Computers & Operations Research 61 (2015) 143–152

Contents lists available at ScienceDirect

Computers & Operations Research

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

Skewed general variable neighborhood search for the location routing scheduling problem

Rita Macedo [a,b,*], Cláudio Alves [c], Saïd Hanafi [b], Bassem Jarboui [d], Nenad Mladenović [b], Bruna Ramos [c], J.M. Valério de Carvalho [c]

[a] Institut de Recherche Technologique Railenium, F-59300 Famars, France
[b] LAMIH-UMR CNRS 8201, Université de Valenciennes et du Hainaut-Cambrésis, Le Mont Houy, 59313 Valenciennes Cedex 9, France
[c] Centro de Investigação Algoritmi, Escola de Engenharia, Universidade do Minho , 4710-057, Braga, Portugal
[d] MODILS, Université de Sfax, Faculté de sciences économiques et de gestion, route de l'aéroport km 4, Sfax 3018, Tunisia

ARTICLE INFO

Available online 31 March 2015

Keywords:
Location
Routing
Scheduling
Variable neighborhood search

ABSTRACT

The integrated location routing scheduling problem is a variant of the well-known location routing problem. The location routing problem consists in selecting a set of depots to open and in building a set of routes from these depots, to serve a set of customers at minimum cost. In this variant, a vehicle can perform more than a single route in the planning period. As a consequence, the routes have to be scheduled within the workdays of each vehicle. The problem arises typically when routes are constrained to have a short duration. It happens for example within the boundaries of small geographic areas or in the transportation of perishable goods. In this paper, we propose a skewed general variable neighborhood search based heuristic to solve it. The algorithm is tested extensively and we show that it is efficient and provides the proven optimal solution in a significant number of cases. Moreover, it clearly outperforms a multi-start VND based heuristic that uses the same neighborhood structures.

1. Introduction

The optimization of integrated problems consists in addressing simultaneously different inter-related problems arising in the companies. This approach allows us to consider explicitly the strong inter-dependencies that arise in real world situations. The aim is to avoid finding solutions that are good or even optimal for an isolated problem, but are globally sub-optimal when considering the whole context where it occurs. The drawback of this integrated vision is that it typically results in the combination of problems that are, by themselves, already very complex.

The Location Routing Problem (LRP) is an example of such an integrated problem arising essentially in the supply chain. The problem consists in determining simultaneously the depots that should be opened or installed, together with the routes that a fleet of vehicles should perform from those depots so as to fulfill the demand of a set of customers and optimize a given objective function. Formally, the LRP combines two well-known NP-hard problems, namely the location problem and the vehicle routing problem. Hence, solving the LRP up to optimality is also NP-hard, but it may lead to significant savings compared to a separated resolution of the two problems, as in the LRP the operational costs are considered in a much broader way.

The LRP has been widely studied in the literature. Different surveys have been proposed by Laporte [25] and more recently by Nagy and Salhi [40], for example. A taxonomy was also proposed in [35], together with a classification scheme that applies to the solution approaches described in the literature. Most of the solution algorithms that are reported for the deterministic version of the problem rely on heuristic approaches. Nagy and Salhi [40] divide them into three groups: the clustering-based methods, the iterative methods and the hierarchical methods. Some examples can be found in [39,47,50,3,7,13]. In contrast, the number of exact methods is much smaller. Some contributions may be found in [26,27]. The number of contributions for variants of the LRP is also growing. The simplest variants assume, for example, constraints on the capacities of the depots [2,8,12,17,23], while others impose constraints on the total distances that each vehicle can perform [9].

In this paper, we address a variant of the LRP that has been considered recently in [28,29,1]. It combines the location problem and the Multi-trip Vehicle Routing Problem (MVRP), a variant of the

* Corresponding author at: Institut de Recherche Technologique Railenium, F-59300 Famars, France. Tel.: +33 3 27 51 19 39; fax: +33 3 27 51 13 50.
E-mail addresses: rita.sgd.macedo@gmail.com (R. Macedo), claudio@dps.uminho.pt (C. Alves), Said.Hanafi@univ-valenciennes.fr (S. Hanafi), bassem_jarboui@yahoo.fr (B. Jarboui), Nenad.Mladenovic@univ-valenciennes.fr (N. Mladenović), bruna.ramos@dps.uminho.pt (B. Ramos), vc@dps.uminho.pt (J.M. Valério de Carvalho).

http://dx.doi.org/10.1016/j.cor.2015.03.011
0305-0548/

classical vehicle routing problem where a vehicle can be assigned to more than one route per planning period. MVRP was first approached in [11]. Some exact [31,32,21,4,5] and heuristic solution methods [22,41,43,45,46] have been proposed in the literature for same variants of this problem, and surveys on this problem can be found in [15,10]. The problem can be seen as an integrated Location Routing Scheduling Problem (LRSP). It consists in choosing a set of depots to open, in building a set of routes from these depots to serve a set of customers and in assigning the routes to a fleet of capacitated vehicles. The difference between the standard LRP is that a vehicle can now perform more than a single route in the planning period, which implies scheduling the routes in the workdays of the vehicles. The problem was originally described in Lin et al. in [28] for the planning of a bill delivery service in a telecommunication company. The authors developed a metaheuristic approach based on simulated annealing to solve this variant, and compared it to the solutions provided by an exact branch-and-bound algorithm.

The work of [28] was later extended by Lin and Kwok in [29] for a multi-objective case with time and capacity constraints on the routes and capacitated depots. The authors consider an optimization criterion involving both the minimization of the total cost and the load and working time imbalance among vehicles. They propose a general tabu search algorithm with neighborhood structures consisting of insert and swap moves, a simple aspiration criterion based on the total cost of a solution, and the possibility of visiting infeasible solutions at the cost of a given penalty. Additionally, the authors develop and test a simulated annealing for this multi-objective version of the problem. They compare their different approaches on a set of real and simulated instances.

In [1], the authors propose two different formulations for the problem: a three-index commodity flow formulation and a set partitioning formulation. Based on the latter, the authors propose a branch-and-price algorithm. They define the concept of pairing, which is the schedule of one vehicle for the planning period, *i.e.* the set of routes one vehicle is scheduled to perform. The generation of pairings is done by solving a pricing problem that is an elementary shortest path problem with resource constraints. The authors propose two heuristic algorithms to solve it and only solve it exactly when no more attractive columns are identified by the heuristic methods. The approach is tested on instances with 5 potential depots and up to 40 customers. The total computing times reported in [1] go up to 6 h.

In [34], Macedo et al. explore a network flow formulation for the problem, strengthened with different families of valid inequalities. The authors compare their model with the three-index commodity flow formulation proposed in [1]. Based on this model, they develop an iterative rounding heuristic that relies on the continuous solutions provided by the linear relaxation of the model. The model is tested on benchmark instances with promising results using a commercial optimization solver.

In this paper, we propose a Variable Neighborhood Search (VNS) algorithm for the LRSP. The algorithm is based on different neighborhoods involving the routes, the workdays and the selection of depots. The algorithm is tested on a large set of benchmark instances and other randomly generated instances. To evaluate the quality of the solutions, we compare them with the lower and upper bounds provided by two integer programming formulations described in [34] and [1]. The computational results show that in many cases the VNS algorithm provides the proven optimal solution, while the optimality gap remains very small for all the remaining instances. In [33], some preliminary results are presented. Here, we provide a complete description of the algorithm with extended computational results showing the efficiency of the approach.

The paper is organized as follows. In Section 2, we formally define the problem, introduce all the related notation and review the two integer programming formulations that will be used to evaluate the quality of the solutions provided by our VNS algorithm. The algorithm is described in Section 3. In Section 4, we report an extensive set of computational experiments performed using benchmark and randomly generated instances. We further describe a multi-start VND based heuristic that uses some of the neighborhood structures used in our VNS. In order to asses the efficiency of the proposed VNS, we compare its solutions with lower bounds obtained by the two models described in Section 2 and with the results obtained by the multi-start VND algorithm. Some final conclusions are drawn in Section 5.

2. The integrated location routing scheduling problem

The integrated location routing scheduling problem is characterized by a set D of n_d depots that may be kept closed or opened, in which case a fixed cost of C_f^d units for each depot $d \in D$ is incurred. Each depot $d \in D$ has a capacity L_d, which may be different from depot to depot. The depots are used as a starting and ending point for the routes of the available vehicles. Each route must start and end at the same depot and each depot d has a set of allocated vehicles V_d, with $V = \cup_{d \in D} V_d$. All the vehicles have the same capacity Q. The workday of a vehicle consists of the set of routes performed during the same planning period. The time length of a workday cannot exceed W units. The set of n customers is denoted by N. Each customer $i \in N$ has an associated demand of b_i units, and it must be served by one and only one vehicle. The use of a vehicle v implies a fixed cost of C_v units. The cost and load of a route r are represented, respectively, by c_r and l_r.

The integrated location routing scheduling problem consists in determining the depots that should be opened, the routes that should be performed, and the way these routes are scheduled within the workdays of the vehicles so as to minimize the total cost involving the fixed costs for opening the depots and using the vehicles, plus the variable costs related to the distances traveled by the vehicles. The main difference between this problem and the standard LRP is that a vehicle can now perform more than a single route during its workday, which contrasts with the standard one-to-one correspondence between routes and vehicles assumed in the standard version.

