Engineering Applications of Artificial Intelligence 23 (2010) 780–794

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

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

GRASP/VND and multi-start evolutionary local search for the single truck and trailer routing problem with satellite depots

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 DC, Colombia
[c] Departamento de Ingeniería Industrial, Universidad de Antioquia, A.A. 1226, Medellin, Colombia

ARTICLE INFO

Article history:
Received 20 May 2009
Received in revised form
12 January 2010
Accepted 16 January 2010
Available online 18 February 2010

Keywords:
Truck and trailer routing problem (TTRP)
Multi-depot vehicle routing problem (MDVRP)
Greedy randomized adaptive search procedures (GRASP)
Evolutionary local search (ELS)
Variable neighborhood descent (VND)
Iterated local search (ILS)

ABSTRACT

In the single truck and trailer routing problem with satellite depots (STTRPSD) a vehicle composed of a truck with a detachable trailer serves the demand of a set of customers reachable only by the truck without the trailer. This accessibility constraint implies the selection of locations to park the trailer before performing the trips to the customers. We propose two metaheuristics based on greedy randomized adaptive search procedures (GRASP), variable neighborhood descent (VND) and evolutionary local search (ELS) to solve this problem. To evaluate these metaheuristics we test them on a set of 32 randomly generated problems. The computational experiment shows that a multi-start evolutionary local search outperforms a GRASP/VND. Moreover, it obtains competitive results when applied to the multi-depot vehicle routing problem (MDVRP), that can be seen as a special case of the STTRPSD.

1. Introduction

This paper presents the single truck and trailer routing problem with satellite depots (STTRPSD), a generalization of the vehicle routing problem (VRP). The VRP is a well known combinatorial optimization problem that aims to find a set of routes of minimum total length to serve the demand of a set of customers using a homogeneous fleet of capacitated vehicles based at a main depot (Laporte, 2007).

In the STTRPSD a single vehicle (a truck with a detachable trailer) based at a main depot serves the demand of a set of customers, reachable only by the truck without the trailer. Therefore, there is a set of parking locations (called trailer points or satellite depots) where it is possible to detach the trailer and to transfer products between the truck and the trailer. In feasible solutions of the STTRPSD, each client is assigned to one trailer point. Consequently, trailer points with assigned customers are said to be open. The first-level trip departing from the main depot is performed by the truck with the trailer and visits the subset of open trailer points. Each customer must be served by exactly one second-level trip (performed by the truck alone), starting and ending at the allocated trailer point. Thus, the total load in a second-level trip should not exceed the truck capacity. The goal of the STTRPSD is to minimize the total length of the trips. The STTRPSD is NP-hard since it includes the VRP (one satellite depot) and the multi-depot VRP (null cost between any two depots) as particular cases. The STTRPSD is a relevant problem that appears for instance in milk collection, where customers (farms) are often served by a single tanker with a removable tank trailer. Trailer points are in general parking locations on main roads, while farms are located on narrow roads inaccessible with the trailer. In reality, there can be several vehicles but usually farms are clustered based on their geographical location (Claassen and Hendriks, 2007), each cluster being assigned to one vehicle. Fig. 1 depicts a feasible solution of the STTRPSD.

Another suitable application for the arc routing counterpart of the STTRPSD appears in the design of park-and-loop routes for postal delivery (Levy and Bodin, 2000), where the postman drives a vehicle from the postal facility to a parking location, loads his sack, and delivers mail by walking the streets forming a loop. Then, he returns to the vehicle, loads his sack again, and delivers by foot in a second loop. Once he has served all the walking loops

* 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, juan_guillermo.villegas@utt.fr, jg.villegas64@uniandes.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).

0952-1976/$ - see front matter © 2010 Elsevier Ltd. All rights reserved.
doi:10.1016/j.engappai.2010.01.013

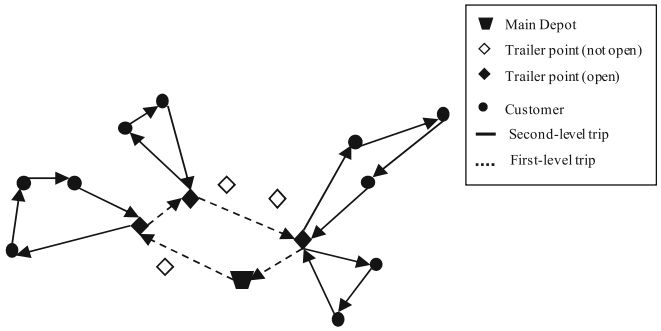

Fig. 1. Feasible solution for the STTRPSD.

nearby, he drives the vehicle to the next parking location. At the end of the day the postman returns to the postal facility.

The STTRPSD is also related to the capacitated arc routing problem with intermediate facilities (CARPIF) (Ghiani et al., 2001). The CARPIF is a variant of the capacitated arc routing problem (CARP) with a single vehicle but multiple trips, in which the vehicle is unloaded at intermediate facilities between two consecutive trips. Applications of the CARPIF appear in waste collection where intermediate facilities are dump sites or incinerators.

The STTRPSD generalizes the VRP. Since exact approaches solve consistently VRPs up to only 100 customers (Baldacci et al., 2007), the preferred solution methods are mainly heuristics (Laporte and Semet, 2002) or metaheuristics (Gendreau et al., 2008). VRP variants appear by incorporating new constraints (e.g., time windows, Cordeau et al., 2002) or by integrating the design of the routes with higher level decisions (e.g. location-routing problems, Nagy and Salhi, 2007). The books by Toth and Vigo (2002) and Golden et al. (2008) provide a wide overview of the VRP, its extensions, solution methods, and practical applications.

It is possible to see the STTRPSD as a variant of the truck and trailer routing problem (TTRP), another extension of the VRP in which a heterogeneous fleet comprised of m trucks and b trailers ($b<m$) is used to satisfy the demand of a set of customers partitioned in *vehicle* and *truck customers*, where the latter are only accessible by truck. The TTRP has been tackled using tabu search (Chao, 2002; Scheuerer, 2006), simulated annealing (Lin et al., 2009) and a mathematical-programming based heuristic (Caramia and Guerriero, 2009). The STTRPSD differs from the TTRP in the use of a single vehicle and the definition of trailer points independent from customer locations. Also, when the STTRPSD is extended to include several vehicles and capacitated satellite depots it transforms into the seldomly studied two-echelon capacitated vehicle routing problem (2E-CVRP) introduced by Gonzalez-Feliu et al. (2007).

The STTRPSD reduces to the multi-depot VRP (MDVRP) when all the distances between satellite depots are null. In the MDVRP, the fleet of vehicles is based at several depots, and the customers must be visited by exactly one route starting and ending at one of the depots. Recently, Baldacci and Mingozzi (2009) proposed a unified exact method capable of solving different variants of the VRP, including the MDVRP. Tabu search (Cordeau et al., 1997) and adaptive large scale neighborhood search (ALNS) (Pisinger and Ropke, 2007) provide the best results among the metaheuristics developed for the MDVRP. Crevier et al. (2007) extended the MDVRP allowing routes that may depart and end at different depots, giving rise to the multi-depot vehicle routing problem with inter-depot routes (MDVRPI).

Hoff and Løkketangen (2007) described a practical application of vehicle routing for milk collection in Norway in which the routes have the structure of a STTRPSD. For their particular problem they use tabu search to design the routes to collect the milk of a set of farms to supply various dairy plants. However, to the best of our knowledge, the STTRPSD has not been formally described in the literature, and no solution algorithm has been designed for it.

The remainder of this paper is organized as follows. Section 2 presents an integer programming formulation of the STTRPSD. Section 3 describes two metaheuristics based on greedy randomized adaptive search procedures and evolutionary local search. Section 4 presents a computational evaluation of the proposed methods. Finally, we conclude and outline future work in Section 5. Appendix A summarizes the notation used through the paper.

2. Integer programming formulation

The STTRPSD is defined on a graph $G=(V,A)$, where $V=\{0\}\cup V_D\cup V_C$ is the node set with the main depot at node 0, $V_D=\{1,2,\ldots,p\}$ is the set of trailer points (satellite depots), and $V_C=\{p+1,p+2,\ldots,p+n\}$ is the set of customers with known demands q_i ($i\in V_C$). A is the arc set, with costs c_{ij} ($i,j\in V$, $i\neq j$) satisfying the triangle inequality. The parameters Q_V and Q_T are the capacities of the truck and the trailer, respectively. To ensure feasibility with one vehicle, the sum of demands does not exceed Q_V+Q_T.

To formulate the STTRPSD we define the following subsets of V: (i) $V_1=V_D\cup\{0\}$, set of nodes that can be visited in the first-level trip; (ii) $V_2=V_C\cup V_D$, set of nodes that can be visited in second-level trips; and (iii) $V^j=V_C\cup\{j\}$ ($\forall j\in V_D$), set of nodes that can be visited in a second-level trip rooted at satellite depot j. The integer programming formulation of the STTRPSD uses binary variables y_{ij} equal to 1 if and only if the truck with the trailer traverses the arc (i,j) ($i,j\in V_1$); and binary variables x^j_{lm} equal to 1 if and only if arc (l,m) ($l,m\in V^j$) is traversed by the truck in a second-level trip departing from trailer point $j\in V_D$. The integer programming formulation of the STTRPSD follows:

$$\min\sum_{i\in V_1}\sum_{j\in V_1}c_{ij}y_{ij}+\sum_{j\in V_D}\sum_{l\in V_2}\sum_{m\in V_2}c_{lm}x^j_{lm} \tag{1}$$

subject to

$$\sum_{i\in V_1}y_{ij}\le 1,\quad \forall j\in V_D \tag{2}$$

$$\sum_{i\in V_1}y_{ji}\le 1,\quad \forall j\in V_D \tag{3}$$

$$\sum_{i\in V_1}y_{ij}=\sum_{k\in V_1}y_{jk},\quad \forall j\in V_D \tag{4}$$

$$\sum_{j\in V_D}y_{j0}=1, \tag{5}$$

$$\sum_{j\in V_D}y_{0j}=1, \tag{6}$$

$$\sum_{i\in V'}\sum_{j\in V'}y_{ij}\le |V'|-1,\quad \forall V'\subseteq V_D;|V'|\ge 2 \tag{7}$$

$$x^j_{lm}\le\sum_{i\in V_1}y_{ij},\quad \forall l,m\in V_2,\ \forall j\in V_D \tag{8}$$

$$\sum_{l\in V_2}\sum_{j\in V_D}x^j_{lm}=1,\quad \forall m\in V_C \tag{9}$$

$$\sum_{l\in V_2}x^j_{lm}=\sum_{o\in V_2}x^j_{mo},\quad \forall m\in V_C,\ \forall j\in V_D \tag{10}$$

$$\sum_{l \in V_C} x^j_{jl} = \sum_{o \in V_C} x^j_{oj}, \quad \forall j \in V_D \tag{11}$$

$$\sum_{l \in V'} \sum_{m \in V'} x^j_{lm} \leq |V'| - \gamma(V'), \quad \forall j \in V_D, \ \forall V' \subseteq V_C, \ |V'| \geq 2 \tag{12}$$

$$y_{ij} \in \{0,1\}, \quad \forall i,j \in V_1, \ i \neq j \tag{13}$$

$$x^j_{lm} \in \{0,1\}, \quad \forall j \in V_D, \ l,m \in V^j, \ l \neq m \tag{14}$$

