	4282.89	3524.29	9.48	49.20	135.14	4168.71
3	80	60	4103.36	3151.56	11.90	49.80	250.12	4067.39	3831.49	3037.69	9.20	49.60	102.11	3801.00	3733.76	3078.56	8.19	51.00	139.54	3694.69
4	80	60	3928.60	3101.20	10.34	52.40	251.63	3910.26	3784.60	3008.80	9.20	52.40	92.25	3748.67	3969.05	3200.65	9.61	53.60	129.17	3709.22
5	80	60	4270.17	3334.57	11.70	49.40	290.37	4254.10	4056.12	3221.72	10.00	51.00	96.26	3992.95	3902.91	3193.51	8.87	51.60	132.89	3795.64
6	80	60	5062.45	4120.05	9.42	57.40	263.55	4988.80	4719.14	3874.74	7.80	60.00	140.83	4647.12	4762.20	3931.20	8.31	60.20	201.58	4644.72
7	100	75	5565.04	4514.04	10.51	60.00	276.24	5455.13	5363.54	4370.34	9.40	61.20	142.91	5229.55	5204.00	4327.60	8.76	62.20	190.85	5120.99
8	100	75	5901.62	4808.22	10.93	65.00	254.65	5433.11	5060.93	4113.13	9.00	60.80	140.96	4958.29	4950.27	4100.87	8.49	61.80	193.85	4856.04
9	100	75	5266.61	4235.21	10.31	58.80	254.74	5158.08	5007.81	4010.61	9.40	60.00	142.41	4915.02	4990.29	4077.29	9.13	60.40	191.41	4693.44
10	100	75	5380.42	4303.02	10.77	63.00	275.09	5276.21	5113.40	4086.20	9.60	63.20	143.63	5031.06	4976.69	4088.49	8.88	62.80	193.28	4935.74
11	100	75	5640.96	4564.96	10.76	60.20	299.17	5532.07	5406.07	4405.87	9.60	64.00	163.77	5362.86	5377.94	4390.54	9.87	65.60	196.09	5313.82
12	100	80	6007.86	4775.86	12.32	60.20	268.24	5685.88	5282.72	4179.92	10.60	61.00	148.51	5218.09	5192.97	4172.17	10.21	63.20	209.06	5030.40
13	100	80	5502.87	4352.47	11.50	62.80	320.41	5453.88	5196.79	4223.79	9.00	64.00	144.54	5130.97	5074.07	4198.27	8.76	63.80	200.66	4869.94
14	100	80	5427.83	4242.63	11.85	64.00	301.68	5313.09	5015.81	3986.21	9.80	63.00	142.57	4934.23	5109.22	4138.22	9.71	64.20	198.47	4864.73
15	100	80	5470.87	4362.87	11.08	61.60	314.51	5389.47	5193.66	4210.46	9.40	61.80	148.42	5069.17	5068.83	4176.63	8.92	63.80	204.10	5043.10
Av.			5079.12	4051.67	11.02	57.45	268.57	4954.13	4733.93	3812.46	9.35	57.93	129.79	4659.82	4683.70	3843.18	8.99	58.87	176.68	4558.52

Table 4: Results on large random instances: comparison of TS, VNS1 and VNS2.