2.1. A three-index commodity flow formulation

Let G be a digraph with a set of nodes representing the customers and the depots, and A be a set of arcs representing the links between these nodes, such that $G = (N \cup D, A)$. The time required to travel through the arc $(i,j) \in A$ is denoted by t_{ij}, while the corresponding unitary cost is represented by C^o.

For each vehicle $v \in V$, there is a binary variable x_{ijv} that determines whether vehicle v uses the arc $(i,j) \in A$ or not. The amount of flow that the vehicle $v \in V$ carries through arc $(i,j) \in A$ is represented by the integer variable y_{ijv}. The choice between opening or not a depot $d \in D$ is represented by the binary variable λ_d. The use of a vehicle $v \in V$ is determined by the binary variable h_v.

The three-index commodity flow model proposed in [1] states as follows:

$$\min \quad z_1(x,y,\lambda,h) = \sum_{d \in D} C_f^d \lambda_d + C_v \sum_{v \in V} h_v + \sum_{v \in V} \sum_{(i,j) \in A} t_{ij} x_{ijv} \tag{1}$$

$$\text{s.t.} \quad \sum_{v \in V} \sum_{j \in (N \cup D)} x_{ijv} = 1, \quad \forall i \in N, \tag{2}$$

$$\sum_{j \in (N \cup D)} x_{ijv} - \sum_{j \in (N \cup D)} x_{jiv} = 0, \quad \forall i \in N \cup D, \ \forall v \in V, \tag{3}$$

$$\sum_{v \in V_d} \sum_{j \in N} y_{djv} \le L_d \lambda_d, \quad \forall d \in D, \tag{4}$$

$$y_{ijv} \le Q x_{ijv}, \quad \forall (i,j) \in A, \ \forall \ v \in V, \tag{5}$$

$$\sum_{j \in N} y_{ijv} - \sum_{j \in N} y_{jiv} + b_i \sum_{j \in (N \cup D)} x_{ijv} = 0, \quad \forall i \in N, \ \forall v \in V, \tag{6}$$

$$\sum_{(i,j) \in A} t_{ij} x_{ijv} \le W h_v, \quad \forall v \in V, \tag{7}$$

$$x_{djv} = 0, \quad \forall d \in D, \ \forall j \in (N \cup D), \ \forall v \in V_t, \ \forall t \in D \setminus \{d\}, \tag{8}$$

$$x_{ijv} \in \{0,1\}, \quad \forall (i,j) \in A, \ \forall v \in V, \tag{9}$$

$$y_{ijv} \ge 0, \quad \forall (i,j) \in A, \ \forall v \in V, \tag{10}$$

$$\lambda_d \in \{0,1\}, \quad \forall d \in D, \tag{11}$$

$$h_v \in \{0,1\}, \quad \forall v \in V. \tag{12}$$

Function (1) represents the optimization criterion of the problem, which consists in the minimization of the fixed and variable costs. Constraints (2)–(3) force the customers to be visited exactly once. The constraints on the capacities of the depots and vehicles are represented respectively through the inequalities (4) and (5)–(6). Constraints (7) ensure that the maximum time length W of each vehicle workday is not exceeded. Constraints (8) ensure that a vehicle travels only using the arcs related to its depot.

2.2. An alternative network flow model

The integrated location routing scheduling problem can be formulated using an alternative network flow formulation, as proposed in [34]. Network flow models have been successfully used to solve difficult NP-hard problems [48,49,30,42,31,24]. The model is based on a set of acyclic digraphs $\Pi_d = (\Delta, \Psi_d)$, $\forall d \in D$, with vertices in $\Delta = \{0, 1, \ldots, W\}$ representing discrete time instants from 0 to the limit W, and arcs representing feasible routes or waiting time periods of length 1. Each arc $(u,v)^r \in \Psi_d$ represents a route r from the time instant u to the time instant v. A route r consists in a sequence N_r of customers (with $N_r \subseteq N$) whose total demand l_r is smaller than or equal to the capacity Q of the vehicles, and whose duration is not longer than a maximum duration W of a vehicle workday. The set of all feasible routes from a depot $d \in D$ is represented by R_d, with $R = \cup_{d \in D} R_d$. We have that $\Psi_d = \{(u,v)^r : 0 \le u < v \le W, r \in R_d\} \bigcup \{(u,v)^o : 0 \le u < v \le W, \ v = \ u+1\}$, $\forall d \in D$.

The model has general integer flow variables x^d_{uvr} associated to each arc $(u,v)^r \in \Psi_d$ of a route $r \in R$ starting at instant u and ending at instant v. The total flow on a graph Π_d, $d \in D$, is represented by the variable z^d. The opening of a depot $d \in D$ is determined by a binary variable λ_d. The network flow model for the integrated location routing and scheduling problem states as follows:

$$\min \quad z_2(x,z,\lambda) = \sum_{d \in D} C^d_f \lambda_d + C_v \sum_{d \in D} z^d + \sum_{d \in D} \sum_{(u,v)^r \in \Psi_d} c_r x^d_{uvr} \tag{13}$$

$$\text{s.t.} \quad \sum_{d \in D} \sum_{(u,v)^r \in \Psi_d \mid i \in N_r} x^d_{uvr} = 1, \quad \forall i \in N, \tag{14}$$

$$-\sum_{(u,v)^r \in \Psi_d} x^d_{uvr} + \sum_{(v,y)^t \in \Psi_d} x^d_{vyt} = \begin{cases} z^d & \text{if } v = 0 \\ 0 & \text{if } v = 1, \ldots, W-1, \\ -z^d & \text{if } v = W \end{cases} \quad \forall d \in D, \tag{15}$$

$$\sum_{(u,v)^r \in \Psi_d} l_r x^d_{uvr} \le L_d, \quad \forall d \in D, \tag{16}$$

$$M \lambda_d \ge z^d, \quad \forall d \in D, \tag{17}$$

$$x^d_{uvr} \ge 0 \quad \text{and integer}, \quad \forall (u,v)^r \in \Psi_d, \quad \forall d \in D, \tag{18}$$

$$z^d \ge 0 \quad \text{and integer}, \quad \forall d \in D, \tag{19}$$

$$\lambda_d \in \{0,1\}, \quad \forall d \in D. \tag{20}$$

The objective function is formulated through the expression (13). Constraints (14) force an unique (but mandatory) visit to each customer. The conservation of flow along the graphs Π_d, $d \in D$, is ensured by constraints (15). The capacities of the depots are enforced through the constraints (16). Let M be a sufficiently large number, for example, an upper bound for the total number of used vehicles. Constraints (17) are used to guarantee the opening of a depot whenever there are routes assigned to it in the solution.

In order to strengthen this formulation, we use the three valid inequalities described in [34]. The first two, (21) and (22), correspond to enforcing a minimum number of depots to open, D^{min}, and a minimum number of vehicles to use, V^{min}. To calculate these two values we solve a one-dimensional bin-packing problem, resorting to dual-feasible functions [11]:

$$\sum_{d \in D} \lambda_d \ge D^{min} \tag{21}$$

$$\sum_{d \in D} \sum_{(u,v)^r \in \Psi_d} x^d_{uvr} \ge V^{min} \tag{22}$$

The last one, (23), enforces that at least one workday is to be performed from an open depot:

$$\sum_{(0,v)^r \in \Psi_d} x^d_{0vr} \ge \lambda_d, \quad \forall d \in D \tag{23}$$

3. A variable neighborhood search algorithm

Variable Neighborhood Search, first introduced by Mladenović and Hansen [36], is a metaheuristic based on the systematic exploration of different neighborhood structures, within a local search routine. Generally speaking, there is a change of neighborhood every time that the local search stops when a local optimum is reached. This avoids the algorithm being trapped, given that a local optimum may not remain optimal when another neighborhood structure is considered. VNS has been successfully applied to a variety of combinatorial optimization problems. It has a simple implementation, when compared with other metaheuristics, given that it typically does not consider many parameters. Surveys of some of these applications, among which of location and of routing problems, can be found in [20]. Several extensions of this metaheuristic can be found in [19].

In this section, we describe the set of neighborhood structures used in our method, as well as the solution evaluation function and the global algorithm.

3.1. Search space and evaluation function

The search space is the space of all possible solutions that can be visited by VNS during the search. Given a solution S, let $m(S)$ be its number of open depots, $v(d)$ be the number of workdays associated to depot d (and consequently the number of used vehicles of that depot), $r(d,v)$ be the number of routes allocated to vehicle $v \in V_d$ and $c(d,v,r)$ be the number of customers on route r allocated to vehicle $v \in V_d$.

In order to represent a solution in an unique way, we use a four-dimensional matrix as a solution, whose elements s_{dvrc} represent the cth customer to be served in the rth route of vehicle v from depot d, for $d=\{1,\dots,m(S)\}$, $v=\{1,\dots,v(d)\}$, $r=\{1,\dots,r(d,v)\}$, $c=\{1,\dots,c(d,v,r)\}$. Each route is represented by a sequence of $c(d,v,r)+2$ nodes to visit $(\tau_0,\tau_1,\dots,\tau_{c(d,v,r)},\tau_{c(d,v,r)+1})$, where τ_0 and $\tau_{c(d,v,r)+1}$ represent depot d and $\tau_{i\in\{1,\dots,c(d,v,r)\}}$ represent the served customers in the route. Let, in addition, t_{τ_i,τ_j} represent the cost of traveling from τ_i to τ_j.