The objective function (1) comprises two terms, the first one represents the length of the first-level trip and the second one the total distance traveled by the truck in the second-level trips. Constraints (2) and (3) state that a trailer point is visited at most once in the first-level trip. Constraints (4) are connectivity constraints for the first-level trip. Constraints (5) and (6) state that the first-level trip departs and ends at the main depot. Constraints (7) are subtour elimination constraints for the first-level trip. Constraints (8) guarantee that second-level trips depart only from trailer points visited in the first-level trip. Constraints (9) state that each customer must be visited exactly once. Constraints (10) guarantee the connectivity of second-level trips. Constraints (11) state that all the second-level trips departing from a trailer point return to it. Constraints (12) prevent subtours in second-level trips, where $\gamma(R)$ is the minimum number of second-level trips needed to serve the demand of the customers in $V' \subseteq V_C$. Binary variables y_{ij} and x^j_{lm} are defined in (13) and (14), respectively.

3. Metaheuristics for the STTRPSD

Since the STTRPSD generalizes the VRP and the MDVRP, it is clear that only small instances of the STTRPSD could be solved efficiently using the formulation defined by (1)–(14). Therefore, we decided to develop metaheuristics based on greedy randomized adaptive search procedures (GRASP) and evolutionary local search (ELS) to solve it.

3.1. GRASP/VND

GRASP (Feo and Resende, 1995) is a memory-less multi-start method in which local search is applied to *ns* initial solutions constructed with a greedy randomized heuristic. Recently, Festa and Resende (2009) surveyed the application of GRASP to solve combinatorial optimization problems in various domains. GRASP can be hybridized in different ways, for instance by replacing the local search with another metaheuristic such as tabu search, simulated annealing, variable neighborhood search, iterated local search, among others (Resende, 2008). Our first metaheuristic for the STTRPSD is a hybrid GRASP/VND in which the local search of GRASP is replaced by variable neighborhood descent (VND) (Hansen and Mladenovic, 2001).

3.1.1. Greedy randomized construction

Route-first, cluster-second based metaheuristics have shown to be effective solving capacitated node routing problems ranging from the classical VRP (Prins, 2004) to some of its extensions, including the VRP with time windows (Labadi et al., 2008), heterogeneous fleet (Prins, 2009b) and pick-up and delivery (Velasco et al., 2009), among others. Following this approach, the greedy randomized construction of the hybrid GRASP/VND is done by means of a tour splitting procedure (hereafter labeled *Split*) that obtains a solution S of the STTRPSD from a giant tour $T=(t_0,t_1,t_2,\ldots,t_j,\ldots,t_n)$ visiting all the customers, where t_j represents the customer in the j-th position and t_0 the main depot.

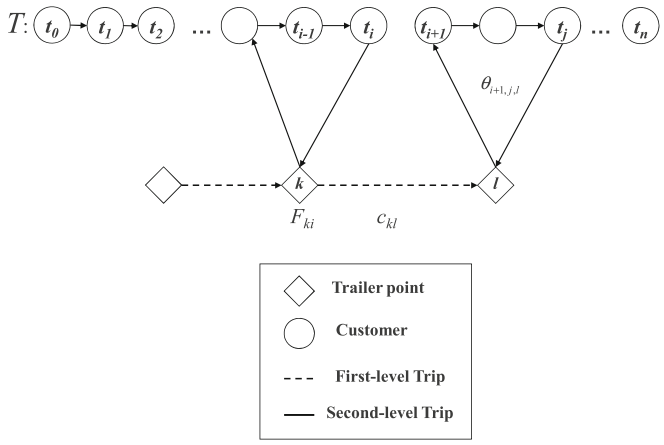

Fig. 2. Principle of the recurrence for the dynamic programming method (*Split*).

Giant tours are constructed with a randomized nearest neighbor method with a restricted candidate list (RCL) of size r that ignores capacity constraints and trailer-point selection. Algorithm 1 presents the pseudocode of the randomized nearest neighbor method, note that it is possible to obtain a deterministic nearest neighbor heuristic when $r=1$, and random permutations of the customers when $r=n$.

Algorithm 1. Randomized nearest neighbor.

Parameters: V_c, r
Output: T
Create an empty giant tour T
$t_0 := 0$
$count := 0$
$i := 0$
repeat
  $RCL := \emptyset$
  $sl := \min(r, n-count)$
  for k=1 to sl **do**
    $l := \underset{j \in V_c - RCL}{\operatorname{argmin}} \{c_{ij}\}$
    $RCL := RCL \cup \{l\}$
  end for
  Select at random $l^* \in RCL$
  $count := count + 1$
  $t_{count} := l^*$
  $V_c := V_c - \{l^*\}$
  $i := l^*$
until $count=n$
return T

It is worth mentioning that the splitting procedure for the STTRPSD is much more complicated than that of the VRP (Prins, 2004). Since the distance between trailer points is included in the objective function, the selection and routing of trailer points are important decisions. For the STTRPSD, *Split* derives a solution of the STTRPSD from T by optimally splitting it into second-level trips. The selection of open trailer points and the construction of the corresponding first-level trip are also included in the tour splitting procedure.

Split uses a dynamic programming method, in which state $[l,j]$ represents an optimal splitting of $(t_0, t_1,\ldots, t_j)$ with trailer point l for the last trip, and [0,0] denotes the initial state. Let F_{lj} denote the cost of state $[l,j]$ and θ_{ijk} the cost of a trip visiting customers $(t_i, t_{i+1},\ldots, t_j)$ from trailer point k. We have the following

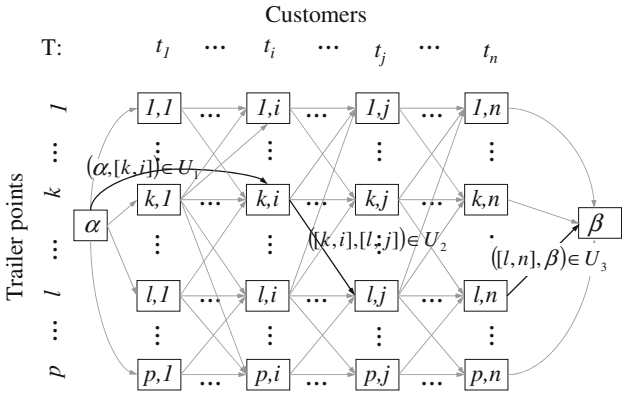

Fig. 3. State graph $H(X,U,W)$ for the STTRPSD.

recurrence relations for any customer t_j and any trailer point l (see Fig. 2 for a graphical example):

$$F_{lj}=\begin{cases}0, & \text{if} \quad l=0 \text{ and } j=0\\ \min\{F_{ki}+c_{kl}+\theta_{i+1,j,l}\}, & i<j:\sum_{u=i+1}^{j} q(t_u)\le Q_V, \quad k=0 \text{ if } i=0 \text{ else } k\in V_D\end{cases} \tag{15}$$

Since states $[l, n]$ ($l\in V_D$) do not include the return cost to the main depot, the cost of an optimal splitting is $z=\min_{l\in V_D}\{F_{ln}+c_{l0}\}$.

The dynamic programming method can be viewed as a shortest path problem in a state graph $H=(X, U, W)$ depicted in Fig. 3. The node set X contains $np+2$ nodes: two copies of the main depot (α and β), acting as source and sink nodes for H, and np nodes for the states $[l,j]$. The arc set U has three types of arcs: U_1 the outgoing arcs of α, U_2 the arcs between internal nodes of the state graph, and U_3 the incoming arcs of β. W is a mapping defining the cost of each arc. Formally, each subset of U and its cost is defined in Eqs. (16)–(21)

$$U_1=\left\{(\alpha,[k,i]): k\in V_D; 1\le i<n, \sum_{u=1}^{i} q(t_u)\le Q_V\right\} \tag{16}$$

$$w(\alpha,[k,i])=c(\alpha,k)+c(k,t_1)+\sum_{u=1}^{i-1} c(t_u,t_{u+1})+c(t_i,k) \tag{17}$$

$$U_2=\left\{([k,i],[l,j]): k,l\in V_D; 1\le i<j<n, \sum_{u=i+1}^{j} q(t_u)\le Q_V\right\} \tag{18}$$

$$w([[k,i],[l,j]])=c(k,l)+c(l,t_{i+1})+\sum_{u=i+1}^{j-1} c(t_u,t_{u+1})+c(t_j,l) \tag{19}$$

$$U_3=\{([l,n],\beta): l\in V_D\} \tag{20}$$

$$w([[l,n],\beta))=c(l,\beta) \tag{21}$$

Like for the VRP, the shortest path can be computed using Bellman's algorithm for directed acyclic graphs (Cormen et al., 2001), whose complexity is proportional to the number of arcs. H has n vertical layers of p trailer points each. In the worst case, each node in layer $i=1,2,\ldots,n-1$ is linked to all the $p(n-i)$ nodes in subsequent layers. Adding the np outgoing arcs from the source node in the worst case and the p incoming arcs of the sink node, we have

$$\begin{aligned}|U|&=np+p+\sum_{i=1}^{n-1} p^2(n-i)=p(n+1)+\sum_{i=1}^{n-1} p^2 i\\ &=p(n+1)+p^2\frac{(n)(n-1)}{2}=O(n^2p^2)\end{aligned} \tag{22}$$

Even though the number of arcs $|U|$ can be huge, a more precise evaluation shows significant savings. Note that each capacity-feasible subsequence $(t_{i+1}, t_{i+2},\ldots,t_j)$ of T gives p^2 arcs between vertical layers i and j, because p depots can be chosen for the trip serving customer t_i and p depots for the trip serving customers t_{i+1} to t_j. If the average number of customers per trip is b, the average number of feasible trips is $O(nb)$ and then the number of arcs in U is $O(nbp^2)$. When the average number of clients per trip is small compared to n or, equivalently, if the average customer demand is relatively large compared to vehicle capacity, then $O(nbp^2)$ is significantly smaller than $O(n^2p^2)$. Thus, *Split* has a worst case complexity of $O(nbp^2)$.

