Computers & Operations Research 39 (2012) 3185–3199

Contents lists available at SciVerse ScienceDirect

Computers & Operations Research

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

Lower and upper bounds for the two-echelon capacitated location-routing problem

Claudio Contardo [a,b,*], Vera Hemmelmayr [a,c], Teodor Gabriel Crainic [a,b]

[a] Centre interuniversitaire de recherche sur les réseaux d'entreprise, la logistique et le transport (CIRRELT), C.P. 6128, succ. Centre-ville, Montréal (PQ), Canada H3C 3J7
[b] École des sciences de la gestion, UQAM, 315 Ste-Catherine Est, Montréal, Canada H2X 3X2
[c] Institute for Transport and Logistics Management, WU, Vienna University of Economics and Business Nordbergstraß 15/D/6/621 C, 1090 Vienna, Austria

ARTICLE INFO

Available online 12 April 2012

Keywords:
Two-echelon capacitated location routing problem
Adaptive large neighbourhood search
Branch-and-cut

ABSTRACT

In this paper, we introduce two algorithms to address the two-echelon capacitated location-routing problem (2E-CLRP). We introduce a branch-and-cut algorithm based on the solution of a new two-index vehicle-flow formulation, which is strengthened with several families of valid inequalities. We also propose an adaptive large-neighbourhood search (ALNS) meta-heuristic with the objective of finding good-quality solutions quickly. The computational results on a large set of instances from the literature show that the ALNS outperforms existing heuristics. Furthermore, the branch-and-cut method provides tight lower bounds and is able to solve small- and medium-size instances to optimality within reasonable computing times.

1. Introduction

The single-sourcing *two-echelon capacitated location-routing problem* (*2E-CLRP*) is an important combinatorial optimization problem arising in freight distribution. The problem can be stated as follows. Given three disjoint sets of nodes representing platforms (first-echelon facilities), satellites (second-level facilities), and customers, one must decide the location of a subset of platforms and a subset of satellites, as well as construct vehicle routes to visit each customer exactly once using a vehicle routed from an open satellite, which is also visited exactly once using a vehicle route from an open platform, at minimum total cost. The utilization of open facilities is limited by echelon-specific capacities. Vehicles belong to two homogeneous fleets, each operating at a particular echelon and with echelon-specific capacities. Fig. 1 illustrates the network layout of the 2E-CLRP.

The 2E-CLRP has been formally introduced by Boccia et al. [4], together with a classification of the different 2E-CLRP problem classes. The authors also proposed two-index and three-index vehicle-flow formulations, as well as a set-partitioning formulation. The third model is not included in their experimental analyses. The first two models contain a polynomial number of variables and constraints and are addressed using a general-purpose optimization solver. These models are shown to be effective for dealing with very small instances with up to 10 customers, 5 satellites and 3 platforms within reasonable computing times. For larger instances, the authors report average gaps above 25%. A tabu search procedure was proposed by Boccia et al. [3] based on the decomposition of the 2E-CLRP into four subproblems, a capacitated facility location problem (CFLP) and a multiple depot vehicle routing problem (MDVRP) addressing a single-echelon CLRP for each level.

Thus, to the best of our knowledge, only two articles address the 2E-CLRP [3,4] (a few others use the name but locate on one level only). No exact procedure has been introduced to effectively deal with medium- and large-size instances. Moreover, a single meta-heuristic has been proposed so far and its performance, in terms of solution quality, appears impaired by the extreme decomposition used. Our goal is to contribute filling these gaps by introducing an exact procedure to deal with medium- and large-size instances and a new meta-heuristic procedure to quickly find good 2E-CLRP solutions. The main contributions of this paper can be summarized as follows:

i. We propose exact and meta-heuristic solution methods based on the same principle of handling the 2E-CLRP as the superposition of two *Capacitated Location Routing Problems* (*CLRP*s), one at each echelon. This allows to better address the interdependencies between location and routing decisions and, thus, enhance the solution quality, as opposed to previous contributions. This also provides the means to address the 2E-CLRP through efficient algorithms for the CLRP applied to each echelon, their respective solutions being combined through simple and efficient rules.

* Corresponding author at: Centre interuniversitaire de recherche sur les réseaux d'entreprise, la logistique et le transport (CIRRELT), C.P. 6128, succ. Centre-ville, Montréal (PQ), Canada H3C 3J7. Tel.: +1 514 340 6053x6033; fax: +1 514 343 7121.
E-mail addresses: claudio.contardo@cirrelt.ca (C. Contardo), vera.hemmelmayr@wu.ac.at (V. Hemmelmayr), teodorgabriel.crainic@cirrelt.ca (T.G. Crainic).

0305-0548
http://dx.doi.org/10.1016/j.cor.2012.04.003

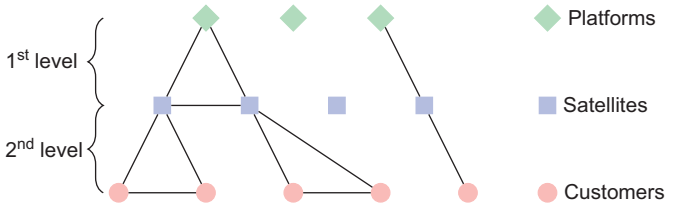

Fig. 1. 2E-CLRP network layout.

ii. We introduce a new two-index vehicle-flow formulation of the 2E-CLRP, which is shown to provide tight lower bounds.
iii. We develop an efficient branch-and-cut algorithm based on the new formulation, providing the means to solve medium-size instances with up to 50 customers and 10 satellites, and to compute tight gaps for larger instances.
iv. We introduce a new *Adaptive Large Neighbourhood Search* (*ALNS*) meta-heuristic solution method, which outperforms the previously proposed heuristic methods from the literature and yields tight upper bounds that lie, on average, no further than a 3.06% from the lower bounds obtained with the exact method.

The solution methods introduced in this paper are complementary in scope. While the exact algorithm provides insightful information about the structure of the 2E-CLRP and optimal solutions for small- to medium-size instances, the very high two-tiered combinatorial nature of the problem makes it difficult to scale for larger instances for which even finding a feasible solution of reasonable quality can be prohibitive. Alternatively, the proposed ALNS provides high-quality solutions quickly. Moreover, the methods validate each other through the lower bounds obtained with the branch-and-cut method and the upper bounds obtained with the meta-heuristic. The remainder of the paper is organized as follows. Section 2 briefly surveys literature on the 2E-CLRP and related problems. Section 3 defines the notation and provides a mathematical formulation of the 2E-CLRP. The branch-and-cut algorithm is described in Section 4 and the ALNS in Section 5. Computational results are presented and analyzed in Section 6. We conclude in Section 7.

2. Literature review

The Introduction reviewed the very limited literature directly targeting the 2E-CLRP. We turn to a brief overview of contributions to other logistic problems the 2E-CLRP generalizes.

The *capacitated location-routing problem* (*CLRP*) is the particular case where the location of a single platform of infinite capacity is known in advance and the costs on the first echelon can be neglected. The problem is to find the optimal satellite locations and to build vehicle routes around those satellites to satisfy the customer demand. Recent algorithms for solving the CLRP include exact methods [2,1,7,6] and heuristics [17–19,15,10,8]. Belenguer et al. [2] introduce a two-index vehicle-flow formulation, which they strengthen with several families of valid inequalities. Their computational experiments show that instances containing up to 50 customers and 5 facilities can be solved to optimality within reasonable computing times. Contardo et al. [7] introduced three new flow formulations from which they derived new valid inequalities. The authors also proposed new improved separation routines for the inequalities introduced in [2]. These refinements allow us to solve larger instances with up to 100 customers and 5 facilities. Baldacci et al. [1] introduced a set-partitioning formulation for the problem, strengthened with new families of valid inequalities. Their algorithm is able to solve to optimality instances containing up to 199 customers and 14 facilities. Contardo et al. [6] strengthened this set-partitioning formulation with new valid inequalities. Their method is able to solve all instances solved by Baldacci et al. [1] plus four more instances.

Prins et al. [19] developed a GRASP (greedy randomized adaptive search procedure) for the CLRP, and then a memetic algorithm with population management [18], which yielded better results. Their cooperative Lagrangian relaxation-granular tabu search method [17] outperformed the previous two. Pirkwieser and Raidl [15] introduced the first hybrid meta-heuristic for the CLRP combining meta-heuristics (in this case a VNS) and integer-linear programming techniques. Their method yielded improved results compared to previous methods. Hemmelmayr et al. [10] introduced an ALNS for the 2E-CVRP and also tested it on CLRP instances. Their computational experiments showed that, even though not initially developed for the CLRP, their method obtained very competitive results compared to the previous methods from the literature. Finally, Contardo et al. [8] introduced a hybrid meta-heuristic combining a GRASP with integer-programming methods based on column generation. Their method yielded very good results, obtaining the tightest gaps on several instances from the literature.

The *two-echelon capacitated vehicle routing problem* (*2E-CVRP*) is also a particular case of the 2E-CLRP, where the location of a single platform is known in advance and the satellites are uncapacitated with no setup costs. A fleet of trucks is routed from the depot to the satellites, and then from these satellites smaller trucks are used to deliver the commodities to the final customers. Several models and algorithms have been designed for the 2E-CVRP with multiple-sourcing at the first echelon, both exact [14,13] and heuristic [9,10]. Perboli et al. [14] designed a branch-and-cut algorithm based on a three-index formulation of the problem. A third index is added to the vehicle-flow variables at the second echelon to specify the satellite serving a node. The formulation is strengthened with subtour-elimination and some flow-conservation constraints. Their algorithm is able to solve to optimality instances containing up to 21 customers. Perboli and Tadei [13] strengthen the previous formulation with new cuts, including capacity cuts, which allows their algorithm to scale and solve instances with up to 50 customers. Crainic et al. [9] developed multi-start heuristics for the solution of the 2E-CVRP, where an intensification phase, aiming to improve feasible solutions by local search, is followed by a diversification phase to avoid local optima. Hemmelmayr et al. [10] developed an ALNS meta-heuristic for the 2E-CVRP and the CLRP which is shown to provide better solutions than previous approaches. The algorithm is based on the destroy-and-repair principle in which two sets of operators (destroy operators and repair operators) are alternated.

Nguyen et al. [12] introduced a GRASP complemented with a path relinking method to address a hybrid class of 2E-CVRPs with single-sourcing at both echelons, in which location decisions are restricted to satellites, the location of a single platform being known in advance. In their method, a GRASP is used to build a set of solutions $\mathcal{P}$. Then, for each pair of solutions $S,T \in \mathcal{P}$, a path-relinking method tries to build better solutions by applying different operators to S with the goal of obtaining T. A learning process is used to guide the GRASP by restricting the opening of satellites to those which seem more promising during the first iterations. In Nguyen et al. [11], the same authors provide a multi-start iterated local search with tabu list and path relinking. The new algorithm outperforms their previous approach and they also report results on the instances by Boccia et al. [3]. In the iterated local search, two search spaces are considered: 2E-CVRP solutions and a giant tour covering the main depot and the customers. A path-relinking procedure is presented that can be used for intensification or post-optimization.

3. Mathematical formulation

The 2E-CVRP is defined on a graph with three disjoint sets of nodes, $\mathcal{P}$ (the platforms), $\mathcal{S}$ (the satellites), and $\mathcal{C}$ (the customers). To each platform $p \in \mathcal{P}$, we associate a fixed cost H_p^1 and a capacity K_p^1. Similarly, to each satellite $s \in \mathcal{S}$, we associate a fixed cost H_s^2 and a capacity K_s^2. Finally, we associate a demand $d_c > 0$ to each customer $c \in \mathcal{C}$. We distinguish two echelons, one containing nodes in $\mathcal{P} \cup \mathcal{S}$ and the other containing nodes in $\mathcal{S} \cup \mathcal{C}$.

At the first echelon, we consider an undirected graph $G^1 = (V^1, E^1)$, with $V^1 = \mathcal{P} \cup \mathcal{S}$ and $E^1 = \{\{u,v\} : u,v \in V^1,\ u \text{ and } v \text{ not both in } \mathcal{P}\}$. Similarly, at the second echelon, we consider an undirected graph $G^2 = (V^2, E^2)$ with $V^2 = \mathcal{S} \cup \mathcal{C}$ and $E^2 = \{\{u,v\} : u,v \in V^2,\ u \text{ and } v \text{ not both in } \mathcal{S}\}$. Two fleets of vehicles are used, one at each echelon. At each echelon, the fleet is homogeneous with vehicle capacities of Q^1 and Q^2 for the first and second echelons, respectively. A routing cost $\gamma_e > 0$ is associated to edge $e \in E^1 \cup E^2$.