During the exploration of the search space we allow infeasible solutions to be visited. There are two possible infeasibilities. The proposed VNS algorithm visits infeasible solutions that exceed the capacity of a depot or that exceed the capacity of a vehicle. Hence, the evaluation function of a solution S considers all fixed and variable costs of the solution, which have been previously described, added to the penalties associated to the violation of the capacity constraints for depots and vehicles. More precisely, we define the evaluation function of a solution S, $z_3(S)$, as the sum of all the costs and penalizations of depots, vehicles and routes:

$$z_3(S) = \text{depot costs} + \text{vehicle costs} + \text{route costs},$$

where

$$\text{depot costs} = \sum_{d=1}^{m(S)} \left(C_f^d + \alpha \max\left\{0, \sum_{v=1}^{v(d)} \sum_{r=1}^{r(d,v)} \sum_{c=1}^{c(d,v,r)} b_{s_{dvrc}} - L_d \right\}\right),$$

$$\text{vehicle costs} = C_v \sum_{d=1}^{m(S)} v(d),$$

$$\text{route costs} = \sum_{d=1}^{m(S)} \sum_{v=1}^{v(d)} \sum_{r=1}^{r(d,v)} \left(\sum_{c=0}^{c(d,v,r)} t_{\tau_c,\tau_{c+1}} + \beta \max\left\{0, \sum_{c=1}^{c(d,v,r)} b_{s_{dvrc}} - Q \right\}\right),$$

and α represents the penalty for exceeding the capacity of a vehicle and β the penalty for exceeding the capacity of a depot. Remind that C_f^d and C_v stand for the fixed costs of opening a depot and using a vehicle, respectively.

Proposition 1. *If a solution S is feasible, then*

$$\max\left\{0, \sum_{v=1}^{v(d)} \sum_{r=1}^{r(d,v)} \sum_{c=1}^{c(d,v,r)} b_{s_{dvrc}} - L_d \right\} = 0, \quad \forall d \in 1,\dots,m(S),$$

$$\max\left\{0, \sum_{c=1}^{c(d,v,r)} b_{s_{dvrc}} - Q \right\} = 0, \quad \forall d \in \{1,\dots,m(S)\},$$

$$v=\{1,\dots,v(d)\},\ r \in \{1,\dots,r(d,v)\},$$

and thus

$$z_1(x,y,\lambda,h) = z_2(x,z,\lambda) = z_3(S).$$

The penalization parameters α and β are updated dynamically, in order to intensify or diversify the search. They are increased when the current solution violates the corresponding constraint, in order to intensify the search by driving the moves through the feasible solution space. When, in the other hand, the current solution respects the corresponding constraint, the parameter is decreased, in order to broaden the search space by allowing the violation of the capacity constraint. This way, the search is diversified. Algorithm 1 describes the updating process of parameters α and β for an iteration with a current solution S.

Algorithm 1. Dynamic update for α and β.

```
Input: S, α, β, ε_α, ε_β, (α, β > 0; ε_α, ε_β ∈ ]0, 1[)
if Σ_{v=1}^{v(d)} Σ_{r=1}^{r(d,v)} Σ_{c=1}^{c(d,v,r)} b_{s_dvrc} ≤ L_d,   ∀d ∈ {1, …, m(S)} then
 | α = α(1 − ε_α);
else
 | α = α(1 + ε_α);
if Σ_{c=1}^{c(d,v,r)} b_{s_dvrc} ≤ Q,   ∀d ∈ {1, …, m(S)}, ∀v ∈ {1, …, v(d)}, ∀i ∈ {1, …, r(d, v)}
then
 | β = β(1 − ε_β);
else
 | β = β(1 + ε_β);
return α, β
```

In order to evaluate each visited solution efficiently, we do not compute entirely the evaluation function at each move, but use updating functions, according to the structure of the corresponding neighborhood.

3.2. Neighborhood structures

We define a set of six different neighborhood structures within the solution space, $\mathcal{N}=\{\mathcal{N}_1,\dots,\mathcal{N}_6\}$. The two first neighborhoods, $\mathcal{N}_1$ and $\mathcal{N}_2$, are used to improve the cost of routes, $\mathcal{N}_3$ and $\mathcal{N}_4$ aim to improve the cost of workdays, whereas $\mathcal{N}_5$ and $\mathcal{N}_6$ are used for the location problem. Neighborhoods $\mathcal{N}_1$, $\mathcal{N}_5$ and $\mathcal{N}_6$ are used in the shaking phase of VNS. Neighborhoods $\mathcal{N}_5$ and $\mathcal{N}_6$ allow the diversification of the search by opening a new depot. However, those moves can perturb strongly good local optima reached during the search. This is why we combine neighborhoods $\mathcal{N}_1$ with $\mathcal{N}_5$ and $\mathcal{N}_6$, in order to have smaller perturbations.

Whenever a route r is shifted from one depot to another depot ($\mathcal{N}_3$, $\mathcal{N}_4$, $\mathcal{N}_5$, $\mathcal{N}_6$), we consider a rearrangement of the customers' visit orders (r'). The depot is removed and the last visited customer becomes the precedent of the first visited customer. The new depot is then inserted in the best position, between two consecutive customers. In the example illustrated in Fig. 1, route $r=(d_1,7,3,5,1,d_1)$ becomes route $r'=(d_2,1,7,3,5,d_2)$.

We use the following notation to explain more clearly the neighborhood structures of our algorithm. Given a solution S, S_d represents the part of the solution concerning an open depot d, S_{dv} the part of the solution concerning the used vehicle v, allocated to depot d and S_{dvr} the part of the solution concerning route r performed by vehicle v, allocated to depot d. This way, S_d represents a set of workdays, S_{dv} a sequence of routes, and S_{dvr} a sequence of customers to visit (and corresponding beginning and ending depot).

3.2.1. Routing neighborhoods

- *Shift move of a customer* ($\mathcal{N}_1$): A neighbor of a solution S is obtained by removing a customer from its position and inserting it in a different position, within the same route or in another route, from the same or a different depot. It corresponds to a *shift* move of a customer. Fig. 2 represents a move within neighborhood $\mathcal{N}_1$. In this example, customer 1 is shifted from route S_{dvr} to route $S_{d^*v^*r^*}$.
- *Swap move of two customers* ($\mathcal{N}_2$): A neighbor of a solution S is obtained by exchanging two customers, from the same route or from different routes, from the same or a different depot. It

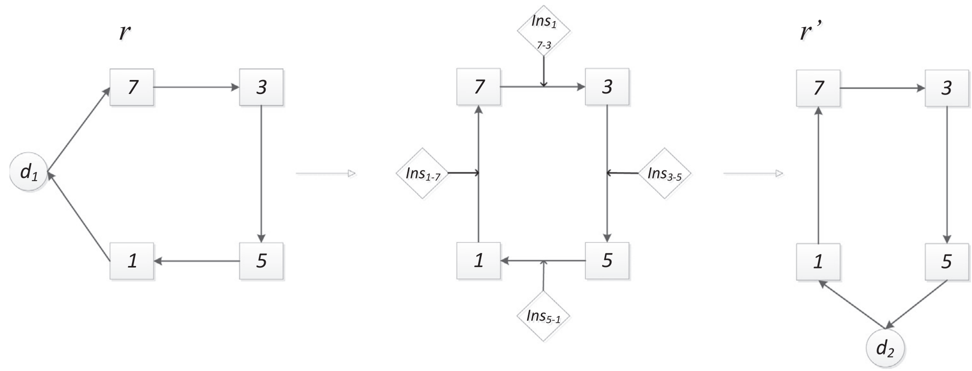

Fig. 1. Rearrangement of routes.

$$S_{dvr} = (d,4,5,1,3,d) \quad S_{d^*v^*r^*} = (d^*,2,6,d^*) \xrightarrow[S' \in \mathcal{N}_1(S)]{} S'_{dvr} = (d,4,5,3,d) \quad S'_{d^*v^*r^*} = (d^*,2,1,6,d^*)$$

Fig. 2. Example of a move in neighborhood $\mathcal{N}_1$.

$$S_{dvr} = (d,4,5,1,3,d) \quad S_{d^*v^*r^*} = (d^*,2,6,d^*) \xrightarrow[S' \in \mathcal{N}_2(S)]{} S'_{dvr} = (d,4,5,6,3,d) \quad S'_{d^*v^*r^*} = (d^*,2,1,d^*)$$

Fig. 3. Example of a move in neighborhood $\mathcal{N}_2$.

$$S_{dv} = (r_1,r_2,r_3,r_4) \quad S_{d^*v^*} = (r_5,r_6) \xrightarrow[S' \in \mathcal{N}_3(S)]{} S'_{dv} = (r_1,r_2,r_4) \quad S'_{d^*v^*} = (r_5,r'_3,r_6)$$

Fig. 4. Example of a move in neighborhood $\mathcal{N}_3$.

$$S_{dv} = (r_1,r_2,r_3,r_4) \quad S_{d^*v^*} = (r_5,r_6) \xrightarrow[S' \in \mathcal{N}_4(S)]{} S'_{dv} = (r_1,r_2,r'_6,r_4) \quad S'_{d^*v^*} = (r_5,r'_3)$$

Fig. 5. Example of a move in neighborhood $\mathcal{N}_4$.

$$S_d = \{(r_1,r_2),(r_3,r_5)\} \quad S_{d^*} = \emptyset \xrightarrow[S' \in \mathcal{N}_5(S)]{} S'_d = \{(r_1,r_2)\} \quad S'_{d^*} = \{(r'_3,r'_5)\}$$

Fig. 6. Example of a move in neighborhood $\mathcal{N}_5$.

$$S_d = \{(r_1,r_2),(r_3,r_5)\} \quad S_{d^*} = \emptyset \xrightarrow[S' \in \mathcal{N}_6(S)]{} S'_d = \emptyset \quad S'_{d^*} = \{(r'_1,r'_2),(r'_3,r'_5)\}$$

Fig. 7. Example of a move in neighborhood $\mathcal{N}_6$.

corresponds to a *swap* move of two customers. Fig. 3 represents a move within neighborhood $\mathcal{N}_2$. In this example, customer 1 of route S_{dvr} is swapped with customer 6 of route $S_{d^*v^*r^*}$.