Moreover, it is possible to implement the dynamic programming method without generating explicitly the auxiliary graph, using for $F=(F_{ij})$ (see Eq. (15)) a state table with p rows and $n+1$ columns indexed from 0 to n. The implementation of *Split*, detailed in Algorithm 2, uses two procedures *Develop* and *TourToSol* (see the Appendix B for the details of these procedures). *Develop*(k,i) scans all arcs leaving state $[k,i]$, to update the labels of its successors. Simultaneously the record of the predecessor of each state is stored in two matrices of the same size of F, *PD* (previous depot) and *PC* (previous customer). For instance, if the predecessor of state $[l,j]$ is state $[k,i]$ then $PD[l,j]=k$ and $PC[l,j]=i$. After having solved the shortest path problem, the procedure *TourToSol*(*PD*,*PC*,*LD*) creates a solution of the STTRPSD by backtracking from state $[LD,n]$ using the information of the last satellite depot visited (*LD*) and the matrices of predecessors.

In general, each trailer point may have several trips. Consider a sequence of customers in T close to one trailer point k, with a total demand exceeding Q_V. After splitting, S will contain two consecutive trips departing from k. Such a solution in which each trailer point has consecutive trips is called *strong splitting*. Consider now two capacity-feasible sequences of customers close to k but separated in the giant tour T by other customers. Since the dynamic programming method follows the order of T, after split S will contain two non-consecutive trips departing from trailer point k, so k will be visited twice in the first-level trip; such a solution is called *weak splitting*. Since the triangle inequality holds the length of the first-level trip can be reduced by making adjacent in T non-consecutive trips with common trailer points (i.e., transforming a weak splitting into a strong one). Then, after the creation of a solution with *TourToSol*, the procedure *EliminateWeakSplitting* scans the first-level trip to eliminate duplicated visits to the same trailer point. The copy of a trailer point whose elimination produces the greatest reduction in the length of the first-level trip is deleted, and its departing trips reassigned to other visit of the same trailer point. The deletion of copies is repeated until the elimination of the weak splitting.

Algorithm 2. Split.

Input: T
Output: S, solution of the STTRPSD
1: $F[0,0]\coloneqq 0$
2: **for** $k\coloneqq 1$ to p **do**
3: **for** $i\coloneqq 0$ to n **do**
4: $F[k,i]\coloneqq\infty$
5: **end for**
6: **end for**
7: *Develop*$(0,0)$
8: **for** $i\coloneqq 1$ to $n-1$ **do**
9: **for** $k\coloneqq 1$ to p **do**
10: *Develop(k,i)*
11: **end for**
12: **end for**
13: $z\coloneqq\infty$

14: **for** $k \coloneqq 1$ to p **do**
15: **if** $F[k,n]+c_{k0} < z$ **then**
16: $z \coloneqq F[k,n]+c_{k0}$
17: $LD \coloneqq k$
18: **end if**
19: **end for**
20: $S \coloneqq TourToSol(PD, PC, LD)$
21: $S \coloneqq EliminateWeakSplitting(S)$
22: **return** S

Fig. 4(a) illustrates *Split* on a simple instance of the STTRPSD with Q_V=3, p=2 trailer points (1 and 2), and n=4 customers (3, 4, 5 and 6) with demands 2, 1, 1 and 1, respectively. The length of each square side in the grid is equal to 1 and the distance between nodes is Euclidean. Fig. 4(b) shows the results of splitting the giant tour T=(0,4,6,3,5). Fig. 4(c) shows the corresponding state graph, where the arcs of the optimal split are in bold. Table 1 illustrates the calculation of the cost of some arcs of Fig. 4(c).

3.1.2. Variable neighborhood descent

Solutions obtained from the GRASP construction phase are improved by a VND. VND is a deterministic version of variable neighborhood search (Hansen and Mladenovic, 2001) in which k_{max} neighborhoods are explored sequentially ($k_{max} \geq 2$). Given the incumbent solution S_0, a subset of the solution space $\mathcal{N}_k(S_0)$ is composed of the solutions reachable from S_0 when neighborhood k is applied to it. Algorithm 3, outlines VND with a best-improvement selection strategy in which the best solution of $\mathcal{N}_k(S_0)$ replaces S_0 if it has a smaller cost (line 4). When the incumbent solution is improved the search is reinitialized from

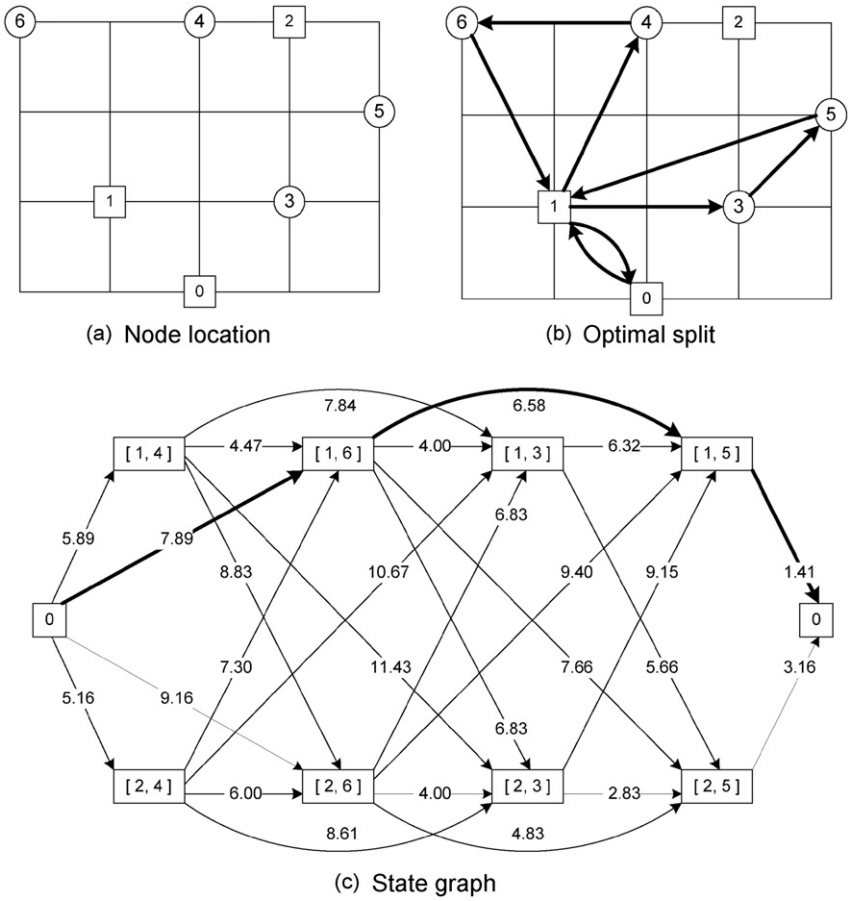

Fig. 4. *Split* example for the 4-customer sequence T=(0,4,6,3,5) on a 2-satellite-depot instance of the STTRPSD. (a) Node location; (b) optimal split; (c) state graph.

Table 1
Example of the cost calculation for the 4-customer 2-satellite-depot instance of the STTRPSD.

Subset	Arc	Trip	Inter-depot distance	Trip distance	Arc cost
U_1	(0,[1,4])	(1,4,1)	c_{01}=1.41	$\theta_{111}=c_{14}+c_{41}=4.47$	5.89
U_1	(0,[2,6])	(2,4,6,2)	c_{02}=3.16	$\theta_{122}=c_{24}+c_{46}+c_{62}=6$	9.16
U_2	([1,4],[1,6])	(1,6,1)	c_{11}=0	$\theta_{221}=c_{16}+c_{61}=4.47$	4.47
U_2	([1,6],[1,5])	(1,3,5,1)	c_{11}=0	$\theta_{341}=c_{13}+c_{35}+c_{51}=6.58$	6.58
U_2	([1,6],[2,5])	(2,3,5,2)	c_{12}=2.83	$\theta_{342}=c_{23}+c_{35}+c_{52}=4.83$	7.66
U_3	([1,5],0)	–	c_{10}=1.41	–	1.41
U_3	([2,5],0)	–	c_{20}=3.16	–	3.16

the first neighborhood. On the contrary, if the best solution in $\mathcal{N}_k(S_0)$ is not better than S_0 the neighborhood index is increased. VND stops when all the neighborhoods have been explored without improving the incumbent solution (line 11). If a first-improvement selection strategy is used, the search of line 4 is stopped as soon as an improving solution is found (i.e., $\exists S \in \mathcal{N}_k(S_0) : f(S) < f(S_0)$).

Algorithm 3. VND with best-improvement selection.

Parameters S_0, neighborhoods structures $\mathcal{N}_1, \ldots, \mathcal{N}_{k_{max}}$
Output: Improved solution S^*
1: $flag \coloneqq true$
2: $k \coloneqq 1$
3: **while** flag **do**
4: $S' \coloneqq \underset{S \in \mathcal{N}_k(S_0)}{\operatorname{argmin}}\{f(S)\}$
5: **if** $f(S') < f(S_0)$ **then**
6: $S_0 \coloneqq S'$
7: $k \coloneqq 1$
8: **else**
9: $k \coloneqq k+1$
10: **end if**
11: **if** $k = k_{max}+1$ **then**
12: $flag \coloneqq false$
13: $S^* \coloneqq S_0$
14: **end if**
15: **end while**
16: **return** S^*

Our VND uses two types of neighborhoods: (i) neighborhoods $\mathcal{N}_1$, $\mathcal{N}_2$ and $\mathcal{N}_3$ are intended to improve routing within the solution; and (ii) neighborhoods $\mathcal{N}_4$ and $\mathcal{N}_5$ modify the set of open trailer points. Moreover, when applied to second-level trips neighborhoods 1–3 only explore feasible solutions satisfying the capacity constraint. A brief description of the neighborhoods follows:

- $\mathcal{N}_1$: Each customer and each trailer point is removed from its current position and reinserted elsewhere. When applied to a customer, the new position could be in the same second-level trip or in another second-level trip; whereas trailer-points changes are made within the first-level trip.
- $\mathcal{N}_2$: Two customers or trailer points are exchanged. The two customers may belong to different second-level trips.
- $\mathcal{N}_3$: A modified 2-opt in which two edges are exchanged. If the edges belong to the same trip $\mathcal{N}_3$ reduces to a simple 2-opt for the traveling salesman problem (Croes, 1958); whereas if the edges belong to a pair of second-level trips with different trailer points, the new trips are assigned to the trailer point from the original trips with the smallest connecting cost. $\mathcal{N}_3$ is inspired by a neighborhood for the capacitated location-routing problem (Prins et al., 2006).
- $\mathcal{N}_4$: Each open trailer point is considered for closure. If the trailer point is closed, its trips are relocated to the remaining trailer points. When relocating a second-level trip, a cheapest insertion of the new trailer point is performed.
- $\mathcal{N}_5$: In contrast to $\mathcal{N}_4$, each closed trailer point is considered for opening, only if there exist second-level trips whose cost decreases when relocated to the new trailer point. Again, when relocating a second-level trip, the new trailer point is inserted in the best position.