Recall that Boccia et al. [4] introduced a three-index vehicle-flow formulation with a cubic number of variables (in terms of the number of nodes). The formulation presents a very high number of symmetries, however, and provides weak lower bounds, making it of little use within a branch-and-bound framework. The bounds provided by the two-index formulation proposed in the same article proved even weaker. Small-size instances only were consequently solved to optimality using these formulations. We therefore introduce a new two-index vehicle-flow formulation for the 2E-CLRP, which is shown to provide tighter gaps in much shorter computational times for medium- to large-size instances, compared the previous formulations [4].

The main observation is that the 2E-CLRP can be decomposed into two different CLRPs, one at each echelon, connected through the satellite nodes. This observation is used in formulating the model and designing the exact and meta-heuristic algorithms. In particular, as detailed in the next section, the new formulation inherits most of the cuts valid for the CLRP [2,7].

Let w_p be a binary variable equal to 1 iff platform $p \in \mathcal{P}$ is selected for opening. Similarly, let z_s be a binary variable equal to 1 iff satellite $s \in \mathcal{S}$ is chosen for opening. For first-echelon edges, let u_e and v_e be the binary variables equal to 1 iff edge $e \in E^1$ is used exactly once and twice (for single-customer routes), respectively. Analogously for second-echelon edges, let x_e and y_e be the binary variables equal to 1 iff edge $e \in E^2$ is used once and twice, respectively. Finally, let $g_s \geq 0$ be the continuous variable capturing the volume of commodity shipped to/from satellite $s \in \mathcal{S}$.

Define the following vertex subsets: (1) $\delta^k(I) \subseteq E^k$, the edge subset at echelon $k \in \{1,2\}$ that contains one endpoint in I, for any vertex set $I \subseteq V^k$;(2) $(J{:}^k I)$, the edge subset of $E^k, k \in \{1,2\}$ with one endpoint in J and the other in I, for two disjoint vertex subsets $J \subseteq V^k$ and $I \subseteq V^k$;(3) $E^k(I)$, the subset of edges in $E^k, k \in \{1,2\}$ with both endpoints in $I \subseteq V^k$. Then, for a given edge set $F \subseteq E^1$, let $u(F) = \sum_{e \in F} u_e$ and $v(F) = \sum_{e \in F} v_e$;similarly for $F \subseteq E^2$, $x(F)$, and $y(F)$. Also, for a given satellite subset $\mathcal{S}' \subseteq \mathcal{S}$, let $g(\mathcal{S}') = \sum_{s \in \mathcal{S}'} g_s$ and $z(\mathcal{S}') = \sum_{s \in \mathcal{S}'} z_s$. Finally, for any customer set $U \subseteq \mathcal{C}$, let $d(U) = \sum_{c \in U} d_c$, $r(U) = \lceil d(U)/Q^2 \rceil$, and $\overline{U} = \mathcal{C} \backslash U$, while for any satellite subset $\mathcal{S}'$, $\overline{\mathcal{S}'} = \mathcal{S} \backslash \mathcal{S}'$. The quantity $r(U)$ represents a lower bound on the number of second-echelon vehicles needed to serve the customers in U. A valid formulation of the 2E-CLRP is the following:

$$\min \sum_{p \in \mathcal{P}} H_p^1 w_p + \sum_{s \in \mathcal{S}} H_s^2 z_s + \sum_{e \in E^1} \gamma_e u_e + 2 \sum_{e \in \delta^1(\mathcal{P})} \gamma_e v_e + \sum_{e \in E^2} \gamma_e x_e + 2 \sum_{e \in \delta^2(\mathcal{S})} \gamma_e y_e \tag{1}$$

$$u(\delta^1(s)) + 2v(\mathcal{P}{:}^1\{s\}) = 2z_s, \quad s \in \mathcal{S} \tag{2}$$

$$x(\delta^2(c)) + 2y(\mathcal{S}{:}^2\{c\}) = 2, \quad c \in \mathcal{C} \tag{3}$$

$$u(\delta^1(T)) + 2v(\mathcal{P}{:}^1 T) \geq 2\left(\frac{g(T)}{Q^1}\right), \quad T \subseteq \mathcal{S},\ |T| \geq 2 \tag{4}$$

$$x(\delta^2(U)) + 2y(\mathcal{S}{:}^2 U) \geq 2r(U), \quad U \subseteq \mathcal{C},\ |U| \geq 2 \tag{5}$$

$$u_{ps} + v_{ps} \leq w_p, \quad s \in \mathcal{S},\ p \in \mathcal{P} \tag{6}$$

$$x_{sc} + y_{sc} \leq z_s, \quad s \in \mathcal{S},\ c \in \mathcal{C} \tag{7}$$

$$u(\mathcal{P}{:}^1\{s\}) + v(\mathcal{P}{:}^1\{s\}) \leq z_s, \quad s \in \mathcal{S} \tag{8}$$

$$x(\mathcal{S}{:}^2\{c\}) + y(\mathcal{S}{:}^2\{c\}) \leq 1, \quad c \in \mathcal{C} \tag{9}$$

$$u((\mathcal{P}\backslash\{p\}) \cup \overline{T}{:}^1 T) + 2v(\mathcal{P}\backslash\{p\}{:}^1 T) \geq 2\left(\frac{g(T) - K_p^1}{Q^1}\right), \quad p \in \mathcal{P},\ T \subseteq \mathcal{S} \tag{10}$$

$$x((\mathcal{S}\backslash\{s\}) \cup \overline{U}{:}^2 U) + 2y(\mathcal{S}\backslash\{s\}{:}^2 U) \geq 2\left(\frac{d(U) - g_s}{Q^2}\right), \quad s \in \mathcal{S},\ U \subseteq \mathcal{C} \tag{11}$$

$$u(\delta^1(T)) \geq 2(u(\{p\}{:}^1 \mathcal{P}') + u(\{p'\}{:}^1 \mathcal{P} \backslash \mathcal{P}')), \quad \begin{matrix} T \subseteq \mathcal{S},\ |T| \geq 2 \\ p,p' \in T,\ \mathcal{P}' \subset \mathcal{P} \end{matrix} \tag{12}$$

$$x(\delta^2(U)) \geq 2(x(\{c\}{:}^2 \mathcal{S}') + x(\{c'\}{:}^2 \mathcal{S} \backslash \mathcal{S}')), \quad \begin{matrix} U \subseteq \mathcal{C},\ |U| \geq 2 \\ c,c' \in U,\ \mathcal{S}' \subset \mathcal{S} \end{matrix} \tag{13}$$

$$0 \leq g_s \leq K_s^2 z_s, \quad s \in \mathcal{S} \tag{14}$$

$$g(\mathcal{S}) = d(\mathcal{C}) \tag{15}$$

$$w_p \in \{0,1\}, \quad p \in \mathcal{P} \tag{16}$$

$$z_s \in \{0,1\}, \quad s \in \mathcal{S} \tag{17}$$

$$u_e \in \{0,1\}, \quad e \in E^1 \tag{18}$$

$$v_{ps} \in \{0,1\}, \quad p \in \mathcal{P},\ s \in \mathcal{S} \tag{19}$$

$$x_e \in \{0,1\}, \quad e \in E^2 \tag{20}$$

$$y_{sc} \in \{0,1\}, \quad s \in \mathcal{S},\ c \in \mathcal{C} \tag{21}$$

Constraints (2) and (3) are the degree constraints for the satellite nodes and the customer nodes at the first and second echelons, respectively. Constraints (4) and (5) are the capacity constraints at both echelons that ensure the connectivity for the tours and make sure that the vehicle capacities are respected. Constraints (6) and (7) ensure that edges incident to a platform (for the first level) or to a satellite (for the second level) may only be used when the corresponding facility is opened. Constraints (8) and (9) are the so-called path elimination constraints for single satellite or single customer routes, respectively. They forbid routes that start at one facility, visit one customer or satellite and go back to a different facility. Constraints (10) and (11) are the facility capacity inequalities for the first and second echelon, respectively. Constraints (12) and (13) are the path elimination constraints that forbid routes that start at one facility and end at another facility. These constraints are complementary to constraints (8) and (9). Constraints 14 ensure that flows going through satellites do not exceed their capacities. These constraints are complementary to constraints (10) to forbid routes from serving a demand higher than the platform capacities. Finally, Constraints (15) ensure that the total volume of commodity going through satellites coincides with the total demand.

This formulation includes $O(|E^1| + |E^2|)$ variables and an exponential number of constraints. Therefore, constraints must be added dynamically in a branch-and-cut fashion within the exact

solver. Note that the use of flow variables g at constraints (4), (10), and (11) links the use of the satellites at the two echelons. In the following section, we introduce a branch-and-cut method using formulation (1)–(21). We introduce separation algorithms to dynamically add cuts, as well as new valid inequalities to strengthen the formulation, thus providing tighter lower bounds.

4. Branch-and-cut method

We have developed a branch-and-cut algorithm based on the previously introduced formulation, which we strengthen with the use of several families of valid inequalities. The branch-and-cut algorithm is based on a relaxation of the original model in which integrality is dropped as well as some constraints that are dynamically included by the use of separation algorithms. The solution of the resulting linear problem provides a lower bound on the optimal solution of the 2E-CLRP. If the solution of such program is not integer, a branching decision splits the problem into two complementary subproblems, and the same procedure is applied to each of them in a recursive manner. If a node is proven to be infeasible or its associated lower bound is larger than the value of the incumbent solution, it is discarded.

Inspired by the success of the branch-and-cut methods for the CLRP by Belenguer et al. [2] and Contardo et al. [7], the algorithm introduced in this paper is based on the decomposition of the 2E-CLRP into two CLRPs, one for each echelon. Indeed, the key observation is that flow variables $(g_s)_{s\in\mathcal{S}}$ at the first echelon correspond to satellite demands, while at the second echelon they correspond to the satellite capacities. Hence, each echelon can be seen as a CLRP by giving variables $(g_s)_{s\in\mathcal{S}}$ the proper role. Therefore, we derive valid inequalities for the 2E-CLRP from those for the CLRP and make use of the separation algorithms described in the previously mentioned papers. In the following, we describe the valid inequalities used in this article, including some derived from the CLRP, plus some newly introduced. We also describe the separation algorithms used at each echelon, the node-selection strategy, the branching strategy, and the general separation strategy used through the search tree.

4.1. Valid inequalities

In this section, we introduce valid inequalities for the 2E-CLRP. They are subdivided into those specific for the first echelon and those specific for the second echelon. We introduce some new families of valid inequalities, and also derive valid inequalities from the two-index vehicle-flow formulation of the CLRP, which have been introduced by Belenguer et al. [2] and Contardo et al. [7]. They include lifted cover inequalities (LCIs), co-circuit constraints (CoCCs), flow-assignment inequalities (FAIs), y-capacity cuts (y-CCs), strengthened facility capacity inequalities (SFCIs), strengthened effective facility capacity inequalities (SEFCIs), location-routing comb inequalities (LRCIs), y-generalized large multistar inequalities (y-GLMs), strengthened comb inequalities (SCIs), and framed capacity inequalities (FrCIs). For details on the inequalities as well as the separation algorithms used to identify violated inequalities, we refer to [2,7].

4.1.1. First-echelon inequalities

Note that variables g on this echelon may be seen as the actual satellite demands. Therefore, all the inequalities valid for the CLRP are also valid for the first echelon of the 2E-CLRP when taking these quantities as demands. However, many of them become non-linear, such as those containing expressions combining the rounding operator $\lceil\cdot\rceil$ with the flow variables g, or those containing products of the demands g with vehicle-flow variables u, v (e.g., several types of multistar inequalities) and thus cannot be introduced into the problem without losing linearity. We have then restricted the inclusion of valid inequalities at the first echelon to LCI, CoCC, and FAI, plus the two following families of valid inequalities:

$$u((\mathcal{P}\setminus\mathcal{P}')\cup\overline{\mathcal{S}}':^1\mathcal{S}')+2v(\mathcal{P}\setminus\mathcal{P}':^1\mathcal{S}')\geq 2\left(\frac{g(\mathcal{S}')-\sum_{p\in\mathcal{P}'}K_p^1 w_p}{Q^1}\right),\quad \mathcal{S}'\subseteq\mathcal{S},\ \mathcal{P}'\subset\mathcal{P} \tag{22}$$

$$u_{st}+v(\mathcal{P}:s)\leq z_s,\quad s,t\in\mathcal{S} \tag{23}$$

We call the first lifted facility capacity inequalities (LFCIs). They are a generalization of constraints (10) and can safely replace them. The second ones are called flow-location inequalities (FLIs). They forbid a node to be linked at the same time to another satellite and to a platform using a single-satellite edge. Fig. 2 illustrates how such a constraint may be violated. In the figure, the left-hand side of constraint (23) is equal to 1.25, which is greater than 1, the maximum allowable value for the right-hand side.

