Computers & Operations Research 38 (2011) 1319–1334

Contents lists available at ScienceDirect

Computers & Operations Research

journal homepage: www.elsevier.com/locate/caor

A GRASP with evolutionary path relinking for the truck and trailer routing problem

Juan G. Villegas [a,b,c,*], Christian Prins [a], Caroline Prodhon [a], Andrés L. Medaglia [b], Nubia Velasco [b]

[a] Laboratoire d'Optimisation des Systèmes Industriels (LOSI), Institut Charles Delaunay, Université de Technologie de Troyes, BP 2060, 10010 Troyes Cedex, France
[b] Centro para la Optimización y Probabilidad Aplicada (COPA), Departamento de Ingeniería Industrial, Universidad de los Andes, A.A. 4976, Bogotá D.C., Colombia
[c] Departamento de Ingeniería Industrial, Universidad de Antioquia, Calle 67 No. 53 - 108, A.A. 1226, Medellin, Colombia

ARTICLE INFO

Available online 30 November 2010

Keywords:
Truck and trailer routing problem (TTRP)
Greedy randomized adaptive search procedures (GRASP)
Variable neighborhood search (VNS)
Path relinking
Vehicle routing problem

ABSTRACT

In the truck and trailer routing problem (TTRP) a heterogeneous fleet composed of trucks and trailers has to serve a set of customers, some only accessible by truck and others accessible with a truck pulling a trailer. This problem is solved using a route-first, cluster-second procedure embedded within a hybrid metaheuristic based on a greedy randomized adaptive search procedure (GRASP), a variable neighborhood search (VNS) and a path relinking (PR). We test PR as a post-optimization procedure, as an intensification mechanism, and within evolutionary path relinking (EvPR). Numerical experiments show that all the variants of the proposed GRASP with path relinking outperform all previously published methods. Remarkably, GRASP with EvPR obtains average gaps to best-known solutions of less than 1% and provides several new best solutions.

1. Introduction

In the truck and trailer routing problem [7] a heterogeneous fleet composed of m_t trucks and m_r trailers ($m_r < m_t$) serves a set of customers $N = \{1,\dots,n\}$ from a main depot, denoted by 0. Each customer $i \in N$ has a non-negative demand q_i; the capacities of the trucks and the trailers are Q_t and Q_r, respectively; and the distance c_{ij} between any two points $i,j \in N \cup \{0\}$ $(i \neq j)$ is known. Some customers with limited maneuvering space or accessible through narrow roads must be served only by a truck, while other customers can be served either by a truck or by a *complete vehicle* (i.e., a truck pulling a trailer). These incompatibility constraints create a partition of N into two subsets: the subset of *truck customers* N_t accessible only by truck; and the subset of *vehicle customers* N_v accessible either by truck or by a complete vehicle. The objective of the TTRP is to find a set of routes of minimum total distance such that: each customer is visited in a route performed by a compatible vehicle; the total demand of the customers visited in a route does not exceed the capacity of the allocated vehicle; and the number of required trucks and trailers is not greater than m_t and m_r, respectively. Being an extension of the well-known vehicle routing problem (VRP), the TTRP is NP-Hard. For updated reviews of the VRP and its extensions the reader is referred to the books by Toth and Vigo [53] and Golden et al. [24], and the introductory tutorial by Laporte [30].

A solution of the TTRP may have three types of routes: *pure truck routes* performed by a truck visiting customers in N_v and N_t; *pure vehicle routes* performed by a complete vehicle serving only customers in N_v; and finally *vehicle routes with subtours*. The latter are composed of a main tour performed by the complete vehicle visiting only customers in N_v, and one or more subtours, in which the trailer is detached at a vehicle customer location, to visit (with the truck) one or more customers in N_t and probably some customers in N_v. The parking place of the trailer is called the root of the subtour.

Fig. 1 depicts a solution of the TTRP with $m_t = 4$, $m_r = 3$, $Q_t = 6$, and $Q_r = 10$. Solid lines represent segments traversed by a complete vehicle and dashed lines represent segments traversed by a truck. Fig. 1 illustrates some of the special features of the TTRP: (i) only the vehicle customers visited in the main tour can be used as roots in vehicle routes with subtours; (ii) the total demand of the customers of a subtour must not exceed Q_t; (iii) the total demand of the customers of all subtours visited on a vehicle route with subtours may exceed Q_t (but not Q_t+Q_r), because at the root of a subtour it is possible to transfer goods between the truck and the trailer; (iv) several subtours may have the same root; (v) the first customer of a vehicle route with subtours cannot be a truck customer, because in that case the trailer would have been detached at the main depot due to the accessibility constraint of the first customer, giving rise to a pure truck route.

* Corresponding author at: Departamento de Ingeniería Industrial, Universidad de Antioquia, Calle 67 No. 53 - 108, A.A. 1226, Medellín, Colombia. Tel: +574 2195575.
E-mail addresses: jvillega@udea.edu.co (J.G. Villegas), christian.prins@utt.fr (C. Prins), caroline.prodhon@utt.fr (C. Prodhon), amedagli@uniandes.edu.co (A.L. Medaglia), nvelasco@uniandes.edu.co (N. Velasco).

0305-0548/$ - see front matter © 2010 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cor.2010.11.011

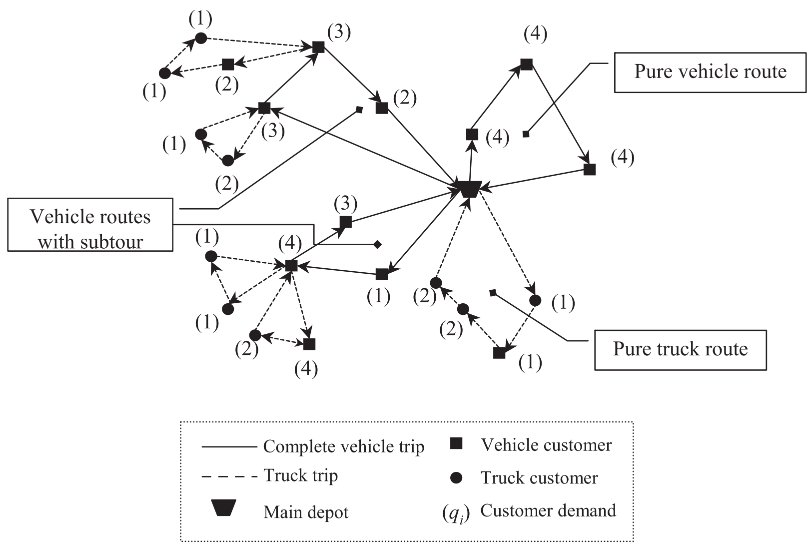

Fig. 1. Feasible solution for the TTRP.

For the solution of the TTRP we present a hybrid metaheuristic based on a greedy randomized adaptive search procedure (GRASP), a variable neighborhood search (VNS) and a path relinking (PR). The remainder of this paper is organized as follows. Section 2 motivates the TTRP with some practical applications and presents the relevant literature on the TTRP and other related problems. Section 3 describes the hybrid metaheuristic and its components. Section 4 presents a computational evaluation of different variants of the hybrid metaheuristic on a set of publicly available test problems and their comparison against other methods from the literature. Finally, Section 5 presents some conclusions. Appendix A summarizes the notation used throughout the paper.

2. Literature review

Practical applications of the TTRP appear mainly in collection and delivery operations in rural areas or crowded cities with accessibility constraints. Semet and Taillard [48] used a tabu search to solve a TTRP with time windows, site dependencies and heterogeneous fleet arising in the distribution operations of a chain of grocery stores in Switzerland. Gerdessen [21] described two possible applications of the TTRP. The first one arises in the distribution of dairy products in the Netherlands, where the use of trucks with trailers is common. However, customers located in crowded cities cannot be served by the complete vehicle, thus the trailers must be left in parking lots before reaching these customers. The second application is related with the distribution of compound animal feed in rural regions, customers reachable through narrow roads or bridges must be served by the truck after leaving the trailer parked in a proper place.

Milk collection is another known practical application of the TTRP. Hoff and Løkketangen [28] presented a tabu search algorithm for the solution of a routing problem for milk collection in Norway. They modeled the milk collection using a multi-depot TTRP variant, where N_v is empty and the parking places for the trailers are not associated with customer locations. Likewise, Caramia and Guerriero [6] used the Heterogeneous Milk Collection with Heterogeneous Fleet (HMCHF) problem to model and optimize the milk collection of an Italian dairy company. The HMCHF problem can be seen as a TTRP variant with multi-compartments, route-length constraints, and heterogeneous trucks and trailers.

Chao [7] introduced the TTRP and also proposed a tabu search metaheuristic based on a cluster-first, route-second approach. The clustering phase solves a relaxed generalized assignment problem (RGAP) to allocate customers to routes. The RGAP is solved by rounding the solution of its linear programming relaxation; this rounding may produce infeasible solutions with overcapacity utilization. The second phase uses a cheapest insertion heuristic to sequence the customers within each route. The insertion heuristic treats pure vehicle routes and pure truck routes as classical traveling salesman problems (TSPs); on the other hand, when constructing vehicle routes with subtours the insertion procedure takes into account the accessibility constraints. In a third step a multiple-neighborhood improvement procedure with a penalized objective function is used to repair infeasible solutions and improve feasible ones. The neighborhoods are reallocations and exchanges of customers between routes, a specialized neighborhood that changes the roots of the subtours, and a 2-opt [18] refinement for every route and subtour. The fourth and final phase is a hybrid tabu search/deterministic annealing method that reuses some of the neighborhoods of the previous phase, and implements a tabu restriction that forbids moves that increase the objective function over a certain threshold. The tabu search has one diversification stage and one intensification stage, executed in sequence. The search is restarted several times from the best solution found so far.

In the same vein, Scheuerer [45] proposed two constructive methods and a tabu search for the TTRP. The first constructive heuristic, called T-Cluster, is a cluster-based insertion heuristic that constructs routes sequentially. The insertion of each customer into a route is followed by a steepest descent improvement procedure with three neighborhoods: a root refining approach (for vehicle routes with subtours [7]), 2-opt [18] and Or-opt [33]. The second constructive heuristic, called T-Sweep is an adaptation for the TTRP of the sweep heuristic of Gillett and Miller [22], followed by the same steepest descent procedure. In both methods, it is possible to produce infeasible solutions, because capacity violation of the last route is allowed when there are unrouted customers and no more vehicles available. Starting from the solution obtained with any of the above constructive heuristics, the tabu search method explores the neighborhood generated using reallocations and exchanges of subsets of customers between routes and subtours, and the procedure that changes the roots of vehicle routes with subtours [7]. After the acceptance of a solution, 2-opt and Or-opt procedures improve each modified tour. The search explores infeasible solutions using a penalized objective function and strategic oscillation following the approach of Cordeau et al. [9].

Neighborhood reduction strategies are used to speed-up the evaluation of moves. The search is restarted using the best solution found so far as an intensification mechanism.

Lin et al. [32] developed a very effective simulated annealing (SA) for the TTRP. Their SA uses an indirect representation of the solutions using a permutation of the customers with additional dummy zeros to separate routes and terminate subtours, along with a vector of binary variables of length $|N_v|$, representing the type of vehicle used to serve each vehicle customer (0 for a complete vehicle, and 1 for a truck). A specialized procedure decodes the permutation into a TTRP solution using the information of the binary vector. Since the decoding procedure may fail to find feasible solutions with respect to the availability of trucks and trailers, a route combination approach is used to reduce the number of required trucks and trailers, and within the simulated annealing heuristic a penalty term is added to the objective function to guide the search towards feasible regions. To solve the problem, the authors use a rather standard simulated annealing procedure with three neighborhoods applied to the indirect representation. For the permutation the neighborhoods are: reinsertion of a randomly selected customer or exchange of the position of a random pair of customers, whereas for the binary vector the neighborhood is defined by flipping the type of vehicle serving a randomly selected vehicle customer. To increase the chance of obtaining high quality solutions, half of the time the move performed is the best of several random trials of the selected neighborhood. By relaxing the truck and trailer availability constraints Lin et al. [31] proposed the relaxed TTRP (RTTRP). Reusing their SA, these authors discovered a non-trivial trade-off between the fleet size and the total distance.