3.2.2. Workday neighborhoods

- *Shift move of a route* ($\mathcal{N}_3$): A neighbor of a solution S is obtained by removing a route r from the workday of a vehicle, and inserting it in a different workday of a vehicle associated to the same depot or a different depot. It corresponds to a *shift* move of a route. Fig. 4 represents a move within neighborhood $\mathcal{N}_3$. In this example, the customers of route r_3 of workday S_{dv} are shifted to workday $S_{d^*v^*}$. Route r'_3 represents a more efficient rearrangement of those customers, which are now visited by a vehicle allocated to depot d^*.
- *Swap move of two routes* ($\mathcal{N}_4$): A neighbor of a solution S is obtained by exchanging two routes from the workday of two vehicles associated to the same depot or to different depots. It corresponds to a *swap* move of two routes. Fig. 5 represents a move within neighborhood $\mathcal{N}_4$. In this example, route r_3 of vehicle S_{dv} is swapped with route r_6 of vehicle $S_{d^*v^*}$. Again, routes r'_3 and r'_6 represent, respectively, a more efficient rearrangement of customers of routes r_3 and r_6, which are now visited by vehicles allocated to other depots.

3.2.3. Location neighborhoods

- *Shift move of a workday* ($\mathcal{N}_5$): A neighbor of a solution S is obtained by opening a closed depot. It consists of removing a workday of a vehicle from its associated depot to a closed depot. Fig. 6 represents a move within neighborhood $\mathcal{N}_5$. In this example, depot d^*, previously closed, is opened, as the workday (r_3, r_5) is removed from depot d and allocated to depot d'. Routes r'_3 and r'_5 represent more efficient rearrangements of the customers visited in r_3 and r_5, respectively.
- *Shift move of a depot* ($\mathcal{N}_6$): A neighbor of a solution S is obtained by opening a new depot and closing another one. It consists of shifting the workdays of all vehicles from their associated depot to a new depot. Fig. 7 represents a move within neighborhood $\mathcal{N}_6$. In this example, depot d', previously closed, is opened, as all the workdays of vehicles allocated to depot d' are removed from it and allocated to vehicles of depot d'. Again, routes r'_1, r'_2, r'_3 and r'_5 represent more efficient rearrangements of the customers visited in r_1, r_2, r_3 and r_5, respectively.

3.3. Skewed general VNS algorithm

We propose the VNS method summarized in Algorithm 2. It starts with an initial solution, generated by a simple greedy heuristic. The local search implemented corresponds to a variable neighborhood descent (VND) algorithm, where neighborhoods $\mathcal{N}_1, \mathcal{N}_2, \mathcal{N}_3$ and $\mathcal{N}_4$ are used sequentially to improve the current solution. This corresponds to a General VNS (GVNS). GVNS has been successfully applied to routing problems, for example to several variants of the travelling salesman problems [38,37].

Neighborhoods $\mathcal{N}_{i \in \{1,5,6\}}$ are used in the shaking phase (Shaking). Neighborhoods $\mathcal{N}_5$ and $\mathcal{N}_6$ allow the diversification of the search by opening a new depot. As those moves can perturb strongly local optima reached during the search, we combine neighborhood $\mathcal{N}_1$ in order to have smaller perturbations. More precisely, we apply k consecutive moves from those neighborhoods in order to perturb the current solution. At each of those k iterations, one of the three neighborhoods is selected according to a probability ($P_{\mathcal{N}_{i \in \{1,5,6\}}}$).

In order to more easily explore valleys that are far from the incumbent, we apply a Skewed version of GVNS (SGVNS). This means that we accept to visit a solution that is worse than the incumbent if this solution is sufficiently different (according to a distance function ρ) from it.

Algorithm 2. SGVNS algorithm.

```
Input: Set of neighborhood structures
    N = {N_{i∈{1,...,6}}}, P_{N_{i∈{1,5,6}}}, k_max, ρ, η
Initialization: find an initial solution S with a greedy
    randomized heuristic;
S* ← S;
repeat
    k ← 1;
    while k ≤ k_max do
        S' ← Shaking(S, N_{i∈{1,5,6}}, P_{N_{i∈{1,5,6}}}, k);
        S'' ← VND(S', N_{i∈{1,...,4}});
        if f(S'') < f(S*) and S'' is feasible then
            S* ← S'';
        if f(S'') < (1 + ηρ(S, S''))f(S) then
            S ← S'';
            k ← 1;
        else
            k ← k + 1;
        Dynamic update for α and β;
until a termination condition is met;
return S*
```

We propose a distance function ρ that expresses the structural difference between two solutions S and $\overline{S}$ in what concerns their open depots. Let π (respectively $\overline{\pi}$) be equal to 1 if depot i is open in solution S (respectively $\overline{S}$) and equal to 0 otherwise. We define $\rho(S,\overline{S}) = \frac{\sum_{i=1}^{n_d} |\pi_i - \overline{\pi}_i|}{m}$.

4. Computational experiments

To evaluate the quality of our method, we conducted a set of computational experiments. The tests were run on a Pentium 4 with 3.6 GHz and 2 GB of RAM.

4.1. Test instances and parameters

The used instances were generated from instances $p01$, $p03$ and $p07$, described in [14]. Instances $p03_2$, $p07_2$, $p07_3$ and $p07_4$ are derived from the instances $p03$ and $p07$, considering a rearrangement of the customers' coordinates. We consider instances with 25, 40, 50, 75 and 100 customers. The ones with 25 and 40 customers are the same as the ones proposed in [1]. For all instances there are five available depots. The duration of every route equals its traveled distance, and distances between customers and depots are rounded to the nearest smaller integer. Each of these instances was tested for two different values of Q and two different values of W.

We further consider a set of benchmark instances for the location routing problem with capacities on depots and vehicles. The considered instances, proposed in [6], do not consider a fixed cost for using vehicles ($C_v = 0$). The difference between the LRP and the LRSP is that in the latter we can perform several routes by incurring on a single fixed cost of using one vehicle, whereas in the LRP each route will have an associated fixed cost for using the corresponding vehicle. This means that for this set of instances the cost of a solution should be the same and thus we can compare our algorithm with other approaches of the literature that tackle the LRP. Note, however, that our algorithm was developed for dealing and taking advantage of the multi-trip aspect of the LRSP.

The SGVNS algorithm (Algorithm 2) was run 5 times. Its termination condition corresponds to a CPU time limit of $(n \times n_d)$ seconds (125 s; 200 s; 250 s; 375 s; 500 s) and we consider the parameters $k_{max} = 10$, $P_{\mathcal{N}_1} = 0.7$, $P_{\mathcal{N}_5} = 0.05$, $P_{\mathcal{N}_6} = 0.25$ and $\eta = 0.1$. The penalization parameters (Algorithm 1) are set to $\alpha = 0.1$, $\varepsilon_\alpha = 0.001$, $\beta = 0.1$, $\varepsilon_\beta = 0.001$.

4.2. Multi-start algorithm

In order to assess the quality of our VNS method, we compare its solutions with the ones obtained with another heuristic approach, a multi-start VND method using some of the same neighborhood structures defined in Section 3. The idea behind this method is to escape from local optima by iteratively restarting the algorithm with a different solution. The multi-start VND method we propose is summarized in Algorithm 3. In a first phase, a random solution is obtained by the same constructive algorithm we use in Algorithm 2 to obtain an initial solution. In a second phase, we apply the local search method also used in Algorithm 2. It consists of a VND where neighborhoods $\mathcal{N}_{i \in \{1,\dots,4\}}$ are used sequentially. At each iteration, if a better solution is found in the VND local search, it is kept as the best one.

Algorithm 3. Multi-start algorithm.

```
Input: Set of neighborhood structures N = {N_{i∈{1,...,4}}}
f* ← ∞;
repeat
    find a random solution S;
    S' ← VND(S, N_{i∈{1,...,4}});
    if f(S') < f* then
        S* ← S';
        f* ← f(S');
until a termination condition is met;
return S*
```

We use the same termination condition as for the SGVNS and the penalization parameters were set to $\alpha = \beta = 1000$.

4.3. Analysis of the performance of SGVNS

In order to further analyze the performance of our method, we compare the solution values obtained with the best lower bounds obtained with the network flow model (13)–(20), strengthened with the valid inequalities described in Section 2.2, and the three-index commodity flow model (1)–(12).

Table 1
Results for 25 and 40 customers.