4.1.2. Second-echelon inequalities

Satellite capacities on the second echelon are given by the flow variables g. One still has, however, that no more than K_s^2 units of flow can be delivered from any given satellite $s\in\mathcal{S}$. Therefore, all the inequalities valid for the two-index vehicle-flow formulation of the CLRP are valid on the second echelon (the ones restricted to the variables z, x and y). Note that when replacing K_s^2 by g_s in those inequalities, many of them become non-linear and thus cannot be added without losing the linearity of the problem. We then make use of the following inequalities: y-CC, SFCI, SEFCI, LRCI, y-GLM, CoCC, LCI, SCI, FAI, and FrCI.

Additionally, we add the following family of location-routing generalized large multistar inequalities (LRGLMs). Let $U\subset\mathcal{C}$ be a customer subset, $\mathcal{S}'\subseteq\mathcal{S}$ be a facility subset and $c\in\mathcal{C}\setminus U$ be a customer not in U. Let us define $\eta(\mathcal{S}',U,c)=\frac{1}{2}x(\mathcal{S}':\{c\})+y(\mathcal{S}':\{c\})+x(U:\{c\})$. The following inequality is valid for the 2E-CLRP:

$$x((\mathcal{S}\setminus\mathcal{S}')\cup\overline{U}:^2U)+2y(\mathcal{S}\setminus\mathcal{S}':^2U)\geq\frac{2}{Q^2}\left(d(U)-g(\mathcal{S}')+\sum_{c\notin U}d_c\eta(\mathcal{S}',U,c)\right) \tag{24}$$

The validity of these inequalities can be derived from the validity proof of the LRGLM introduced in Contardo et al. [7], by replacing the satellite capacities $(K_s^2)_{s\in\mathcal{S}}$ by the tighter capacities $(g_s)_{s\in\mathcal{S}}$. The dominance with respect to the original LRGLM comes from the inclusion of constraints (14).

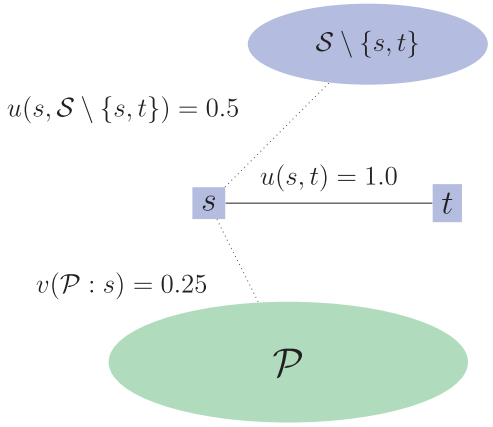

Fig. 2. Violation of a FLI.

4.2. Separation algorithms

For the separation of the inequalities directly translated from the CLRP, we make use of the separation algorithms introduced in Belenguer et al. [2] and Contardo et al. [7]. Note that for a given satellite subset $\mathcal{S}' \subseteq \mathcal{S}$ the degree constraints (2) in the first echelon imply the following identity:

$$u(\delta^1(\mathcal{S}')) + 2v(\mathcal{P}:{}^1\mathcal{S}') + 2u(E^1(\mathcal{S}')) = 2z(\mathcal{S}') \quad (25)$$

which differs from the classical identity of other vehicle routing problems in that the right-hand side of the expression above is replaced by $2|\mathcal{S}'|$. As a consequence, separation algorithms must be adapted to make use of the right expression when necessary.

For the separation of the LFCI (22), we also make use of the separation algorithms for the SFCI, by setting the satellite demands to $(g_s)_{s \in \mathcal{S}}$. The FLI (23) are inspected for violation one by one at each iteration, as they are polynomial in number. Finally, for the LRGLM (24), we use the separation algorithms for the LRGLM for the CLRP as described in Contardo et al. [7], by replacing the actual satellite capacities $(z_s K_s^2)_{s \in \mathcal{S}}$ used in the original inequalities by the stronger ones $(g_s)_{s \in \mathcal{S}}$.

4.3. Node selection strategy

As our objective is to obtain the tightest possible lower bound for the problem, we use a best-bound strategy. Thus, after the exploration of the current node and the creation of the two children subproblems, the next node to explore is the uninspected node with the smallest lower bound.

4.4. Branching strategy

The branching strategy is a hybrid mixing branching on variables and on cutsets. For the branching on cutsets, we add additional slack variables to the problem and the y-capacity cuts on the second echelon are added as identities. We then branch on these slack variables, similar to Belenguer et al. [2] and Contardo et al. [7]. The branching is performed in the following order: (1) location variables w, (2) location variables z, (3) cutsets on the second echelon, (4) vehicle-flow variables u, v and (5) vehicle-flow variables x, y.

4.5. Separation strategy

We distinguish between first and second echelon inequalities. We decided following preliminary computational experiments, to separate inequalities on the second echelon in the first place. Then, when we are no longer able to identify any violated inequality on the second echelon, we run the separation algorithms on the first echelon.

We implemented a dynamic separation strategy as explained in Contardo et al. [7]. We differentiate between the cuts that are needed to impose feasibility of integer solutions and those which can be seen as cuts to strengthen the problem. For each of the latter, we consider a counter representing the number of times the corresponding family of cuts has been successfully separated and added to the problem. We keep track of this counter during the branching tree. At certain depths (as suggested by Contardo et al. [7] for the CLRP, we first check at depth 10, and then at multiples of 5), we deactivate from a branch (and thus from all its children) the cuts for which the counter is zero, and for the remaining ones we reset the counter to zero. Using this strategy, we rapidly deactivate cuts that do not seem promising during certain branches of the tree, but we keep them where they seem to be useful. As a matter of fact, we have observed that cuts are rarely added after depth 25 in the branching tree.

5. Adaptive large neighbourhood search

ALNS was proposed by Ropke and Pisinger [20] for the pickup and delivery problem. It extends the large neighbourhood search algorithm of Shaw [22] and is also inspired by the ruin-and-recreate principle [21]. A general ALNS for several classes of routing problems was developed in Pisinger and Ropke [16], where the authors show that ALNS can outperform existing solution methods. In ALNS, there are two types of operators. Destroy operators that remove a certain number of elements from a solution and repair operators that reinsert these elements in the partial solution. ALNS works with an adaptive weight adjustment, where each operator is chosen based on its past success.

The ALNS we propose for the 2E-CLRP is based on the same decomposition principle detailed in the branch-and-cut section. An ALNS is thus called recursively to solve the CLRP on each echelon. Similar to the branch-and-cut, we initiate with the second echelon. Then, whenever the second echelon is solved and the new solution leads to a change in the satellite configuration, the first echelon is solved again by calling the ALNS procedure.

The ALNS called at each echelon is based on a previously proposed solution method for the 2E-CVRP and the CLRP [10], with a different strategy to determine the solution serving as the new incumbent. This strategy is detailed in Section 5.3, when we refer to the large destroy operators. Parameters were also modified to achieve best performance for the new problem as described in Section 6.2.

The same destroy and repair operators are used to address the CLRP on either the first or the second level. In the following, in order not to overload the presentation, we describe the operators in terms of the second echelon only. When the operators are used on the first level, the platforms correspond to satellites and the satellites correspond to customers.

We introduce a two-stage algorithm that deals with two types of destroy operators. There are large-impact, *large*, operators, D_L, that change the current configuration of opened satellites and small-impact, *small*, operators, D_S, that remove only a certain number of customers, but do not explicitly open or close satellites. The operators from set D_L are called every ω iterations without improvements. For these operators, the algorithm starts from the best found solution. Only if the new solution is within $\eta\%$ of the best found solution, it can serve as the new incumbent with a probability of Θ. Moreover, a solution yielded by these operators is accepted, even if it is not improving. The goal of this procedure is to provide the means for the algorithm to explore different configurations of satellites with the small operators of set D_S. After a certain number of non-improving iterations, the configuration is changed and the algorithm starts from either the incumbent or the best found solution.

The customers that were removed by the destroy operators are then inserted by means of an insertion operator to minimize the objective function value. After the destroy and repair operators are executed, the first level is improved by recursively calling again the ALNS to solve the first-level CLRP. For the instances containing only one platform (like those used in Nguyen et al. [12]), a simplified procedure based on moving and swapping single satellites is executed. In these instances, the first level is not a CLRP and hence it is not necessary to call the ALNS to solve a CLRP at every iteration.

The destroy and repair operators are selected by a roulette wheel mechanism. Every operator j has a score π_j. This score is updated by adding σ, every time operator j finds a new global best solution. The probability of being selected in the roulette wheel is based on $\pi_j / \sum_{k=1}^{p} \pi_k$, where p is the number of operators

considered. Algorithm 1 displays the pseudocode for our ALNS procedure detailed in the rest of this section.

Algorithm 1. Basic steps of the ALNS algorithm.

procedure ALNS-2ECLRP
  $s,s',s^* \leftarrow InitialSolution, InitializeScores(\boldsymbol{\pi})$
  repeat
    if ω iterations without improvement **then**
      $N^- \leftarrow ChooseDestroyOperator(D_L,\boldsymbol{\pi})$
      if $f(s') < (1+\eta)f(s^*)$ **then**
        $s' \leftarrow s^*$ with probability $(1-\Theta)$
      else
        $s' \leftarrow s^*$
      end if
    else
      $N^- \leftarrow ChooseDestroyOperator(D_S,\boldsymbol{\pi})$
    end if
    $N^+ \leftarrow ChooseRepairOperator(R,\boldsymbol{\pi})$
    $s' \leftarrow DestroyAndRepair(s',N^-,N^+)$
    solve the first-level CLRP by **ALNS-2ECLRP**
    if ω iterations without improvement **then**
      $s' \leftarrow LocalSearch(s')$
      $s \leftarrow s'$
    else if $f(s') < (1+\theta)f(s^*)$ **then**
      $s' \leftarrow LocalSearch(s')$
    endif
    if $f(s') < f(s)$ **then**
      $s \leftarrow s'$
    end if
    if $f(s) < f(s^*)$ **then**
      $s^* \leftarrow s$
    end if
    Update scores $\boldsymbol{\pi}$
  until the stopping condition is met
  return s^*
end procedure

5.1. Search space

We allow the exploration of infeasible solutions during the search. More precisely, violations of the constraints on vehicle, satellite, and platform capacities are penalized by a weighted penalty function. The objective function is $f(s) = c(s) + \alpha d(s) + \beta e(s)$, where $c(s)$ corresponds to the routing cost and the opening cost of satellites or platforms, $d(s)$ represents violations of the vehicle capacity, and $e(s)$ represents violations of the satellite or platform capacity. The parameters α and β are the corresponding weights, which are adjusted dynamically during the search. If a violation occurs, the corresponding weight is multiplied by a factor $\delta > 1$, if the solution is feasible, it is divided by δ. The weights are restricted to an interval $[\iota;\kappa]$ that guarantees that the search starts with a reasonably high weight and also prevents the weight from going to infinity.

5.2. Initial solution

For the initial solution, we open the configuration of satellites that yields the lowest cost and can serve the total customer demand. Then, customers are randomly assigned to a satellite with a bias towards the shortest distance and vehicle routes are constructed by means of the Clarke and Wright [5] Savings Algorithm. To construct a first level solution, a random platform is opened and satellites are assigned to it, no matter if platform capacity is violated. The vehicle routes at that platform are build with the Clarke and Wright [5] Savings Algorithm.

5.3. Destroy and repair operators

We use eight destroy operators and four repair operators in our algorithm, based on the ones in Hemmelmayr et al. [10]. Three of the destroy operators explicitly open or close a satellite. These are *Satellite Removal*, *Satellite Opening*, and *Satellite Swap*. They are the large operators of set D_L. *Satellite Removal* chooses one random satellite. This satellite is closed by removing all the customer routes originating from it. Furthermore, the satellite is removed from the first level routes. In the *Satellite Opening* operator, a random satellite is opened. The q customers that have the minimum distance to this satellite are removed from their current routes. *Satellite Swap* closes one satellite and opens another one. The satellite that will be closed is chosen randomly and the customer routes originating from it are removed. The satellite that will be opened is chosen randomly with a probability that is inversely proportional to the distance to the closed satellite.

The destroy operators from the set D_S remove only a limited number of customers, but do not explicitly change the satellite configuration. However, they can close satellites if all the customers of a satellite are removed and no customer is inserted at this satellite any more. It can also happen that a satellite is opened by the diversification mechanism in *Route Removal*. The operator *Random Removal* is a very simple operator that removes q random customers. *Worst Removal* selects the q "worst" customers. These are the customers that are in the most expensive insertion positions, i.e., the positions where the difference between the cost with the customer in the solution compared to the cost without the customer in the solutions is large. We normalize this gain by dividing it by the average cost of the ingoing arcs. Moreover, a perturbation factor d is added, $d \in [0.8,1.2]$. This value has been chosen following the recommended settings used by Hemmelmayr et al. [10]. In the *Related Removal* operator, a random seed customer and the $q-1$ customers that are nearest to the seed customer are removed. In the *Route Removal*, one route is removed. All customers that were contained in the removed route are put in the customer pool. In order to avoid cycling, it is forbidden to open a new route at the corresponding satellite. Another route is opened at a random satellite. This mechanism prevents the rare cases where all satellites are closed, because all customers are served by a single route originating from the only open satellite. Moreover, it is important for diversification. *Route Redistribution* is the largest destroy operator that removes between 1 and 3 random routes. The routes are selected based on the distance of the customers that are served by that route between their current satellite and any other open satellite. This selection mechanism reflects the idea that customers close to several satellites may benefit more from a reassignment than those that are only close to one satellite. A perturbation factor $d \in [0.8,1.2]$ is also added.