Caramia and Guerriero [5] designed a mathematical-programming based heuristic that also employs the cluster-first, route-second approach. Their method solves two subproblems sequentially. The first one, called customer-route assignment problem (CAP) assigns the customers to valid routes seeking to reduce the size of the fleet. Then, given the assignment of customers to routes, the route-definition problem (RDP) minimizes the tour length of each route using a TSP like model without subtour elimination constraints. Since the RDP may produce solutions with disconnected subtours, an edge-insertion heuristic is used to properly connect subtours to the main route. For pure truck routes and pure vehicle routes the heuristic builds a single tour; while for vehicle routes with subtours, the heuristic selects the root of each subtour and constructs the main tour. The authors embedded these two models within an iterative mechanism that adds new constraints to the CAP based on the information of the RDP solution. This restarting mechanism is intended to diversify the search, and includes a tabu search mechanism that forbids (in the CAP) customers route assignments already explored in previous iterations of the algorithm.

In the literature several researchers have studied other vehicle routing problems with trailers that deviate from the TTRP. Semet [47] presented the partial accessibility constrained vehicle routing problem (PACVRP), in which each parking place for the trailer is restricted to have only one departing subtour. He provided an integer programming formulation and developed a cluster-first, route-second approach for the PACVRP. His article mainly discusses the clustering phase modeled with an extended generalized assignment problem, which is solved using Lagrangian relaxation embedded within branch and bound. Gerdessen [21] tackled the vehicle routing problem with trailers (VRPT) using constructive and local search heuristics. The VRPT differs from the TTRP in several simplifying assumptions: in the VRPT there are no accessibility constraints, instead a different service time is incurred if a customer is visited by a complete vehicle or by a truck; all the customers have unit demand, and each trailer is parked exactly once.

Drexl [13] proposed the vehicle routing problem with trailer and transshipments VRPTT. In the VRPTT the customers have time windows; the routing cost is vehicle dependent (i.e., if one arc is traversed by a complete vehicle it has a different cost than if it is traversed just by a truck); parking places (transshipment locations) differ from customer locations; and the assumption of a fixed truck-trailer assignment is dropped, so that a trailer may be pulled by any compatible truck in different routes. The author used a branch-and-cut method to solve the VRPTT and showed that only very small instances of this problem can be solved to optimality.

Recently, Villegas et al. [54] have studied the single truck and trailer routing problem with satellite depots (STTRPSD) in which a truck with a detachable trailer based at a main depot has to serve the demand of a set of customers accessible only by truck. Therefore, before serving the customers, the trailer is detached in appropriated parking places (called trailer points of satellite depots) where goods are transferred between the truck and the trailer. To solve the STTRPSD they proposed a multi-start evolutionary local search and a hybrid metaheuristic based on GRASP and variable neighborhood descent (VND). In their computational experiments, on a set of randomly generated instances, multi-start evolutionary local search outperformed GRASP/VND in terms of solution quality and running time.

3. GRASP/VNS with path relinking

GRASP is a memory-less multi-start metaheuristic in which a local search is applied to initial solutions obtained with a greedy randomized heuristic [17]. Even though GRASP has not been widely used for the solution of vehicle routing problems [20], GRASP-based hybrid metaheuristics have achieved competitive results in different routing problems such as the VRP [36], the capacitated location-routing problem [14] and the capacitated arc-routing problem with time windows [39].

Resende and Ribeiro [43] reported that the performance of GRASP can be enhanced by using reactive fine tuning mechanisms, multiple neighborhoods, and path relinking. Along this line, our hybrid metaheuristic includes VNS as the local search component and uses PR in different strategies. A description of the components and the general structure of the hybrid GRASP/VNS with path relinking follows.

3.1. Greedy randomized construction

Contrary to most of the solution methods for TTRP-related problems [5–7,28,45,47,48] that use a natural cluster-first, route-second approach, in this paper we use a route-first, cluster-second (RFCS) procedure for the randomized construction of GRASP. Even though in the 80s Beasley [1] introduced route-first, cluster-second heuristics for the VRP, it was only twenty years later that Prins [35] unveiled its potential as a component of metaheuristics for routing problems. The fundamental idea is to take a giant tour $T=(0,t_1,\dots,t_i,\dots,t_n,0)$ visiting all the customers and break it into VRP feasible routes using a tour splitting procedure. We follow the same spirit for the TTRP.

The randomized route-first, cluster-second heuristic follows three steps. First, a randomized nearest neighbor heuristic with a *restricted candidate list* (RCL) of size κ constructs a giant tour $T=(0,t_1,\dots,t_i,\dots,t_n,0)$, where t_i represents the customer in the i-th position of the tour. Note that T visits all the customers in N, ignoring the capacity of the vehicles and the accessibility constraints of customers in N_t. Second, we define an auxiliary acyclic graph $H=(X,U,W)$ where the set of nodes X contains a dummy node 0 and n nodes numbered 1 through n, where node i represents customer t_i (i.e., the customer in the i-th position of T); the arc set U contains one arc $(i-1,j)$ if and only if the subsequence $(t_i,\dots,t_j)$ can be served in a feasible route; and the weight $w_{i-1,j}$ of arc $(i-1,j)$ is the total distance of the corresponding route. Third, the shortest path between nodes 0 and n in H represents a TTRP solution S, where the cost of the shortest path corresponds to the total distance of S and the arcs in the shortest path represent its routes.

Note that to adapt the route-first, cluster-second (RFCS) approach for the TTRP it is necessary to take into account its complicating elements, namely, the accessibility constraints and the heterogeneous fixed fleet. We manage the accessibility constraints when building H and take into account the heterogeneous fixed fleet while solving the shortest path on H.

The arc set U in H has three types of arcs, each one representing a type of route. Before adding arc $(i-1,j)$ to U we perform a feasibility test for route $R_{ij}=(0,t_i,\ldots,t_j,0)$. Let $Q_{ij}=\sum_{u=i}^{j} q_{t_u}$ be the total demand of R_{ij}. If $Q_{ij}<Q_t$ then R_{ij} is feasible (regardless of the type of customers assigned to it), and it is a pure truck route. On the other hand, if $Q_t<Q_{ij}\le Q_t+Q_r$, R_{ij} is feasible if t_i is a vehicle customer, and the type of route represented by arc $(i-1,j)$, depends on the customers assigned to R_{ij}. If all customers belong to N_v then R_{ij} is a pure vehicle route (without subtours). But, if there is at least one customer in N_t, then R_{ij} is a vehicle route with subtours. Finally, if $Q_{ij}>Q_t+Q_r$ the route is infeasible and the arc dropped from set U.

Since the triangle inequality holds, the cost of pure truck routes and pure vehicle routes is easily calculated with $c(R_{ij})=c_{0t_i}+\sum_{u=i}^{j-1} c_{t_u,t_{u+1}}+c_{t_j0}$, because its structure corresponds to a single tour. On the contrary, the total distance of vehicle routes with subtours is calculated using an optimization subproblem that selects the parking places for the trailer, and builds the main tour and subtours. This subproblem can be seen as a restricted version of the single truck and trailer routing problem with satellite depots (STTRPSD) [54], in which the satellite depots correspond to the vehicle customers included in R_{ij}.

To solve the restricted STTRPSD associated with the vehicle route with subtours R_{ij} we use a dynamic programming method in which state [l,m] $(i\le l\le j, l: t_l\in N_v; l\le m\le j)$ represents the use of vehicle customer t_l as the root of a subtour ending at customer t_m. The initial state [i,i] represents the departure of the complete vehicle from the main depot to the first vehicle customer. By definition we include states $[l,l]\forall l: t_l\in N_v$ to represent the movement of the complete vehicle from the root of a subtour to the next vehicle customer performing an empty subtour of null cost.

Let F_{lm} be the cost of state [l,m], and θ_{lkm} be the cost of a subtour $ST=(t_l,t_k,\ldots,t_m,t_l)$ rooted at vehicle customer t_l and visiting customer t_k to t_m. In order to find the parking places for the trailer in route R_{ij} we use a recurrence relation with three cases. The first case represents the initial state ($l=i$ and $m=i$) and corresponds to the trip of the complete vehicle from the depot to the first vehicle customer t_i. Its cost is given by $F_{lm}=c_{0t_i}$. Then, the second case ($l=i$ and $i<m\le j$) analyses the possibility of performing all the subtours based at vehicle customer t_i. The cost of these states is given by $F_{lm}=\min_k\{F_{ik}+\theta_{i,k+1,m}|k<m:\sum_{u=k+1}^{m} q_{t_u}\le Q_t\}$.

Finally, the general case $(i<l\le j, t_l\in N_v; l\le m\le j)$ has two terms. The first term includes the alternative of having several subtours rooted at vehicle customer t_l, while the second term includes the movement of the complete vehicle from vehicle customer t_h to perform a subtour rooted at vehicle customer t_l. The cost of these states is given by:

$$F_{lm}=\min\left\{\min_k\left\{F_{lk}+\theta_{l,k+1,m}|k<m:\sum_{u=k+1}^{m} q_{t_u}\le Q_t\right\},\right.$$
$$\left.\min_{h,k}\left\{F_{hk}+c_{t_h,t_l}+\theta_{l,k+2,m}|h<l: t_h\in N_v; k=l-1:\sum_{u=k+2}^{m} q_{t_u}\le Q_t\right\}\right\}$$

In all cases, the additional conditions over k check the capacity constraints of the truck while performing the subtours; and the conditions over h assure that the main tour only visits vehicle customers. The recursive equation that includes the three cases is presented in Appendix B.

Since states [l,m] do not include the return from the last vehicle customer to the main depot, the cost of the route is finally calculated as $c(R_{ij})=\min_{i\le l\le j: t_l\in N_v}\{F_{lj}+c_{t_l,0}\}$. To solve the STTRPSD we use a similar approach to that used by Villegas et al. [54]. See Appendix B for additional details of the solution procedure.

Once we have generated the auxiliary graph H, we solve a shortest path problem to find the optimal partition of T into a TTRP feasible solution. The limited fleet is taken into account at this stage, thus we solve a resource-constrained shortest path problem (RCSPP) in H, where the resources are the available trucks and trailers. Each arc $(i-1,j)$ in U has three attributes: the distance of

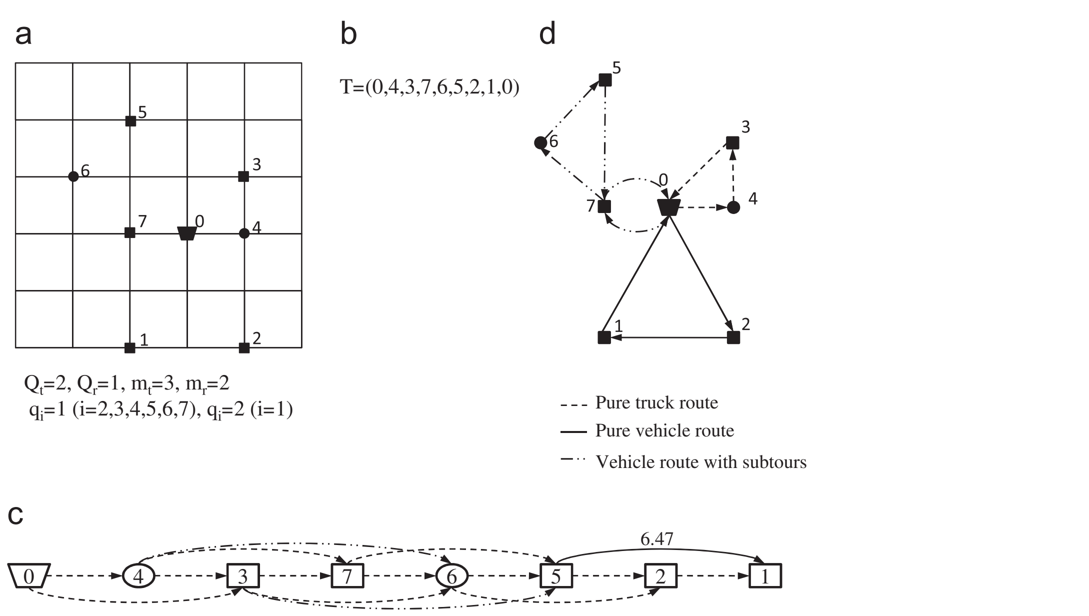

Fig. 2. Example of the route-first cluster-second procedure for the TTRP used in the greedy randomized construction. (a) Problem information; (b) giant tour; (c) auxiliary graph; (d) TTRP solution.

the route it represents $w_{i-1,j}=c(R_{ij})$, the consumption of trucks $\alpha_{i-1,j}$, and the consumption of trailers $\beta_{i-1,j}$. The quantities $\alpha_{i-1,j}$ and $\beta_{i-1,j}$ depend on the type of route in the following way: $\alpha_{i-1,j}=1, \beta_{i-1,j}=0$ if R_{ij} is a pure truck route, while $\alpha_{i-1,j}=\beta_{i-1,j}=1$ if R_{ij} is a pure vehicle route or a vehicle route with subtours.

