EURO J Comput Optim (2018) 6:85–115
https://doi.org/10.1007/s13675-017-0089-z

ORIGINAL PAPER

New exact solution approaches for the split delivery vehicle routing problem

Gizem Ozbaygin[1] · **Oya Karasan**[1] ·
Hande Yaman[1]

Received: 11 July 2016 / Accepted: 5 September 2017 / Published online: 13 September 2017

Abstract In this study, we propose exact solution methods for the split delivery vehicle routing problem (SDVRP). We first give a new vehicle-indexed flow formulation for the problem and then a relaxation obtained by aggregating the vehicle-indexed variables over all vehicles. This relaxation may have optimal solutions where several vehicles exchange loads at some customers. We cut off such solutions, in a nontraditional way, either by extending the formulation locally with vehicle-indexed variables or by node splitting. We compare these approaches using instances from the literature and new randomly generated instances. Additionally, we introduce two new extensions of the SDVRP by restricting the number of splits and by relaxing the depot return requirement and modify our algorithms to handle these extensions.

Keywords Split delivery · Vehicle routing problem · Valid inequalities · Extended formulations · Exact approaches

Mathematics Subject Classification 90 (specifically 90-04 and 90-08) · 90B · 90C

✉ Hande Yaman
hyaman@bilkent.edu.tr

Gizem Ozbaygin
ozbaygin@bilkent.edu.tr

Oya Karasan
karasan@bilkent.edu.tr

[1] Department of Industrial Engineering, Bilkent University, Ankara, Turkey

1 Introduction

The split delivery vehicle routing problem (SDVRP) is a relaxation of the classical capacitated vehicle routing problem (CVRP), where the demand of a customer can be split and delivered by multiple vehicles. The task is to find a set of least cost delivery routes for a vehicle fleet starting and ending at the depot so that each customer belongs to at least one route, the demand of every customer is fully satisfied, and the total demand assigned to any (vehicle) route does not exceed the vehicle capacity. The SDVRP is formally defined by Dror and Trudeau (1989) with the motivation that permitting split deliveries can result in considerable transportation cost savings. The problem is shown to be NP-hard by Dror and Trudeau (1990), and despite being a relaxation of the classical CVRP, it is not easier to tackle as the amounts to be delivered to each customer by each vehicle is also unknown. In the past 25 years, several different exact and heuristic solution approaches as well as complexity-related analyses are proposed, and real-life problems are modeled and solved as variants of SDVRP. The first heuristic method is a two-stage local search algorithm developed by Dror and Trudeau (1989). The subsequent studies (Archetti et al. 2006; Boudia et al. 2007; Mota et al. 2007; Chen et al. 2007; Jin et al. 2008; Khmelev and Kochetov 2015; Aleman et al. 2010; Aleman and Hill 2010; Wilck IV and Cavalier 2012a, b; Silva et al. 2015; Berbotto et al. 2014) focus on hybrid methods and metaheuristics. Archetti and Speranza (2012) provides a comprehensive discussion on heuristics. Various other heuristic methods exist in the literature that are used for solving more special routing problems incorporating the split delivery option within their framework such as Yi and Kumar (2007), Gulczynski et al. (2010), Ozdamar and Demir (2012), Sahin et al. (2013), Chen et al. (2014), Wang et al. (2014).