Finally, we use four insertion operators. Their goal is to select a satellite among the set of open satellites, a route, and an insertion position for every customer that has to be inserted. The opening of a new route is considered too, unless this is forbidden because the *Route Removal* operator was applied to the considered satellite in the same iteration. In *Greedy Insertion*, the customers are inserted in a random order into the position that minimizes the total insertion cost over all satellites and routes. There are two versions of *Greedy Insertion*. *Greedy Insertion Perturbation* uses an additional perturbation factor d, $d \in [0.8,1.2]$, that provides diversification. In the *Greedy Insertion Forbidden* operator, the satellite from which the customer has just been removed, cannot be selected for insertion in the same iteration. *Greedy Insertion Perturbation* and *Greedy Insertion Forbidden* guarantee that the insertion position is

not determined too greedily based on second-level insertion cost. *Regret Insertion* uses a more sophisticated insertion scheme. Customers for which the difference between the best and the next best insertion positions is large, are favoured for insertion. In a regret-k heuristic, customer i is chosen for insertion from the set of untreated customers U, according to $i := \arg\max_{i \in U}(\sum_{h=2}^{k} \Delta f_i^h - \Delta f_i^1)$, where Δf_i^h is the cost of inserting customer i at the h-th cheapest position. When a customer is inserted, the insertion positions of the remaining customers, that have not been inserted yet, have to be recomputed.

5.4. Local search

The goal of local search is to improve the CLRP solution on the second level. It is performed after the *Satellite Removal*, *Satellite Swap* or *Satellite Opening* operators and for promising solutions. Promising solutions are solutions for which the objective value is within $\eta\%$ of the best found solution. The following operators are used within the local search framework: split, move, swap, 2-opt, and 2-opt*. They are performed sequentially, in a first improvement manner. For more details on each of these operators, we refer to [10].

6. Computational results

Our methods were coded in C++, compiled with the Intel C++ compiler v11.0, and run on an Intel Xeon E5462, 3.0 GHz processor with 16 GB of memory. For the solution of linear and integer problems, we used CPLEX 12.2.

6.1. Test instances

We have tested our methods on several sets of standard instances from the literature. Nguyen et al. [12] introduced two sets of instances that contain only one platform at the first level, i.e., $|\mathcal{P}| = 1$. The first set is an extension of the set "Prodhon" from the CLRP and contains 30 instances with 20–200 customers and 5–10 satellites. The second set contains 24 newly generated instances and is referred to as set "Nguyen". The number of customers in these instances ranges from 25 to 200 and the number of satellites from 5 to 10. The last three sets of instances are instances used by Nguyen et al. [11], generated by Sterle [23] according to the specifications explained in Boccia et al. [3] and contain a total of 93 instances. Note that these sets do not correspond to the instances sets used in Boccia et al. [3,4], which were not available from the authors, but have been regenerated. The three sets of instances, I_1,I_2, and I_3, differ in the location of the satellites and platforms. The number of customers in the instances ranges from 8 to 200, the number of satellites ranges from 3 to 20, and the number of platforms ranges from 2 to 5. In total, our experiments are run on 147 instances.

6.2. Parameter settings

For the branch-and-cut method, the parameters associated to the separation of each family of inequalities are set as in Contardo et al. [7] for both echelons. The search strategy is set to best-bound to guide the search towards the best possible lower bound. Finally, we use as initial cut-off value the best known upper bound according to Tables 4–6 as described in the next subsection.

For the ALNS, the parameters were set according to the experimental tests. The parameters for solution acceptance were set to the following values. The threshold η, which defines that incumbent solutions that have an objective function value within $\eta\%$ of the value of the best solution found, can be accepted, was set to 1%. The probability of acceptance, Θ, was set to 0.5. For the CLRP, values for ω in the range [100; 2000] yielded the best performance. To solve the 2E-CLRP, ω was set to 1000. The number of customers to remove is a random integer between ρ and τ. We set ρ to 1 and τ to $\lceil 0.6|\mathcal{C}| \rceil$. For the weighted penalty function, δ was set to 1.1, ι to 5 and κ to 10,000. The parameter that is added to the score π_j every time a new best solution is found, σ, was set to 1. As a stopping condition, we choose the number of iterations. We decided that 500,000 iterations offer a good trade-off between runtime and solution quality. A regret-3 heuristic was used in *Regret Insertion* and θ, which is the threshold that identifies promising solutions that are selected for local search, was set to 0.2.

6.3. Numerical results

In the calibration phase of the ALNS, we tested the performance of the method under several different parameter settings. Some of these settings yielded the best results for some instances, without always providing good overall results. We report in Tables 4–6 in the Appendix the best results obtained by the ALNS during this testing phase, and highlight those in which we observe an improvement of the current best known solutions as reported in Nguyen et al. [12,11]. From now on, we refer to the best known results as the results reported in those tables. Note that in order to make fair comparisons to other methods and to assess the performances of our algorithms, these values are only taken as references.

Due to the randomness incorporated in the ALNS, we have performed 10 runs of our method on each instance. We compare in Table 1 the ALNS meta-heuristic we propose against the methods GRASP+PR [12] and MS-ILS+PR [11]. In these papers, the authors do not report the average solution quality of their algorithms, but only the best solutions found after five runs for each instance. For the sets I_1,I_2 and I_3, Nguyen et al. [11] only report results for the instances with 50 and more customers, which we also include in our comparison. In this table, headers gap_{avg} and gap_{min} correspond to the average and the minimum gaps, respectively, for each algorithm, when available. The gap between a solution of value z and a best known solution of value z^* is computed as $(z-z^*)/z^* \times 100$. Header t_{avg} stands for the average CPU time spent, in seconds, by each method (times are not scaled and comparisons based on time should be considered with care). The results reported by Nguyen et al. [12,11] are based on experiments run on an Intel Pentium IV CPU running at 3.4 GHz with 1 GB of total RAM.

As one can see from the results obtained, our method is robust and clearly outperforms the previous heuristics of Nguyen et al. [12,11]. Indeed, our ALNS provides solutions on average more than 10 times better (in terms of the gaps with respect to the best known solutions) than the solutions obtained by MS-ILS+PR. Although it is not possible to make a fair comparison with method GRASP+PR

Table 1
Aggregated results of the proposed ALNS.

Set	GRASP+PR		MS-ILS+PR		ALNS		
	Gap_{min}	t_{avg}	Gap_{min}	t_{avg}	Gap_{avg}	Gap_{min}	t_{avg}
Prodhon	1.15	14.2	0.88	178.3	0.73	0.21	465.82
Nguyen	1.60	13.1	0.80	112.20	0.45	0.12	191.97
I_1[a]	–	–	3.15	917.10	0.53	0.00	839.60
I_2[a]	–	–	2.72	928.00	0.60	0.00	913.70
I_3[a]	–	–	1.68	935.10	0.49	0.00	909.85
Average			1.44	426.71		0.11	538.25
# of BKS	8/84		14/84		70/84 (133/147)[b]		

[a] Restricted to instances containing 50 or more customers.
[b] All instances included.

(due to the smaller number of instances reported in that paper), the differences between GRASP+PR and MS-ILS+PR suggest that when comparing to ALNS the differences should be even larger. Moreover, out of the 84 instances considered in the comparison, our method was able to find the best known solutions in a total of 70 instances, against 8 for GRASP+PR and 14 for MS-ILS+PR.

To better evaluate the strength of the formulation and branch-and-cut method we propose, we also implemented the three-index formulation introduced by Boccia et al. [4]. In such a formulation, a third index is added to the vehicle-flow variables to take into account the vehicle performing each particular trip. This formulation contains $O(|\mathcal{S}||E^1|+|\mathcal{C}||E^2|)$ variables and constraints but, as indicated previously, presents many symmetries when the fleet is homogeneous, providing weak bounds in large computing times, as shown by our computational results.

We report in Table 2 the aggregated results obtained by our branch-and-cut algorithm and by our implementation of the three-index vehicle-flow formulation (both methods run on the same machine). Header *#opt* stands for the number of instances that were solved to optimality. Headers gap_{lr} and t_{lr} correspond to the gap (in %) and the CPU time (in seconds) spent for the solution of the linear relaxation of the problem. Analogously, columns *gap* and *t* stand for the gap and CPU time spent after a maximum of two hours of computation. Given a lower bound z_{lb} and an upper bound z_{ub} the gap is computed as $(z_{ub}-z_{lb})/z_{ub}\times 100$. Finally, Column *gap** corresponds to the optimality gap between the final lower bound and the average solution value found by the ALNS.

As shown in Table 2, our algorithm can solve more and larger instances than the previous three-index formulation of Boccia et al. [4]. For the instances that remain unsolved, our branch-and-cut method provides much tighter gaps in lower computing times. Also, it is possible to establish a comparison between our ALNS and the branch-and-cut algorithm. Indeed, if we consider the average upper bounds provided by the ALNS and the lower bounds obtained with the branch-and-cut method, they are at an average distance of 3.06%, which leaves little space for further improvements and also validates both algorithms introduced in this paper.

In Table 3, we report a sensitivity analysis of the branch-and-cut method with respect to the quality of the initial upper bound. In this table, we report results restricted to a small number of instances. We have chosen 20 instances from the three sets, containing small, medium and large instances. We tested and compared two different settings. In the columns *With UB*, we report the results obtained by our branch-and-cut method under the default setting when the best known solutions are taken as initial cut-off values. Under the header *Without UB*, we report the same results but without considering any initial cut-off value. In both cases, we report the final lower bound, the number of nodes inspected, the total CPU time, as well as the final gap with respect

Table 2
Aggregated results of the B&C algorithm.

Set	B&C						3-Index model[a]		
	#opt	Gap_{lr}	t_{lr}	*Gap*	*t*	*Gap**	*#opt*	*Gap*	*t*
Prodhon	8/30	7.22	963.27	3.55	5644.80	4.21	0/30	21.83	7200
Nguyen	11/24	8.87	508.23	3.07	4190.05	3.47	0/24	23.45	7200
I_1	17/31	11.33	574.65	2.96	3400.84	3.11	6/31	19.24	5905.62
I_2	19/31	8.41	186.97	2.47	3007.53	2.65	6/31	17.55	5629.40
I_3	20/31	9.31	80.15	1.85	2668.83	2.01	6/31	17.05	5639.27
Average		9.05	457.08	2.76	3750.33	3.06		19.64	6266.69
Optimality	75/147						18/147		

[a] Instances with 150 or more customers could not be loaded into memory.

Table 3
Sensitivity analysis of the B&C algorithm.

Instance	z_{UB}	With UB				Without UB			
		LB	*#N*	*t*	*Gap*	LB	*#N*	*t*	*Gap*
I_1-20×8×3	848.31	848.31	31,055	1412.31	0.00	848.31	38,489	2409.29	0.00
I_2-20×8×3	758.06	758.06	682	6.05	0.00	758.06	1161	18.08	0.00
I_3-20×8×3	643.89	643.89	67	1.67	0.00	643.89	102	2.72	0.00
I_1-50×10×5	1132.63	1104.98	4373	7411.64	2.44	1103.60	4178	7415.68	2.56
I_2-50×10×5	1256.44	1222.10	4165	7420.76	2.73	1221.99	3935	7419.09	2.74
I_3-50×10×5	1207.31	1179.15	3565	7469.87	2.33	1176.75	3527	7465.06	2.53
I_1-100×10×5	2124.90	1958.10	53	7480.91	7.85	1959.88	58	7508.93	7.77
I_2-100×10×5	2231.21	2072.44	192	7413.89	7.12	2073.91	216	7473.96	7.05
I_3-100×10×5	2178.35	2057.15	225	7518.35	5.56	2056.28	198	7503.27	5.60
I_1-150×10×5	1883.44	1711.06	16	7425.67	9.15	1711.04	16	7401.28	9.15
I_2-150×10×5	1728.05	1553.03	52	7575.30	10.13	1550.90	53	7577.74	10.25
I_3-150×10×5	1274.44	1198.77	152	7474.79	5.94	1198.88	188	7535.21	5.93
ppw-20×5-1a	89,075	89,075	203	2.77	0.00	89,075	175	3.73	0.00
ppw-50×5-1a	130,843	128,870	5354	7326.08	1.51	128,999	4333	7337.29	1.41
ppw-100×10-1a	353,133	325,164	99	7468.46	7.92	323,051	71	7464.60	8.52
ppw-200×10-1a	549,718	484,115	2	7266.78	11.93	484,115	2	7271.33	11.93
25-5MN	78,947	78,947	13	0.93	0.00	78,947	36	1.11	0.00
50-5MN	123,484	123,484	164	46.47	0.00	123,484	160	109.48	0.00
100-10MN	201,275	192,108	1362	7467.13	4.55	192,174	908	7518.53	4.52
200-10MN	324,006	294,931	7	7356.73	8.97	294,931	7	7369.34	8.97
Average			2590	5277.33	4.41		2891	5340.29	4.45

to the best known solution. As shown in this table, the impact of adding a cut-off value to the algorithm is not important (the final gap decreases from 4.45% to 4.41%), the improvement being marginal with respect to the situation without an initial cut-off value. This shows the robustness of our method which produces almost the same lower bound even if no upper bound is known in advance. However, the algorithm following a best-bound strategy does not always find a feasible solution, which also demonstrates the need of a heuristic such as the ALNS we introduce in this paper.