Inst	*W*	*Q*	*NF model*			*Multi-start VND*						*SGVNS*					
			lb	*ub*	*gap*	z^b	t^b	gap^b	z^{av}	t^{av}	gap^{av}	z^b	t^b	gap^b	z^{av}	t^{av}	gap^{av}
$C_{25,1}$	140	50	4698	4751	1.1%	4751	28.3	1.1%	4757.8	73.3	1.3%	4751	7.2	1.1%	4751.0	8.1	1.1%
$C_{25,1}$	160	50	4562	4646	1.8%	4751	8.5	4.1%	4751.0	32.8	4.1%	4646	35.8	1.8%	4646.0	30.8	1.8%
$C_{25,1}$	140	70	4439	4590	3.4%	4677	8.7	5.4%	4678.0	49.8	5.4%	4580	4.5	3.2%	4580.0	14.6	3.2%
$C_{25,1}$	160	70	4372	4460	2.0%	4492	60.2	2.7%	4582.6	54.6	4.8%	4460	21.7	2.0%	4460.0	33.5	2.0%
$C_{25,2}$	140	50	4764	4764	0.0%	4766	41.3	0.0%	4852.6	46.9	1.9%	4764	41.5	0.0%	4764.0	32.4	0.0%
$C_{25,2}$	160	50	4608	4759	3.3%	4760	89.1	3.3%	4761.2	80.5	3.3%	4759	25.7	3.3%	4759.2	38.5	3.3%
$C_{25,2}$	140	70	4553	4699	3.2%	4700	39.1	3.2%	4702.6	34.8	3.3%	4699	22	3.2%	4700.6	33.2	3.2%
$C_{25,2}$	160	70	4474	4474	0.0%	4486	108.6	0.3%	4646.0	83.5	3.8%	4474	53.7	0.0%	4477.0	40.4	0.1%
$C_{25,3}$	140	60	4806	4806	0.0%	5043	107.2	4.9%	5059.4	83.3	5.3%	4817	32.8	0.2%	4817.0	18.9	0.2%
$C_{25,3}$	160	60	4699	4800	2.1%	4815	76.5	2.5%	4870.0	80.7	3.6%	4800	18.3	2.1%	4800.0	26.4	2.1%
$C_{25,3}$	140	80	4607	4721	2.5%	4732	114.5	2.7%	4739.6	67.7	2.9%	4721	13.7	2.5%	4721.0	23.2	2.5%
$C_{25,3}$	160	80	4467	4722	5.7%	4722	102.9	5.7%	4732.8	54.4	6.0%	4721	12.9	5.7%	4721.0	53.3	5.7%
$C_{25,4}$	140	60	4677	4767	1.9%	4767	103.9	1.9%	4777.2	56.5	2.1%	4767	0.4	1.9%	4767.0	6.0	1.9%
$C_{25,4}$	160	60	4539	4767	5.0%	4767	123.4	5.0%	4769.6	68.3	5.1%	4767	0.3	5.0%	4767.0	6.1	5.0%
$C_{25,4}$	140	80	4446	4687	5.4%	4684	21.7	5.4%	4684.0	60.7	5.4%	4684	0.2	5.4%	4684.0	11.4	5.4%
$C_{25,4}$	160	80	4319	4684	8.5%	4684	3.0	8.5%	4685.2	27.6	8.5%	4511	32.4	4.4%	4516.6	53.6	4.6%
$C_{25,5}$	140	60	4779	4779	0.0%	4999	80.5	4.6%	4999.8	49.1	4.6%	4779	6.7	0.0%	4779.0	22.2	0.0%
$C_{25,5}$	160	60	4709	4774	1.4%	4774	76.5	1.4%	4787.4	62.4	1.7%	4774	10.8	1.4%	4774.0	13.2	1.4%
$C_{25,5}$	140	80	4645	4715	1.5%	4715	41.6	1.5%	4715.0	41.9	1.5%	4715	44.4	1.5%	4716.4	19.5	1.5%
$C_{25,5}$	160	80	4509	4509	0.0%	4715	22.6	4.6%	4715.0	38.0	4.6%	4509	0.8	0.0%	4633.2	18.2	2.8%
$C_{25,6}$	140	50	4687	4755	1.5%	4755	106.7	1.5%	4762.4	71.1	1.6%	4755	9.4	1.5%	4755.0	7.2	1.5%
$C_{25,6}$	160	50	4540	4540	0.0%	4755	10.2	4.7%	4759.8	45.6	4.8%	4606	3.3	1.5%	4647.4	42.8	2.4%
$C_{25,6}$	140	70	4480	4480	0.0%	4699	107.7	4.9%	4699.8	67.0	4.9%	4480	68.4	0.0%	4480.4	68.1	0.0%
$C_{25,6}$	160	70	4403	4474	1.6%	4478	81.5	1.7%	4479.2	30.9	1.7%	4477	8.8	1.7%	4477.0	20.8	1.7%
$C_{25,7}$	140	50	5083	5083	0.0%	5084	33.4	0.0%	5102.0	78.1	0.4%	5083	47.2	0.0%	5083.0	24.7	0.0%
$C_{25,7}$	160	50	4864	4864	0.0%	4920	44.7	1.2%	4942.2	70.6	1.6%	4864	3.5	0.0%	4864.0	15.3	0.0%
$C_{25,7}$	140	70	4772	4772	0.0%	4785	64.6	0.3%	4820.0	77.3	1.0%	4772	43.3	0.0%	4772.0	23.1	0.0%
$C_{25,7}$	160	70	4648	4771	2.6%	4772	101.8	2.7%	4776.8	58.6	2.8%	4771	46.5	2.6%	4771.0	30.5	2.6%
$C_{25,8}$	140	50	4889	4889	0.0%	5053	61.8	3.4%	5078.6	61.0	3.9%	4889	1.6	0.0%	4889.0	22.6	0.0%
$C_{25,8}$	160	50	4723	4860	2.9%	4860	60.8	2.9%	4871.6	62.1	3.1%	4860	58.5	2.9%	4860.0	27.1	2.9%
$C_{25,8}$	140	70	4604	4742	3.0%	4742	5.5	3.0%	4753.2	35.4	3.2%	4742	122.3	3.0%	4743.2	73.7	3.0%
$C_{25,8}$	160	70	4540	4540	0.0%	4599	83.4	1.3%	4696.8	46.7	3.5%	4540	80.5	0.0%	4540.0	25.8	0.0%
$C_{40,1}$	140	50	6848	6918	1.0%	7189	53.8	5.0%	7233.2	91.8	5.6%	6920	173.7	1.1%	6980.8	87.7	1.9%
$C_{40,1}$	160	50	6682	6910	3.4%	7062	124.3	5.7%	7155.2	102.9	7.1%	6920	151	3.6%	6927.8	112.1	3.7%
$C_{40,1}$	140	70	6530	6815	4.4%	6858	107.5	5.0%	7005.0	126.7	7.3%	6813	146.4	4.3%	6813.0	109.2	4.3%
$C_{40,1}$	160	70	6426	6815	6.1%	6803	115.5	5.9%	6879.4	103.4	7.1%	6615	154.8	2.9%	6620.8	93.1	3.0%
$C_{40,2}$	140	50	6880	6930	0.7%	7195	188.6	4.6%	7317.6	86.5	6.4%	6956	28.1	1.1%	7113.8	89.0	3.4%
$C_{40,2}$	160	50	6708	6924	3.2%	7147	112.5	6.5%	7169.8	79.4	6.9%	6924	126.8	3.2%	6931.6	108.6	3.3%
$C_{40,2}$	140	70	6525	6818	4.5%	6935	78.3	6.3%	7025.8	70.3	7.7%	6813	178.1	4.4%	6814.6	127.8	4.4%
$C_{40,2}$	160	70	6433	6832	6.2%	6831	190.4	6.2%	6849.2	149.4	6.5%	6614	182	2.8%	6639.6	68.9	3.2%
$C_{40,3}$	140	60	8546	8711	1.9%	8939	13.4	4.6%	8960.2	75.6	4.8%	8715	33.1	2.0%	8855.0	91.8	3.6%
$C_{40,3}$	160	60	8490	8490	0.0%	8736	70.9	2.9%	8752.6	93.9	3.1%	8713	179	2.6%	8730.6	149.6	2.8%
$C_{40,3}$	140	80	8397	8397	0.0%	8648	168.3	3.0%	8755.4	103.6	4.3%	8411	11.7	0.2%	8547.6	43.0	1.8%
$C_{40,3}$	160	80	8392	8392	0.0%	8398	61.0	0.1%	8510.8	100.6	1.4%	8403	187.5	0.1%	8413.6	67.5	0.3%
$C_{40,4}$	140	60	6993	7250	3.7%	7551	147.1	8.0%	7648.6	124.9	9.4%	7239	196.3	3.5%	7244.8	147.7	3.6%
$C_{40,4}$	160	60	6831	7289	6.7%	7351	2.6	7.6%	7441.2	77.5	8.9%	7092	64	3.8%	7178.2	129.3	5.1%
$C_{40,4}$	140	80	6727	6969	3.6%	7244	111.8	7.7%	7263.8	117.9	8.0%	6959	56.5	3.4%	6970.8	107.6	3.6%
$C_{40,4}$	160	80	6621	7011	5.9%	7206	163.5	8.8%	7237.4	122.1	9.3%	6912	103.7	4.4%	6941.8	128.7	4.8%
$C_{40,5}$	140	60	6908	7243	4.8%	7384	141.4	6.9%	7444.2	80.1	7.8%	7207	50	4.3%	7210.0	109.9	4.4%
$C_{40,5}$	160	60	6786	7065	4.1%	7254	197.9	6.9%	7284.8	134.7	7.4%	6976	41.7	2.8%	6995.6	94.0	3.1%
$C_{40,5}$	140	80	6616	6973	5.4%	7156	80.9	8.2%	7172.0	113.1	8.4%	6877	124.2	3.9%	6888.2	72.5	4.1%
$C_{40,5}$	160	80	6552	6865	4.8%	6954	112.6	6.1%	7061.4	102.9	7.8%	6730	136.3	2.7%	6832.2	95.2	4.3%
$C_{40,6}$	140	50	7022	7218	2.8%	7411	101.7	5.5%	7458.8	107.6	6.2%	7246	114.3	3.2%	7261.6	92.6	3.4%
$C_{40,6}$	160	50	6831	7039	3.0%	7251	96.5	6.1%	7287.2	129.6	6.7%	7042	46.1	3.1%	7064.2	75.1	3.4%
$C_{40,6}$	140	70	6404*	–	–	7059	18.6	10.2%	7130.4	102.9	11.3%	6908	32.7	7.9%	6919.6	74.4	8.1%
$C_{40,6}$	160	70	6310*	–	–	6914	135.0	9.6%	6962.4	122.7	10.3%	6762	75.6	7.2%	6858.0	91.3	8.7%
$C_{40,7}$	140	50	7237	7316	1.1%	7592	104.9	4.9%	7600.6	84.8	5.0%	7333	104.3	1.3%	7351.4	139.4	1.6%
$C_{40,7}$	160	50	7076	7104	0.4%	7364	188.0	4.1%	7442.6	88.7	5.2%	7244	77	2.4%	7293.6	100.6	3.1%
$C_{40,7}$	140	70	6454*	–	–	7230	188.0	12.0%	7255.2	140.0	12.4%	6982	30.7	8.2%	7004.6	121.3	8.5%
$C_{40,7}$	160	70	6355*	–	–	6984	157.3	9.9%	7121.4	125.0	12.1%	6957	102.1	9.5%	6962.2	72.1	9.6%
$C_{40,8}$	140	50	6844	7025	2.6%	7254	99.0	6.0%	7324.8	103.5	7.0%	7223	80.4	5.5%	7236.0	137.5	5.7%
$C_{40,8}$	160	50	6768	7005	3.5%	7203	74.3	6.4%	7215.0	107.1	6.6%	7015	196.5	3.6%	7022.6	109.3	3.8%
$C_{40,8}$	140	70	6335*	–	–	6887	170.5	8.7%	6928.0	88.3	9.4%	6857	34.2	8.2%	6867.8	109.6	8.4%
$C_{40,8}$	160	70	6226*	–	–	6867	27.5	10.3%	6887.2	75.9	10.6%	6645	183	6.7%	6962.2	72.1	11.8%