The first exact algorithm to solve the SDVRP is a constraint relaxation branch-and-bound algorithm due to Dror et al. (1994). The problem is formulated as an integer linear program, and effective valid inequalities are derived. Branch and bound is used for achieving integrality, while the valid inequalities are added to cut off the solutions that are inadmissible for the SDVRP. In Sierksma and Tijssen (1998), the problem of scheduling helicopter flights to exchange crews is modeled as an SDVRP. The authors propose an integer linear programming formulation in which all feasible flight schedules are enumerated in advance and solve its linear relaxation by means of column generation. A similar column generation approach is suggested by Jin et al. (2008) for the SDVRP with large demands. The undirected version of SDVRP is considered by Belenguer et al. (2000). The authors provide an integer programming model and a relaxation of the SDVRP and prove that all constraints in this relaxation are facet-defining for the convex hull of the incidence vectors of the SDVRP solutions. Computational experiments with 25 instances indicate that their cutting plane approach can solve instances with up to 50 customers optimally. A dynamic program with finite state and action spaces is given in Lee et al. (2006). Test instances containing at most 9 customers are solved with this method. A two-stage algorithm with valid inequalities (TSVI) is introduced by Jin et al. (2007). The first stage creates clusters while respecting vehicle capacity restrictions, and establishes a lower bound on the optimal cost. The second stage computes an upper bound by solving a TSP on each cluster. TSVI iteratively executes these steps until the lower bound in the first stage and the upper bound from the second stage are equal and solve instances with up to 21 customers. Ceselli et al. (2009a) and Moreno et al. (2010) provide extended formulations to compute lower bounds for the SDVRP. In both studies, Dantzig–Wolfe decomposition principle is employed and column generation procedures are implemented to solve the resulting master problems. Computational experiments show that Moreno et al. (2010) can in general produce tighter lower bounds than Ceselli et al. (2009a).

The first branch-and-price-and-cut method for the SDVRP is developed by Archetti et al. (2011) applying a similar decomposition to Desaulniers (2010), who proposes a branch-and-price-and-cut technique for the SDVRP with time windows. The algorithm is tested on a large set of benchmark instances for both limited and unlimited vehicle fleet cases. The majority of the best available lower bounds and some of the best available upper bounds are improved. Although one instance with 144 customers is solved to optimality, the second largest instance optimized contains 48 customers. Two exact branch-and-cut solution methodologies are given in Archetti et al. (2014) where the optimality of 17 instances in the literature and a new instance involving 100 customers is established.

In this paper, we propose a new arc flow formulation for the SDVRP that uses variables with vehicle indices. To decrease the size and to eliminate the symmetry, we aggregate the variables over all vehicles. This resulting relaxation is similar to one of the relaxations in Archetti et al. (2014). We give a family of valid inequalities that includes the generalized capacity inequalities of Belenguer et al. (2000) as a special case and show that these inequalities are not sufficient to obtain a formulation. To eliminate solutions of the relaxation infeasible for SDVRP, we propose to locally extend the relaxation either by adding vehicle-indexed variables for some customer nodes or by node splitting. Our computational experiments reveal that iterating for an optimal solution of the SDVRP with our methods can be performed effectively as long as the relaxation can be solved effectively.

Though split deliveries save costs, they come at the expense of additional handling time. We introduce the problem SDVRP with restricted number of splits to the literature. We extend our exact solution methodologies to solve this variant. Against intuition, it is not any easier to solve this restricted version of the SDVRP. Another variation we handle is the SDVRP with open routes where the depot return requirement is relaxed. Though some theoretical results no longer are valid for this variation, our computational experiments reveal favorable results.

The rest of this paper is organized as follows. In Sect. 2, the arc flow formulation and its relaxed version are presented along with some simplifications for the relaxation. We propose a family of valid inequalities that generalize the generalized capacity inequalities of Belenguer et al. (2000) and give an example where these inequalities cannot cut off the optimal solution of the relaxation that is infeasible for the SDVRP. In Sect. 3, the methods to eliminate the optimal solutions of the relaxed model that are not feasible for the SDVRP as well as the exact solution algorithms are introduced. The results of the computational experiments are given in Sect. 4. Section 5 is reserved for the two extensions of SDVRP along with their computational results. Section 6 provides insights into the behavior of the proposed algorithms. Finally, Sect. 7 provides a summary of our findings.