7. Conclusions

We presented new algorithms to efficiently reach good-quality lower and upper bounds for the 2E-CLRP. We introduced a compact two-index vehicle-flow formulation, proposed several families of valid inequalities and embedded them into a branch-and-cut solver. To the best of our knowledge, this is the first time an exact method has been proposed for this problem class. The method is able to solve to optimality small- and medium-size instances containing up to 50 customers, and provides tight lower bounds for the instances that cannot be solved. We also introduced an adaptive large neighbourhood search meta-heuristic capable of providing good upper bounds in short computing times. The proposed ALNS outperforms previous heuristics for the 2E-CVRP with single-sourcing constraints in terms of upper bound quality and also provides good quality upper bounds for the instances of 2E-CLRP. Moreover, our ALNS was able to find the value of the best known solutions on 133 instances out of 147. When comparing our methods, we observe that the lower bounds obtained by the branch-and-cut method lie no further than 3.06% on average below the average solution values found by the ALNS, which validates the robustness and quality of both approaches.

As an avenue of future research, we believe that exploring other meta-heuristic approaches combining integer-programming methods and meta-heuristics could lead to better upper bounds. On the other hand, we believe that embedding the inequalities used in this paper into a branch-and-cut-and-price solver could result in a more robust exact method being able to scale better on large instances. Also, the methodologies used in this paper can be used to solve some related problems combining location with routing decisions.

Acknowledgements

Financial support from the Austrian Science Fund (FWF-project no. J3047) and the Natural Sciences and Engineering Research Council of Canada (NSERC) are gratefully acknowledged. The authors would like to thank Jean-François Cordeau for his valuable comments, as well as two anonymous referees for their insightful questions and comments.

Appendix A

In this Appendix, we provide detailed results for both algorithms introduced in this paper and provide a brief discussion on our results.

Tables 4–6 report the values of the best solutions found by the proposed ALNS during the parameter calibration phase. Although some of these results might not be reproducible by our calibrated algorithm, we believe that they are useful as a benchmark for future methods. In these tables, headers labelled z^* stand for the best known solutions reported by Nguyen et al. [12,11], headers z^*_{BKS} stand for the best solutions found by our ALNS, and *gap* stands for the gap between the two solutions, computed as $(z^*_{ALNS}-z^*_{NPP})/z^*_{NPP}\times 100$.

Tables 7–11 report detailed results obtained by our ALNS. In these tables, header z^*_{BKS} stands for the best known solution as reported in Tables 4–6. Header z^*_{avg} stands for the average solution

Table 4
Best known results for the set Prodhon.

Instance	z^*	z^*_{BKS}	*Gap*
ppw-20 × 5-1a	89,075	89,075	0.00
ppw-20 × 5-1b	61,863	61,863	0.00
ppw-20 × 5-2a	85,290	**84,478**	−0.95
ppw-20 × 5-2b	60,838	60,838	0.00
ppw-50 × 5-1a	134,855	**130,843**	−2.98
ppw-50 × 5-1b	101,530	101,530	0.00
ppw-50 × 5-2a	132,159	**131,825**	−0.25
ppw-50 × 5-2b	110,547	**110,332**	−0.19
ppw-50 × 5-2BIS	122,654	**122,599**	−0.04
ppw-50 × 5-2bBIS	105,776	**105,696**	−0.08
ppw-50 × 5-3a	128,379	128,379	0.00
ppw-50 × 5-3b	104,006	104,006	0.00
ppw-100 × 5-1a	320,130	**318,761**	−0.43
ppw-100 × 5-1b	258,205	**256,878**	−0.51
ppw-100 × 5-2a	234,179	**231,305**	−1.23
ppw-100 × 5-2b	195,426	**194,729**	−0.36
ppw-100 × 5-3a	245,944	**244,194**	−0.71
ppw-100 × 5-3b	195,254	**194,110**	−0.59
ppw-100 × 10-1a	358,939	**353,133**	−1.62
ppw-100 × 10-1b	302,584	**297,167**	−1.79
ppw-100 × 10-2a	306,303	**305,154**	−0.38
ppw-100 × 10-2b	264,389	**263,876**	−0.19
ppw-100 × 10-3a	313,249	**310,200**	−0.97
ppw-100 × 10-3b	266,383	**261,796**	−1.72
ppw-200 × 10-1a	554,598	**549,718**	−0.88
ppw-200 × 10-1b	452,286	**445,802**	−1.43
ppw-200 × 10-2a	502,173	**498,199**	−0.79
ppw-200 × 10-2b	425,311	**423,031**	−0.54
ppw-200 × 10-3a	533,732	**531,548**	−0.41
ppw-200 × 10-3b	418,800	**402,130**	−3.98
Average			−0.77

Table 5
Best known results for the set Nguyen.

Instance	z^*	z^*_{BKS}	*Gap*
25-5N	80,370	80,370	0.00
25-5Nb	64,562	64,562	0.00
25-5MN	78,947	78,947	0.00
25-5MNb	64,438	64,438	0.00
50-5N	138,126	**137,815**	−0.23
50-5Nb	111,062	**110,094**	−0.87
50-5MN	123,484	123,484	0.00
50-5MNb	105,401	105,401	0.00
50-10N	116,132	**115,725**	−0.35
50-10Nb	87,315	87,315	0.00
50-10MN	135,748	**135,519**	−0.17
50-10MNb	110,613	110,613	0.00
100-5N	196,910	**193,228**	−1.87
100-5Nb	159,086	**158,927**	−0.10
100-5MN	207,119	**204,682**	−1.18
100-5MNb	166,115	**165,744**	−0.22
100-10N	215,792	**210,449**	−2.48
100-10Nb	156,401	**155,489**	−0.58
100-10MN	205,964	**201,275**	−2.28
100-10MNb	170,706	**170,625**	−0.05
200-10N	353,685	**347,395**	−1.78
200-10Nb	262,072	**256,171**	−2.25
200-10MN	332,345	**324,006**	−2.51
200-10MNb	292,523	**287,076**	−1.86
Average			−0.78

Table 6
Best known results for the sets I_1,I_2 and I_3.

Instance	I_1		I_2		I_3	
	z^*	z^*_{BKS}	z^*	z^*_{BKS}	z^*	z^*_{BKS}
$8\times3\times2$		**575.70**		**575.70**		**578.33**
$8\times4\times2$		**549.34**		**604.13**		**450.71**
$9\times3\times2$		**878.69**		**386.15**		**454.63**
$10\times4\times2$		**806.72**		**629.38**		**540.61**
$10\times5\times3$		**696.94**		**551.45**		**745.48**
$10\times8\times3$		**596.56**		**504.20**		**412.91**
$15\times10\times2$		**732.48**		**709.10**		**605.11**
$15\times10\times3$		**686.71**		**777.49**		**546.61**
$15\times4\times2$		**1064.52**		**827.81**		**688.87**
$15\times5\times3$		**933.75**		**1075.22**		**1001.28**
$15\times8\times3$		**730.36**		**652.58**		**578.23**
$20\times10\times2$		**937.40**		**766.24**		**744.82**
$20\times10\times3$		**761.28**		**744.26**		**728.17**
$20\times10\times4$		**1149.72**		**793.00**		**1190.95**
$20\times8\times2$		**1029.59**		**772.29**		**846.07**
$20\times8\times3$		**848.31**		**758.06**		**643.89**
$25\times10\times2$		**1030.40**		**961.59**		**834.23**
$25\times10\times3$		**1062.30**		**987.57**		**820.12**
$25\times10\times4$		**1607.94**		**1125.56**		**1057.63**
$25\times8\times2$		**950.87**		**912.02**		**951.56**
$25\times8\times3$		**870.69**		**979.62**		**774.36**
$50\times10\times5$	1207.31	**1132.63**	1265.73	**1256.44**	1208.43	**1207.31**
$50\times8\times5$	1171.69	**1162.44**	1123.42	**1121.13**	1171.35	**1162.44**
$75\times10\times5$	1561.5	**1540.23**	1718.25	**1691.15**	1732.33	**1721.47**
$75\times15\times5$	1700.32	**1686.21**	1751.14	**1742.25**	1491.31	**1483.14**
$100\times10\times5$	2192.14	**2124.9**	2290.64	**2231.21**	2238.7	**2178.35**
$100\times20\times5$	1989.48	**1973.08**	2039.25	**1996.34**	2053.12	**2035.37**
$150\times10\times5$	1953.55	**1883.44**	1768.79	**1728.05**	1307.19	**1274.44**
$150\times20\times5$	1905.81	**1869.53**	1664.2	**1630.29**	1266.83	**1235.86**
$200\times10\times5$	2601.33	**2443.80**	2292.47	**2147.51**	1822.5	**1766.46**
$200\times20\times5$	2307.53	**2219.54**	2097.74	**2049**	2604.56	**2531.21**

Table 7
Results of the ALNS on the instances of set Prodhon.

Instance	z^*_{BKS}	z^*_{avg}	Gap_{avg}	z^*_{min}	Gap_{min}	t_{avg}	t_{min}
ppw-20 × 5-1a	89,075	89,075.0	0.00	89,075	0.00	26.3	2.8
ppw-20 × 5-1b	61,863	61,863.0	0.00	61,863	0.00	26.1	0.2
ppw-20 × 5-2a	84,478	84,478.0	0.00	84,478	0.00	28.9	2.8
ppw-20 × 5-2b	60,838	60,838.0	0.00	60,838	0.00	29.1	0
ppw-50 × 5-1a	130,843	131,036.6	0.15	130,843	0.00	96.9	8.2
ppw-50 × 5-1b	101,530	101,530.0	0.00	101,530	0.00	94.2	8.7
ppw-50 × 5-2a	131,825	131,878.3	0.04	131,825	0.00	130.6	37.2
ppw-50 × 5-2b	110,332	110,332.0	0.00	110,332	0.00	160.7	21.1
ppw-50 × 5-2BIS	122,599	122,626.5	0.02	122,599	0.00	121.2	43.2
ppw-50 × 5-2bBIS	105,696	105,719.9	0.02	105,696	0.00	131.6	57.4
ppw-50 × 5-3a	128,379	128,379.0	0.00	128,379	0.00	88.4	17.8
ppw-50 × 5-3b	104,006	104,006.0	0.00	104,006	0.00	114.9	14.2
ppw-100 × 5-1a	318,761	320,511.2	0.55	319,137	0.12	646.9	445
ppw-100 × 5-1b	256,878	258,540.8	0.65	257,349	0.18	1179.9	607.9
ppw-100 × 5-2a	231,305	231,305.0	0.00	231,305	0.00	316	43.7
ppw-100 × 5-2b	194,729	194,771.0	0.02	194,729	0.00	1641	602.4
ppw-100 × 5-3a	244,194	244,418.0	0.09	244,194	0.00	375.5	167.9
ppw-100 × 5-3b	194,110	194,239.4	0.07	194,110	0.00	377	210.9
ppw-100 × 10-1a	353,133	365,036.1	3.37	358,068	1.40	158.9	65.6
ppw-100 × 10-1b	297,167	303,089.8	1.99	297,167	0.00	155.2	118.7
ppw-100 × 10-2a	305,154	307,762.9	0.85	305,402	0.08	270	163
ppw-100 × 10-2b	263,876	266,642.1	1.05	265,138	0.48	348.2	225.9
ppw-100 × 10-3a	310,200	318,499.8	2.68	313,517	1.07	215.7	106
ppw-100 × 10-3b	261,796	270,326.5	3.26	264,096	0.88	256.9	181.6
ppw-200 × 10-1a	549,718	559,774.8	1.83	552,816	0.56	1039.2	648.7
ppw-200 × 10-1b	445,802	459,033.1	2.97	448,236	0.55	1811.9	927.6
ppw-200 × 10-2a	498,199	498,659.4	0.09	498,199	0.00	576.4	339.3
ppw-200 × 10-2b	423,031	423,517.6	0.12	423,048	0.00	1723.8	1161.5
ppw-200 × 10-3a	531,548	535,823.8	0.80	534,569	0.57	741.3	285.2
ppw-200 × 10-3b	402,130	407,070.2	1.23	404,284	0.54	1091.9	439.1
Average			0.73		0.21	465.82	231.79

Table 8
Results of the ALNS on the instances of set Nguyen.