3.1.3. Hybrid GRASP/VND for the STTRPSD

Algorithm 4 outlines the hybrid GRASP/VND for the STTRPSD. In the main cycle, repeated *ns* times, a giant tour T is constructed with a randomized nearest neighbor heuristic with restricted candidate list of size r. Then, a solution S is produced by applying *Split* to T, and finally S is improved with VND.

Algorithm 4. GRASP/VND for the STTRPSD.

Parameters: *ns*, *r*
Output: S^*
1: $f^* \coloneqq \infty$
2: **for** $i \coloneqq 1$ to *ns* **do**
3: $T \coloneqq RandomizedNearestNeighbor(V_C, r)$
4: $S \coloneqq Split(T)$
5: $S \coloneqq VND(S)$
6: **if** $f(S) < f^*$ **then**
7: $f^* = f(S)$
8: $S^* \coloneqq S$
9: **end if**
10: **end for**
11: **return** S^*

3.2. Multi-start evolutionary local search

Evolutionary local search (ELS) (Wolf and Merz, 2007) can be seen as an evolutionary extension of iterated local search (ILS) (Lourenço et al., 2003), in which a single solution is mutated to obtain *nc* children that are further improved using local search. Following a (1+*nc*) selection paradigm, the solution of the next generation is the best among the parent and its *nc* children (Bäck et al., 1991). The main loop of mutation and local search is repeated during *ni* generations.

Although ELS was originally introduced for the solution of optimization problems in telecommunications (Wolf and Merz, 2007), metaheuristics based on ELS have achieved very good results in different routing problems such as the VRP (Prins, 2009a) and the split delivery CARP (Belenguer et al., 2009).

We implemented a multi-start variant of ELS, introduced by Prins and Reghioui (2009), in which ELS is restarted from *nr* initial solutions obtained by strongly perturbing the current best solution S^*. By reusing the *Split* and VND procedures described in Sections 3.1.1 and 3.1.2, it is possible to derive the multi-start evolutionary local search for the STTRPSD outlined in Algorithm 5, where the new elements are the *Mutate* and *Concat* operators.

Note that, within the ELS inner loop (lines 12–30 of Algorithm 5), the method alternates between STTRPSD solutions and giant tours. While mutation is applied to giant tours, VND is applied to solutions. When a parent is replaced, a giant tour is derived from it with the procedure (*Concat*) illustrated in Fig. 5.

Concat generates a giant tour by creating chains of customers from the second-level trips departing from each open trailer point and concatenating them using the order of the trailer points visited in the first-level trip. As illustrated in Fig. 6, the mutation operator (*Mutate*), exchanges *b* pairs of customers in T.

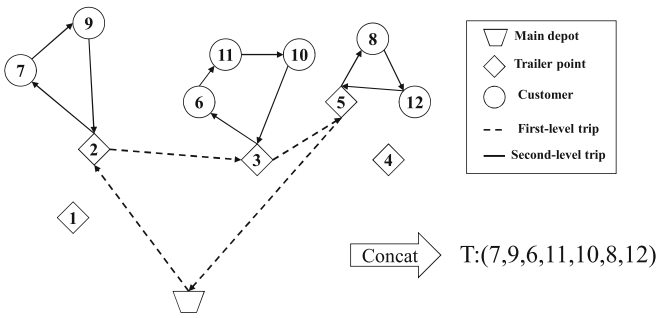

Fig. 5. Principle of procedure *Concat*.

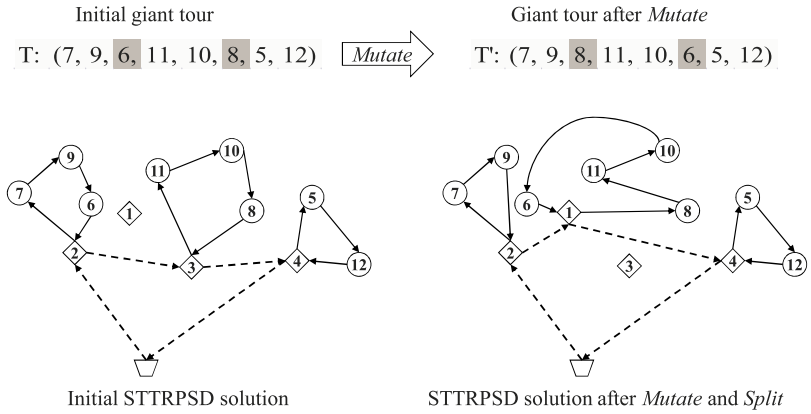

Fig. 6. Principle of procedure *Mutate*.

The parameter b is dynamically controlled and varies from 1 to b_{max}. When the parent of ELS is updated the value of b is reset to 1, on the contrary if the parent is not changed the value of b is increased (with a maximum limit b_{max}). We decided to perform the mutation operator in the giant tour because the resulting solution is always feasible and a simple exchange of a pair of customers in T is sometimes enough to perturb the solution structure.

In the proposed multi-start evolutionary local search, the first initial solution is found by splitting the giant tour produced with a deterministic nearest neighbor algorithm (r=1, in Algorithm 1), while the next initial solutions ($i>1$) are obtained by perturbing and splitting T^* (deduced from S^* by *Concat*). The perturbation is in this case the same mutation operator, but with the exchange of b_{pert} pairs of customers ($b_{pert} \gg b_{max}$).

Algorithm 5. Multi-start evolutionary local search for the STTRPSD.

Parameters: nr, ni, nc, b_{max}, b_{pert}
Output: S^*
1: $f^* := \infty$
2: $b := 1$
3: **for** $i := 1$ to nr **do**
4: **if** i=1 **then**
5: $T := NearestNeighbor(V_C)$
6: **else**
7: $T := Mutate(T^*, b_{pert})$
8: **end if**
9: $S := Split(T)$
10: $S := VND(S)$
11: $T := Concat(S)$
12: **for** $j := 1$ to ni **do**
13: $\hat{f} := f(S)$
14: **for** $k := 1$ to nc **do**
15: $T' := Mutate(T, b)$
16: $S' := Split(T')$
17: $S' := VND(S')$
18: **if** $f(S') < \hat{f}$ **then**
19: $\hat{f} := f(S')$
20: $\hat{S} := S'$
21: **end if**
22: **end for**
23: **if** $\hat{f} < f(S)$ **then**
24: $S := \hat{S}$
25: $T := Concat(S)$
26: $b := 1$
27: **else**
28: $b := min(b+1, b_{max})$
29: **end if**
30: **end for**
31: **if** $f(S) < f^*$ **then**
32: $f^* = f(S)$
33: $S^* := S$
34: $T^* := Concat(S^*)$
35: **end if**
36: **end for**
37: **return** S^*

4. Computational study

4.1. Implementation and test problems

The proposed methods were tested on a set of 32 randomly generated Euclidean instances with the following characteristics: $n \in \{25, 50, 100, 200\}$, $p \in \{5, 10, 20\}$, and $Q_V \in \{1000, 2000\}$. The coordinates for trailer points and customers are randomly generated in a square grid of size 100×100. There are two problem types based on the distribution of customers and trailer points: clustered (c) and randomly distributed (rd). The customer demands are drawn from a discrete uniform distribution in the interval [1,200]. The data set is available at http://hdl.handle.net/1992/1125 and Table 2 describes each instance in the testbed. All the metaheuristics were implemented in Java and compiled using Eclipse JDT version 3.3.2. The experiments described in this section were performed on a computer with an Intel Pentium D 945 processor running at 3.4 GHz with 1024 MB of RAM on a Windows XP Professional environment.

4.2. Preliminary experiments

We conducted two experiments to analyze the building blocks of the metaheuristics. The first experiment explores the greedy randomized construction procedures. Initial giant tours for the GRASP/VND are obtained using the randomized nearest neighbor (RNN) method using a RCL of size r=2 or 3. Alternatively, giant tours can be generated through random permutations of customers when r=n (RAND). While generating a solution we can call (or not) the transformation of weak splittings into strong ones (line 21 of Algorithm 2). Combining these options we have a total of six randomized heuristics with the characteristics summarized in Table 3.

Table 2
Characteristics of the 32 randomly generated instances.

Instance	n	p	Q_V	Problem type
1	25	5	1000	*c*
2	25	5	2000	*c*
3	25	5	1000	*rd*
4	25	5	2000	*rd*
5	25	10	1000	*c*
6	25	10	2000	*c*
7	25	10	1000	*rd*
8	25	10	2000	*rd*
9	50	5	1000	*c*
10	50	5	2000	*c*
11	50	5	1000	*rd*
12	50	5	2000	*rd*
13	50	10	1000	*c*
14	50	10	2000	*c*
15	50	10	1000	*rd*
16	50	10	2000	*rd*
17	100	10	1000	*c*
18	100	10	2000	*c*
19	100	10	1000	*rd*
20	100	10	2000	*rd*
21	100	20	1000	*c*
22	100	20	2000	*c*
23	100	20	1000	*rd*
24	100	20	2000	*rd*
25	200	10	1000	*c*
26	200	10	2000	*c*
27	200	10	1000	*rd*
28	200	10	2000	*rd*
29	200	20	1000	*c*
30	200	20	2000	*c*
31	200	20	1000	*rd*
32	200	20	2000	*rd*

Table 3
Preliminary experiments with the greedy randomized construction methods.