2 Formulation, relaxation and valid inequalities

Let $G = (N, A)$ be a directed complete graph with the set of nodes $N = \{0, 1, \dots, n\}$ and the set of arcs $A = \{a = (i, j) : i, j \in N, i \neq j\}$. Suppose that the depot is located at node 0 and each node $i \in N\backslash\{0\}$ represents a customer location. There are m identical vehicles available at the depot to serve the customers, each having a capacity of Q units. We define $K = \{1, \dots, m\}$. The cost of traversing arc $a \in A$ is c_a and the demand of customer $i \in N\backslash\{0\}$ is $0 < d_i \leq Q$. We assume that the costs are nonnegative and they satisfy the triangle inequality.

2.1 An exact flow-based formulation with vehicle indices

We first present a flow-based formulation with vehicle indices. We use the following decision variables:

- $y_a^k = \begin{cases} 1 & \text{if vehicle } k \in K \text{ travels on arc } a \in A, \\ 0 & \text{otherwise,} \end{cases}$
- g_a^k = the amount of flow carried on arc $a \in A$ by vehicle $k \in K$,
- w_i^k = fraction of the demand of customer $i \in N\backslash\{0\}$ delivered by vehicle $k \in K$.

For a given set $S \subset N$, let $\delta^-(S)$ denote the set of arcs (i, j) with $i \in N\backslash S$ and $j \in S$ and $\delta^+(S)$ denote the set of arcs (i, j) with $i \in S$ and $j \in N\backslash S$. We use $\delta^-(i)$ and $\delta^+(i)$ for $\delta^-(\{i\})$ and $\delta^+(\{i\})$. For a vector $\alpha \in R^{|U|}$ and $U' \subseteq U$, we let $\alpha(U') = \sum_{u \in U'} \alpha_u$.

(SDVRP)

$$\min \sum_{a \in A} \sum_{k \in K} c_a y_a^k \tag{1}$$

$$g^k(\delta^-(i)) - g^k(\delta^+(i)) = d_i w_i^k \qquad i \in N\backslash\{0\}, k \in K, \tag{2}$$

$$y^k(\delta^-(i)) - y^k(\delta^+(i)) = 0 \qquad i \in N\backslash\{0\}, k \in K, \tag{3}$$

$$y^k(\delta^+(0)) = 1 \qquad k \in K, \tag{4}$$

$$\sum_{k \in K} w_i^k = 1 \qquad i \in N\backslash\{0\}, \tag{5}$$

$$g_a^k \leq Q y_a^k \qquad a \in A, k \in K, \tag{6}$$

$$w_i^k \geq 0 \qquad i \in N\backslash\{0\}, k \in K, \tag{7}$$

$$g_a^k \geq 0 \qquad a \in A, k \in K, \tag{8}$$

$$y_a^k \in \{0, 1\} \qquad a \in A, k \in K. \tag{9}$$

The objective function (1) aims to minimize the global transportation cost. Constraints (2) and (3) require commodity flow and vehicle flow conservation for every customer

Table 1 Some results with the vehicle-indexed model

Instance	Number of nodes	Number of vehicles	Lower bound	Upper bound	Gap (%)	Time (s)	Nodes in b&c tree
eil22	22	4	375	375	0	108.57	115,403
eil23	23	3	569	569	0	9.87	14,475
eil30	30	3	510	510	0	2149.89	1,065,899
eil33	33	4	819.64	835	1.84	7200	1,364,276

and for every vehicle. Constraint (4) forces all the vehicles to leave the depot for service, and (5) guarantees that the demand of each customer is fully satisfied. Constraint (6) is the coupling constraints ensuring that the flow on an arc carried by a vehicle does not exceed the vehicle capacity. Finally, (7)–(9) specify variable restrictions.

The formulation given in (1)–(9) contains $O(n^2m)$ variables and $O(n^2m)$ constraints. Due to its large size and due to the symmetry induced by the homogeneous fleet of vehicles, it can be solved to optimality for small size problems. Table 1 shows our results with a time bound of 7200 s regarding the four smallest instances in Belenguer et al. (2000). It can be observed that the number of nodes in the branch-and-cut tree is quite large even for these instances.