Instance	z^*_{BKS}	z^*_{avg}	Gap_{avg}	z^*_{min}	Gap_{min}	t_{avg}	t_{min}
25-5N	80,370	80,370	0.00	80,370	0.00	38	0.1
25-5Nb	64,562	64,562	0.00	64,562	0.00	36.4	0
25-5MN	78,947	78,947	0.00	78,947	0.00	50.6	0.5
25-5MNb	64,438	64,438	0.00	64,438	0.00	30.6	0
50-5N	137,815	138,093.2	0.20	137,815	0.00	74.2	33.9
50-5Nb	110,094	110,094	0.00	110,094	0.00	113.8	32.3
50-5MN	123,484	123,484	0.00	123,484	0.00	105.2	13.1
50-5MNb	105,401	105,401	0.00	105,401	0.00	77.5	23.1
50-10N	115,725	115,843	0.10	115,725	0.00	90.3	21.3
50-10Nb	87,315	87,315	0.00	87,315	0.00	102	20.7
50-10MN	135,519	135,556.6	0.03	135,519	0.00	76.3	39
50-10MNb	110,613	110,636.1	0.02	110,613	0.00	49.2	19.1
100-5N	193,228	19,4012.3	0.41	193,228	0.00	224.5	154.3
100-5Nb	158,927	159,029.9	0.06	158,927	0.00	234.9	133
100-5MN	204,682	204,819.6	0.07	204,682	0.00	271.2	135.7
100-5MNb	165,744	165,863.1	0.07	165,744	0.00	220.9	112.9
100-10N	210,449	216,285.5	2.77	212,847	1.14	150.8	70.8
100-10Nb	155,489	156,261.6	0.50	155,489	0.00	177.2	70.3
100-10MN	201,275	202,491.9	0.60	201,275	0.00	155.4	104.7
100-10MNb	170,625	170,985.4	0.21	170,625	0.00	178.7	114.2
200-10N	347,395	351,770	1.26	347,395	0.00	420.5	237.2
200-10Nb	256,171	258,397	0.87	256,171	0.00	492.4	340.8
200-10MN	324,006	329,913.7	1.82	326,454	0.76	547.3	354.7
200-10MNb	287,076	292,357.6	1.84	289,742	0.93	689.5	481.7
Average			0.45		0.12	191.97	104.72

Table 9
Results of the ALNS on the instances of set I_1.

Instance	z^*_{BKS}	z^*_{avg}	Gap_{avg}	z^*_{min}	Gap_{min}	t_{avg}	t_{min}
I_1-8 × 3 × 2	575.701	575.70	0.00	575.70	0.00	15.3	0.2
I_1-8 × 4 × 2	549.338	549.34	0.00	549.34	0.00	18.0	0.2
I_1-9 × 3 × 2	878.69	878.69	0.00	878.69	0.00	36.4	0.0
I_1-10 × 4 × 2	806.719	806.72	0.00	806.72	0.00	30.1	0.0
I_1-10 × 5 × 3	696.938	696.94	0.00	696.94	0.00	22.8	0.0
I_1-10 × 8 × 3	596.563	596.56	0.00	596.56	0.00	40.4	4.2
I_1-15 × 10 × 2	732.483	732.48	0.00	732.48	0.00	48.6	2.4
I_1-15 × 10 × 3	686.714	686.71	0.00	686.71	0.00	49.0	1.7
I_1-15 × 4 × 2	1064.52	1064.52	0.00	1064.52	0.00	47.5	0.3
I_1-15 × 5 × 3	933.747	933.75	0.00	933.75	0.00	49.1	0.2
I_1-15 × 8 × 3	730.362	730.36	0.00	730.36	0.00	55.7	0.9
I_1-20 × 10 × 2	937.401	937.40	0.00	937.40	0.00	62.1	10.3
I_1-20 × 10 × 3	761.285	761.29	0.00	761.29	0.00	65.5	0.9
I_1-20 × 10 × 4	1149.72	1149.72	0.00	1149.72	0.00	66.6	5.9
I_1-20 × 8 × 2	1029.59	1029.59	0.00	1029.59	0.00	67.0	6.3
I_1-20 × 8 × 3	848.31	848.31	0.00	848.31	0.00	64.8	1.7
I_1-25 × 10 × 2	1030.4	1030.40	0.00	1030.40	0.00	74.9	6.7
I_1-25 × 10 × 3	1062.3	1062.30	0.00	1062.30	0.00	75.1	2.8
I_1-25 × 10 × 4	1607.94	1607.94	0.00	1607.94	0.00	76.3	12.3
I_1-25 × 8 × 2	950.866	950.87	0.00	950.87	0.00	71.8	2.7
I_1-25 × 8 × 3	870.692	870.69	0.00	870.69	0.00	76.4	0.1
I_1-50 × 10 × 5	1132.63	1133.74	0.10	1132.63	0.00	341.8	178.8
I_1-50 × 8 × 5	1162.44	1163.45	0.09	1162.44	0.00	321.5	119.6
I_1-75 × 10 × 5	1540.23	1540.91	0.04	1540.23	0.00	507.2	219.9
I_1-75 × 15 × 5	1686.21	1700.38	0.84	1686.21	0.00	527.3	247.6
I_1-100 × 10 × 5	2124.9	2136.82	0.56	2124.90	0.00	659.9	499.5
I_1-100 × 20 × 5	1973.08	1983.05	0.51	1973.08	0.00	705.2	605.2
I_1-150 × 10 × 5	1883.44	1893.92	0.56	1883.44	0.00	1187.8	718.9
I_1-150 × 20 × 5	1869.53	1889.07	1.05	1869.53	0.00	1203.1	901.5
I_1-200 × 10 × 5	2443.8	2461.77	0.74	2443.80	0.00	1431.0	1135.6
I_1-200 × 20 × 5	2219.54	2238.06	0.83	2219.54	0.00	1511.5	1073.8
Restricted average[a]			0.53		0.00	839.60	570.00
Total average			0.17		0.00	307.76	185.81

[a] Restricted to instances containing 50 or more customers.

value found by our method. Header gap_{avg} stands for the gap between our average values and the best known solutions. It is computed as $(z^*_{avg}-z^*_{BKS})/z^*_{BKS} \times 100$. Header z^*_{min} stands for the best solution found in the 10 runs. Header gap_{min} stands for the gap between the best known solution and our best solution, which is computed as $(z^*_{min}-z^*_{BKS})/z^*_{BKS} \times 100$. Finally, t_{avg} and t_{min}

Table 10
Results of the ALNS on the instances of set I_2.

Instance	z^*_{BKS}	z^*_{avg}	Gap_{avg}	z^*_{min}	Gap_{min}	t_{avg}	t_{min}
I_2-8 × 3 × 2	575.701	575.7	0.00	575.7	0.00	15.9	0.0
I_2-8 × 4 × 2	604.127	604.13	0.00	604.13	0.00	24.5	0.0
I_2-9 × 3 × 2	386.152	386.15	0.00	386.15	0.00	17.5	0.0
I_2-10 × 4 × 2	629.381	629.38	0.00	629.38	0.00	18.4	0.0
I_2-10 × 5 × 3	551.452	551.45	0.00	551.45	0.00	27.3	0.0
I_2-10 × 8 × 3	504.203	504.2	0.00	504.2	0.00	39.6	0.0
I_2-15 × 10 × 2	709.103	709.1	0.00	709.1	0.00	53.7	1.0
I_2-15 × 10 × 3	777.488	777.49	0.00	777.49	0.00	56.5	0.5
I_2-15 × 4 × 2	827.81	827.81	0.00	827.81	0.00	45.3	0.4
I_2-15 × 5 × 3	1075.22	1075.22	0.00	1075.22	0.00	56.1	0.0
I_2-15 × 8 × 3	652.575	652.58	0.00	652.58	0.00	49.5	1.9
I_2-20 × 10 × 2	766.238	766.24	0.00	766.24	0.00	62.3	2.3
I_2-20 × 10 × 3	744.265	744.27	0.00	744.27	0.00	62.0	3.0
I_2-20 × 10 × 4	792.996	793	0.00	793	0.00	68.5	0.6
I_2-20 × 8 × 2	772.29	772.29	0.00	772.29	0.00	65.5	14.2
I_2-20 × 8 × 3	758.059	758.06	0.00	758.06	0.00	65.6	1.4
I_2-25 × 10 × 2	961.59	961.59	0.00	961.59	0.00	81.5	3.7
I_2-25 × 10 × 3	987.57	987.57	0.00	987.57	0.00	79.6	0.8
I_2-25 × 10 × 4	1125.56	1125.56	0.00	1125.56	0.00	79.0	4.0
I_2-25 × 8 × 2	912.017	912.02	0.00	912.02	0.00	77.9	9.3
I_2-25 × 8 × 3	979.615	979.62	0.00	979.62	0.00	77.9	0.0
I_2-50 × 10 × 5	1256.44	1256.44	0.00	1256.44	0.00	410.8	96.5
I_2-50 × 8 × 5	1121.13	1121.13	0.00	1121.13	0.00	386.6	51.5
I_2-75 × 10 × 5	1691.15	1691.15	0.00	1691.15	0.00	638.7	208.6
I_2-75 × 15 × 5	1742.25	1748.62	0.37	1742.25	0.00	670.1	446.6
I_2-100 × 10 × 5	2231.21	2242.28	0.50	2231.21	0.00	870.9	535.5
I_2-100 × 20 × 5	1996.34	2018.74	1.12	1996.34	0.00	872.0	592.8
I_2-150 × 10 × 5	1728.05	1734.64	0.38	1728.05	0.00	1004.6	683.8
I_2-150 × 20 × 5	1630.29	1654.83	1.51	1630.29	0.00	1070.3	832.3
I_2-200 × 10 × 5	2147.51	2158.24	0.50	2147.51	0.00	1477.2	1002.8
I_2-200 × 20 × 5	2049.01	2081.58	1.59	2049.01	0.00	1736.0	1369.6
Restricted average[a]			0.60		0.00	913.7	582.0
Total average			0.19		0.00	331.01	189.13

[a] Restricted to instances containing 50 or more customers.

Table 11
Results of the ALNS on the instances of set I_3.

Instance	z^*_{BKS}	z^*_{avg}	Gap_{avg}	z^*_{min}	Gap_{min}	t_{avg}	t_{min}
I_3-8 × 3 × 2	578.325	578.33	0.00	578.33	0.00	15.3	0.1
I_3-8 × 4 × 2	450.713	450.71	0.00	450.71	0.00	17.2	0.0
I_3-9 × 3 × 2	454.627	454.63	0.00	454.63	0.00	15.6	0.0
I_3-10 × 4 × 2	540.605	540.61	0.00	540.61	0.00	20.3	0.0
I_3-10 × 5 × 3	745.484	745.48	0.00	745.48	0.00	30.5	0.1
I_3-10 × 8 × 3	412.906	412.91	0.00	412.91	0.00	20.4	0.0
I_3-15 × 10 × 2	605.108	605.11	0.00	605.11	0.00	48.6	0.0
I_3-15 × 10 × 3	546.612	546.61	0.00	546.61	0.00	53.0	2.8
I_3-15 × 4 × 2	688.869	688.87	0.00	688.87	0.00	45.1	0.3
I_3-15 × 5 × 3	1001.28	1005.02	0.37	1001.28	0.00	50.5	2.7
I_3-15 × 8 × 3	578.225	578.23	0.00	578.23	0.00	50.8	2.3
I_3-20 × 10 × 2	744.815	744.82	0.00	744.82	0.00	62.7	2.8
I_3-20 × 10 × 3	728.169	728.17	0.00	728.17	0.00	65.0	8.2
I_3-20 × 10 × 4	1190.95	1190.95	0.00	1190.95	0.00	63.1	5.6
I_3-20 × 8 × 2	846.066	846.07	0.00	846.07	0.00	68.0	7.0
I_3-20 × 8 × 3	643.892	643.89	0.00	643.89	0.00	63.9	6.5
I_3-25 × 10 × 2	834.226	834.34	0.01	834.23	0.00	75.0	24.3
I_3-25 × 10 × 3	820.123	820.12	0.00	820.12	0.00	75.3	2.9
I_3-25 × 10 × 4	1057.63	1057.63	0.00	1057.63	0.00	74.3	7.9
I_3-25 × 8 × 2	951.56	951.59	0.00	951.56	0.00	74.0	40.1
I_3-25 × 8 × 3	774.356	774.36	0.00	774.36	0.00	73.7	5.4
I_3-50 × 10 × 5	1207.31	1207.31	0.00	1207.31	0.00	341.2	36.9
I_3-50 × 8 × 5	1162.44	1164.64	0.19	1162.44	0.00	353.2	107.6
I_3-75 × 10 × 5	1721.47	1723.41	0.11	1721.47	0.00	691.0	406.3
I_3-75 × 15 × 5	1483.14	1483.42	0.02	1483.14	0.00	634.7	392.6
I_3-100 × 10 × 5	2178.35	2183.31	0.23	2178.35	0.00	922.4	584.5
I_3-100 × 20 × 5	2035.37	2048.35	0.64	2035.37	0.00	755.1	585.1
I_3-150 × 10 × 5	1274.44	1282.13	0.60	1274.44	0.00	1040.0	710.1
I_3-150 × 20 × 5	1235.86	1254.38	1.50	1235.86	0.00	1062.9	829.4
I_3-200 × 10 × 5	1766.46	1776.47	0.57	1766.46	0.00	1690.0	1079.6
I_3-200 × 20 × 5	2531.21	2557.05	1.02	2531.21	0.00	1608.0	1157.7
Restricted average[a]			0.49		0.00	909.85	588.98
Total average			0.17		0.00	327.77	193.83

[a] Restricted to instances containing 50 or more customers.

stand for the average times spent by the whole algorithm and until finding the best solution at each run, respectively.