In general, shortest path problems with resource constraints can be solved using a generalization of Bellman's algorithm with several labels per node [10]. In our case, let, $\Lambda=(\delta,\tau,\rho,\eta,\lambda)$ be a label associated with any given node $i \in X$ that represents a partial shortest path ending at node i. The label has five attributes: cost δ, truck consumption τ, trailer consumption ρ, father node η, and father label λ. Let $\mathcal{L}_i$ be the set of labels of node i, and let $\Gamma(i)$ be the set of successors of node i, $\Gamma(i)=\{j \in X : (i,j) \in U\}$.

For two labels $\Lambda_1=(\delta_1,\tau_1,\rho_1,\eta_1,\lambda_1)$, $\Lambda_2=(\delta_2,\tau_2,\rho_2,\eta_2,\lambda_2)$, we say that Λ_1 dominates in the Pareto sense Λ_2 (denoted $\Lambda_1 \preceq \Lambda_2$) if and only if $\delta_1 \le \delta_2 \wedge \tau_1 \le \tau_2 \wedge \rho_1 \le \rho_2$, and at least one of the inequalities is strict [15]. That is, label Λ_2 is dominated by label Λ_1 because it is possible to reach node j with the same distance and less resource consumption, or with a shorter distance and the same resource consumption.

We use Algorithm 1 to solve the RCSPP. Since by construction H is acyclic, the outmost *for* loop takes the nodes in increasing order, and for a given node i, it scans the set of successors $\Gamma(i)$ (lines 3–23). In the inmost *forall* loop (lines 4–22) all the labels of node i are extended, lines 11–17 perform a domination test, and remove all dominated labels of $\mathcal{L}_j$ if any exists. Finally, line 19 adds non-dominated labels to $\mathcal{L}_j, (j \in \Gamma(i))$. The number of arcs in U is bounded by $O(n^2)$. If we assume that there are no two labels with the same distance, the maximum number of non-dominated labels for each node can be bounded by $O(m_t m_r)$ because m_t and m_r are integers and the consumption is done one unit at a time. Thus, the non-domination test of lines 11–17 is performed in the worst case $O(m_t^2 m_r^2)$ times for each arc. The previous arguments prove that Algorithm 1 runs in $O(n^2\ m_t^2\ m_r^2)$.

Algorithm 1. Labeling algorithm for the resource-constrained shortest path problem

Input: Auxiliary graph H
Output: Shortest path from node 0 to n
1: Create a label $\Lambda_0=(0,0,0,0,\emptyset)$; $\mathcal{L}_0 := \mathcal{L}_0 \cup \{\Lambda_0\}$
2: **for** $i=0$ to $n-1$ **do**
3: **for all** $j \in \Gamma(i)$ **do**
4: **for all** $\overline{\Lambda}=(\overline{\delta},\overline{\tau},\overline{\rho},\overline{\eta},\overline{\lambda}) \in \mathcal{L}_i$ **do**
5: $ld := \overline{\delta}+w_{ij}$
6: $lt := \overline{\tau}+\alpha_{ij}$
7: $lr := \overline{\rho}+\beta_{ij}$
8: **if** $lt \le m_t$ and $lr \le m_r$ **then**
9: Create a label $\hat{\Lambda} := (ld,lt,lr,i,\overline{\Lambda})$
10: $nondom := true$
11: **for all** $\Lambda \in \mathcal{L}_j$ **do**
12: **if** $\hat{\Lambda} \preceq \Lambda$ **then**
13: $\mathcal{L}_j := \mathcal{L}_j \backslash \{\Lambda\}$
14: **else if** $\Lambda \preceq \hat{\Lambda}$ **then**
15: $nondom := false$
16: **end if**
17: **end for**
18: **if** *nondom* **then**
19: $\mathcal{L}_j := \mathcal{L}_j \cup \{\hat{\Lambda}\}$
20: **end if**
21: **end if**
22: **end for**
23: **end for**
24: **end for**

After solving the RCSPP, it is possible to derive the minimum-cost TTRP solution by selecting the label $\Lambda^*=\text{argmin}_{\Lambda \in \mathcal{L}_n}\delta_\Lambda$, (i.e., the cheapest label of node n) and backtracking from it using the information stored in η_{Λ^*} and λ_{Λ^*}. However, the algorithm for the RCSPP may fail to find a feasible solution and in that case $\mathcal{L}_n=\emptyset$. This occurs when it is not possible to reach node n with at most m_t trucks and m_r trailers. In this case, we relax the fleet-size constraints and solve a classical shortest path problem to find an infeasible solution. The infeasibility of the resulting solution is treated later in the improvement phase and the path relinking procedure.

Fig. 2 illustrates the greedy randomized construction for the TTRP. For the sake of clarity we only include in the auxiliary graph the cost of the arcs in the shortest path. The length of each square side in the grid is equal to 1 and the distance between nodes is Euclidean. All customers have unitary demands, except customer 1 with $q_1=2$. Using the information of the problem (Fig. 2(a)) and the sequence of the giant tour (Fig. 2(b)), the route-first cluster-second procedure first builds the auxiliary graph (Fig. 2(c)). After solving the shortest path problem from node 0 to node 1 in this graph, the route-first, cluster-second procedure builds the TTRP solution of Fig. 2(d) using the information of the arcs in the shortest path. Table 1 gives the details of each arc in the auxiliary graph including its tail and head, and the information of the associated route.

3.2. Variable neighborhood search for the TTRP

The improvement phase of the hybrid metaheuristic is a VNS procedure [25]. Taking an initial solution S_0, VNS performs a classical *variable neighborhood descent* (hereafter VND) step, and then repeats *ni* main iterations of shaking and improvement alternating between solutions and giant tours. Within VNS we accept infeasible solutions, provided that its infeasibility $\Phi(S)$ does not exceed a given threshold μ. The infeasibility of a given TTRP solution S is calculated using the following expression:

$$\Phi(S)=\max\left\{0,\frac{ut(S)}{m_t}-1\right\}+\max\left\{0,\frac{ur(S)}{m_r}-1\right\}$$

where $ut(S)$ and $ur(S)$ represent the number of trucks and trailers required by S. At each call of VNS, the value of μ is initialized at μ_{max} and decreased after each iteration by $\mu=\mu-\mu_{max}/ni$.

Let $T(S)$ be the giant tour associated with a given TTRP solution S, $T(S)$ is obtained by concatenating all the routes of S in a single string. The shaking procedure works on $T(S)$ by randomly exchanging b pairs of customers with procedure *perturb*$(T(S),b)$. Then, we derive a new TTRP solution from the perturbed giant tour by using the RFCS approach described above. The value of b is controlled dynamically between 1 and b_{max}, depending on the feasibility and the objective function of the current solution. If the current solution is feasible and updates the best solution visited so far, the value of b is reset to 1 to search in its neighborhood. Whereas, if the current solution is infeasible or the best solution is not improved, b is increased to search in regions far from it. With this mechanism VNS acts as a reparation operator for infeasible solutions and as an improvement procedure for feasible ones.

On the other hand, the procedure *VND*(S) explores sequentially the following five neighborhoods of a given TTRP solution S using a best-improvement strategy:

- *Modified Or-opt*. For a given chain of customers $(r_i,\ldots,r_{i+l-1})$ of length $l=1,2,3$, check all possible reinsertions of the chain and its reverse $(r_{i+l-1},\ldots,r_i)$ within the same route or subtour. The difference with classical Or-opt is the simultaneous evaluation of the reversal of the chain.
- *Node exchange* (in single routes/subtours and between pairs of routes/subtours). Given a pair of customers u and v served by routes (or subtours) R_u and R_v exchange their positions. If $R_u \ne R_v$, in addition to classical capacity constraints, it is

Table 1
Arc information of the auxiliary graph for the route-first cluster-second example of Fig. 2.

Arc		Route				
Tail	Head	Type	Capacity	Load	Cost	Structure
0	4	PTR	2	1	2.00	0–4–0
0	3	PTR	2	2	3.41	(*) 0–4–3–0
4	3	PTR	2	1	2.83	0–3–0
4	7	PTR	2	2	4.65	0–3–7–0
4	6	VRWS	3	3	7.48	0–3–7–0 (Main tour) 7–6–7 (Subtour)
3	7	PTR	2	1	2.00	0–7–0
3	6	PTR	2	2	4.65	0–7–6–0
3	5	VRWS	3	3	6.83	(*) 0–7–0 (Main tour) 7–6–5–7 (Subtour)
7	6	PTR	2	1	4.47	0–6–0
7	5	PTR	2	2	5.89	0–6–5–0
6	5	PTR	2	1	4.47	0–5–0
6	2	PTR	2	2	8.94	0–5–2–0
5	2	PTR	2	1	4.47	0–2–0
5	1	PVR	3	3	6.47	(*) 0–2–1–0
2	1	PTR	2	2	4.47	0–1–0

PTR: pure truck route, PVR: pure vehicle route, VRWS: vehicle route with subtours,
(*): routes of the optimal splitting.

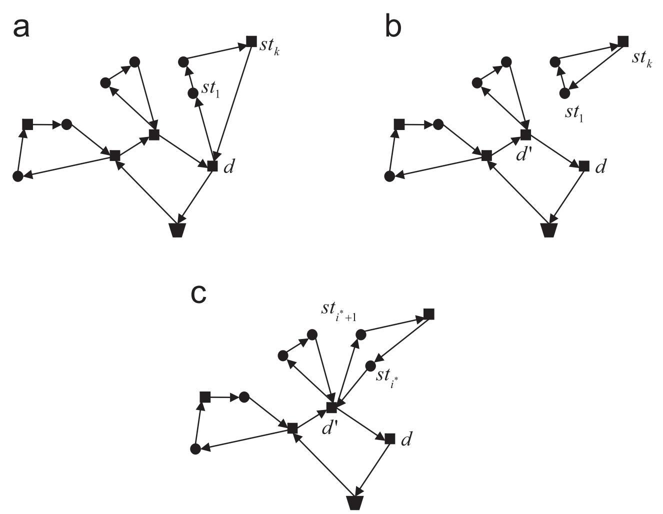

Fig. 3. Example of the root refining neighborhood [7]. (a) Initial Subtour. (b) Subtour as TSP. (c) Best insertion of the new root.

necessary to verify the accessibility constraints of u in R_v and v in R_u. Moreover when R_u or R_v is a subtour the capacity of the associated vehicle route is also checked.