Name	Giant tour	r	Split	Avg. dev. (%)	Avg. mod. CV (%)
RH1	RNN	2	Weak	−49.53	2.88
RH2	RNN	2	Strong	−50.70	2.81
RH3	RNN	3	Weak	−41.87	3.53
RH4	RNN	3	Strong	−43.26	3.50
RH5	RAND	*n*	Weak	94.05	8.67
RH6	RAND	*n*	Strong	91.31	8.81

For each instance in the test set we ran 100 times each randomized heuristic (RH) and computed the average cost and its standard deviation. In Table 3 the column *avg. dev.* presents for each method the average deviation with respect to the average cost calculated over all methods. The column *avg. mod. CV.* reports the average value of a modified coefficient of variation, in which the standard deviation of the solutions for each method is divided by the average cost of the solutions over all methods. While the former column measures the quality of the solutions, the latter measures their diversity.

Because of the good balance between diversity and solution quality, RH2 (r=2 with strong splitting) was selected as the greedy randomized construction block of GRASP/VND. With r=2 it is possible to obtain better solutions at the expense of just a small loss in diversity, compared to RH3 and RH4 (r=3). On the other hand, RH5 and RH6 (i.e., random permutations) provide a higher diversity, but at the expense of affecting the quality of the solutions. Finally, this experiment shows that the elimination of weak splittings slightly improves the cost of the solutions without affecting much the diversity.

Table 4
Preliminary experiments with the VND procedure.

Name	k_{max}	Option	Avg. improvement (%)	Avg. time (ms)
VND1	3	First-improvement	35.0	182.89
VND2	3	Best-improvement	39.6	172.68
VND3	5	First-improvement	39.0	233.76
VND4	5	Best-improvement	42.0	207.37

A second experiment, analyzes the contribution of neighborhoods 4 and 5 to the performance of VND. For each instance, we applied two versions of VND to 100 solutions generated with RH2. The first version uses only the routing neighborhoods (k_{max}=3), while the second version includes also neighborhoods 4 and 5 (k_{max}=5). Both options were tested with first-improvement and best-improvement selection criteria. Table 4 reports the average improvement over the initial solution (in %) and the average computation time (in ms) for each method. The use of k_{max}=5 with best-improvement (VND4) produces the best solutions overall. A comparison between the selection criteria shows a marginal (quality) gain between 3% and 4.6% by using the best over the first-improvement strategy; in addition, the best-improvement strategy also proves to be faster than the first-improvement strategy. This experiment also highlights the importance of the selection of trailer points for the quality of the solutions. A comparison of VND2 against VND4 shows that the marginal effect of exploring neighborhoods 4 and 5 is 2.4%. Based on these results we selected VND4 as the local search component of the metaheuristics, even if it takes 20% longer than VND2.

4.3. Parameter fine-tuning

To have a fair comparison we allocated a fixed "budget" of 2500 calls of the VND to each metaheuristic to see which one makes the best use of the VND.

For GRASP/VND the selection of the parameters is very simple: ns=2500, and r=2 according to the results of Section 4.2. On the contrary, for the multi-start evolutionary local search it is necessary to select the values for b_{max} and b_{pert} and to distribute the 2500 calls of the VND such that $nr \times ni \times nc = 2500$. By setting different values of nc, we implemented two versions of the multi-start evolutionary local search method: (1) a multi-start ELS (henceforth labeled MS-ELS) whit $nc > 1$; and (2) a multi-start ILS (henceforth labeled MS-ILS) with nc=1.

To select the parameters we used a simple (yet effective) fine-tuning procedure: beginning from a promising configuration, one or two parameters were modified, then the current and new configuration were compared using the sign test Conover (1998) on a subset of 12 problems. If the new configuration was better we updated the parameters, but if there was no evidence that one configuration was better, the one with the smaller running time was declared the winner and its parameters kept. We stopped the procedure after testing a maximum of 10 configurations. After the fine-tuning procedure the parameters were set to: nr=50, ni=10, nc=5, b_{max}=4 and b_{pert}=10 for the MS-ELS option; and nr=125, ni=20, nc=1, b_{max}=4 and b_{pert}=10 for the MS-ILS option.

4.4. Results

We used three heuristics as benchmarks to compare the performance of the metaheuristics. The first one is a simple cluster-first, route-second (henceforth labeled CFRS) approach in which each customer is assigned to its nearest trailer point; then

the customers assigned to each open trailer point are routed using an insertion heuristic (i.e., a VRP is solved for each open trailer point); and finally, the first-level trip that visits the set of open trailer points is derived from the same insertion heuristic. The second heuristic enhances the solutions constructed with cluster-first, route-second by using VND4 (CFRS+VND). The third and last heuristic, called iterated route-first, cluster-second (henceforth labeled IRFCS) repeats RH2 2500 times.

Tables 5 and 6 presents the results of the three benchmark heuristics and the proposed metaheuristics on the set of 32 randomly generated instances, respectively. Additionally, Table 6 includes the results of a hybrid GRASP × ELS proposed in Villegas et al. (2009). In both tables we present the best, worst and average results over 10 runs for all the randomized methods; the last two rows report the number of times each method found the best-known solution (NBKS) and the average deviation (in %) above the best-known solution (BKS). The last column reports the cost of the best-known solution for each problem, and every BKS is highlighted in bold type when found by a given method. Table 7 shows the average running times (in minutes) for each method over all instances. We do not report exact running times for CFRS and CFRS+VND since they take on average less than 1 s.

Even though the CFRS heuristic is extremely fast, its average deviation above the best solutions is large (26.17%) compared to the average deviation of CFRS+VND (4.70%) which also runs in less than 1 s. IRFCS produces, on average, solutions of better quality (21.98%) than CFRS and still with short running times. However, CFRS frequently outperforms IRFCS in the clustered problems. Also, the results of IRFCS have a large variability, with a gap of 4.53% between its best and worst results.

The metaheuristics proposed in this article outperform the previous hybrid GRASP × ELS reported in Villegas et al. (2009). Among the proposed metaheuristics, multi-start evolutionary local search is both faster and more accurate than GRASP/VND. The MS-ILS version found 29 out of 32 best known solutions and was 2.8 times faster than GRASP/VND; while the MS-ELS version was 3.5 times faster, and found 25 out of 32 best known solutions. Also important is the fact that the multi-start evolutionary local search reports very small average gaps with respect to best known solutions. While MS-ILS and MS-ELS achieved average gaps as small as 0.25% and 0.31%, respectively; the GRASP/VND and GRASP × ELS report larger average gaps of 0.52% and 0.91%, respectively. Despite of the slight differences, remarkably these experiments highlight the robustness of the proposed metaheuristics: none of them has deviations to BKS greater than 1.0% even under their worst performance.

The comparison of GRASP/VND against IRFCS shows the effect of the VND on the quality of the solutions at the expense of computational effort. For instance, VND represents more than 98% of the running time of GRASP/VND. On the contrary, MS-ELS and MS-ILS exploit better the local search. First and foremost, they scale much better than GRASP/VND as the number of customers increases (Fig. 7). Second, as shown in Fig. 8, they have better time-to-target distributions (Aiex et al., 2007) than GRASP/VND. Fig. 8 was constructed by applying each metaheuristic 200 times to a fixed instance and recording the time needed to find a

Table 5
Results of the heuristics on the 32-instance testbed.

Instance	n	p	Type	CFRS	CFRS+VND	IRFCS			BKS
						Best	Worst	Average	
1	25	5	*c*	444.08	**405.46**	420.34	435.62	427.48	405.46
2	25	5	*c*	444.08	391.62	390.60	407.58	398.56	374.79
3	25	5	*rd*	696.73	585.96	596.17	644.50	618.61	584.03
4	25	5	*rd*	640.01	526.27	530.48	568.88	548.32	508.48
5	25	10	*c*	460.28	**386.45**	398.41	409.12	404.86	386.45
6	25	10	*c*	460.28	386.45	391.43	406.42	401.46	380.86
7	25	10	*rd*	789.70	582.64	597.13	628.87	613.02	573.96
8	25	10	*rd*	789.70	582.64	521.67	559.78	542.26	506.37
9	50	5	*c*	625.67	583.41	641.15	655.86	646.42	583.07
10	50	5	*c*	574.17	560.17	594.72	616.69	608.57	516.98
11	50	5	*rd*	1177.25	**870.51**	994.62	1031.36	1012.40	870.51
12	50	5	*rd*	980.57	787.79	895.60	935.34	907.91	766.03
13	50	10	*c*	471.43	**387.83**	424.53	438.86	433.50	387.83
14	50	10	*c*	460.36	381.32	415.45	426.20	421.57	367.01
15	50	10	*rd*	1034.77	847.49	892.42	953.81	931.37	811.28
16	50	10	*rd*	1013.20	758.95	855.75	893.63	874.40	731.53
17	100	10	*c*	705.19	640.01	724.13	755.20	742.97	614.02
18	100	10	*c*	665.76	555.31	679.70	702.49	691.22	547.44
19	100	10	*rd*	1544.01	1416.60	1569.15	1612.74	1593.06	1275.76
20	100	10	*rd*	1290.79	1167.97	1378.20	1440.09	1408.73	1097.28
21	100	20	*c*	820.00	668.04	768.01	789.28	781.21	642.61
22	100	20	*c*	808.61	643.16	692.03	727.02	714.43	581.56
23	100	20	*rd*	1392.01	1192.83	1374.35	1435.41	1410.23	1143.10
24	100	20	*rd*	1342.10	1138.84	1321.92	1373.19	1348.44	1060.75
25	200	10	*c*	1004.80	849.63	1032.04	1068.24	1049.39	822.52
26	200	10	*c*	878.59	734.63	936.67	961.79	949.52	714.33
27	200	10	*rd*	2391.08	2026.04	2305.69	2344.00	2322.63	1761.10
28	200	10	*rd*	1951.77	1515.01	2028.96	2081.73	2050.88	1445.94
29	200	20	*c*	1098.70	950.21	1134.92	1169.78	1152.02	909.46
30	200	20	*c*	1036.27	862.35	1040.35	1069.29	1056.52	815.51
31	200	20	*rd*	2251.76	1691.43	2142.39	2191.40	2167.92	1614.18
32	200	20	*rd*	2019.44	1559.43	1956.08	2004.72	1975.18	1413.32
NBKS				0	4	0	–	–	
Avg. deviation above BKS (%)				26.17	4.70	17.45	21.98	19.81	

Table 6
Results of the metaheuristics on the 32-instance testbed.