Tables 12–16 report detailed results of our branch-and-cut method. In these tables, columns labelled z_{UB} stand for the upper bound value, as reported in Tables 4–6. Columns labelled z_{lr} stand for the lower bound at the linear relaxation. Columns labelled gap_{lr} stand for the gap at the linear relaxation, computed as $(z_{UB}-z_{lr})/z_{UB} \times 100$. Columns labelled t_{lr} represent the CPU time (in seconds) spent for solving the linear relaxation. Analogously, columns labelled z, gap and t represent the final lower bound, the final gap and the total CPU time (in seconds) after a maximum time of two hours. Note that our branch-and-cut method did not improve any of the upper bounds found by the proposed ALNS, which corroborates the fact that the ALNS produces very tight

Table 12
Results of the B&C algorithm on the set Prodhon.

Instance	z_{UB}	z_{lr}	Gap_{lr}	t_{lr}	z	Gap	t	#N	Gap^*
ppw-20 × 5-1a	89,075	82,642.7	7.22	0.24	89,075	0.00	2.77	203	0.00
ppw-20 × 5-1b	61,863	61,793.2	0.11	0.17	61,863	0.00	0.22	4	0.00
ppw-20 × 5-2a	84,478	79,253.1	6.18	0.23	84,478	0.00	4.98	380	0.00
ppw-20 × 5-2b	60,838	59,044.2	2.95	0.23	60,838	0.00	0.29	1	0.00
ppw-50 × 5-1a	130,843	116,292	11.12	18.44	128,870	1.51	7326.08	5354	1.65
ppw-50 × 5-1b	101,530	94,577	6.85	2.28	101,530	0.00	5688.08	10,952	0.00
ppw-50 × 5-2a	131,825	123,481	6.33	5.31	131,825	0.00	1670.06	5352	0.04
ppw-50 × 5-2b	110,332	104,488	5.30	1.98	110,332	0.00	75.27	833	0.00
ppw-50 × 5-2Bis	122,599	117,815	3.90	22.64	122,023	0.47	7283.15	7900	0.49
ppw-50 × 5-2bBis	105,696	88,717.5	16.06	3.20	90,196.6	14.66	7410.63	6249	14.68
ppw-50 × 5-3a	128,379	119,734	6.73	7.74	127,291	0.85	7289.17	5134	0.85
ppw-50 × 5-3b	104,006	101,702	2.22	2.16	104,006	0.00	41.33	499	0.00
ppw-100 × 5-1a	318,761	301,246	5.49	694.48	311,192	2.37	7353.31	579	2.91
ppw-100 × 5-1b	256,878	241,312	6.06	252.07	251,596	2.06	7423.35	140	2.69
ppw-100 × 5-2a	231,305	221,840	4.09	178.18	228,186	1.35	7396.83	769	1.35
ppw-100 × 5-2b	194,729	189,324	2.78	33.83	193,425	0.67	7440.09	994	0.69
ppw-100 × 5-3a	244,194	220,717	9.61	55.71	240,856	1.37	7336.53	1145	1.46
ppw-100 × 5-3b	194,110	183,744	5.34	27.01	192,564	0.80	7355.21	405	0.86
ppw-100 × 10-1a	353,133	306,764	13.13	280.34	325,164	7.92	7468.46	99	10.92
ppw-100 × 10-1b	297,167	261,305	12.07	81.83	275,997	7.12	7407.30	361	8.94
ppw-100 × 10-2a	305,154	272,802	10.60	330.54	283,090	7.23	7437.82	411	8.02
ppw-100 × 10-2b	263,876	239,525	9.23	38.08	247,263	6.30	7327.44	2913	7.27
ppw-100 × 10-3a	310,200	270,124	12.92	335.92	286,718	7.57	7422.85	101	9.98
ppw-100 × 10-3b	261,796	235,561	10.02	44.22	247,049	5.63	7556.61	215	8.61
ppw-200 × 10-1a	549,718	484,092	11.94	4623.58	484,115	11.93	7266.78	2	13.52
ppw-200 × 10-1b	445,802	402,796	9.65	1681.13	407,921	8.50	7351.46	3	11.13
ppw-200 × 10-2a	498,199	479,141	3.83	7215.22	479,141	3.83	7215.22	1	3.91
ppw-200 × 10-2b	423,031	409,470	3.21	1487.28	411,382	2.75	7361.75	2	2.87
ppw-200 × 10-3a	531,548	487,224	8.34	7201.81	487,224	8.34	7201.81	1	9.07
ppw-200 × 10-3b	402,130	389,005	3.26	4272.17	389,005	3.26	7229.17	1	4.44
Average			7.22	963.27		3.55	5644.80		4.21

Table 13
Results of the B&C algorithm on the set Nguyen.

Instance	z_{UB}	z_{lr}	Gap_{lr}	t_{lr}	z	Gap	t	$\#N$	Gap^*
25-5N	80,370	76,143.7	5.26	0.37	80,370	0.00	0.77	5	0.00
25-5Nb	64,562	63,587.4	1.51	0.17	64,562	0.00	0.26	1	0.00
25-5MN	78,947	72,436.5	8.25	0.31	78,947	0.00	0.93	13	0.00
25-5MNb	64,438	63,820.3	0.96	0.36	64,438	0.00	0.68	2	0.00
50-5N	137,815	132,439.0	3.90	2.55	137,815	0.00	1450.09	6849	0.20
50-5Nb	110,094	98,710.5	10.34	1.53	110,094	0.00	1195.50	3876	0.00
50-5MN	123,484	110,486.0	10.53	4.93	123,484	0.00	46.47	164	0.00
50-5MNb	105,401	101,104.0	4.08	3.14	105,401	0.00	47.16	278	0.00
50-10N	115,725	101,887.0	11.96	3.40	115,311	0.36	7318.03	13,902	0.46
50-10Nb	87,315	81,114.5	7.10	2.93	87,315	0.00	1378.04	11,481	0.00
50-10MN	135,519	122,272.0	9.78	3.79	135,519	0.00	79.13	452	0.028
50-10MNb	110,613	102,301.0	7.51	2.76	110,613	0.00	23.61	111	0.02
100-5N	193,228	174,725.0	9.58	158.82	183,570	5.00	7380.52	885	5.38
100-5Nb	158,927	145,960.0	8.16	20.89	154,145	3.01	7436.82	2594	3.07
100-5MN	204,682	176,807.0	13.62	69.39	194,968	4.75	7466.65	707	4.81
100-5MNb	165,744	153,433.0	7.43	18.38	160,221	3.33	7402.31	1272	3.40
100-10N	210,449	178,544.0	15.16	134.87	194,187	7.73	7478.19	301	10.22
100-10Nb	155,489	143,997.0	7.39	46.54	151,477	2.58	7461.93	1125	3.06
100-10MN	201,275	174,704.0	13.20	52.20	192,108	4.55	7467.13	1362	5.13
100-10MNb	170,625	151,322.0	11.31	32.12	166,010	2.70	7413.37	1144	2.91
200-10N	347,395	303,357.0	12.68	6452.22	303,357	12.68	7232.67	1	13.76
200-10Nb	256,171	231,195.0	9.75	1314.04	235,288	8.15	7452.34	12	8.94
200-10MN	324,006	290,392.0	10.37	2485.98	294,931	8.97	7356.73	7	10.60
200-10MNb	287,076	249,836.0	12.97	1385.72	259,079	9.75	7471.75	8	11.38
Average			8.87	508.23		3.07	4190.05		3.47

Table 14
Results of the B&C algorithm on the set I_1.

Instance	z_{UB}	z_{lr}	Gap_{lr}	t_{lr}	z	Gap	t	#N	Gap^*
I_1-8 × 3 × 2	575.70	575.70	0.00	0.00	575.70	0.00	0.01	1	0.00
I_1-8 × 4 × 2	549.34	534.37	2.72	0.02	549.34	0.00	0.04	2	0.00
I_1-9 × 3 × 2	878.69	874.49	0.48	0.03	878.69	0.00	0.05	1	0.00
I_1-10 × 4 × 2	806.72	752.79	6.69	0.03	806.72	0.00	0.06	1	0.00
I_1-10 × 5 × 3	696.94	660.28	5.26	0.07	696.94	0.00	0.11	3	0.00
I_1-10 × 8 × 3	596.56	538.76	9.69	0.18	596.56	0.00	0.37	26	0.00
I_1-15 × 10 × 2	732.48	672.82	8.14	0.51	732.48	0.00	1.37	32	0.00
I_1-15 × 10 × 3	686.71	601.28	12.44	0.31	686.71	0.00	1.01	46	0.00
I_1-15 × 4 × 2	1064.52	928.81	12.75	0.22	1064.52	0.00	0.56	44	0.00
I_1-15 × 5 × 3	933.75	829.52	11.16	0.14	933.75	0.00	0.39	22	0.00
I_1-15 × 8 × 3	730.36	667.28	8.64	0.21	730.36	0.00	0.67	33	0.00
I_1-20 × 10 × 2	937.40	817.88	12.75	0.52	937.40	0.00	390.76	14,546	0.00
I_1-20 × 10 × 3	761.29	690.44	9.31	0.88	761.29	0.00	2.26	18	0.00
I_1-20 × 10 × 4	1149.72	967.72	15.83	0.49	1149.72	0.00	68.05	4377	0.00
I_1-20 × 8 × 2	1029.59	874.66	15.05	1.45	1012.89	1.62	7292.83	34,556	1.62
I_1-20 × 8 × 3	848.31	787.66	7.15	0.32	848.31	0.00	1412.31	31,055	0.00
I_1-25 × 10 × 2	1030.40	855.45	16.98	1.99	1028.69	0.17	7260.62	53,013	0.17
I_1-25 × 10 × 3	1062.30	931.55	12.31	1.39	1049.68	1.19	7277.37	31,511	1.19
I_1-25 × 10 × 4	1607.94	1347.55	16.19	1.67	1535.00	4.54	7303.17	19,872	4.54
I_1-25 × 8 × 2	950.87	871.44	8.35	0.69	950.87	0.00	204.41	7542	0.00
I_1-25 × 8 × 3	870.69	775.31	10.96	1.18	870.69	0.00	251.24	6319	0.00
I_1-50 × 10 × 5	1132.63	935.55	17.40	52.19	1104.98	2.44	7411.64	4373	2.54
I_1-50 × 8 × 5	1162.44	991.17	14.73	29.03	1148.23	1.22	7410.61	5162	1.31
I_1-75 × 10 × 5	1540.23	1340.40	12.97	130.08	1433.29	6.94	7469.92	1479	6.98
I_1-75 × 15 × 5	1686.21	1427.27	15.36	128.32	1578.61	6.38	7502.19	532	7.16
I_1-100 × 10 × 5	2124.90	1854.75	12.71	788.34	1958.10	7.85	7480.91	53	8.36
I_1-100 × 20 × 5	1973.08	1690.46	14.32	561.42	1818.41	7.84	7442.96	153	8.30
I_1-150 × 10 × 5	1883.44	1594.73	15.33	2013.00	1711.06	9.15	7425.67	16	9.66
I_1-150 × 20 × 5	1869.53	1523.65	18.50	1872.83	1578.38	15.57	7360.21	11	16.45
I_1-200 × 10 × 5	2443.80	2188.39	10.45	5017.94	2190.89	10.35	7245.36	3	11.00
I_1-200 × 20 × 5	2219.54	1854.16	16.46	7208.84	1854.16	16.46	7208.84	1	17.15
Average			11.33	574.65		2.96	3400.84		3.11

Table 15
Results of the B&C algorithm on the set I_2.