			VNS3						VNS4						VNS5					
Inst.	*n*	*m*	**obj**	**cost**	**wt**	**v**	**t**	**best**	**obj**	**cost**	**wt**	**v**	**t**	**best**	**obj**	**cost**	**wt**	**v**	**t**	**best**
1	80	60	3695.35	3226.15	5.87	42.60	208.63	3606.40	3592.16	3194.96	4.97	41.6	264.38	3541.62	3552.99	3170.39	4.78	42.80	277.84	3489.34
2	80	60	4000.95	3521.15	6.00	44.00	202.63	3952.35	3992.43	3563.63	5.36	44.2	254.11	3925.24	3977.95	3557.95	5.25	45.20	262.01	3925.24
3	80	60	3422.08	3060.68	4.52	44.40	209.36	3406.11	3479.65	3113.25	4.58	44.8	263.30	3436.35	3466.03	3105.43	4.51	45.00	272.81	3427.51
4	80	60	3687.78	3185.38	6.28	44.60	205.72	3653.90	3678.68	3194.08	6.06	44.4	255.38	3599.62	3635.77	3167.77	5.85	45.00	265.21	3579.07
5	80	60	3671.98	3315.18	4.46	45.60	209.58	3637.12	3752.87	3364.87	4.85	45.6	254.61	3623.37	3709.97	3347.97	4.53	46.80	267.60	3623.37
6	100	75	4591.89	4082.89	5.09	53.20	302.15	4457.04	4592.02	4055.22	5.37	54.4	393.09	4468.37	4564.76	4057.16	5.08	55.60	430.55	4437.35
7	100	75	5090.93	4579.73	5.11	51.80	310.39	4910.11	5025.51	4542.31	4.83	51.6	389.38	4923.48	4998.91	4530.71	4.68	51.80	431.03	4923.48
8	100	75	4705.60	4223.60	4.82	50.20	302.88	4607.60	4685.30	4183.10	5.02	49.8	390.57	4606.38	4665.60	4172.60	4.93	50.20	429.49	4569.48
9	100	75	4528.93	4051.33	4.78	54.40	302.47	4481.97	4531.11	4057.71	4.73	55	373.69	4407.08	4525.87	4062.07	4.64	55.60	415.69	4407.08
10	100	75	4636.24	4121.24	5.15	56.00	297.51	4546.40	4576.17	4125.77	4.50	54.6	401.72	4521.63	4547.17	4116.57	4.31	55.40	448.72	4479.62
11	100	80	4954.40	4492.20	4.62	53.00	314.69	4879.77	4926.67	4492.67	4.34	51.8	401.07	4886.76	4905.57	4486.57	4.19	52.20	448.30	4877.47
12	100	80	4843.07	4301.07	5.42	57.80	308.38	4761.22	4805.18	4332.98	4.72	57	385.02	4736.37	4797.66	4331.26	4.66	57.20	415.61	4736.37
13	100	80	4684.54	4280.34	4.04	54.00	312.04	4640.35	4656.95	4190.75	4.66	54.6	392.03	4585.83	4592.91	4205.11	3.88	55.00	418.28	4554.63
14	100	80	4478.62	4035.62	4.43	59.00	302.36	4375.20	4443.69	4009.09	4.35	59	383.23	4398.5	4414.90	4025.70	3.89	59.20	410.99	4383.77
15	100	80	4791.58	4361.98	4.30	51.80	311.61	4701.41	4802.51	4325.71	4.77	53.6	394.25	4761.08	4781.53	4314.73	4.67	54.20	425.92	4721.35
Av			4385.60	3922.57	4.99	50.83	273.36	4307.80	4369.39	3916.41	4.87	50.8	346.39	4294.78	4342.51	3910.13	4.66	51.41	374.67	4275.68

Table 5: Results on large random instances: comparison of VNS3, VNS4 and VNS5.

attempts to use empty vehicles.

The results of Table 5 show that the new features designed for VNS (i.e., the new neighborhoods, the adjusting procedure and vehicle waiting minimization) are quite effective with respect to the objective function value and the patient waiting. In fact, on average, the objective function values of VNS3, VNS4 and VNS5 are 4385.6, 4369.4 and 4342.5, respectively, while the average waiting times of the patients are 4.99, 4.87 and 4.66, respectively. VNS3 produces solutions with objective function values better than VNS5 on two instances out of 15 only (i.e., instances 3 and 5). With respect to the best objective function value (**best**), VNS5 outperforms VNS3 in 13 instances and VNS4 in 10 instances (in the remaining 5 instances, VNS4 and VNS5 attain the same results). The computational times increase about 100 seconds on average from VNS3 to VNS5. On average, the number of used vehicles is slightly bigger in VNS5, since both the neighborhood eliminate and the adjusting procedure attempt to employ empty vehicles, if needed.

In summary, among the algorithms, VNS5 generally has the best behavior in terms of objective function values. It dominates VNS4 on all the large instances (recall that VNS4 corresponds to VNS5 in which the adjusting procedure is disabled), while it is outperformed by VNS3 on two instances. In the next section, results for VNS3 and VNS5 on the real-life instance and on the small instance set are presented.

6.2 Results on the real-life instance and comparison with state-of-the-art MILP solver

In this section, the computational results on the real-life instance and a comparison on the small instances with a state-of-the-art MILP solver are reported. As previously stated, the real-life instance contains 246 transportation services and 313 vehicles distributed over 17 depots. Patient preferences impose that 188 of the 246 transportation requests must be transported only by vehicles of a given organization. We point out that in the real case, as prescribed by the preference constraints, the Local Healthcare Agency ASL 11 directly managed 58 transportation services out of 246, dispatching them to the non-profit organizations. The other 188 transportation services where directly managed by the non-profit organizations. Hence, on the basis of the patients assigned by the preference constraints and by the healthcare agency, the non-profit organizations routed their vehicles in order to accomplish all the assigned transportation services. The decision process, involving the dispatching of the transportation services to the non-profit organizations and the assignment to the vehicles and the routing, was performed by hand, both at the central level (ASL 11) and at the non-profit organization level. Hence, in the real-life instance, 58 transportation services can be assigned to any compatible vehicle, while 188 transportation services can be only assigned to compatible vehicles belonging to the preferred non-profit organizations.