Instance	n	p	Type	GRASP × ELS			GRASP/VND			MS-ELS			MS-ILS			BKS
				Best	Worst	Average	Best	Worst	Average	Best	Worst	Average	Best	Worst	Average	
1	25	5	*c*	**405.46**	405.46	405.46	**405.46**	405.46	405.46	**405.46**	405.46	405.46	**405.46**	405.46	405.46	405.46
2	25	5	*c*	**374.79**	374.79	374.79	**374.79**	374.79	374.79	**374.79**	374.79	374.79	**374.79**	374.79	374.79	374.79
3	25	5	*rd*	**584.03**	584.03	584.03	**584.03**	584.03	584.03	**584.03**	584.03	584.03	**584.03**	584.03	584.03	584.03
4	25	5	*rd*	**508.48**	508.48	508.48	**508.48**	508.48	508.48	**508.48**	508.48	508.48	**508.48**	508.48	508.48	508.48
5	25	10	*c*	**386.45**	386.45	386.45	**386.45**	386.45	386.45	**386.45**	386.45	386.45	**386.45**	386.45	386.45	386.45
6	25	10	*c*	**380.86**	380.86	380.86	**380.86**	380.86	380.86	**380.86**	380.86	380.86	**380.86**	380.86	380.86	380.86
7	25	10	*rd*	**573.96**	573.96	573.96	**573.96**	573.96	573.96	**573.96**	573.96	573.96	**573.96**	573.96	573.96	573.96
8	25	10	*rd*	**506.37**	506.37	506.37	**506.37**	506.37	506.37	**506.37**	506.37	506.37	**506.37**	506.37	506.37	506.37
9	50	5	*c*	**583.07**	585.50	583.38	**583.07**	583.07	583.07	**583.07**	583.41	583.10	**583.07**	583.07	583.07	583.07
10	50	5	*c*	**516.98**	516.98	516.98	**516.98**	516.98	516.98	**516.98**	516.98	516.98	**516.98**	516.98	516.98	516.98
11	50	5	*rd*	**870.51**	870.51	870.51	**870.51**	870.51	870.51	**870.51**	870.51	870.51	**870.51**	870.51	870.51	870.51
12	50	5	*rd*	**766.03**	766.03	766.03	**766.03**	766.03	766.03	**766.03**	766.03	766.03	**766.03**	766.03	766.03	766.03
13	50	10	*c*	389.07	389.76	389.46	**387.83**	387.83	387.83	**387.83**	387.83	387.83	**387.83**	387.83	387.83	387.83
14	50	10	*c*	**367.01**	367.01	367.01	**367.01**	367.01	367.01	**367.01**	367.01	367.01	**367.01**	367.01	367.01	367.01
15	50	10	*rd*	**811.28**	811.28	811.28	**811.28**	811.28	811.28	**811.28**	811.28	811.28	**811.28**	811.28	811.28	811.28
16	50	10	*rd*	**731.53**	735.66	731.94	**731.53**	731.53	731.53	**731.53**	731.53	731.53	**731.53**	731.53	731.53	731.53
17	100	10	*c*	615.00	618.29	616.62	**614.02**	614.61	614.22	614.20	614.31	614.30	**614.02**	615.32	614.41	614.02
18	100	10	*c*	548.36	550.08	549.38	**547.44**	547.44	547.44	**547.44**	548.11	547.64	**547.44**	548.11	547.57	547.44
19	100	10	*rd*	1284.48	1297.80	1292.89	1278.90	1290.52	1285.45	**1275.76**	1285.38	1280.65	1280.02	1286.14	1282.79	1275.76
20	100	10	*rd*	1099.22	1114.45	1106.90	**1097.28**	1107.37	1102.39	**1097.28**	1097.28	1097.28	**1097.28**	1103.43	1097.90	1097.28
21	100	20	*c*	**642.61**	646.00	643.71	**642.61**	643.93	643.12	**642.61**	643.93	642.79	**642.61**	642.61	642.61	642.61
22	100	20	*c*	**581.56**	585.07	582.69	**581.56**	581.56	581.56	**581.56**	583.71	581.78	**581.56**	583.71	582.18	581.56
23	100	20	*rd*	1148.79	1165.28	1155.47	1146.58	1151.17	1148.80	**1143.10**	1151.29	1147.11	**1143.10**	1150.34	1146.66	1143.10
24	100	20	*rd*	1063.85	1077.30	1072.91	1068.25	1076.44	1071.18	**1060.75**	1066.54	1064.04	**1060.75**	1064.99	1062.84	1060.75
25	200	10	*c*	835.91	851.10	843.97	829.89	838.37	833.45	827.10	836.39	830.99	**822.52**	835.02	828.37	822.52
26	200	10	*c*	728.68	740.14	734.04	717.46	725.16	722.42	715.37	729.98	723.17	**714.33**	725.99	719.99	714.33
27	200	10	*rd*	1816.64	1845.04	1829.48	1787.24	1816.84	1808.32	**1761.10**	1807.95	1787.63	1763.30	1805.70	1783.24	1761.10
28	200	10	*rd*	1471.65	1512.90	1490.73	1469.35	1486.87	1480.18	1454.90	1475.95	1461.06	**1445.94**	1470.70	1458.15	1445.94
29	200	20	*c*	923.55	935.17	929.53	918.29	922.75	920.24	912.87	920.76	916.17	**909.46**	923.22	913.51	909.46
30	200	20	*c*	820.99	838.58	830.95	819.28	822.15	820.94	**815.51**	824.51	820.64	820.67	824.84	822.19	815.51
31	200	20	*rd*	1660.42	1675.39	1667.49	1637.01	1662.91	1655.92	1620.47	1648.79	1632.10	**1614.18**	1649.12	1631.61	1614.18
32	200	20	*rd*	1444.70	1479.96	1459.87	1423.72	1449.94	1442.01	1420.45	1449.77	1432.01	**1413.32**	1438.97	1424.77	1413.32
NBKS				17			21			25			29			
Avg. deviation above BKS(%)				0.57	1.29	0.91	0.29	0.68	0.52	0.08	0.58	0.31	0.03	0.54	0.25	

Table 7
Average times (over 10 runs) of the proposed methods on the set of randomly generated instances.

Instance	n	p	Average running time (min)				
			IRFCS	GRASP × ELS	GRASP/VND	MS-ELS	MS-ILS
1	25	5	0.01	0.24	0.20	0.17	0.19
2	25	5	0.01	0.23	0.21	0.18	0.19
3	25	5	0.01	0.25	0.21	0.19	0.21
4	25	5	0.01	0.23	0.18	0.16	0.17
5	25	10	0.02	0.29	0.22	0.20	0.22
6	25	10	0.02	0.27	0.25	0.19	0.21
7	25	10	0.02	0.28	0.25	0.21	0.23
8	25	10	0.02	0.25	0.21	0.19	0.21
9	50	5	0.03	1.73	1.23	0.94	1.18
10	50	5	0.04	1.70	1.20	0.78	0.95
11	50	5	0.04	1.48	1.29	0.85	1.02
12	50	5	0.04	1.52	1.24	0.83	0.98
13	50	10	0.05	1.95	1.23	1.16	1.37
14	50	10	0.07	1.95	1.27	1.02	1.25
15	50	10	0.05	1.64	1.39	0.99	1.20
16	50	10	0.07	1.54	1.29	0.86	1.05
17	100	10	0.14	12.61	8.48	4.45	5.80
18	100	10	0.19	13.21	8.72	5.09	6.49
19	100	10	0.15	9.14	9.70	3.82	4.96
20	100	10	0.20	9.47	9.06	3.54	4.34
21	100	20	0.29	12.38	9.45	4.03	4.82
22	100	20	0.49	13.73	9.35	5.08	6.00
23	100	20	0.29	10.03	10.43	4.09	5.02
24	100	20	0.48	10.61	9.89	4.11	4.91
25	200	10	0.49	62.65	60.86	15.44	19.24

Table 7 (*continued*)

Instance	*n*	*p*	Average running time (min)				
			IRFCS	GRASP × ELS	GRASP/VND	MS-ELS	MS-ILS
26	200	10	0.60	73.34	60.95	15.39	19.47
27	200	10	0.47	52.23	63.92	13.90	17.19
28	200	10	0.58	58.13	64.33	13.02	15.48
29	200	20	0.78	71.43	64.70	18.09	22.91
30	200	20	1.19	80.67	62.91	19.55	24.41
31	200	20	0.78	57.27	68.20	16.39	20.47
32	200	20	1.19	62.95	68.59	16.18	20.14
Average			0.28	19.54	18.79	5.35	6.63

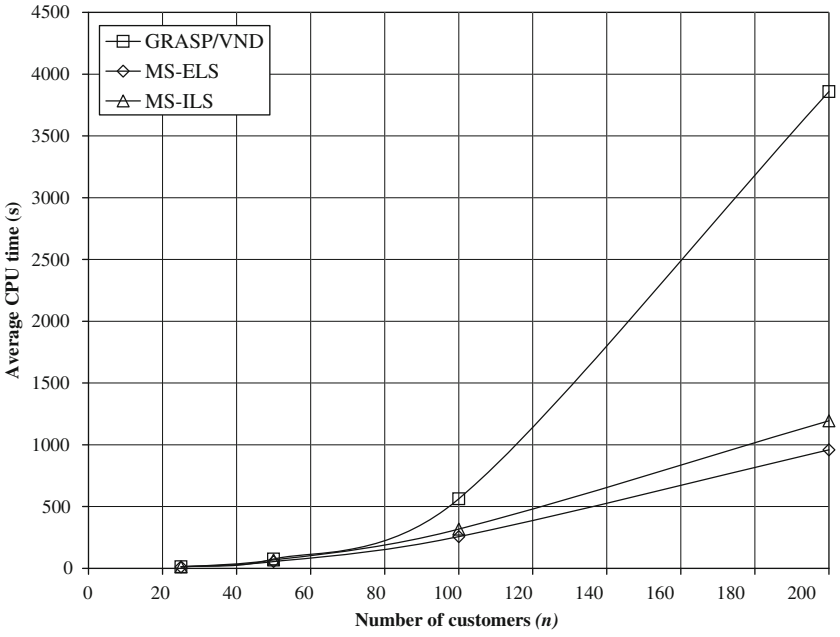

Fig. 7. Average running times for each metaheuristic.