- *2-opt* (in single routes/subtours, or pairs of routes (subtours) of the same type). Remove a pair of arcs (u,v) and (w,y) and add two other arcs. For single routes add arcs (u,w) and (v,y); if the arcs belong to different routes we also consider the addition of arcs (u,y) and (w,v) and select the best of the two options. Moreover, if the arcs belong to a pair of subtours we allocate the resulting subtours to the best root among those of the original subtours, provided the capacities of the associated vehicle routes are not exceeded.
- *Node relocation* (in single routes/subtours and between pairs of routes/subtours). Given a customer u served in route/subtour R_u and two consecutive nodes v,w in a route/subtour R', insert u between v and w. If $R_u \neq R'$ we check the conditions for a valid insertion of u in R'.
- *Root refining*. For each subtour we apply the root refining procedure described by Chao [7], where we try to change the root of each subtour and simultaneously modify its routing. Formally, for a subtour $ST=(d,st_1,\ldots,st_k,d)$ the operator removes arcs (d,st_1) and (st_k,d) and adds arc (st_k, st_1) to create a TSP tour. Then the position i^* of the new root d' is found using best insertion and the new subtour becomes $ST'=(d',st_{i^*+1},\ldots,st_1,st_k,\ldots,st_{i^*},d')$. To be feasible the new root d' must be served in pure vehicle routes or main tours of vehicle routes with subtours having enough residual capacity to insert the total demand of the subtour. Fig. 3 illustrates this neighborhood.

Using the elements described above, Algorithm 2 outlines the VNS component of the proposed metaheuristic.

Algorithm 2. VNS for the TTRP

```
Input: Initial solution S_0, parameters: μ_max, b_max, ni
Output: Improved solution S*
    S_0 := VND(S_0)
    b := 1
    μ := μ_max
    S* := ∅
    if Φ(S_0) = 0 then
        S* := S_0
    end if
    S := S_0
    for i = 1 to ni do
        T' := perturb(T(S), b)
        S' := RFCS(T')
        S' := VND(S')
        if Φ(S') ≤ μ and f(S') < f(S) then
            S := S'
        end if
        if Φ(S') = 0 and f(S') < f* then
            S* := S'
            b := 1
        else
            b := min{b+1, b_max}
        end if
        μ := μ − μ_max/ni
    end for
    return S*
```

3.3. Path relinking

Path relinking (PR) was introduced in the context of tabu search (TS) as a mechanism to combine intensification and diversification

Forward path

	Giant tour	Distance to T(S_f)
T(S_0)	0 7 6 5 3 4 1 2 0	6
T_1	0 1 2 7 6 5 3 4 0	5
T_2	0 1 2 3 4 7 6 5 0	4
T_3	0 1 2 3 4 5 7 6 0	3
T_4	0 1 2 3 4 5 6 7 0	0
T(S_f)	0 1 2 3 4 5 6 7 0	

Backward path

	Giant tour	Distance to T(S_f)
T(S_f)	0 1 2 3 4 5 6 7 0	6
T_1	0 7 1 2 3 4 5 6 0	5
T_2	0 7 6 1 2 3 4 5 0	4
T_3	0 7 6 5 1 2 3 4 0	3
T_4	0 7 6 5 3 4 1 2 0	0
T(S_0)	0 7 6 5 3 4 1 2 0	

Shifting blocks

Fig. 4. Example of the path relinking operator.

[23]. PR generates new solutions by exploring trajectories connecting elite solutions previously produced during the search. Hybridizing PR with GRASP improves the performance of the latter by tackling the memory-less criticism faced by the basic GRASP scheme [42].

Even though the use of PR in metaheuristics for vehicle routing is rather scarce, it has been applied with relative success. Hybrid metaheuristics combining PR and other methods have been used to solve the classical VRP [27,51], the multi-objective dial-a-ride problem [34], the multi-compartment VRP [16], and the VRP with time windows [26], among others. Particularly, GRASP/PR hybrids have been used to solve different routing problems such as the capacitated location-routing problem [38], the team orienteering problem [52], a combined production-distribution problem [3], and the capacitated arc-routing problem with time windows [39], among others.

Resende and Ribeiro [41] give an overview of several ways on how to hybridize PR with GRASP. However, the distance measure, the management of the set of elite solutions, and the design of the PR operator are independent from the hybridization mechanism. A brief description of these components for the TTRP follows.

3.3.1. Distance measure and pool management

GRASP with PR maintains a pool of elite solutions $\boldsymbol{ES}$. For inclusion in $\boldsymbol{ES}$ a solution S must be better than the worst solution of the pool, but to preserve the diversity of $\boldsymbol{ES}$, the distance $d(\boldsymbol{ES},S)$ between S and the pool must be greater than a given threshold Δ, where $d(\boldsymbol{ES},S)=\min_{S' \in \boldsymbol{ES}}\{d(S,S')\}$. However, the latter condition is overridden when the best solution of $\boldsymbol{ES}$ is updated. With the same diversity objective, S replaces $S_w = min_{S' \in \boldsymbol{ES}: f'(S') > f'(S)} d(S',S)$; i.e., the closer solution in $\boldsymbol{ES}$ that is worst than S. Note that since the pool may contain infeasible solutions, to guide the search toward feasible solutions we use a modified objective function $f' = M_1 \cdot \max\{0, ut(S) - m_t\} + M_2 \cdot \max\{0, ur(S) - m_r\} + f(S)$, where M_1 and M_2 are real numbers such that $M_1 \gg M_2 \gg 0$, and $f(S)$ is the total distance of the routes of S.

Initially $\boldsymbol{ES}$ is filled with $|\boldsymbol{ES}|$ solutions generated with GRASP/VND and $\boldsymbol{ES}$ is always kept ordered according to $f'(S)$. Note that, values for M_1 and M_2 are not explicitly needed because the pool is lexicographically sorted using an order consistent with the one imposed by $f'(S)$. This lexicographic order gives priority to feasible solutions, among feasible solutions to those with smaller distances, and among infeasible solutions to those with smaller infeasibility with respect to the use of trucks, and then to those with smaller infeasibility with respect to the use of trailers.

Different metrics can be used to define distances between solutions of vehicle routing problems. For instance, Ho and Gendreau [27] count the number of differing edges; Sörensen and Sevaux [50] measure the similarity between trips using the edit distance and then solve a linear assignment problem to match the trips of the solutions. Then the cost of the assignment becomes the distance between solutions. Finally, in route-first cluster-second based metaheuristics it is possible to measure the distance between solutions using their corresponding giant tours [37]. In that case, different metrics for distance on permutations could be used [46,49]. Among them, we decided to use the distance for R-permutations [4], also known as the adjacency or broken-pairs distance.

Given two solutions S and S' and their corresponding giant tours $T(S)$ and $T(S')$, the broken-pairs distance counts the number of consecutive pairs that differ from one giant tour to the other, that is $d(T(S),T(S'))$ is defined as the number of times t_{i+1} does not immediately follow t_i in $T(S')$, for $i = 0, \ldots, |N|$. For example, if we have $T(S) = (0,1,2,3,4,5,0)$ and $T(S') = (0,5,3,4,1,2,0)$, the distance $d(T(S),T(S'))$ is 4, because pairs (0,1), (2,3), (4,5) and (5,0), of $T(S)$ are not in $T(S')$.

3.3.2. Path relinking operator

To transform the initial solution S_0 into the guiding solution S_f, the PR operator repairs from left to right the broken pairs of $T(S_0)$, creating a path of giant tours with non-increasing distance to $T(S_f)$. A broken pair is repaired by shifting to the left (in $T(S_0)$) blocks of consecutive customers in such a way that at least one broken pair is repaired without creating new broken pairs. Fig. 4 illustrates the PR operator. To increase the chance of finding high quality solutions, the PR operator uses the back-and-forward strategy [41], exploring the forward path from S_0 to S_f, and also the backward path from S_f to S_0. All the giant tours in both paths are split using the route-first cluster-second approach described above to generate a set $\boldsymbol{P}$ of TTRP solutions.

3.3.3. Path relinking strategies and overview of the method

Originally, Laguna and Martí [29] proposed PR as intensification mechanism after each GRASP iteration. Our first GRASP/VNS with PR described in Algorithm 3 follows this approach. In this hybrid method the PR operator explores the paths between a local optimum obtained by GRASP/VNS and a solution randomly chosen from $\boldsymbol{ES}$. The difference with the classical approach is that we apply VND to all feasible solutions produced by the PR operator and test them for insertion in $\boldsymbol{ES}$.

Algorithm 3. GRASP/VNS with PR as intensification mechanism

Input: TTRP, **parameters:** κ, ns, μ_{max},b_{max},ni, $|\boldsymbol{ES}|$, Δ
Output: TTRP solution S^*
for $i = 1$ to $|\boldsymbol{ES}|$ **do**
  $T := RandomizedNearestNeighbor(N,\kappa)$
  $S := RFCS(T)$
  $S := VND(S)$
  Insert S in $\boldsymbol{ES}$
end for
for $i = 1$ to ns **do**
  $T := RandomizedNearestNeighbor(N,\kappa)$

$S := RFCS(T)$
$S := VNS(S,\mu_{max},b_{max},ni)$
Select at random $S' \in \boldsymbol{ES}$
$\boldsymbol{P} := PathRelinkingOperator(S,S')$
for all $\bar{S} \in \boldsymbol{P} : \Phi(\bar{S}) = 0$ **do**
$\bar{S} := VND(\bar{S})$
if $d(\boldsymbol{ES},\bar{S}) \geq \Delta$ **then**
Try to insert $\bar{S}$ in $\boldsymbol{ES}$
end if
end for
end for
$S^* := \text{argmin}_{S \in \boldsymbol{ES}} f(S)$
return S^*

Another possibility is to use the PR operator as post-optimization procedure, after the *ns* iterations of GRASP/VNS. In this case, each local optimum produced by GRASP/VNS is just checked for inclusion in ***ES***. After the main GRASP/VNS loop, the procedure *PathRelinking*(***ES***), applies PR to all pairs of elite solutions in ***ES*** not yet relinked. The resulting feasible solutions (***RS***) are further improved with VND and ***ES*** updated. The post-optimization procedure is iterated as long as there are new solutions in ***ES***. Algorithm 4 summarizes this variant of GRASP/VNS with PR.

Algorithm 4. GRASP/VNS with PR as post-optimization mechanism

Input: TTRP, **parameters:** κ, ns, μ_{max},b_{max},ni, $|\boldsymbol{ES}|$, Δ
Output: TTRP solution S^*
for $i=1$ to ns **do**
$T := RandomizedNearestNeighbor(N,\kappa)$
$S := RFCS(T)$
$S := VNS(S,\mu_{max},b_{max},ni)$
if $d(\boldsymbol{ES},S) \geq \Delta$ **then**
Try to insert S in $\boldsymbol{ES}$
end if
end for
$new := true$
repeat
$\mathbf{ES}_0 := \boldsymbol{ES}$
$\boldsymbol{RS} := PathRelinking(\boldsymbol{ES})$
for all $\bar{S} \in \boldsymbol{RS}$ **do**
if $\Phi(\bar{S}) = 0$ **then**
$\bar{S} := VND(\bar{S})$
else
$\boldsymbol{RS} := \boldsymbol{RS} \backslash \{\bar{S}\}$
end if
end for
$\mathbf{ES} := Update(\mathbf{ES},\mathbf{RS})$
if $\mathbf{ES} \backslash \mathbf{ES}_0 = \emptyset$ **then**
$new := false$
end if
until $new = false$
$S^* := \text{argmin}_{S \in \boldsymbol{ES}} f(S)$
return S^*

More recently, Resende and Werneck [44] and Resende et al. [40] introduced evolutionary path relinking (EvPR), a variant in which the *PathRelinking*(***ES***), procedure is used periodically during the search. Consequently, our GRASP/VNS with EvPR for the TTRP (outlined in Algorithm 5) keeps the intensification mechanism, and evolves the elite set every γ iterations.