Table 6 reports the results attained by the algorithms VNS3 and VNS5 on the real-life instance in comparison with the actual solution really implemented by ASL 11 and by the non-profit organizations (denoted as *Real case* and reported in the second row of the table). Observe that, in the Real case, the routing of all the transportation services required 140 vehicles (column 5 of the table) and the total

	obj	cost	wt	v	t	best	best cost
Real case	-	10276.1	-	140	-	-	-
VNS3	9677.19	9299.29	6.14	129.2	9690.07	9610.13	9244.38
VNS5	9596.6	9219.95	6.12	130.2	12885.9	9504.18	9128.93

Table 6: Results on the real-life instance.

(monetary) cost paid by ASL 11 was 10276.1 Euros (column 3 of the table). On the real-life instance, the VNS algorithms have been executed maintaining all the preference constraints of the patients, setting $\psi = 0.25$ in the evaluation function and using the following parameters: $IT_{max} = 1000000$, $q_{max} = 3$, $t_{max} = 40$, $w_{max} = 100$. In Table 6, for VNS3 and VNS5 (in rows 3 and 4 respectively), columns 2–6 report the average results over five runs, while columns 7 and 8 respectively report the best objective function value and the best monetary cost attained over the 5 runs. Observe that, VNS3 and VNS5 respectively produce solutions with monetary costs lower than the one obtained in the Real case of about 976 and 1056 Euros on average respectively (see column **cost**). Considering the **best cost** (column 8), the monetary costs of VNS3 and VNS5 are lower than the Real case of about 1031 and 1147 Euros, respectively. Furthermore, VNS3 and VNS5 employ on average a smaller number of vehicles (about 130 vs. 140 of the Real case). Comparing VNS3 and VNS5, we observe that VNS5 attains the best results with respect to average and best objective function values and costs. In both algorithms, patient waiting times and the number of used vehicles are almost the same (VNS5 employs one vehicle more than VNS3 on average, but, as already stated, this is mainly due to the neighborhood eliminate and the adjusting procedure).

In Table 7, a comparison of algorithms VNS3 and VNS5 with the solutions provided by CPLEX

12.5, running on the formulation of Section 3.2 is presented on the small instances. (In fact, no feasible solution was found by CPLEX on the large instances with a time limit of 24 hours.) In order to evaluate VNS3 and VNS5 even with a limit on the duration of the routes, in the small instances a temporal limit on the route duration, RD, has been inserted ($RD = 360$ minutes in all the instances except for instance $I10-4$ in which $RD = 370$). A constraint on the route duration of the following form

$$A^p_{dep(p)} - B^p_{dep(p)} \leq RD \qquad \forall p \in M_k \;\; k \in K$$

has been inserted in the MILP formulation, too. On these instances, VNS3, VNS5 and CPLEX have been executed setting $\psi = 0$ (i.e., the objective function coincides with the monetary cost). The following parameters have been used in VNS3 and VNS5: $IT_{max} = 30000$, $q_{max} = 2$, $t_{max} = 20$, $w_{max} = 20$, and $\tau_0 = 1$. In Table 7, the second and the third column respectively report on the number of transportation services and vehicles of each instance. Columns 4 (7) and 5 (8) respectively report the best and the average objective function values of the solutions found by VNS3 (VNS5) over 5 runs of the algorithms. Columns 6 and 9 show the average computational time. Column 10 reports the cost of the solution attained by CPLEX with a time limit of 1 hour (a * indicates that the solution found by Cplex is optimal). In the table, the entries in bold highlight the best results. Observe that, VNS5 finds the optimal solutions nearly in all the instances in which Cplex solves the instance at the optimality (3 instances out of 4), and outperforms Cplex in all the other instances. VNS3 finds the optimal solutions in 3 out of 4 instances, too. Furthermore note that, Cplex is not able to find any feasible solution within the time limit for the instance $I35-20$. The last row of the table reports average results for VNS3 and VNS5. Observe that, on average VNS5 obtains better results with respect to both average and the best objective function values, with a small increase of the computational time