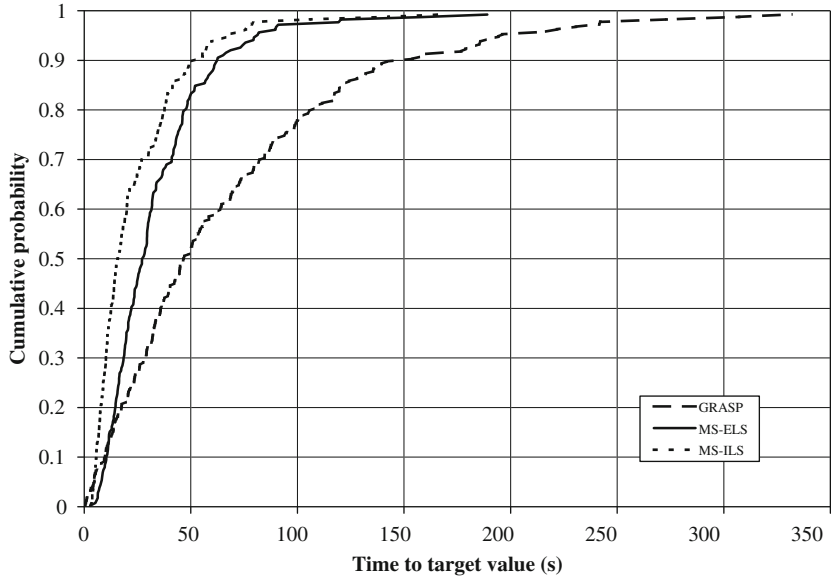

Fig. 8. Time-to-target plot for a STTRPSD with n=100 and p=20.

Table 8
Results for the MDVRP.

Instance				ALNS[a]			CGL[b]			GRASP/VND			MS-ELS			MS-ILS		
	n	p	Best known	Cost[c]	Best[d]	T[e]	Cost[f]	Best[g]	T[h]	Cost[i]	Best[j]	T[k]	Cost[i]	Best[j]	T[k]	Cost[i]	Best[j]	T[k]
P01	50	4	576.87	**576.87**	576.87	0.48	**576.87**	576.87	3.24	610.79	592.21	1.26	**576.87**	576.87	0.77	**576.87**	576.87	0.89
P02	50	4	473.53	**473.53**	473.53	0.47	473.87	473.53	3.46	556.35	529.64	1.40	**473.53**	473.53	1.01	**473.53**	473.53	1.12
P03	75	5	641.19	**641.19**	641.19	1.07	645.15	641.19	5.66	694.08	648.68	3.92	643.50	641.19	1.79	643.44	641.19	2.09
P04	100	2	1001.04	**1006.09**	1001.04	1.47	1006.66	1001.59	7.79	1071.49	1055.26	8.02	1014.07	1005.67	2.86	1008.10	1003.62	3.50
P05	100	2	750.03	752.34	751.26	2.00	753.34	750.03	8.21	803.93	769.37	8.66	**751.49**	751.15	2.63	752.54	751.15	3.13
P06	100	3	876.5	883.01	876.7	1.55	**877.84**	876.5	7.65	963.10	924.68	8.16	883.73	878.90	2.95	884.42	880.69	3.51
P07	100	4	881.97	**889.36**	881.97	1.47	891.95	885.8	7.71	955.76	925.80	8.70	893.05	887.16	2.97	892.59	888.65	3.55
P12	80	2	1318.95	1319.13	1318.95	1.25	**1318.95**	1318.95	5.57	1409.02	1326.85	3.69	**1318.95**	1318.95	1.92	**1318.95**	1318.95	2.20
P15	160	4	2505.42	2519.64	2505.42	4.22	2534.13	2505.42	14.06	2809.46	2553.80	28.65	2521.75	2505.42	7.83	**2508.05**	2505.42	9.30
P18	240	6	3702.85	3736.53	3702.85	6.98	**3710.49**	3702.85	24.85	4075.48	4209.56	96.84	3727.80	3702.85	18.81	3737.64	3702.85	22.72
P21	360	9	5474.84	**5501.58**	5474.84	9.70	5535.99	5474.84	48.16	6187.00	5903.63	325.23	5555.78	5504.04	44.35	5522.02	5490.55	52.96
Average time						2.79			12.40			44.96			7.99			9.54
Avg. deviation above BKS (%)				0.40	0.02		0.50	0.04		9.98	4.96		0.61	0.18		0.49	0.18	

[a] Results of the best variant of Pisinger and Ropke ALNS (Pisinger and Ropke, 2007) with 50 000 iterations.
[b] Results of Cordeau et al. (1997) tabu search.
[c] Average cost over 10 runs.
[d] Best solution found over 10 runs.
[e] Average time in minutes on a 3 GHz Pentium 4 over 10 runs.
[f] Cost of the solution found with standard parameters settings in a single run.
[g] Best solution found during sensitivity analysis with different parameter settings.
[h] Time in minutes on a Sun Sparcstation 10.
[i] Average cost over 10 runs.
[j] Best solution found over 10 runs.
[k] Average time in minutes on a 3.4 GHz Pentium D over 10 runs.

solution with objective function at least as good as a given target value. By analyzing the time-to-target distributions, GRASP/VND needs approximately 150 s to have a probability of 90% to find a solution as good as the given target value; on the other hand, MS-ILS needs just 50 s and MS-ELS only 60 s to reach the same quality.

4.5. Multi-depot VRP

Since the STTRPSD is a new problem, no published solution method is available for comparison. Because the MDVRP is a special case of the STTRPSD where $c_{ij}=0$, $\forall i,j \in V_1$, we tested the proposed methods on the MDVRP, and compare them against the best published metaheuristics designed to solve it.

Without changing their parameters GRASP/VND, MS-ELS and MS-ILS were applied 10 times for each instance in the set of test problems commonly used for the MDVRP (Cordeau et al., 1997) (available at http://neumann.hec.ca/chairedistributique/data/mdvrp/). Given that the STTRPSD does not have route-length constraints, we only solve the 11 MDVRP instances without such constraints. Table 8 presents the results of this experiment. The best known solutions taken from Pisinger and Ropke (2007) were used to calculate the average deviations given in the last row of Table 8 and best average results for each instance are in bold. As a reference, the results of Pisinger and Ropke (2007) ALNS and Cordeau et al. (1997) tabu search are included in Table 8.

Although the methods presented in this work are not tailored to solve the MDVRP, the proposed multi-start evolutionary local search performs well and stands as a competitive alternative. Overall, ALNS is the best method, with the smallest deviation (0.40%) and the best average performance on 6 out of the 11 instances, closely followed by MS-ILS and tabu search with the best average performance in 4 out of 11 instances. Remarkably, MS-ILS achieved an average deviation of 0.49% against 0.50% of tabu search. Not far along is MS-ELS which is also effective in the solution of the MDVRP with an average deviation of 0.61%. Finally, GRASP/VND does not perform as well in the MDVRP: despite the fact that the best results have an average deviation of 4.96%, its average deviation reaches 9.98%. In terms of computing times, the methods presented in this paper are slower than ALNS and tabu search, specially on problems P18 and P21 with over 200 customers. Nevertheless, our algorithms could be easily accelerated for the MDVRP, for instance, by implementing the Split procedure in $O(nbp)$ instead of the $O(nbp^2)$ of Eq. (22). Because the target of this paper is the STTRPSD and not the MDVRP, we decided not to change our code to tailor it to the MDVRP.

5. Conclusion and future work

In this paper, we introduced the STTRPSD, a generalization of the VRP arising from practical applications such as milk collection and postal services. To solve the STTRPSD we proposed two metaheuristics: a hybrid GRASP/VND and a multi-start evolutionary local search. Both of them perform very well when compared against cluster-first route-second, iterated route-first cluster-second and VND heuristics. The results of the computational experiments on a set of 32 randomly generated instances also unveil the robustness of the proposed metaheuristics, all of them achieving gaps to best known solutions of less than 1% even in the worst case. Among the proposed methods, the multi-start evolutionary local search is more accurate, faster, and scales better (as the number of customers increases) than the GRASP/VND. Finally, when tested on the MDVRP, a special case of the STTRPSD, the proposed multi-start evolutionary local search obtains results that are competitive to those achieved by the state-of-the art ALNS (Pisinger and Ropke, 2007) and tabu search (Cordeau et al., 1997). Future research directions include the derivation of a lower bound to evaluate the quality of solutions found with heuristics and metaheuristics, the development of population metaheuristics, and the extension of the methods to the case with multiple vehicles.

Acknowledgments

This research was partially supported by the Champagne-Ardenne Regional Council (France) and by Colciencias (Colombia).

Appendix A. Nomenclature

A.1. Notation for the integer programming formulation

G	Graph for the formulation of the STTRPSD, $G=(V,A)$.
V	Set of nodes of G.
V_C	Subset of V representing the customers.
V_D	Subset of V representing the satellite depots.
V_1	Subset of V representing the nodes that can be visited in the first-level trip.
V_2	Subset of V representing the nodes that can be visited in second-level trips.
V^j	Subset of V representing the nodes that can be visited in second-level trips departing from satellite depot j.
A	Set of arcs of G.
c_{ij}	Cost (length) of the arc between nodes i and j ($i,j \in V$).
n	Number of customers.
p	Number of satellite depots.
q_i	Demand of customer i ($i \in V_C$).
Q_V	Capacity of the truck.
Q_T	Capacity of the trailer.
$\gamma(V')$	Minimum number of second-level trips needed to serve the demand of the customers in $V' \subseteq V_C$, $\gamma(V') = \left\lceil \frac{\sum_{i \in V'} q_i}{Q_V} \right\rceil$
x^j_{lm}	Binary variable, $x^j_{lm}=1$ if arc (l,m) is traversed by the truck in a second-level trip departing from satellite depot j ($l,m \in V^j$); $x^j_{lm}=0$ otherwise.
y_{ij}	Binary variable, $y_{ij}=1$ if arc (i,j) is traversed in the first-level trip ($i,j \in V_1$); $y_{ij}=0$ otherwise.

A.2. Notation used in the metaheuristics

A.2.1. General notation

S	A solution of the STTRPSD.
$f(S)$	Objective function of solution S calculated using Eq. (1).
S^*	Best solution found during the execution of a method.
f^*	Cost of the best solution.

A.2.2. GRASP

ns	Number of iterations.
r	Cardinality of the restricted candidate list.

A.2.3. Split

T	Giant tour visiting all the customers in V_C, $T=(t_0,t_1\ldots,t_n)$.
t_j	Customer in position j of giant tour T.
F_{lj}	Cost of state $[l,j]$ in the dynamic programming method.
θ_{ijk}	Cost of the second-level trip $(k,t_i,\ldots,t_j,k)$, visiting the customers from position i to position j of giant tour T from satellite depot k.