Algorithm 5. GRASP/VNS with evolutionary path relinking

Input: TTRP, **parameters:** κ, ns, μ_{max},b_{max},ni, $|\boldsymbol{ES}|$, Δ
Output: TTRP solution S^*
for $i=1$ to $|\boldsymbol{ES}|$ **do**
$T := RandomizedNearestNeighbor(N,\kappa)$
$S := RFCS(T)$
$S := VND(S)$
Insert S in $\boldsymbol{ES}$
end for
for $i=1$ to ns **do**
$T := RandomizedNearestNeighbor(N,\kappa)$
$S := RFCS(T)$
$S := VNS(S,\mu_{max},b_{max},ni)$
Select at random $S' \in \boldsymbol{ES}$
$\boldsymbol{P} := PathRelinkingOperator(S,S')$
for all $\bar{S} \in \boldsymbol{P} : \Phi(\bar{S}) = 0$ **do**
$\bar{S} := VND(\bar{S})$
if $d(\boldsymbol{ES},\bar{S}) \geq \Delta$ **then**
Try to insert $\bar{S}$ in $\boldsymbol{ES}$
end if
end for
if $i \bmod \gamma = 0$
$new := true$
repeat
$\mathbf{ES}_0 := \boldsymbol{ES}$
$\boldsymbol{RS} := PathRelinking(\boldsymbol{ES})$
for all $\bar{S} \in \boldsymbol{RS}$ **do**
if $\Phi(\bar{S}) = 0$ **then**
$\bar{S} := VND(\bar{S})$
else
$\boldsymbol{RS} := \boldsymbol{RS} \backslash \{\bar{S}\}$
end if
end for
$\mathbf{ES} := Update(\mathbf{ES},\mathbf{RS})$ **then**
if $\mathbf{ES} \backslash \mathbf{ES}_0 = \emptyset$ **then**
$new := false$
end if
until $new = false$
end if
end for
$S^* := \text{argmin}_{S \in \boldsymbol{ES}} f(S)$
return S^*

4. Computational experiments

We implemented the three variants of the proposed metaheuristic (GRASP/VNS with PR as post-optimization, GRASP/VNS with PR as intensification and GRASP/VNS with EvPR) using Java and compiled them using Eclipse JDT 3.5.1. We ran the experiments of this section on a computer with an Intel Xeon running at 2.67 GHz under Windows 7 Enterprise Edition (64 bits) with 4 GB of RAM. Table 2 summarizes the characteristics of each problem in the 21-instance test bed described by Chao [7], where the size of the problems range from $n = 50$ to 199 and for each problem size there are three values for the fraction of truck customers (25%, 50% and 75%).

All the variants of GRASP/VNS with PR share the size of the RCL (κ) and the number of iterations (ns) of GRASP; the maximum infeasibility threshold (μ_{max}), the maximum number of pairs (b_{max}) and the number of iterations (ni) of VNS; and the distance threshold (Δ) and size of the elite set ($|\mathbf{ES}|$) of PR. Additionally, EvPR is applied

Table 2
Test problems for the TTRP.

Problem number	Customers			Trucks		Trailers		Demand–capacity ratio
	n	$\lvert N_v \rvert$	$\lvert N_t \rvert$	m_t	Q_t	m_r	Q_r	
1	50	38	12					
2	50	25	25	5	100	3	100	0.971
3	50	13	37					
4	75	57	18					
5	75	38	37	9	100	5	100	0.974
6	75	19	56					
7	100	75	25					
8	100	50	50	8	150	4	100	0.911
9	100	25	75					
10	150	113	37					
11	150	75	75	12	150	6	100	0.931
12	150	38	112					
13	199	150	49					
14	199	100	99	17	150	9	100	0.923
15	199	50	149					
16	120	90	30					
17	120	60	60	7	150	4	100	0.948
18	120	30	90					
19	100	75	25					
20	100	50	50	10	150	5	100	0.903
21	100	25	75					

Table 3
Parameters of the proposed GRASP/VNS with PR variants.

Method	GRASP		VNS			Path relinking		
	κ	ns	μ_{max}	b_{max}	ni	$\lvert \mathbf{ES} \rvert$	Δ	γ
GRASP/VNS	2	60	0.25	6	200	–	–	–
GRASP/VNS with PR (intensification)	2	60	0.25	6	200	5	$\max\{10, m_t + m_r\}$	–
GRASP/VNS with PR (post-optimization)	2	60	0.25	6	200	5	$\max\{10, m_t + m_r\}$	–
GRASP/VNS with EvPR	2	60	0.25	6	200	5	$\max\{10, m_t + m_r\}$	20

Table 4
Results of the proposed metaheuristics in the test problems of [7].

Problem			GRASP/VNS			GRASP/VNS +PR (intensification)			GRASP/VNS +PR (post-optimization)			GRASP/VNS + EvPR		
Number	n	BKS	Best	Avg.	Time	Best	Avg.	Time	Best	Avg.	Time	Best	Avg.	Time
1	50	564.68	**564.68**	568.31	0.91	**564.68**	566.22	1.07	**564.68**	566.38	0.98	**564.68**	565.99	1.17
2	50	611.53	614.27	617.53	1.00	**611.53**	614.46	1.18	**611.53**	614.81	1.08	**611.53**	614.23	1.29
3	50	618.04	**618.04**	619.07	0.86	**618.04**	618.04	0.98	**618.04**	618.24	0.91	**618.04**	618.04	1.05
4	75	798.53	802.41	815.16	1.86	802.41	805.72	2.37	799.34	805.62	2.08	**798.53**	803.51	2.69
5	75	839.62	841.81	857.79	1.99	**839.62**	844.99	2.59	840.74	848.18	2.19	**839.62**	841.63	2.82
6	75	930.64	989.71	1040.19	1.95	952.43	967.77	2.57	946.66	970.18	2.36	940.59	961.47	2.89
7	100	830.48	830.62	832.27	4.11	**830.48**	830.55	5.55	**830.48**	830.71	4.66	**830.48**	830.48	6.05
8	100	872.56	881.53	885.01	4.35	874.95	878.48	5.99	874.73	879.02	5.12	**872.56**	876.21	6.96
9	100	912.02	916.63	930.55	5.67	915.29	920.22	7.66	915.46	921.39	6.25	914.23	918.45	8.38
10	150	1039.07	1050.76	1062.03	9.95	1047.25	1051.40	16.01	1047.59	1054.40	13.10	1046.71	1050.11	18.84
11	150	1093.37	1114.64	1122.43	11.02	1095.94	1108.45	18.30	1097.75	1105.37	14.63	**1093.37**	1100.95	21.20
12	150	1152.32	1159.88	1174.89	14.00	1155.09	1163.67	23.21	1153.04	1159.11	18.44	**1152.32**	1158.88	25.78
13	199	1287.18	1319.38	1332.55	18.71	1304.77	1314.52	36.33	1301.22	1310.78	26.15	1298.89	1305.83	43.94
14	199	1339.36	1380.86	1395.50	20.07	1357.05	1367.50	39.96	1351.23	1362.02	30.02	**1339.36**	1354.04	45.57
15	199	1420.72	1454.10	1462.23	25.14	1430.38	1443.45	52.53	**1420.72**	1436.29	37.12	1423.91	1437.52	59.83
16	120	1002.49	1003.99	1005.88	8.13	**1002.49**	1003.51	12.12	**1002.49**	1003.82	9.99	**1002.49**	1003.07	14.73
17	120	1026.20	1045.08	1050.86	8.09	1042.53	1042.99	11.86	1042.53	1044.76	9.41	1042.46	1042.61	13.17
18	120	1098.15	1121.07	1128.51	7.73	1114.33	1121.00	11.16	1113.18	1120.02	9.01	1113.07	1118.63	12.69
19	100	813.30	817.11	820.94	3.82	814.73	820.45	4.74	813.72	820.35	4.18	813.50	819.81	5.21
20	100	848.93	860.12	861.34	4.21	860.12	860.12	5.28	860.12	860.12	4.49	860.12	860.12	5.62
21	100	909.06	912.35	913.62	4.59	**909.06**	909.60	5.83	**909.06**	910.33	5.06	**909.06**	909.06	6.31
Avg. gap above BKS			1.29%	2.26%		0.60%	1.11%		0.48%	1.09%		0.36%	0.84%	
NBKS			2			7			7			12		
Avg. time (min)					7.53			12.73			9.87			14.58

Table 5
Comparison of GRASP/VNS with EvPR against other approaches from the literature.

Problem				Chao [7] (tabu search)		Scheuerer [45] (tabu search)				Lin et al. [32] (simulated annealing)				Caramia and Guerriero [5] (math. prog. heuristic)		GRASP/VNS + EvPR			
Number	n	BKS	Reference	Avg. Cost	Gap	Best cost	Gap	Avg. Cost	Gap	Best cost	Gap	Avg. Cost	Gap	Cost	Gap	Best cost	Gap	Avg. Cost	Gap
1	50	564.68	[45]	565.02	0.06	566.80	0.38	567.98	0.59	566.82	0.38	568.86	0.74	566.80	0.38	**564.68**	0.00	565.99	0.23
2	50	611.53	[32]	662.84	8.39	615.66	0.68	619.35	1.28	612.75	0.20	617.48	0.97	620.15	1.41	**611.53**	0.00	614.23	0.44
3	50	618.04	[45]	664.73	7.56	620.78	0.44	629.59	1.87	**618.04**	0.00	620.50	0.40	632.48	2.34	**618.04**	0.00	618.04	0.00
4	75	798.53	[45]	857.84	7.43	801.60	0.38	809.13	1.33	808.84	1.29	817.71	2.40	803.32	0.60	**798.53**	0.00	803.51	0.62
5	75	839.62	[45]	949.98	13.14	**839.62**	0.00	858.98	2.31	**839.62**	0.00	858.95	2.30	842.50	0.34	**839.62**	0.00	841.63	0.24
6	75	930.64	[32]	1084.82	16.57	936.01	0.58	949.89	2.07	934.11	0.37	942.60	1.29	938.18	0.81	940.59	1.07	961.47	3.31
7	100	830.48	[45]	837.80	0.88	**830.48**	0.00	832.91	0.29	**830.48**	0.00	838.50	0.97	832.56	0.25	**830.48**	0.00	830.48	0.00
8	100	872.56	[32]	906.16	3.85	878.87	0.72	881.26	1.00	875.76	0.37	882.70	1.16	878.87	0.72	**872.56**	0.00	876.21	0.42
9	100	912.02	[32]	1000.27	9.68	942.31	3.32	955.95	4.82	912.64	0.07	921.97	1.09	980.42	7.50	914.23	0.24	918.45	0.71
10	150	1039.07	[45]	1076.88	3.64	1039.23	0.02	1052.65	1.31	1053.90	1.43	1074.38	3.40	1060.41	2.05	1046.71	0.74	1050.11	1.06
11	150	1093.37	(a)	1170.17	7.02	1098.84	0.50	1107.47	1.29	1093.57	0.02	1108.88	1.42	1170.70	7.07	**1093.37**	0.00	1100.95	0.69
12	150	1152.32	(a)	1217.01	5.61	1175.23	1.99	1184.58	2.80	1155.44	0.27	1166.59	1.24	1178.34	2.26	**1152.32**	0.00	1158.88	0.57
13	199	1287.18	[45]	1364.50	6.01	1288.46	0.10	1296.33	0.71	1320.21	2.57	1340.98	4.18	1288.46	0.10	1298.89	0.91	1305.83	1.45
14	199	1339.36	(a)	1464.20	9.32	1371.42	2.39	1384.13	3.34	1351.54	0.91	1367.91	2.13	1372.52	2.48	**1339.36**	0.00	1354.04	1.10
15	199	1420.72	(b)	1544.21	8.69	1459.55	2.73	1488.71	4.79	1436.78	1.13	1454.91	2.41	1470.21	3.48	1423.91	0.22	1437.52	1.18
16	120	1002.49	[45]	1064.89	6.22	**1002.49**	0.00	1003.00	0.05	1004.47	0.20	1007.26	0.48	1004.69	0.22	**1002.49**	0.00	1003.07	0.06
17	120	1026.20	[32]	1104.67	7.65	1042.35	1.57	1042.79	1.62	1026.88	0.07	1035.23	0.88	1042.35	1.57	1042.46	1.58	1042.61	1.60
18	120	1098.15	[32]	1202.00	9.46	1129.16	2.82	1141.94	3.99	1099.09	0.09	1110.13	1.09	1129.16	2.82	1113.07	1.36	1118.63	1.86
19	100	813.30	[32]	887.22	9.09	813.50	0.02	813.98	0.08	814.07	0.09	823.01	1.19	813.50	0.02	813.50	0.02	819.81	0.80
20	100	848.93	[45]	963.06	13.44	**848.93**	0.00	852.89	0.47	855.14	0.73	859.06	1.19	**848.93**	0.00	860.12	1.32	860.12	1.32
21	100	909.06	[45]	952.29	4.76	**909.06**	0.00	914.04	0.55	**909.06**	0.00	915.38	0.70	**909.06**	0.00	**909.06**	0.00	909.06	0.00
Average cost				1025.74				970.84				968.24		970.65				961.46	
Avg. gap above BKS					7.55%		0.89%		1.74%		0.48%		1.51%		1.74%		0.36%		0.84%
NBKS						5				4				2		12			

(a) GRASP/VNS + EvPR, (b) GRASP/VNS with PR (post-optimization).