Instance	z_{UB}	z_{lr}	Gap_{lr}	t_{lr}	z	Gap	t	#N	Gap^*
I_2-8 × 3 × 2	575.70	575.70	0.00	0.01	575.70	0.00	0.03	1	0.00
I_2-8 × 4 × 2	604.13	596.68	1.23	0.01	604.13	0.00	0.03	4	0.00
I_2-9 × 3 × 2	386.15	386.15	0.00	0.00	386.15	0.00	0.01	1	0.00
I_2-10 × 4 × 2	629.38	602.86	4.21	0.06	629.38	0.00	0.10	2	0.00
I_2-10 × 5 × 3	551.45	551.45	0.00	0.01	551.45	0.00	0.02	1	0.00
I_2-10 × 8 × 3	504.20	503.16	0.21	0.10	504.20	0.00	0.11	1	0.00
I_2-15 × 10 × 2	709.10	651.21	8.16	0.28	709.10	0.00	1.27	122	0.00
I_2-15 × 10 × 3	777.49	702.21	9.68	0.29	777.49	0.00	1.62	97	0.00
I_2-15 × 4 × 2	827.81	799.99	3.36	0.13	827.81	0.00	0.25	4	0.00
I_2-15 × 5 × 3	1075.22	1048.49	2.49	0.18	1075.22	0.00	0.36	8	0.00
I_2-15 × 8 × 3	652.58	591.31	9.39	0.28	652.58	0.00	1.52	216	0.00
I_2-20 × 10 × 2	766.24	684.32	10.69	0.51	766.24	0.00	4.71	300	0.00
I_2-20 × 10 × 3	744.27	688.77	7.46	0.42	744.27	0.00	1.74	56	0.00
I_2-20 × 10 × 4	793.00	735.41	7.26	0.40	793.00	0.00	1.04	22	0.00
I_2-20 × 8 × 2	772.29	698.42	9.57	0.37	772.29	0.00	5.93	584	0.00
I_2-20 × 8 × 3	758.06	679.76	10.33	0.30	758.06	0.00	6.05	682	0.00
I_2-25 × 10 × 2	961.59	857.44	10.83	1.72	954.19	0.77	7303.45	23,742	0.77
I_2-25 × 10 × 3	987.57	846.89	14.25	1.29	927.53	6.08	7266.33	26,128	6.08
I_2-25 × 10 × 4	1125.56	948.87	15.70	1.13	1125.56	0.00	2871.52	38,299	0.00
I_2-25 × 8 × 2	912.02	838.39	8.07	0.85	912.02	0.00	1267.23	36,505	0.00
I_2-25 × 8 × 3	979.62	908.59	7.25	0.59	979.62	0.00	8.18	739	0.00
I_2-50 × 10 × 5	1256.44	1117.28	11.08	9.31	1222.10	2.73	7420.76	4165	2.73
I_2-50 × 8 × 5	1121.13	990.19	11.68	7.15	1098.20	2.05	7389.18	7401	2.05
I_2-75 × 10 × 5	1691.15	1520.07	10.12	61.75	1601.08	5.33	7414.64	2874	5.33
I_2-75 × 15 × 5	1742.25	1477.08	15.22	91.61	1663.78	4.50	7531.82	488	4.85
I_2-100 × 10 × 5	2231.21	2020.73	9.43	776.75	2072.44	7.12	7413.89	192	7.57
I_2-100 × 20 × 5	1996.34	1778.54	10.91	219.90	1850.29	7.32	7563.92	252	8.34
I_2-150 × 10 × 5	1728.05	1463.78	15.29	464.69	1553.03	10.13	7575.30	52	10.47
I_2-150 × 20 × 5	1630.29	1423.48	12.69	597.70	1489.57	8.63	7460.51	76	9.99
I_2-200 × 10 × 5	2147.51	1912.36	10.95	2098.34	1930.59	10.10	7303.56	12	10.55
I_2-200 × 20 × 5	2049.01	1780.07	13.13	1459.93	1804.10	11.95	7418.46	17	13.33
Average			8.41	186.97		2.47	3007.53		2.65

Table 16
Results of the B&C algorithm on the set I_3.

Instance	z_{UB}	z_{lr}	Gap_{lr}	t_{lr}	z	Gap	t	$\#N$	Gap^*
I_3-8 × 3 × 2	578.33	539.21	6.76	0.03	578.33	0.00	0.04	2	0.00
I_3-8 × 4 × 2	450.71	422.33	6.30	0.04	450.71	0.00	0.05	1	0.00
I_3-9 × 3 × 2	454.63	454.63	0.00	0.01	454.63	0.00	0.01	1	0.00
I_3-10 × 4 × 2	540.61	525.79	2.74	0.04	540.61	0.00	0.08	6	0.00
I_3-10 × 5 × 3	745.48	681.98	8.52	0.05	745.48	0.00	0.13	16	0.00
I_3-10 × 8 × 3	412.91	412.91	0.00	0.02	412.91	0.00	0.03	1	0.00
I_3-15 × 10 × 2	605.11	539.46	10.85	0.40	605.11	0.00	1.26	71	0.00
I_3-15 × 10 × 3	546.61	495.00	9.44	0.46	546.61	0.00	1.63	160	0.00
I_3-15 × 4 × 2	688.87	636.67	7.58	0.04	688.87	0.00	0.27	59	0.00
I_3-15 × 5 × 3	1001.28	897.85	10.33	0.31	1001.28	0.00	1.06	88	0.37
I_3-15 × 8 × 3	578.23	520.55	9.97	0.33	578.23	0.00	0.86	36	0.00
I_3-20 × 10 × 2	744.82	692.91	6.97	0.52	744.82	0.00	5.25	414	0.00
I_3-20 × 10 × 3	728.17	648.27	10.97	0.40	728.17	0.00	4.17	230	0.00
I_3-20 × 10 × 4	1190.95	997.77	16.22	0.53	1190.95	0.00	273.39	8855	0.00
I_3-20 × 8 × 2	846.07	776.12	8.27	0.68	846.07	0.00	42.26	2518	0.00
I_3-20 × 8 × 3	643.89	593.92	7.76	0.45	643.89	0.00	1.67	67	0.00
I_3-25 × 10 × 2	834.23	768.98	7.82	1.24	834.23	0.00	73.44	4509	0.01
I_3-25 × 10 × 3	820.12	760.13	7.32	1.22	820.12	0.00	2.64	37	0.00
I_3-25 × 10 × 4	1057.63	873.48	17.41	0.45	1057.63	0.00	12.52	326	0.00
I_3-25 × 8 × 2	951.56	799.63	15.97	2.19	910.07	4.36	7304.93	22,689	4.36
I_3-25 × 8 × 3	774.36	711.48	8.12	1.01	774.36	0.00	144.79	6122	0.00
I_3-50 × 10 × 5	1207.31	1062.72	11.98	15.61	1179.15	2.33	7469.87	3565	2.33
I_3-50 × 8 × 5	1162.44	991.17	14.73	28.98	1148.23	1.22	7398.12	5151	1.41
I_3-75 × 10 × 5	1721.47	1528.74	11.20	48.96	1637.59	4.87	7394.83	3048	4.98
I_3-75 × 15 × 5	1483.14	1310.82	11.62	49.65	1411.30	4.84	7637.12	540	4.86
I_3-100 × 10 × 5	2178.35	1984.71	8.89	287.95	2057.15	5.56	7518.35	225	5.78
I_3-100 × 20 × 5	2035.37	1827.62	10.21	187.86	1889.23	7.18	7578.47	330	7.77
I_3-150 × 10 × 5	1274.44	1149.44	9.81	148.58	1198.77	5.94	7474.79	152	6.50
I_3-150 × 20 × 5	1235.86	1102.34	10.80	303.63	1162.72	5.92	7512.05	105	7.31
I_3-200 × 10 × 5	1766.46	1593.40	9.80	498.75	1658.02	6.14	7449.47	54	6.67
I_3-200 × 20 × 5	2531.21	2273.53	10.18	904.33	2304.52	8.96	7430.33	70	9.88
Average			9.31	80.15		1.85	2668.83		2.01

upper bounds which are difficult to improve. Columns labelled $\#N$ stand for the number of nodes inspected by the branch-and-cut algorithm. Finally, column labelled gap^* corresponds to the gap between the final lower bound obtained by the branch-and-cut method with respect to the average solution value of the ALNS.

References

[1] Baldacci R, Mingozzi A, Wolfler-Calvo R. An exact method for the capacitated location-routing problem. Operations Research 2011;59:1284–96.

[2] Belenguer JM, Benavent E, Prins C, Prodhon C, Wolfler-Calvo R. A branch-and-cut algorithm for the capacitated location routing problem. Computers and Operations Research 2010;38:931–41.

[3] Boccia M, Crainic TG, Sforza A, Sterle C. A metaheuristic for a two echelon location-routing problem. In: Festa P, editor. Experimental algorithms. Lecture notes in computer science, vol. 6049. Springer; 2010. p. 288–301.

[4] Boccia M, Crainic TG, Sforza A, Sterle C. Location-routing models for designing a two-echelon freight distribution system. Technical Report CIRRELT-2011-06, Université de Montréal; 2011.

[5] Clarke G, Wright JW. Scheduling of vehicles from a central depot to a number of delivery points. Operations Research 1964;12:568–81.

[6] 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.

[7] Contardo C, Cordeau J-F, Gendron B. A computational comparison of flow formulations for the capacitated location-routing problem. Technical Report CIRRELT-2011-47, Université de Montréal; 2011.

[8] 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.

[9] Crainic TG, Mancini S, Perboli G, Tadei R. Multi-start heuristics for the two-echelon vehicle routing problem. In: Merz P, Hao J-K, editors. Evolutionary computation in combinatorial optimization: proceedings of the 11th European conference (EvoCOP 2011), Torino, Italy, April 27–29, 2011. Lecture notes in computer science, vol. 6622; 2011. p. 179–90.

[10] Hemmelmayr VC, Cordeau J-F, Crainic TG. An adaptive large neighborhood search heuristic for two-echelon vehicle routing problems arising in city logistics. Technical Report CIRRELT-2011-42, Université de Montréal; 2011.

[11] Nguyen V-P, Prins C, Prodhon C. A multi-start iterated local search with tabu list and path relinking for the two-echelon location-routing problem. Engineering Applications of Artificial Intelligence 2012;25:56–71.

[12] Nguyen V-P, Prins C, Prodhon C. Solving the two-echelon location routing problem by a grasp reinforced by a learning process and path relinking. European Journal of Operational Research 2012;216:113–26.

[13] Perboli G, Tadei R. New families of valid inequalities for the two-echelon vehicle routing problem. Electronic Notes in Discrete Mathematics 2010;36:639–46.

[14] Perboli G, Tadei R, Vigo D. The two-echelon capacitated vehicle routing problem: models and math-based heuristics. Transportation Science 2011;45:364–80.

[15] Pirkwieser S, Raidl GR. Variable neighborhood search coupled with ILP-based very large-neighborhood searches for the periodic location-routing problem. In: Hybrid metaheuristics – seventh international workshop, HM 2010. Lecture notes in computer science, vol. 6373, Vienna; 2010. p. 174–89.

[16] Pisinger D, Ropke S. A general heuristic for vehicle routing problems. Computers and Operations Research 2007;34:2403–35.

[17] Prins C, Prodhon C, Ruiz A, Soriano P, Wolfler-Calvo R. Solving the capacitated location-routing problem by a cooperative Lagrangian relaxation-granular tabu search heuristic. Transportation Science 2007;41:470–83.

[18] Prins C, Prodhon C, Wolfler-Calvo R. A memetic algorithm with population management (MA|PM) for the capacitated location-routing problem. In: Gottlieb J, Raidl GR, editors. EvoCOP 2006. Lecture notes in computer science, vol. 3906. Berlin Heidelberg: Springer-Verlag; 2006. p. 183–94.

[19] Prins C, Prodhon C, Wolfler-Calvo R. Solving the capacitated location-routing problem by a GRASP complemented by a learning process and path relinking. 4OR – Quarterly Journal of Operations Research 2006;4:221–38.

[20] Ropke S, Pisinger D. An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. Transportation Science 2006;40:455–72.

[21] Schrimpf G, Schneider J, Stamm-Wilbrandt H, Dueck G. Record breaking optimization results using the ruin and recreate principle. Journal of Computational Physics 2000;159(2):139–71.

[22] Shaw P. Using constraint programming and local search methods to solve vehicle routing problems. In: Maher M, Puget J-F, editors. Principles and practice of constraint programming CP98. Lecture notes in computer science, vol. 1520. Berlin/Heidelberg: Springer; 1998. p. 417–31.

[23] Sterle C. Private communication; 2011.