H Auxiliary graph, $H=(X,U,W)$.
X Nodes of H, representing the states of the dynamic programming method.
U Arcs of H.
W Mapping defining the cost of the arcs of H.
PD Matrix with the record of the preceding satellite depot in the dynamic programming method.
PD Matrix with the record of the preceding customer in the dynamic programming method.
LD Last satellite depot used in the solution of the shortest path problem in H.

A.2.4. VND

$\mathcal{N}_k(S_0)$ Subset of the solution space composed of the solutions reachable from solution S_0 when neighborhood k is applied to it.
k_{max} Number of neighborhoods.

A.2.5. Multi-Start evolutionary local search

nr Number of restarts.
ni Number of iterations of ELS.
nc Number of children obtained by mutation in each iteration.
b_{max} Maximum number of pairs for the mutation operator.
b_{pert} Number of pairs for the perturbation of the best tour in each restart.
T^* Giant tour of the best solution.
T' Giant tour obtained with the mutation operator.
S' Child generated after mutation and tour splitting.
$\hat{S}$ Best child generated in each iteration.
$\hat{f}$ Cost of the best child.

Appendix B. Procedures used in *Split*

Procedure *Develop*(k,i) scans the successors of state [k,i] to update their labels and the predecessor records PD and PC. It uses a function, *MaxRank*(i) that returns the largest index j such that the second-level trip (L,t_i…,t_j,L) is feasible for any $L \in V_D$.

Algorithm 6. *Develop*[k,i]

Input: T, k, i
$last \coloneqq MaxRank(i+1)$
$cost \coloneqq F(k,i)$
$w \coloneqq t_{i+1}$
$u \coloneqq w$
for $j \coloneqq i$ to $last$ **do**
  $v \coloneqq t_j$
  $cost \coloneqq cost + c_{uv}$
  $u \coloneqq v$
  for $L \coloneqq 1$ to p **do**
    $Fnew \coloneqq cost + c_{kl} + c_{Lw} + c_{vL}$
    if $Fnew \le F[L,j]$ **then**
      $F[L,j] \coloneqq Fnew$
      $PD[L,j] \coloneqq k$
      $PC[L,j] \coloneqq i$
    end if
  end for
end for

After solving the shortest path problem in the auxiliary graph H using Algorithm 2, the solution S associated with a giant tour T can be deduced by backtracking from state [LD,n]. This is done with procedure *TourToSol* using the information stored in LD, PD and PC.

Algorithm 7. *TourToSol*(LD,PD,PC)

Input: LD, PD, PC
Output: S, solution of the STTRPSD
Create an empty solution of the STTRPSD S
Create *flt* an empty first-level trip departing from the main depot
$L \coloneqq LD$
$lisd \coloneqq L$
Insert L at the beginning of *flt*
$j \coloneqq n$
repeat
  $k \coloneqq PD[L,j]$
  $i \coloneqq PC[L,j]$
  if $lisd \neq L$ **then**
    Insert L at the beginning of *flt*
    $lisd \coloneqq L$
  end if
  Create a second-level trip $slt \coloneqq (L, t_{i+1}, \ldots, t_j, L)$
  add *slt* to S
  $L \coloneqq k$
  $j \coloneqq i$
until $j \coloneqq 0$
add *flt* to S
return S

References

Aiex, R., Resende, M., Ribeiro, C., 2007. TTTplots: a Perl program to create time-to-target plots. Optimization Letters 1, 335–366.
Baldacci, R., Mingozzi, A., 2009. A unified exact method for solving different classes of vehicle routing problems. Mathematical Programming Series A 120, 347–380.
Baldacci, R., Toth, P., Vigo, D., 2007. Recent advances in vehicle routing exact algorithms. 4OR—A Quarterly Journal of Operations Research 5, 269–298.
Bäck, T., Hoffmeister, F., Schwefel, H.-P., 1991. A survey of evolution strategies. In: Belew, R.K., Booker, L.B. (Eds.), Proceedings of the 4th International Conference on Genetic Algorithms, San Diego, Morgan Kaufmann, pp. 2–9.
Belenguer, J.-M., Benavent, E., Labadi, N., Prins, C., Reghioui, M., 2009. Split delivery capacitated arc routing problem: lower bound and metaheuristic. Transportation Science, forthcoming.
Caramia, M., Guerriero, F., 2009. A heuristic approach for the truck and trailer routing problem. Journal of the Operational Research Society, in press, doi:10.1057/jors.2009.59.
Chao, I.-M., 2002. A tabu search method for the truck and trailer routing problem. Computers & Operations Research 29, 33–51.
Claassen, G., Hendriks, T., 2007. An application of special ordered sets to a periodic milk collection problem. European Journal of Operational Research 180, 754–769.
Conover, W.J., 1998. Practical Nonparametric Statistics. Wiley, New York.
Cordeau, J.-F., Desaulniers, G., Desrosiers, J., Solomon, M.M., Soumis, F., 2002. VRP with time windows. In: Toth, P., Vigo, D. (Eds.), The Vehicle Routing Problem. SIAM, pp. 157–193.
Cordeau, J.-F., Gendreau, M., Laporte, G., 1997. A tabu search heuristic for periodic and multi-depot vehicle routing problems. Networks 30, 105–119.
Cormen, T.H., Leiserson, C.E., Rivest, R.L., 2001. Introduction to Algorithms. The MIT Press, Cambridge.
Crevier, B., Cordeau, J.-F., Laporte, G., 2007. The multi-depot vehicle routing problem with inter-depot routes. European Journal of Operational Research 176, 756–773.
Croes, G.A., 1958. A method for solving traveling-salesman problems. Operations Research 6, 791–812.
Feo, T., Resende, M., 1995. Greedy randomized adaptive search procedures. Journal of Global Optimization 6, 109–133.
Festa, P., Resende, M., 2009. An annotated bibliography of GRASP, part II: applications. International Transactions in Operational Research 16, 131–172.
Gendreau, M., Potvin, J.-Y., Bräysy, O., Hasle, G., Løkketangen, A., 2008. Metaheuristics for the vehicle routing problem and its extensions: a categorized bibliography. In: Golden, B., Raghavan, S., Wasil, E. (Eds.), The Vehicle Routing Problem: Latest Advances and New Challenges. Springer, Berlin, pp. 143–170.
Ghiani, G., Improta, G., Laporte, G., 2001. The capacitated arc routing problem with intermediate facilities. Networks 37, 134–143.

Golden, B., Raghavan, S., Wasil, E., 2008. The Vehicle Routing Problem: Latest Advances and New Challenges. Springer, Berlin.

Gonzalez-Feliu, J., Perboli, G., Tadei, R., Vigo, D., 2007. The two-echelon capacitated vehicle routing problem. Technical Report, Control and Computer Engineering Department, Politecnico di Torino.

Hansen, P., Mladenovic, N., 2001. Variable neighbourhood search: principles and applications. European Journal of Operational Research 130, 449–467.

Hoff, A., Løkketangen, A., 2007. 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, Thailand, June 10–15.

Labadi, N., Prins, C., Reghioui, M., 2008. A memetic algorithm for the vehicle routing problem with time windows. RAIRO Operations Research 42, 415–431.

Laporte, G., 2007. What you should know about the vehicle routing problem. Naval Research Logistics 54, 811–819.

Laporte, G., Semet, F., 2002. Classical heuristics for the capacitated VRP. In: Toth, P., Vigo, D. (Eds.), The Vehicle Routing Problem. SIAM, pp. 109–128.

Levy, L., Bodin, L., 2000. Scheduling of local delivery carriers for the united states postal service. In: Dror, M. (Ed.), Arc Routing: Theory, Solutions and Applications. Kluwer, Dordrecht, pp. 419–442.

Lin, S.-W., Yu, V.F., Chou, S.-Y., 2009. Solving the truck and trailer routing problem based on a simulated annealing heuristic. Computers & Operations Research 36, 1683–1692.

Lourenço, H., Martin, O., Stützle, T., 2003. Iterated local search. In: Glover, F., Kochenberger, G. (Eds.), Handbook of Metaheuristics. Kluwer, Dordrecht, pp. 321–353.

Nagy, G., Salhi, S., 2007. Location-routing: issues, models and methods. European Journal of Operational Research 177, 649–672.

Pisinger, D., Ropke, S., 2007. A general heuristic for vehicle routing problems. Computers & Operations Research 34, 2403–2435.

Prins, C., 2004. A simple and effective evolutionary algorithm for the vehicle routing problem. Computers Operations Research 31, 1985–2002.

Prins, C., 2009a. A GRASP $\times$ evolutionary local search hybrid for the vehicle routing problem. In: Pereira, F., Tavares, J. (Eds.), Bio-inspired Algorithms for the Vehicle Routing Problem. Springer, Berlin, pp. 35–53.

Prins, C., 2009b. Two memetic algorithms for heterogeneous fleet vehicle routing problems. Engineering Applications of Artificial Intelligence 22, 916–928.

Prins, C., Prodhon, C., Wolfler-Calvo, R., 2006. 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 4, 221–238.

Prins, C., Reghioui, M., 2009. A hybrid GRASP $\times$ evolutionary local search for the split delivery vehicle routing problem. In: Odysseus 2009, Cezme, Izmir, Turkey, May 26–29.

Resende, M.G.C., 2008. Metaheuristic hybridization with greedy randomized adaptive search procedures. In: Chen, Z-L., Raghavan, S. (Eds.), TutORials in Operations Research. INFORMS, pp. 295–319.

Scheuerer, S., 2006. A tabu search heuristic for the truck and trailer routing problem. Computers & Operations Research 33, 894–909.

Toth, P., Vigo, D., 2002. The Vehicle Routing Problem. SIAM, Philadelphia.

Velasco, N., Castagliola, P., Dejax, P., Guéret, C., Prins, C., 2009. A memetic algorithm for a pick-up and delivery problem by helicopter. In: Pereira, F., Tavares, J. (Eds.), Bio-inspired Algorithms for the Vehicle Routing Problem. Springer, Berlin, pp. 173–190.

Villegas, J.G., Medaglia, A.L., Prins, C., Prodhon, C., Velasco, N., 2009. GRASP/evolutionary local search hybrids for a truck and trailer routing problem. In: MIC 2009: The VIII Metaheuristics International Conference, Hamburg, July 13–16.

Wolf, S., Merz, P., 2007. Evolutionary local search for the super-peer selection problem and the p-hub median problem. In: Bartz-Beielstein, T., Blesa Aguilera, M.J., Blum, C., Naujoks, B., Roli, A., Rudolph, G., Sampels, M. (Eds.), Lecture Notes in Computer Science, vol. 4771. Springer, Heidelberg, pp. 1–15.