Table 6
Data for the Friedman test on the ranking of the average performance of each metaheuristic.

Problem number	Chao [7]		Scheuerer [45]		Lin et al. [32]		GRASP/VNS + EvPR	
	Avg. Cost	Rank	Avg. cost	Rank	Avg. cost	Rank	Avg. cost	Rank
1	565.02	1	567.98	3	568.86	4	565.99	2
2	662.84	4	619.35	3	617.48	2	614.23	1
3	664.73	4	629.59	3	620.50	2	618.04	1
4	857.84	4	809.13	2	817.71	3	803.51	1
5	949.98	4	858.98	3	858.95	2	841.63	1
6	1084.82	4	949.89	2	942.60	1	961.47	3
7	837.80	3	832.91	2	838.50	4	830.48	1
8	906.16	4	881.26	2	882.70	3	876.21	1
9	1000.27	4	955.95	3	921.97	2	918.45	1
10	1076.88	4	1052.65	2	1074.38	3	1050.11	1
11	1170.17	4	1107.47	2	1108.88	3	1100.95	1
12	1217.01	4	1184.58	3	1166.59	2	1158.88	1
13	1364.50	4	1296.33	1	1340.98	3	1305.83	2
14	1464.20	4	1384.13	3	1367.91	2	1354.04	1
15	1544.21	4	1488.71	3	1454.91	2	1437.52	1
16	1064.89	4	1003.00	1	1007.26	3	1003.07	2
17	1104.67	4	1042.79	3	1035.23	1	1042.61	2
18	1202.00	4	1141.94	3	1110.13	1	1118.63	2
19	887.22	4	813.98	1	823.01	3	819.81	2
20	963.06	4	852.89	1	859.06	2	860.12	3
21	952.29	4	914.04	2	915.38	3	909.06	1
Average Rank		3.81		2.29		2.43		1.48
Sum of Ranks		80		48		51		31
Squared Sum of Ranks		6400		2304		2601		961

every γ iterations. We also included in the computational experiment a GRASP/VNS (without PR) as a base case (benchmark) to analyze the contributions of PR. After some preliminary experimentation we set the parameters of the different variants of the hybrid metaheuristic to the values summarized in Table 3.

Table 4 presents the best and average results over 10 runs of the GRASP/VNS with PR variants and the GRASP/VNS benchmark. The column labeled *Time* reports the average running time in minutes for each method. We also include the best-known solutions (*BKS*) for each instance, taken from Lin et al. [32] and Scheuerer [45] and updated with some new best-known solutions found by the proposed GRASP/VNS with PR. The last rows of the table summarize the average gap above best-known solutions, the number of times each method found the best-known solution (*NBKS*), and the average running time. Values in bold in the table indicate that the BKS was found by a given method.

All the variants of GRASP/VNS with PR largely outperform the base case GRASP/VNS (without PR), highlighting the contribution of PR to the quality of solutions. Remarkably, the use of PR as a post-optimization mechanism offers a good trade-off between running time and solution quality. The post-optimization with PR approximately halved the average gap to BKS of GRASP/VNS with an increase of only 30% in the running time. Moreover, this variant was able to improve the BKS of problem 15.[1] Above all, GRASP/VNS with EvPR stands as the best performing method, having an average gap to BKS as small as 0.84% obtaining 12 out of 21 BKS, and improving the BKS of problems 11, 12 and 14.[1] However, this outstanding performance is achieved at the price of almost doubling the running time of the benchmark GRASP/VNS.

Table 5 presents the comparison of the proposed hybrid metaheuristic against other methods from the literature. In this table we only compare against the best variant, namely GRASP/VNS with EvPR. Table 4 includes the results of the tabu search of Chao [7] and Scheuerer [45], the simulated annealing of Lin et al. [32], and the mathematical-programming-based heuristic of Caramia and Guerriero [5]. Depending of the availability of results we report the best and average results over 10 runs of each metaheuristic. For the heuristic of Caramia and Guerriero, we only report the results of a single run of their deterministic method. Column *BKS* presents the best known solution for each problem reported for the first time in the paper cited in column *Ref*, column *Gap* reports the gap with respect to BKS (in %) for each instance and each method. The last rows of Table 5 summarize the average cost over the 21 test problems, the average gap above best known solutions (*BKS*), and the number of times each method found the best-known solution (*NBKS*).

As can be seen in Table 5, GRASP/VNS with EvPR outperforms all the methods from the literature, achieving a small average gap to BKS of 0.84% and obtaining 12 out of 21 BKS with a single set of parameters. Our method improved the BKS for four large problems. GRASP/VNS with EvPR almost halved the average gap to BKS of the simulated annealing heuristic of Lin et al. [32], the previous best method with an average gap to BKS of 1.51%, and the second-best methods by Scheuerer [45] and Caramia and Guerriero [5], which obtained the same average gap to BKS of 1.74%. Finally, the tabu search of Chao [7] with an average gap to BKS of 7.55% is clearly outperformed by GRASP/VNS with EvPR.

It is important to note that the worst performance of EvPR (as for the other variants of the hybrid metaheuristic) is obtained on problem 6. By analyzing some statistics during the search, we observed that due to the very tight demand to capacity ratio (0.974) it is very difficult to find feasible solutions with the proposed RFCS approach. In contrast, the method by Caramia and Guerriero [5] is better adapted to solve this problem since it has a packing step that produces a feasible solution if any exists.

Some authors used the average cost over the 21 instances as a measure to compare the performance of different metaheuristics for the TTRP [5,32]. Nonetheless, this is not a good measure because it favors methods with good results in the larger test problems with overall large cost due solely to their size. For instance, the cost of the BKS of problem 15 is 2.5 times the cost of the BKS of problem 1. Then an improvement of 1% of the BKS of problem 15 will have 2.5 times more impact in this measure than the same relative

[1] Detailed solutions available at http://hdl.handle.net/1992/1127

improvement on problem 1. Hence, to have a better comparison of the algorithms, we followed Garcia et al. [19] and used the Friedman test to analyze the average results of the randomized metaheuristics.

The null hypothesis of the Friedman test is that each ranking of the algorithms within each problem is equally likely, so there is no difference between them. As can be seen in Table 6, GRASP/VNS with EvPR consistently ranks in the first two positions. The Friedman test was performed according to the procedure described by Conover [8] and the analysis led to the rejection of the null hypothesis with a level of significance $\alpha = 1\%$. Moreover, the paired comparisons unveiled that GRASP/VNS with EvPR is better than each one of the other methods with the same level of significance.

Since the method of Caramia and Guerriero [5] is deterministic, only one run is enough to characterize its performance. To have a fair comparison against it, in Table 7 we compared their results against those of the best and worst runs (i.e., the seeds that produce the smallest and biggest average deviations above BKS, respectively). In the

Table 7
Comparison of a single run of GRASP/VNS with EvPR and Caramia and Guerriero's heuristic.

Problem				Caramia and Guerrriero		Best seed		Worst seed	
Number	n	BKS	Reference	Cost	Gap	Cost	Gap	Cost	Gap
1	50	564.68	[45]	566.80	0.38	566.09	0.25	566.82	0.38
2	50	611.53	[32]	620.15	1.41	613.61	0.34	615.18	0.60
3	50	618.04	[45]	632.48	2.34	618.04	0.00	618.04	0.00
4	75	798.53	[45]	803.32	0.60	799.34	0.10	798.87	0.04
5	75	839.62	[45]	842.50	0.34	843.05	0.41	842.47	0.34
6	75	930.64	[32]	938.18	0.81	948.42	1.91	980.34	5.34
7	100	830.48	[45]	832.56	0.25	830.48	0.00	830.48	0.00
8	100	872.56	[32]	878.87	0.72	879.51	0.80	881.17	0.99
9	100	912.02	[32]	980.42	7.50	914.23	0.24	920.07	0.88
10	150	1039.07	[45]	1060.41	2.05	1051.70	1.22	1053.22	1.36
11	150	1093.37	(a)	1170.70	7.07	1106.41	1.19	1099.05	0.52
12	150	1152.32	(a)	1178.34	2.26	1152.32	0.00	1162.16	0.85
13	199	1287.18	[45]	1288.46	0.10	1309.69	1.75	1307.35	1.57
14	199	1339.36	(a)	1372.52	2.48	1347.61	0.62	1361.63	1.66
15	199	1420.72	(b)	1470.21	3.48	1424.31	0.25	1438.61	1.26
16	120	1002.49	[45]	1004.69	0.22	1002.82	0.03	1002.49	0.00
17	120	1026.20	[32]	1042.35	1.57	1042.53	1.59	1042.63	1.60
18	120	1098.15	[32]	1129.16	2.82	1113.07	1.36	1122.88	2.25
19	100	813.30	[32]	813.50	0.02	821.85	1.05	821.85	1.05
20	100	848.93	[45]	848.93	0.00	860.12	1.32	860.12	1.32
21	100	909.06	[45]	909.06	0.00	909.06	0.00	909.06	0.00
Avg. gap above BKS					1.74%		0.69%		1.05%
Number of times better than Caramia and Guerriero						13		13	

(a) GRASP/VNS + EvPR, (b) GRASP/VNS with PR (post-optimization).

Table 8
Comparison of the running time of GRASP/VNS with EvPR and other published methods.

Problem		GRASP/VNS with EvPR		Scheuerer [45]		Lin et al. [32]	
Number	n	Avg. time	Scaled time ($\times 2.15$)	Avg. time	Speed-up	Avg. time	Speed-up
1	50	1.17	2.51	9.51	3.79	6.80	2.71
2	50	1.29	2.77	9.60	3.47	6.67	2.41
3	50	1.05	2.27	11.24	4.95	5.59	2.46
4	75	2.69	5.79	18.49	3.20	16.32	2.82
5	75	2.82	6.07	15.16	2.50	14.42	2.37
6	75	2.89	6.22	18.62	2.99	13.65	2.19
7	100	6.05	13.02	33.60	2.58	24.96	1.92
8	100	6.96	14.97	25.66	1.71	24.03	1.61
9	100	8.38	18.03	30.47	1.69	21.75	1.21
10	150	18.84	40.54	60.94	1.50	63.61	1.57
11	150	21.20	45.62	56.17	1.23	60.33	1.32
12	150	25.78	55.49	63.71	1.15	51.70	0.93
13	199	43.94	94.56	165.41	1.75	119.56	1.26
14	199	45.57	98.08	132.06	1.35	113.75	1.16
15	199	59.83	128.76	154.10	1.20	93.87	0.73
16	120	14.73	31.69	43.14	1.36	41.46	1.31
17	120	13.17	28.35	33.73	1.19	38.81	1.37
18	120	12.69	27.31	31.78	1.16	31.34	1.15
19	100	5.21	11.21	28.84	2.57	29.58	2.64
20	100	5.62	12.09	24.57	2.03	28.47	2.36
21	100	6.31	13.58	26.84	1.98	24.03	1.77
Average		14.58	31.38	47.32		39.56	
Geometric mean					1.96		1.66

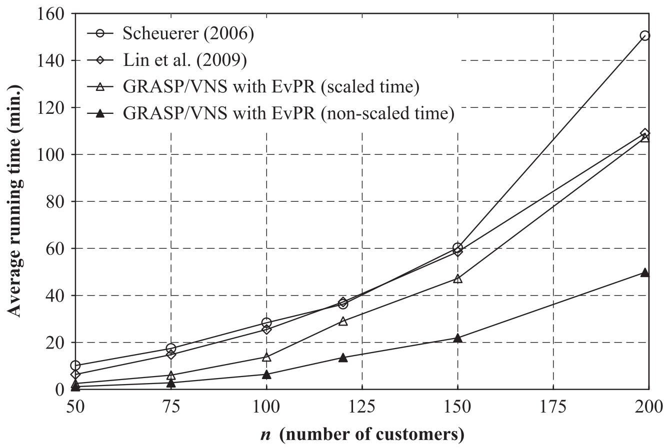

Fig. 5. Comparison of average running time of GRASP/VNS with EvPR and other published methods.

last row of Table 7 it is possible to see that GRASP/VNS with EvPR is consistently better than Caramia and Guerriero's method in 13 out of 21 problems, regardless of the seed. Even though, the best seed has a much smaller gap to BKS of 0.69% compared to 1.74% of their method, the worst seed is still better than their method with a gap to BKS of 1.05%. This experiment also highlights the robustness of GRASP/VNS with EvPR, that is, the performance of a single run is very close to the average performance.