2.2 A flow-based relaxation

In this section, we present a relaxed model obtained by aggregating the decision variables over all vehicles, i.e., by letting $f_a = \sum_{k\in K} g_a^k$ and $x_a = \sum_{k\in K} y_a^k$ for every arc $a \in A$. Our aim is to decrease the size of the vehicle-indexed formulation and to eliminate symmetry. The relaxed model is as follows:

(R-SDVRP)

$$\min \sum_{a\in A} c_a x_a \tag{10}$$

$$\text{s.t. } f(\delta^-(i)) - f(\delta^+(i)) = d_i \qquad i \in N\backslash\{0\}, \tag{11}$$

$$x(\delta^-(i)) - x(\delta^+(i)) = 0 \qquad i \in N\backslash\{0\}, \tag{12}$$

$$x(\delta^+(0)) = m, \tag{13}$$

$$f_a \le Qx_a \qquad a \in A, \tag{14}$$

$$f_a \ge 0 \qquad a \in A, \tag{15}$$

$$x_a \in \mathbb{Z}_+ \qquad a \in A. \tag{16}$$

Similar to the exact model, the objective is to minimize the total cost of transportation. Constraint (11) ensures that the demand of every customer is fulfilled. Vehicle flow conservation is enforced by constraint (12), and constraint (13) guarantees that exactly m vehicles are dispatched from the depot for service. Constraint (14) relates variables x_a and f_a based on the vehicle capacity. Domain restrictions on the decision variables are imposed by (15) and (16).

2.3 An optimality property

A *k-split cycle* is defined in Dror and Trudeau (1989) as a subgraph on a set of customers $i_1, \ldots, i_k \subset N\backslash\{0\}$ with $k \geq 2$ in which there exist $1 \leq h \leq k$ vehicle routes such that i_t and i_{t+1} are on the same route for every $t = 1 \ldots, k-1$ and that i_1 and i_k are on the same route. Accordingly, the authors establish the *k-split cycle property*, which guarantees the existence of an optimal SDVRP solution free of *k-split cycles* for any $k \geq 2$ under the condition that the cost matrix satisfies the triangle inequality. Based on the k-split cycle property, we can impose binary requirements on the x_a variables for arcs a with customers at both endpoints, i.e., $a \in A\backslash(\delta^-(0) \cup \delta^+(0))$. This helps in reducing computation times. In Proposition 2.1, we prove that if the costs are symmetric, then we can also restrict the x variables associated either with the arcs originating from the depot or with those ending at the depot to take 0–1 values. Based on initial computational trials, we prefer to apply the restriction to the arcs emanating from the depot.

Proposition 2.1 *If the costs are symmetric and if they satisfy the triangle inequality, then there exists an optimal SDVRP solution x for which $x_a \in \{0, 1\}$ for all $a \in A\backslash\delta^-(0)$.*

Proof First note that since the cost matrix is symmetric, one can reverse the direction of any route and attain an alternative optimal solution. Also, there exists an alternative optimal solution in which a customer on a dedicated route, i.e., a route with a single customer, is visited only once. To show this, assume that i is a customer who is visited by routes C_1 and C_2 where C_1 is a dedicated route. Since $d_i \leq Q$ and the costs satisfy triangle inequality, it is possible to attain another solution with the same cost by excluding i from route C_2.