			VNS3			VNS5			CPLEX
Inst.	n	m	**best**	**obj**	**t**	**best**	**obj**	**t**	**obj**
$I10-4$	10	4	404.96	421.25	2.16	**395.19**	412.53	2.40	**395.19***
$I15-6$	15	6	**786.71**	789.77	4.48	**786.71**	786.71	4.64	**786.71***
$I20-14$	20	14	**1026.66**	1046.27	3.21	**1026.66**	1043.630	3.92	1034.92
$I20-15$	20	15	**769.29**	776.16	3.45	**769.29**	773.93	3.69	**769.29***
$I22-16$	22	16	1055.24	1063.14	3.50	**1048.88**	1062.73	4.36	1075.88
$I25-18$	25	18	1063.02	1074.15	4.05	**1016.51**	1059.06	4.59	1040.37
$I25-18b$	25	18	**951.40**	960.35	4.11	952.54	956.66	4.71	**951.40***
$I30-20$	30	20	1206.92	1270.82	4.89	**1197.58**	1205.59	5.82	1257.28
$I30-20b$	30	20	1197.40	1205.68	5.00	**1190.06**	1203.85	5.50	1252.92
$I35-20$	35	20	1249.21	1345.96	6.14	**1249.17**	1265.89	7.12	-
$I35-20b$	35	20	1361.16	1376.04	5.81	**1357.83**	1369.95	6.62	1415.99
Av			1006.54	1029.96	4.25	999.13	1012.37	4.85	

Table 7: Comparison between VNS5 and CPLEX on the small instances. (A * indicates that the solution found by Cplex is optimal.)

(about 0.6 seconds on average). On the single instances, see columns 4 and 7 of the table, VNS5 finds the optimal or the best solution in 10 instances out 11, while VNS3 attains the optimal or the best results in 4 instances. On average (columns 5 and 8), VNS5 always produces better results than VNS3.

7 Conclusion

In this work, a DARP problem arising from a real-world healthcare application has been addressed. A Tabu Search algorithm and different versions of Variable Neighborhood Search algorithm are proposed able to tackle all the characteristics of the problem. Computational results on large real-life and

random instances show the effectiveness of the proposed approach, both in terms of costs and quality of service, as certified by the management of ASL 11 that provided the data.

Future research directions include the study of the integrated problem of finding a visit timetable that minimizes the overall routing costs. In this case, the problem consists in determining a timetable of the healthcare services of the patients, in such a way that the transportation costs are minimized, always respecting service quality constraints.

Acknowledgements

The authors are grateful to Dr. Renato Colombai and Dr. Gabriele Marconcini, from the Local Health Care Agency (ASL 11) of Empoli, for their collaboration to the problem definition and to provide the data used in the experimentation. The research has been partially supported by the grant estione delle risorse critiche in ambito ospedalieroritical resource management in hospitals of the Regione Toscana - PAR FAS 2007-2013 1.1.a.3.- B51J10001140002.

The authors thank the Associate Editor and the two anonymous Reviewers for their valuable suggestions and comments.

References

[1] http://www.misericordiaonline.org/servizi/contratti/page/AQRToscana.pdf

[2] A. Agnetis, G. De Pascale, P. Detti, J. Raffaelli, P. Chelli, R. Colombai, G. Marconcini, E. Porfido, and A. Coppi, Applicazione di tecniche di operations management per minimizzare il costo

di trasporto di pazienti, MECOSAN, Italian Quart. of Health Care Management Economics and Politics, 84, 2012.

[3] O.I. Alsalloum, G.K. Rand, Extensions to emergency vehicle location models, Comput. Oper. Res., 33 (9), 2725–2743, 2006.

[4] A. Attanasio, J.-F. Cordeau, G. Ghiani, G. Laporte, Parallel tabu search heuristics for the dynamic multi-vehicle dial-a-ride problem. Parallel Computing 30, 377–387, 2004.

[5] J. F. Bard, A. I. Jarrah, Integrating commercial and residential pickup and delivery networks: A case study, Omega, 41 (4), 706–720, 2013.

[6] A. Beaudry, G. Laporte, T. Melo, S. Nickel, Dynamic transportation of patients in hospitals, OR Spectrum, 32 (1), 77–107, 2010.

[7] G. Berbeglia, J.-F. Cordeau, I. Gribkovskaia, G. Laporte, Static pickup and delivery problems: a classification scheme and survey. TOP, 15, 1–31, 2007.

[8] A. Bettinelli, A. Ceselli, and G. Righini, *A branch-and-cut-and-price algorithm for the multi-depot heterogeneous vehicle routing problem with time windows*, Transportation Research Part C, 19, 723–740, 2011.

[9] J. Bowers, B. Lyons, G. Mould, *Developing a resource allocation model for the Scottish patient transport service*, Operations Research for Health Care, 1, 4, 84–94, 2012.