To compare the running times of GRASP/VNS with EvPR against those reported in the literature we chose the tabu search of Scheuerer [45] and the simulated annealing of Lin et al. [32]. The tabu search of Chao [7] was discarded because it has a large gap to BKS and, unfortunately, Caramia and Guerriero [5] did not report running times.

To have a fair comparison of the running times we scaled the time spent by GRASP/VNS with EvPR to the reference computer used by Scheuerer [45] and Lin et al. [32]. Both of them used an Intel Pentium IV PC running at 1.5 GHz. Scheuerer [45] reported a speed factor of around 326 Mflop/s (millions of floating-point operations per second) for this computer using the Linpack benchmark [11]. Using the Java version of the Linpack benchmark [12], we obtained a speed factor of approximately 702 Mflop/s for our Intel Xeon running at 2.67 GHz. Using these values we derived a scaling factor of 2.15 for our running times.

Table 8 shows that GRASP/VNS with EvPR has in general shorter running times. The column labeled *Speed-up* reports the ratio of the running times of the published algorithms over the scaled time of GRASP/VNS with EvPR. Following the arguments outlined by Bixby [2], we report in the last row the geometric mean of these ratios as a conservative estimate of the speed-up factor achieved by GRASP/VNS with EvPR with respect to the tabu search of Scheuerer [45] and the simulated annealing of Lin et al. [32]. The comparison of Table 8 is further illustrated through Fig. 5. However, this running time comparison must be taken with care since the operating system, computer configuration, and programming language varies for each method.

5. Conclusions

In this paper we proposed a hybrid metaheuristic based on GRASP, VNS and path relinking to solve the truck and trailer routing problem. The constructive phase and the path relinking operator are based on a route-first, cluster-second approach. During the search the proposed metaheuristic explores infeasible solutions while the VNS component plays the role of repairing operator and improving mechanism. The computational experiments on a set of 21 standard test instances from the literature unveils the accuracy of the proposed GRASP/VNS with path relinking and the contribution of path relinking to the quality of solutions. Moreover, the proposed hybrid metaheuristic outperforms all previous published methods, and exhibits a very small variability when comparing the results of a single run against the average results over several runs.

After exploring different hybridization alternatives for the path relinking component, GRASP/VNS with evolutionary path relinking emerged as the overall winner, achieving a small average gap to best-known solutions of 0.84%, finding 12 out of 21 best known solutions, and improving 3 of them with a single set of parameters. The GRASP/VNS with path relinking as post-optimization mechanism variant is 32% faster than GRASP/VNS with evolutionary path relinking without a significant sacrifice on the quality of the results, achieving an average gap to best-known solution of 1.09%, and improving the best-known solution of one of the larger problems.

Further research directions include the development of lower bounds and exact methods to solve the TTRP, and the study of the multi-objective TTRP in which the fleet size (number of trucks and number of trailers), and the total distance are used as objective functions.

Acknowledgments

The authors would like to thank Maria Elsa Correal and Maria Isabel Restrepo at the Universidad de Los Andes for their statistical advice. This work was partly founded by the Champagne-Ardenne Regional Council (France) and by Colciencias, (Colombia). This support is gratefully acknowledged.

Appendix A. Notation

The symbols used throughout the paper are summarized in Tables A1–A4.

Appendix B. Solution of the restricted STTRPSD

The cost of vehicle routes with subtours is obtained by solving a restricted version of the STTRPSD (Villegas et al. [54]). Firstly, we recall the notation. The solution of the restricted STTRPSD is obtained with a dynamic programming method in which state $[l,m](i \le l \le j; l : t_l \in N_v; l \le m \le j)$ represents the use of vehicle customer t_l as the root of a subtour ending at customer t_m. F_{lm} denotes the cost of state $[l,m]$, and θ_{lkm} is the cost of a subtour $ST=(t_l,t_k,\ldots,t_m,t_l)$ rooted at vehicle customer t_l and visiting customer

Table A1
Notation for the definition of the TTRP.

Symbol	Description
m_t	Number of available trucks
m_r	Number of available trailers
Q_t	Truck capacity
Q_r	Trailer capacity
$N=\{1,\ldots,n\}$	Set of customers
N_v	Subset of vehicle customers
N_t	Subset of truck customers
q_i	Demand for customer i
c_{ij}	Distance between nodes i and j

Table A2
Notation for the route-first, cluster-second procedure.

Symbol	Description
$T=(0,t_1,\ldots,t_i,\ldots,t_n,0)$	Giant tour
$H=(X,U,W)$	Auxiliary graph for the tour splitting procedure
X	Set of nodes of the auxiliary graph
U	Set of arcs of the auxiliary graph
W	Weight of the arcs in the auxiliary graph
(i,j)	Arc of the auxiliary graph
w_{ij}	Cost of arc (i,j)
$R_{ij}=(0,t_i,\ldots,t_j,0)$	Route serving from customer t_i to customer t_j of giant tour T
$c(R_{ij})$	Cost (total distance) of route R_{ij}
Q_{ij}	Total demand of route R_{ij}
$[l,m]$	State of the dynamic programming method used for the cost of vehicle routes with subtours
F_{lm}	Cost of state $[l,m]$
θ_{lkm}	Cost of subtour $ST=(t_l,t_k,\ldots,t_m,t_l)$
α_{ij}	Truck consumption of arc $(i,j)\in U$
β_{ij}	Trailer consumption of arc $(i,j)\in U$
$\Lambda=(\delta,\tau,\rho,\nu,\lambda)$	Label for the solution of the resource-constrained shortest path problem
δ	Distance of label Λ
τ	Truck consumption of label Λ
ρ	Trailer consumption of label Λ
ν	Father node of label Λ
λ	Father label of label Λ
$\mathcal{L}_i$	Set of labels of node i in the auxiliary graph
$\Gamma(i)$	Set of successors of node i in the auxiliary graph

Table A3
Notation of VNS and path relinking.

Symbol	Description
S	TTRP solution
S_0	Initial solution
b	Number of pairs of customers exchanged in the perturbation procedure of VNS
$\Phi(S)$	Infeasibility of solution S
μ	Infeasibility threshold
$ut(S)$	Number of trucks used in solution S
$ur(S)$	Number of trailers used in solution S
$T(S)$	Giant tour of solution S
ES	Pool of elite solutions
$d(S,S')$	Distance between solutions S and S'
$d(\mathbf{ES},S)$	Distance between solution S and the pool **ES**
$f'(S)$	Modified objective function for the ordering of the pool
S_f	Final solution of the path relinking operator
P, **RS**	Sets of solutions produced by path relinking

Table A4
Parameters of the hybrid metaheuristic.

Symbol	Description
ns	Number of GRASP iterations
κ	Cardinality of the restricted candidate list of GRASP
ni	Number of VNS iterations
μ_{max}	Maximum infeasibility threshold of VNS
b_{max}	Maximum number of pairs for the shaking of VNS
Δ	Minimum distance threshold in path relinking
$\vert$**ES**$\vert$	Cardinality of the pool of elite solutions
γ	Frequency of evolutionary path relinking

t_k to t_m. It is possible to find the structure of the route and its cost using the following recurrence relation:

$$F_{lj}=\begin{cases} c_{0t_i} & \text{if } l=i \text{ and } m=i \\ \min\limits_{k<m:\sum_{u=k+1}^{m} q_{t_u}\le Q_t}\{F_{lk}+\theta_{i,k+1,m}\} & \text{if } l=i \text{ and } i<m\le j \\ \min\left\{\min\limits_{k<m:\sum_{u=k+1}^{m} q_{t_u}\le Q_t}\{F_{lk}+\theta_{l,k+1,m}\},\ \min\limits_{\substack{h<l:t_h\in N_v,\\ k=l-1:\sum_{u=k+2}^{m} q_{t_u}\le Q_t}}\{F_{hk}+c_{t_h,t_l}+\theta_{l,k+2,m}\}\right\} & \text{if } i<l\le j: t_l\in N_v, \text{ and } l\le m\le j \end{cases}$$

Since states $[l,m]$, do not include the return from the last vehicle customer to the main depot, the cost of the route is finally calculated as $c(R_{ij})=\min_{i\le l\le j:t_l\in N_v}\{F_{lj}+c_{t_l,0}\}$.

Following the same approach as Villegas et al. [54], the dynamic programming method for the STTRPSD can be represented by an auxiliary graph $G=(V,A,Z)$. The node set V is composed of the nodes

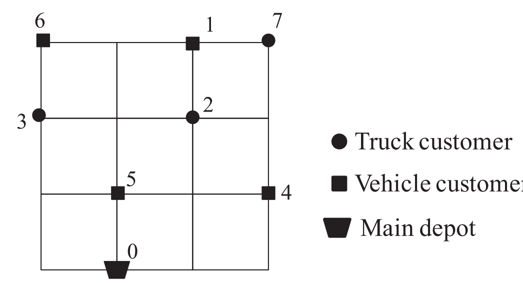

Fig. B1. Data for the solution of the restricted STTRPSD.

representing states $[l,m]$, and a dummy node ω representing the return to the main depot.

The arc set A contains three types of arcs. Arcs $([l,k],[l,m]),i\le l\le j,t_l\in N_v; k<m\le j$ represent a subtour that serves customers $(t_{k+1},\ldots,t_m)$ rooted at vehicle customer t_l without moving the trailer that is already parked at vehicle customer t_l; the cost of these arcs is given by $\theta_{l,k+1,m}$. Arcs of the form $([h,k],[l,m]),i\le h<l\le j,t_h,t_l\in N_v; k=l-1\le j:\sum_{u=k+2}^{m} q_{t_u}\le Q_t$ represent a subtour that serves customers $(t_{k+2},\ldots,t_m)$ rooted at vehicle customer t_l coming from vehicle customer t_h after performing the subtour that ends at customer t_k; the cost of these arcs is $c_{t_h,t_l}+\theta_{l,k+2,m}$. Finally, we have arcs of the form $([l,j],\omega),t_l\in N_v$ representing the return of the complete vehicle to the main depot after serving the last customer of the route in a subtour rooted at vehicle customer t_l. The cost of these arcs is $c_{t_l,0}$.

We obtain the structure of the route and its cost by finding in G the shortest path from state $[i,i]$ to node ω. As shown in Villegas et al. [54], this problem can be solved efficiently without generating explicitly the auxiliary graph G. A small example follows. The location of the customers and its type is given in Fig. B1, the length of each square side in the grid is equal to 1 and the distance between nodes is Euclidean.

Given a giant tour $T=(0,6,5,3,1,2,4,7,0)$ for a TTRP with $Q_r=2$, $Q_t=3$ and unitary demands. We are interested in the structure and cost of route $R_{2,6}=(0,5,3,1,2,4,0)$. The auxiliary graph G is given in Fig. B2(a), the arcs in bold correspond to the shortest path, and the associated solution is given in Fig. B2(b). Note that in the auxiliary graph we replaced state $[l,m]$ with $[t_l,t_m]$ to simplify the presentation.