Note that the only exact method in the literature for this variant of the LRP is the one proposed in [1]. We were not able to compare our solutions with the ones proposed in [1] as there are some incoherences in the reported results. This may be justified by the use of some unexplained parameter. Hence, we compare our results with the lower bounds provided by the previously described mixed integer programming models.

Table 2
Results for 50, 75 and 100 customers.

Inst	*W*	*Q*	3IM	*Multi-start VND*						*SGVNS*					
			lb	z^b	t^b	gap^b	z^{av}	t^{av}	gap^{av}	z^b	t^b	gap^b	z^{av}	t^{av}	gap^{av}
$C_{50,1}$	140	50	4715	5659	91.8	20.0%	5842.6	89.1	23.9%	5172	34.4	9.7%	5276.6	108.9	11.9%
$C_{50,1}$	160	50	4613	5439	135.8	17.9%	5621.8	126.3	21.9%	4890	118.6	6.0%	4981.2	97.6	8.0%
$C_{50,1}$	140	70	4581	5332	33.3	16.4%	5433.4	93.9	18.6%	5041	132.4	10.0%	5045.6	70.1	10.1%
$C_{50,1}$	160	70	4488	5325	131.4	18.6%	5364	160.0	19.5%	4818	40.8	7.4%	4818.8	82.9	7.4%
$C_{50,2}$	140	50	6546	7494	240.4	14.5%	7539.2	172.3	15.2%	7016	27.2	7.2%	7097.6	121.6	8.4%
$C_{50,2}$	160	50	6444	7295	215.3	13.2%	7371.8	136.6	14.4%	6991	70.3	8.5%	7003	112	8.7%
$C_{50,2}$	140	70	6303	7031	113.4	11.6%	7082.4	153.2	12.4%	6759	137	7.2%	6796.2	148.9	7.8%
$C_{50,2}$	160	70	6217	6877	32.4	10.6%	6894.6	87.4	10.9%	6593	33.3	6.0%	6618	104.6	6.5%
$C_{50,3}$	140	60	4914	5683	148.1	15.6%	5784.8	156.6	17.7%	5151	185.5	4.8%	5151.8	132.6	4.8%
$C_{50,3}$	160	60	4801	5405	115.0	12.6%	5536.8	146.0	15.3%	5138	47.3	7.0%	5142.8	80.7	7.1%
$C_{50,3}$	140	80	4713	5272	236.4	11.9%	5303	128.1	12.5%	5023	84.6	6.6%	5024.6	111.2	6.6%
$C_{50,3}$	160	80	4621	5043	15.5	9.1%	5159.2	81.0	11.6%	4798	201	3.8%	4802.8	134.7	3.9%
$C_{50,4}$	140	60	4751	5968	15.6	25.6%	5992.8	157.2	26.1%	5479	144.5	15.3%	5483	117	15.4%
$C_{50,4}$	160	60	4650	5740	56.9	23.4%	5837.8	105.6	25.5%	5237	232.1	12.6%	5336.4	140.6	14.8%
$C_{50,4}$	140	80	4511	5546	89.0	22.9%	5601.2	116.6	24.2%	5114	208.7	13.4%	5114	164.6	13.4%
$C_{50,4}$	160	80	4433	5412	50.1	22.1%	5450.2	103.6	22.9%	4877	188.1	10.0%	5012.2	140.1	13.1%
$C_{75,1}$	140	50	5358	7132	74.9	33.1%	7173.4	141.6	33.9%	6141	272.7	14.6%	6145.4	214.9	14.7%
$C_{75,1}$	160	50	5215	6694	291.3	28.4%	6855.6	221.7	31.5%	5912	242	13.4%	5921.4	217.9	13.5%
$C_{75,1}$	140	70	5000	6007	225.4	20.1%	6128.4	210.2	22.6%	5438	101.8	8.8%	5443	208.5	8.9%
$C_{75,1}$	160	70	4876	5741	26.3	17.7%	6000	127.9	23.1%	5236	181.9	7.4%	5286.6	155.4	8.4%
$C_{75,2}$	140	50	5298	7129	75.2	34.6%	7214.8	206.6	36.2%	6181	122.3	16.7%	6195	171	16.9%
$C_{75,2}$	160	50	5158	6960	289.6	34.9%	7014.6	149.0	36.0%	5946	8	15.3%	5956.4	190.1	15.5%
$C_{75,2}$	140	70	4936	6207	279.1	25.7%	6236.2	165.7	26.3%	5463	274	10.7%	5470.6	230.1	10.8%
$C_{75,2}$	160	70	4816	5976	313.0	24.1%	6093.6	231.3	26.5%	5244	251.8	8.9%	5339.4	198.7	10.9%
$C_{75,3}$	140	60	5270	6520	65.6	23.7%	6588	130.7	25.0%	5788	325	9.8%	5792.8	249.2	9.9%
$C_{75,3}$	160	60	5128	6199	120.5	20.9%	6363	268.1	24.1%	5568	207	8.6%	5622.8	230.8	9.6%
$C_{75,3}$	140	80	5002	5656	240.4	13.1%	5722.6	259.4	14.4%	5183	179.1	3.6%	5355.8	116	7.1%
$C_{75,3}$	160	80	4878	5656	208.4	15.9%	5690.6	214.7	16.7%	5171	58.4	6.0%	5178	227.9	6.2%
$C_{75,4}$	140	60	5135	6619	180.6	28.9%	6707.4	222.3	30.6%	5888	273.4	14.7%	6022	263.1	17.3%
$C_{75,4}$	160	60	5001	6390	35.3	27.8%	6536.6	162.3	30.7%	5874	276.2	17.5%	5879.6	250.1	17.6%
$C_{75,4}$	140	80	4857	5932	193.6	22.1%	5944.8	152.1	22.4%	5486	277	13.0%	5493.8	137.7	13.1%
$C_{75,4}$	160	80	4743	5796	22.0	22.2%	5881.2	125.1	24.0%	5256	195.1	10.8%	5427.2	165.4	14.4%
$C_{100,1}$	140	50	5433	7370	453.0	35.7%	7485.6	335.3	37.8%	6178	91.3	13.7%	6268.6	225.1	15.4%
$C_{100,1}$	160	50	5276	7313	16.0	38.6%	7370	206.5	39.7%	5925	240.4	12.3%	6042	396.9	14.5%
$C_{100,1}$	140	70	5147	6233	315.0	21.1%	6274.4	280.7	21.9%	5524	146.5	7.3%	5709	331.5	10.9%
$C_{100,1}$	160	70	5009	6125	45.4	22.3%	6175.6	247.2	23.3%	5526	110.6	10.3%	5543.2	246.3	10.7%
$C_{100,2}$	140	50	5466	7340	317.0	34.3%	7445.8	240.6	36.2%	6121	118.2	12.0%	6139.8	322.8	12.3%
$C_{100,2}$	160	50	5306	7032	11.0	32.5%	7177	152.1	35.3%	5867	145.5	10.6%	5918.2	260.7	11.5%
$C_{100,2}$	140	70	5152	6208	425.8	20.5%	6251.2	272.0	21.3%	5468	138.9	6.1%	5488.6	318.3	6.5%
$C_{100,2}$	160	70	5013	6072	13.6	21.1%	6166.4	302.9	23.0%	5453	197.1	8.8%	5480.2	280.3	9.3%
$C_{100,3}$	140	60	5325	7385	146.8	38.7%	7472.8	154.5	40.3%	6154	364.5	15.6%	6176.8	229.7	16.0%
$C_{100,3}$	160	60	5179	7165	242.3	38.3%	7356.6	322.0	42.0%	5916	108.8	14.2%	5980.6	308.7	15.5%
$C_{100,3}$	140	80	4955	6259	346.8	26.3%	6279	300.9	26.7%	5462	464.4	10.2%	5532	316.9	11.6%
$C_{100,3}$	160	80	4832	6058	2.0	25.4%	6144.2	112.3	27.2%	5461	467.3	13.0%	5490	382	13.6%
$C_{100,4}$	140	60	5339	7328	254.7	37.3%	7409	220.1	38.8%	6165	455.7	15.5%	6184.8	314.8	15.8%
$C_{100,4}$	160	60	5195	7193	214.3	38.5%	7279.4	222.6	40.1%	5923	144	14.0%	5982	168.9	15.1%
$C_{100,4}$	140	80	4964	6263	94.3	26.2%	6287.6	156.8	26.7%	5537	487.4	11.5%	5668	359.4	14.2%
$C_{100,4}$	160	80	4840	6155	226.0	27.2%	6174.6	169.1	27.6%	5517	386.7	14.0%	5529.4	354.5	14.2%