[10] K. Braekers, A. Caris, G.K. Janssens, Exact and meta-heuristic approach for a general heterogeneous dial-a-ride problem with multiple depots. Transportation Research Part B, 67, the6–186, 2014.

[11] L. Brotcorne, G. Laporte, F. Semet, Ambulance location and relocation models, European J. Oper. Res., 147 (3), 451–463, 2003.

[12] T.A. Carnes, S.G. Henderson, D.B. Shmoys, M. Ahghari, R.D. MacDonald, Mathematical programming guides air-ambulance routing at ornge. Interfaces, 43 (3), 232–239, 2013.

[13] J. C. N. Climaco, J. M. F. Craveirinha, and M. M. B. Pascoal. An automated reference point-like approach for multicriteria shortest path problems. Journal of Systems Science and Systems Engineering, 15, 314–329, 2006.

[14] A. Coppi, P. Detti, J. Raffaelli, A planning and routing model for patient transportation in healthcare, Electronic Notes in Discrete Mathematics, 41, 125–132, 2013.

[15] J.-F. Cordeau, A Branch-and-Cut Algorithm for the Dial-a-Ride Problem. Operations Research, 54(3), 573–586, 2006.

[16] J.-F. Cordeau and G. Laporte, A tabu search heuristic for the static multi-vehicle dial-a-ride problem. Transportation Research Part B, 37, 579–594, 2003.

[17] J.-F. Cordeau and G. Laporte. The dial-a-ride problem: models and algorithms. Annals of Operations Research, 153:29–46, 2007.

[18] P. Detti, F. Papalini, G. Zabalo Manrique de Lara, A multi-depot dial-a-ride problem for patients transportation in healthcare, Proceedings of the 11th Metaheuristic International Conference (MIC), Agadir, 7–10 June, 2015.

[19] G. Erdogan, E. Erkut, A. Ingolfsson, G. Laporte, Scheduling ambulance crews for maximum coverage, J. Oper. Res. Soc. 61 (4), 543–550, 2010.

[20] T. Hanne, T. Melo, S. Nickel, Bringing robustness to patient flow management through optimized patient transports in hospitals, Interfaces 39 (3), 241–255, 2009.

[21] R. Liu, X. Xie, T. Garaix, Hybridization of tabu search with feasible and infeasible local searches for periodic home health care logistics, Omega, 47, 17–32, 2014.

[22] H. Ma, B. Cheang, A. Lim, L. Zhang, Y. Zhu, An investigation into the vehicle routing problem with time windows and link capacity constraints, Omega, 40 (3), 336–347, 2012.

[23] E. Melachrinoudis, A. B. Ilhana, and H. Min, A dial-a-ride problem for client transportation in a health-care organization, Computers & Operations Research, 34, 742–759, 2007.

[24] S. Muelas, A. LaTorre, J.-M. Pena, A variable neighborhood search algorithm for the optimization of a dial-a-ride problem in a large city, Expert Systems with Applications, 40, 14, 5516–5531, 2013.

[25] J. Paquette, J.-F. Cordeau, G. Laporte, M.M.B. Pascoal, Combining multicriteria analysis and tabu search for dial-a-rial problems.Transportation Research B, 46, 100–119, 2012.

[26] S. N. Parragh, K. F. Doerner, and R. F. Hartl, Variable neighborhood search for the dial-a-ride problem, Computers & Operations Research, 37, 6, 1129–1138, 2010.

[27] S. N. Parragh, J. F. Cordeau, K. F. Doerner, and R. F. Hartl, Models and algorithms for the heterogeneous dial-a-ride problem with driver-related constraints. OR Spectrum 34, 3 , 593–633, 2012.

[28] S. N. Parragh and V. Schmid, Hybrid column generation and large neighborhood search for the dial-a-ride problem, Computers & Operations Research, 40, 1, 490–497, 2013.

[29] B. Rekiek, A. Delchambre, H.A. Saleh, Handicapped person transportation: an application of the grouping genetic algorithm, Engineering Applications of Artificial Intelligence, 19:511–520, 2006.

[30] S. Ropke and J.F. Cordeau, Branch and Cut and Price for the Pickup and Delivery Problem with Time Windows, Transportation Science, 43, 267–286, 2009.

[31] P. Toth, D. Vigo, Heuristic algorithms for the handicapped persons transportation problem, Transportation Science, 3, 60–71, 1997.

[32] M. Savelsbergh and M. Sol, DRIVE: Dynamic Routing of Independent Vehicles, Operations Research, 46, 474–490, 1998.

[33] Z. Zhang, M. Liu, A. Lim, A memetic algorithm for the patient transportation problem, Omega, 54, 60–71, 2015.