Table B1 details the calculation of the cost of the arcs in the shortest path. Note that the cost of the route is 11.30, while the cost

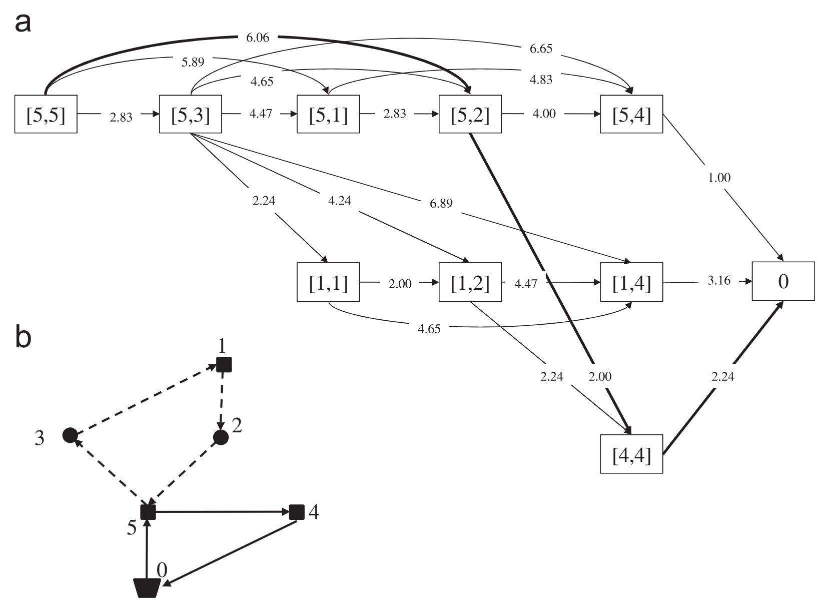

Fig. B2. Example of the restricted STTRPSD used to find the cost of vehicle routes with subtours: (a) Auxiliary graph. (b) Vehicle route with subtours.

Table B1
Example of the cost of the arcs of G for the STTRPSD.

Arc	Cost	Calculation	Value
([5,5],[5,2])	$\theta_{2,3,5} = c_{t_2,t_3} + c_{t_3,t_4} + c_{t_4,t_5} + c_{t_5,t_2}$	$c_{5,3} + c_{3,1} + c_{1,2} + c_{2,5} = 1.41 + 2.24 + 1.00 + 1.41$	6.06
([5,2],[4,4])	$c_{t_2,t_6} + \theta_{6,6,6}$	$c_{5,4} + 0 = 2.00 + 0$	2.00
([4,4],0)	$c_{t_6,0}$	$c_{4,0}$	2.24
Total			10.30

of the shortest path is 10.30 this is because it is necessary to add the cost of state $[i,i] = [2,2]$, that represents the departure from the main depot, in the recursion this is the first state with $F_{2,2} = c_{0,t_2} = c_{0,5} = 1.00$. Then, $c(R_{2,6}) = 11.30$.

References

[1] Beasley JE. Route-first cluster-second methods for vehicle routing. Omega 1983;11:403–8.
[2] Bixby R. Solving real-world linear programs: a decade and more of progress. Operations Research 2002;50(1):3–15.
[3] Boudia M, Louly M, Prins C. A reactive GRASP and path relinking for a combined production-distribution problem. Computers & Operations Research 2007;34(11): 3402–19.
[4] Campos V, Laguna M, Martí R. Context-independent scatter and tabu search for permutation problems. INFORMS Journal on Computing 2005;17(1):111–22.
[5] Caramia M, Guerriero F. A heuristic approach for the truck and trailer routing problem. Journal of the Operational Research Society 2010;61:1168–80.
[6] Caramia M, Guerriero F. A milk collection problem with incompatibility constraints. Interfaces 2010;40(2):130–43.
[7] Chao I-M. A tabu search method for the truck and trailer routing problem. Computers & Operations Research 2002;29:33–51.
[8] Conover W. Practical nonparametric statistics. Wiley; 1998.
[9] Cordeau J-F, Gendreau M, Laporte G. A tabu search heuristic for periodic and multi-depot vehicle routing problems. Networks 1997;30:105–19.
[10] Desrosiers J, Soumis F, Desrochers M. Routing with time windows by column generation. Networks 1984;14(4):545–65.
[11] Dongarra JJ. Performance of various computers using standard linear equations software. Technical Report CS89-95, University of Tennessee, Computer Science; 2009.
[12] Dongarra JJ, Wade R, McMahan P. Linpack benchmark—Java version. ⟨http://www.netlib.org/benchmark/linpackjava/⟩; 2010.
[13] Drexl M. On some generalized routing problems. PhD thesis, Faculty of Business and Economics, RWTH Aachen University; 2007.
[14] Duhamel C, Lacomme P, Prins C, Prodhon C. A GRASP X ELS approach for the capacitated location-routing problem. Computers & Operations Research 2010;37(11):1912–23.
[15] Ehrgott M. Multicriteria optimization. Springer; 2000.
[16] El-Fallahi A, Prins C, Wolfler-Calvo R. A memetic algorithm and a tabu search for the multi-compartment vehicle routing problem. Computers & Operations Research 2008;35:1725–41.
[17] Feo TA, Resende MGC. Greedy randomized adaptive search procedures. Journal of Global Optimization 1995;6(2):109–33.
[18] Flood MM. The traveling-salesman problem. Operations Research 1956;4(1): 61–75.
[19] García S, Molina D, Lozano M, Herrera F. A study on the use of non-parametric tests for analyzing the evolutionary algorithms's behaviour: a case study on the CEC' 2005 special session on real parameter optimization. Journal of Heuristics 2009;15:617–44.
[20] Gendreau M, Potvin J-Y, Bräysy O, Hasle G, Løkketangen A. Metaheuristics for the vehicle routing problem and its extensions: a categorized bibliography. In: Golden B, Raghavan S, Wasil E, editors. The vehicle routing problem: latest advances and new challenges. Springer; 2008. p. 143–70.
[21] Gerdessen JC. Vehicle routing problem with trailers. European Journal of Operational Research 1996;93:135–47.
[22] Gillett BE, Miller LR. A heuristic algorithm for the vehicle-dispatch problem. Operations Research 1974;22(2):340–9.
[23] Glover F, Laguna M. Tabu search. Kluwer; 1997.
[24] Golden BL, Raghavan S, Wasil E. The Vehicle routing problem: latest advances and new challenges. Springer; 2008.
[25] Hansen P, Mladenovic N. Variable neighbourhood search: principles and applications. European Journal of Operational Research 2001;130:449–67.
[26] Hashimoto H, Yagiura M. A path relinking approach with an adaptive mechanism to control parameters for the vehicle routing problem with time windows. In van Hemert JI, Cotta C, editors, EvoCOP, Lecture notes in computer science, vol. 4972. Springer; 2008. p 254–65.
[27] Ho SC, Gendreau M. Path relinking for the vehicle routing problem. Journal of Heuristics 2006;12:55–72.
[28] Hoff A, Løkketangen A. A tabu search approach for milk collection in western Norway using trucks and trailers. In TRISTAN VI: The sixth triennial symposium on transportation analysis. Phuket, Tailand; June 10–15, 2007.
[29] Laguna M, Marti R. GRASP and path relinking for 2-layer straight line crossing minimization. INFORMS Journal on Computing 1999;11(1):44–52.

[30] Laporte G. What you should know about the vehicle routing problem. Naval Research Logistics 2007;54:811–9.
[31] Lin S-W, Yu VF, Chou S-Y. A note on the truck and trailer routing problem. Expert Systems with Applications 2010;37(1):899–903.
[32] Lin S-W, Yu VF, Chou S-Y. Solving the truck and trailer routing problem based on a simulated annealing heuristic. Computers & Operations Research 2009;36(5):1683–92.
[33] Or I. Traveling salesman-type combinatorial optimization problems and their relation to the logistics of regional blood banking. PhD thesis, Department of Industrial Engineering and Management Science, Northwestern University, Evanston, IL; 1976.
[34] Parragh SN, Doerner KF, Hartl RF, Gandibleux X. A heuristic two-phase solution approach for the multi-objective dial-a-ride problem. Networks 2009;54(4): 227–42.
[35] Prins C. A simple and effective evolutionary algorithm for the vehicle routing problem. Computers & Operations Research 2004;31:1985–2002.
[36] Prins C. A GRASP X evolutionary local search hybrid for the vehicle routing problem. In: Pereira FB, Tavares J, editors. Bio-inspired algorithms for the vehicle routing problem. Springer; 2009. p. 35–53.
[37] Prins C. Two memetic algorithms for heterogeneous fleet vehicle routing problems. Engineering Applications of Artificial Intelligence 2009;22:916–28.
[38] Prins C, Prodhon C, Wolfler-Calvo R. Solving the capacitated location-routing problem by a GRASP complemented by a learning process and a path relinking. 4OR—A Quarterly Journal of Operations Research 2006;4:221–38.
[39] Reghioui M, Prins C, Labadi N. GRASP with path relinking for the capacitated arc routing problem with time windows. In: Giacobini M, Brabazon A, Cagnoni S, Caro GD, Drechsler R, Farooq M, Fink A, Lutton E, Machado P, Minner S, O'Neill M, Romero J, Rothlauf F, Squillero G, Takagi H, Uyar S, Yang S, editors. EvoWorkshops, Lecture notes in computer science, vol. 4448. Springer; 2007. p. 722–31.
[40] Resende MGC, Martí R, Gallego M, Duarte A. GRASP and path relinking for the maxmin diversity problem. Computers & Operations Research 2010;37(3): 498–508.
[41] Resende MGC, Ribeiro CC. GRASP with path-relinking: recent advances and applications. In: Ibaraki T, Nonobe K, Yagiura M, editors. Metaheuristics: progress as real problem solvers. Springer; 2005. p. 29–63.
[42] Resende MGC, Ribeiro CC. GRASP. AT&T Labs Research Technical Report, 2008.
[43] Resende MGC, Ribeiro CC. Greedy randomized adaptive search procedures: advances and applications. In: Gendreau M, Potvin J-Y, editors, Handbook of metaheuristics, 2nd ed., Springer; 2010, p. 281–317.
[44] Resende MGC, Werneck RF. A hybrid heuristic for the p-median problem. Journal of Heuristics 2004;10:59–88.
[45] Scheuerer S. A tabu search heuristic for the truck and trailer routing problem. Computers & Operations Research 2006;33:894–909.
[46] Schiavinotto T, Stutzle T. A review of metrics on permutations for search landscape analysis. Computers & Operations Research 2007;34(10):3143–53.
[47] Semet F. A two-phase algorithm for the partial accessibility constrained vehicle routing problem. Annals of Operations Research 1995;61:45–65.
[48] Semet F, Taillard E. Solving real-life vehicle routing problems efficiently using tabu search. Annals of Operations Research 1993;41:469–88.
[49] Sörensen K. Distance measures based on the edit distance for permutation-type representations. Journal of Heuristics 2006;13(1):35–47.
[50] Sörensen K, Sevaux M. A practical approach for robust and flexible vehicle routing using metaheuristics and Monte Carlo sampling. Journal of Mathematical Modelling and Algorithms 2009;8(4):387–407.
[51] Sörensen K, Sevaux M. Distance-based path relinking for the vehicle routing problem. In: TRISTAN VII: seventh triennial symposium on transportation analysis. Tromsø, Norway, June 20–25, 2010. p. 744–7.
[52] Souffriau W, Vansteenwegen P, Vanden Berghe G, VanOudheusden D. A path relinking approach for the team orienteering problem. Computers & Operations Research 2010;37(11):1853–9.
[53] Toth P, Vigo D. The vehicle routing problem. Philadelphia: SIAM; 2002.
[54] Villegas JG, Prins C, Prodhon C, Medaglia AL, Velasco N. GRASP/VND and multi-start evolutionary local search for the single truck and trailer routing problem with satellite depots. Engineering Applications of Artificial Intelligence 2010;23(5):780–94.