Table 3
Average results.

#cust	*#inst*	*NFM*	*Multi-start VND*				*VNS*			
		gap	t^b	gap^b	t^{av}	gap^{av}	t^b	gap^b	t^{av}	gap^{av}
25	32	1.9%	63.1	3.0%	57.8	3.5%	27.5	1.8%	27.7	1.9%
40	32	3.2%	112.6	6.6%	104.2	7.4%	106.3	3.9%	100.9	4.5%
50	16	–	107.5	16.6%	125.8	18.3%	117.9	8.5%	116.8	9.2%
75	16	–	165.1	24.6%	186.8	26.5%	202.9	11.2%	201.7	12.2%
100	16	–	195.2	30.2%	231.0	31.7%	254.2	11.8%	301.1	13.0%
Total	112	–	117.3	13.0%	124.6	14.1%	120.7	6.2%	126.1	6.8%

The network flow model is stronger than the three-index commodity flow model [34], as it considers implicitly the constraints regarding the feasibility of routes, which are previously generated in a pre-processing phase and correspond to variables of the model. This, on the other hand, makes it more difficult to run the model for larger instances. Thus, we compare instances with 25 and 40 customers with the lower bounds obtained with the network flow model (13)–(20) (Table 1) and the remaining ones with the three-index commodity flow model (1)–(12) (Table 2). The lower bounds for instances marked with an (⁎) are also obtained with model (1)–(12). A complete comparison between these two models can be found in [34].

The first three columns of Tables 1 and 2 report the instance name, *Inst*, and the considered values of *W* and *Q*. Columns *lb* and *ub* of Table 1 correspond to the values of the best lower and upper bound obtained with the flow model. The third column of the table represents the corresponding gap ($gap = (ub - lb)/lb$). Similarly, column *lb* of Table 2 corresponds to the values of the best lower bound obtained with the three-index commodity flow model. Both models were run in CPLEX 12.5 for 3600 s.

Table 4
Location-routing instances.

Instances			*BKS*	*SGVNS*			
name	*n*	*m*		z^b	z^{av}	gap^b	gap^{av}
coordChrist50	50	5	565.6	565.6	565.6	0.0%	0.0%
coordChrist75	75	10	848.0	848.8	853.4	0.1%	0.6%
coordChrist100	100	10	833.4	833.5	841.1	0.0%	0.9%
coordDas88	88	8	355.8	355.8	358.4	0.0%	0.7%
coordDas150	150	10	44,011.7	44,135.5	44,392.7	0.3%	0.9%
coordGaspelle	21	5	424.9	424.9	424.9	0.0%	0.0%
coordGaspelle2	22	5	585.1	585.1	585.1	0.0%	0.0%
coordGaspelle3	29	5	512.1	512.1	512.1	0.0%	0.0%
coordGaspelle4	32	5	562.2	562.2	562.2	0.0%	0.0%
coordGaspelle5	32	5	504.3	504.3	504.3	0.0%	0.0%
coordGaspelle6	36	5	460.4	460.4	460.4	0.0%	0.0%
coordMin27	27	5	3062.0	306.0	3062.0	0.0%	0.0%
coordMin134	134	8	5709.0	5726.1	5815.7	0.3%	1.9%

The results obtained with the multi-start VND method and our SGVNS algorithm are reported in the following columns of Tables 1 and 2.

For every instance $C_{a,b}$, a stands for the number of customers. Columns z^b and z^{av} represent the best and average solution obtained by the corresponding method, in five runs. The CPU time for finding z^b (t^b) and the average time for finding the final solutions (t^{av}) are also reported. Columns gap^b and gap^{av} represent the gap between z^b and z^{av} and the best corresponding lower bound ($gap = (z - lb)/lb$).

For 23% of the instances with 25 and 40 customers, the network flow model finds the optimal solution within the time limit. The VNS algorithm finds the optimal value for 67% of those instances, while the multi-start VND method does not find it for any of them.

In what concerns the gaps and the computational times, Table 3 reports the average values presented in Tables 1 and 2. The first five rows report the average values for each corresponding group of instances of 25, 40, 50, 75 and 100 customers and the last row represents the total average values. It is important to note that the reported gaps are in fact upper bounds for the value of the real optimality gaps, since the integer programming models that we use provide only a lower bound for the value of the optimal integer solution. Note also that, for the larger instances (50, 75 and 100 customers), this upper bound is obtained with a weaker model, and this can explain the increase of the SGVNS gap for these instances.

The average gap obtained by the multi-start VND method is around 2.1 times greater than the one obtained with the SGVNS, in a similar average computational time. Note that the ratios between these gaps tend to increase with the size of the instances. For the best gaps, they are around 1.7, 1.7, 2.0, 2.2 and 2.6 for instances of 25, 40, 50, 75 and 100 customers, respectively, and for the average gaps they are around 1.8, 1.6, 2.0, 2.2 and 2.4.

Table 4 reports the results obtained with the SGVNS for the location routing instances proposed in [6]. The first three columns report the instance name and the number of customers (n) and depots (m). In column *BKS*, the best known results from the literature are presented. These correspond to the best solutions among the metaheuristics proposed in [44,18,51,16]. The remainder of the columns report the results obtained with the SGVNS, namely the best (z^b) and average (z^{av}) solutions obtained in the five runs of the algorithm and the corresponding gaps (gap^b, gap^{av}), with $gap^{b/av} = (z^{b/av} - BKS)/BKS$. The SGVNS obtained the best known result for the majority of the instances with average gap^b and gap^{av} equal to 0.1% and 0.4%, respectively.

5. Conclusion

Location routing is an integrated problem that combines two difficult subproblems, the vehicle routing problem and the location problem. We explore a variant of this problem in which vehicles can perform several consecutive routes during one planning period. We propose a skewed general VNS algorithm for the location routing scheduling problem and compare our results with lower bounds obtained with a three-index commodity flow model from the literature, and a network flow model, strengthened with some valid inequalities (upper bounds of the optimality gaps). Our method proved to be efficient, clearly outperforming a multi-start VND approach. It was able to provide good results in short amounts of time, providing the optimal solution for the majority of the instances where the network flow model was able to find them. It also performed well when tested with location routing instances from the literature, even though it was developed for dealing and taking advantage of the multi-trip aspect of the location routing scheduling problem.

Acknowledgements

This work was supported by the Centre National de la Recherche Scientifique (CNRS), by the International Campus on Safety and Intermodality in Transportation (CISIT), by the Laboratoire d'Automatique, de Mécanique et d'Informatique industrielles et Humaines (LAMIH) of the Université de Valenciennes et du Hainaut-Cambrésis, by FEDER funding through the Programa Operacional Factores de Competitividade - COMPETE and by National funding through the Portuguese Science and Technology Foundation (FCT) in the scope of the project PTDC/EGE-GES/116676/2010 (Ref. COMPETE: FCOMP-01-0124-FEDER-020430) and through the grant SFRH/BPD/78024/2011. The work of the fifth author was conducted at National Research University Higher School of Economics and supported by RSF Grant 14-41-00039.

References

[1] Akca Z, Berger R, Ralphs T. Modeling and solving location routing and scheduling problems. Technical report; 2008.
[2] Akca Z, Berger R, Ralphs T. A branch-and-price algorithm for combined location and routing problems under capacity restrictions. In: Chinneck J, Kristjansson B, Saltzman M, Sharda R, Voß S, editors. Operations research and cyber-infrastructure, operations research/computer science interfaces series, vol. 47. US: Springer; 2009. p. 309–30.
[3] Albareda-Sambola M, Díaz J, Fernández E. A compact model and tight bounds for a combined location-routing problem. Comput Oper Res 2005;32 (3):407–28.
[4] Azi N, Gendreau M, Potvin J-Y. An exact algorithm for a single-vehicle routing problem with time windows and multiple routes. Eur J Oper Res 2007;178 (3):755–66.
[5] Azi N, Gendreau M, Potvin J-Y. An exact algorithm for a vehicle routing problem with time windows and multiple use of vehicles. Eur J Oper Res 2010;202(3):756–63.
[6] Barreto S. Análise e modelização de problemas de localização-distribuição [Ph. D. thesis]. Universidade de Aveiro; 2004.
[7] Barreto S, Ferreira C, Paixão J, Santos B. Using clustering analysis in a capacitated location-routing problem. Eur J Oper Res 2007;179(3):968–77.
[8] Belenguer J-M, Benavent E, Prins C, Prodhon C, Calvo R. A branch-and-cut method for the capacitated location-routing problem. Comput Oper Res 2011;38(6):931–41.
[9] Berger R, Coullard C, Daskin M. Location-routing problems with distance constraints. Transp Sci 2007;41(1):29–43.
[10] Cattaruzza D. Vehicle routing for city logistics [Ph.D. thesis]. Ecole Nationale Supérieure des Mines de Saint-Etienne; 2014.
[11] Clautiaux F, Alves C, Valério de Carvalho J. A survey of dual-feasible and superadditive functions. Ann Oper Res 2010;179:317–42.
[12] Contardo C, Cordeau J-F, Gendron B. A branch-and-cut-and-price algorithm for the capacitated location-routing problem. Technical report CIRRELT-2011-44. Université de Montréal; 2011.
[13] Contardo C, Cordeau J-F, Gendron B. A GRASP + ILP-based metaheuristic for the capacitated location-routing problem. Technical report CIRRELT- 2011-52. Université de Montréal; 2011.

[14] Cordeau J-F, Gendreau M, Laporte G. A tabu search heuristic for periodic and multi-depot vehicle routing problems. Networks 2007;30:105–19.
[15] Şen A, Bülbül K. A survey on multiple vehicle routing problem. In: VI International logistics and supply chain congress' 2008, Istanbul, Turkey; 2008. p. 6–7.
[16] Derbel H, Jarboui B, Chabchoub H, Hanafi S, Mladenovic N. A variable neighborhood search for the capacitated location-routing problem. In: 2011 4th international conference on logistics (LOGISTIQUA); May 2011. p. 514–9.
[17] Derbel H, Jarboui B, Hanafi S, Chabchoub H. Genetic algorithm with iterated local search for solving a location-routing problem. Expert Syst Appl 2012;39 (3):2865–71.
[18] Duhamel C, Lacomme P, Prins C, Prodhon C. A Graspels approach for the capacitated location-routing problem. Comput Oper Res 2010;37 (11):1912–23; metaheuristics for Logistics and Vehicle Routing.
[19] Hansen P, Mladenović N. Variable neighborhood search: principles and applications. Eur J Oper Res 2001;130(3):449–67.
[20] Hansen P, Mladenović N, Moreno-Pérez J. Variable neighbourhood search: methods and applications. 4OR A Quarterly Journal of Operations Research 2008;6(4):319–60.
[21] Hernandez F, Feillet D, Giroudeau R, Naud O. A new exact algorithm to solve the multi-trip vehicle routing problem with time windows and limited duration. 4OR 2014;12(3):235–59.
[22] Iori M, Salazar-Gonzalez J-J, Vigo D. An exact approach for the vehicle routing problem with two-dimensional loading constraints. Transp Sci 2007;41 (2):253–64.
[23] Jarboui B, Derbel H, Hanafi S, Mladenović N. Variable neighborhood search for location routing. Comput Oper Res 2013;40(1):47–57.
[24] Lancia G, Rinaldi F, Serafini P. A time-indexed lp-based approach for min-sum job-shop problems. Annals OR 2011:175–98.
[25] Laporte G. Location-routing problems. In: Golden B, Assad A, editors. Vehicle routing: methods and studies, studies in management science and systems, vol. 16. North-Holland, Amsterdam: Elsevier Science Publishers B.V.; 1988.
[26] Laporte G, Nobert Y, Arpin D. An exact algorithm for solving a capacitated location-routing problem. Ann Oper Res 1986;6(9):291–310.
[27] Laporte G, Nobert Y, Taillefer S. Solving a family of multi-depot vehicle routing and location-routing problems. Transp Sci 1988;22(3):161–72.
[28] Lin C, Chow C, Chen C. A location-routing-loading problem for bill delivery services. Comput Ind Eng 2002;43:5–25.
[29] Lin C, Kwok CR. Multi-objective metaheuristics for a location-routing problem with multiple use of vehicles on real data and simulated data. Eur J Oper Res 2006;175:1833–49.
[30] Macedo R, Alves C, Valério de Carvalho J. Arc-flow model for the two-dimensional guillotine cutting stock problem. Comput Oper Res 2010;37 (6):991–1001.
[31] Macedo R, Alves C, Valério de Carvalho J, Clautiaux F, Hanafi S. Solving the vehicle routing problem with time windows and multiple routes exactly using a pseudo-polynomial model. Eur J Oper Res 2011;214(3):536–45.
[32] Macedo R, Hanafi S, Clautiaux F, Alves C, Valério de Carvalho J. Generalized disaggregation algorithm for the vehicle routing problem with time windows and multiple routes. In: ICORES 2012, 1st international conference on operations research and entreprise systems, Vilamora, Portugal, 4–6 February, 2012. p. 305–12, best paper award.
[33] Macedo R, Hanafi S, Jarboui B, Mladenović N, Alves C, Valério de Carvalho J. Variable neighborhood search for the location routing problem with multiple routes. In: Proceedings of the international conference on industrial engineering and systems management (IESM 2013); 2013.
[34] Macedo R, Ramos B, Alves C, Valério de Carvalho J, Hanafi S, Mladenović N. Integer programming based approaches for multi-trip location routing. In: The CMS2014 special volume of lecture notes in economics and mathematical systems, 11th international conference in computational management science. Springer; 2014 (accepted for publication).
[35] Min H, Jayaraman V, Srivastava R. Combined location-routing problems: a synthesis and future research directions. Eur J Oper Res 1998;108(1):1–15.
[36] Mladenović N, Hansen P. Variable neighborhood search. Comput Oper Res 1997;24:1097–100.
[37] Mladenović N, Todosijević R, Urošević D. An efficient general variable neighborhood search for large travelling salesman problem with time windows. Yugosl J Oper Res 2013;23(1):19–30.
[38] Mladenović N, Urošević D, Hanfi S, Ilić A. A general variable neighborhood search for the one-commodity pickup-and-delivery travelling salesman problem. Eur J Oper Res 2012;220(1):270–85.
[39] Nagy G, Salhi S. Nested heuristic methods for the location-routeing problem. J Oper Res Soc 1996;47(9):1166–74.
[40] Nagy G, Salhi S. Location-routing: issues, models and methods. Eur J Oper Res 2007;177(2):649–72.
[41] Oliveira A, Vieira O. Adaptive memory programming for the vehicle routing problem with multiple trips. Comput Oper Res 2007;34(1):28–47.
[42] Pessoa A, Uchoa E, de Aragão M, Rodrigues R. Exact algorithm over an arc-time-indexed formulation for parallel machine scheduling problems. Math Program Comput 2010:1–32.
[43] Petch R, Salhi S. A multi-phase constructive heuristic for the vehicle routing problem with multiple trips. Discrete Appl Math 2003;133:69–92.
[44] Prins C, Prodhon C, Calvo R. A memetic algorithm with population management (ma-pm) for the capacitated location-routing problem. In: Gottlieb J, Raidl G, editors. Evolutionary computation in combinatorial optimization, lecture notes in computer science, vol. 3906. Berlin, Heidelberg: Springer; 2006. p. 183–94.
[45] Salhi S, Petch RJ. A ga based heuristic for the vehicle routing problem with multiple trips. J Math Model Algorithms 2007;6:591–613.
[46] Taillard E, Laporte G, Gendreau M. Vehicle routeing with multiple use of vehicles. J Oper Res Soc 1996;47(8):1065–70.
[47] Tuzun D, Burke L. A two-phase tabu search approach to the location routing problem. Eur J Oper Res 1999;116(1):87–99.
[48] Valério de Carvalho J. Exact solution of bin-packing problems using column generation and branch-and-bound. Ann Oper Res 1999;86:629–59.
[49] Valério de Carvalho J. LP models for bin packing and cutting stock problems. Eur J Oper Res 2002;141(2):253–73.
[50] Wu T-H, Low C, Bai J-W. Heuristic solutions to multi-depot location-routing problems. Comput Oper Res 2002;29(10):1393–415.
[51] Yu VF, Lin S-W, Lee W, Ting C-J. A simulated annealing heuristic for the capacitated location routing problem. Comput Ind Eng 2010;58 (2):288–99; scheduling in Healthcare and Industrial Systems.