Assume that x is an optimal solution to a given SDVRP instance that is free of k-split cycles (for any $k \geq 2$) and that customers receiving dedicated service are not split nodes. If $x_{0i} \leq 1$ for every customer i, then we are done. Otherwise, we shall iteratively construct another optimal solution satisfying the proposed condition. Take a customer i for which $x_{0i} = \mu$, where $\mu \geq 2$. Pick any one of these μ routes, say C_1, and let j_1 be the last customer on this route (where i is the first customer). Note that $j_1 \neq i$ otherwise customer i would be a split node receiving dedicated service. If $x_{0j_1} = 0$, then reversing the direction of route C_1 will result in an alternative optimal solution with x_{0i} decremented and no x_a for $a \in \delta^+(0)$ incremented beyond value 1. Otherwise, let $C_1, \ldots, C_l$ be a sequence of routes such that for any two consecutive routes C_t and C_{t+1} for $t = 1, \ldots, l-1$, the last customer of C_t and the first customer of C_{t+1} are identical. Moreover, let l be the largest possible such number. Consider any two routes C_p and C_q such that $q > p+1$. These two routes cannot intersect, otherwise routes $C_p, C_{p+1}, \ldots, C_q$ will constitute a $(q-p+1)$-split cycle and we know that the optimal solution is free of such cycles. In particular, this implies that if j_l is the last customer in route C_l, then $x_{0j_l} = 0$, otherwise we violate either the fact that l is not the largest possible consecutive route number or that there is no k-split cycle. Now, reversing all the routes $C_1, \ldots, C_l$ will result in an optimal solution with x_{0i} decremented and no x_a for $a \in \delta^+(0)$ incremented beyond value 1. Repeating this procedure for every customer i with $x_{0i} \geq 2$, an alternative optimal solution can be attained in which $x_a \in \{0, 1\}$ for all $a \in A \setminus \delta^-(0)$. □

2.4 Comparison with existing relaxations

Next, we compare our relaxed model to other relaxed models in the literature. A similar model to R-SDVRP is given by Archetti et al. (2014). Different from our model, Archetti et al. (2014) do not force all vehicles to be used. They use additional variables to keep the number of visits to each node and put upper bounds on these variables. Using the k-split cycle property, they restrict the variables associated with the arcs between customer pairs to be 0–1. In addition, they force the flow on return arcs to the depot to be zero.

Note that if we project out the flow variables in R-SDVRP, we obtain the fractional capacity inequalities

$$x(\delta^-(S)) \geq \frac{d(S)}{Q} \tag{17}$$

for all $S \subseteq N \setminus \{0\}$ [see Gouveia (1995) and Letchford and Salazar-González (2006) for more projection results]. Hence, R-SDVRP is equivalent to a directed version of the relaxation used by Belenguer et al. (2000). These authors depict a solution of their relaxation for the instance *eil30* which is not feasible for SDVRP. In Fig. 1, we depict the solution found by solving our relaxation. We obtain the same solution, but we also have the flow values on the arcs. We report these values for the arcs adjacent to node 18. Three vehicles visit node 18, one of which is empty upon arrival while the other two are not. The empty vehicle returns to the depot after passing through node 18, while the nonempty vehicles arrive at node 18 with 4500 and 625 units of load and leave the node with 3175 and 1800 units of load, respectively. This can only be possible if 1175 units of load is unloaded from the first vehicle and loaded on the second vehicle, while the vehicles are at node 18. This is not admissible for the SDVRP.

2.5 Framed capacity inequalities

In Belenguer et al. (2000), the authors propose to cut off the infeasible solution given in Fig. 1 using a valid inequality. In this section, we present a family of valid inequalities that generalizes the inequalities used by Belenguer et al. (2000). These inequalities are called “framed capacity inequalities,” and their undirected variants are proposed for the CVRP [see, e.g., the review by Naddef and Rinaldi (2002)].

Proposition 2.2 *Let $H \subseteq N \setminus \{0\}$ and $S_1, \ldots, S_t$ be disjoint nonempty subsets of H. Define $b(S_1, \ldots, S_t)$ to be the optimal value of the bin packing problem with items $1, \ldots, t$ of size $d(S_1), \ldots, d(S_t)$ (if there exists u with $d(S_u) > Q$, then as done by Belenguer et al., we consider the demand of set S_u to be $d(S_u) - \left\lfloor \frac{d(S_u)}{Q} \right\rfloor Q$ in the bin packing problem and add $\left\lfloor \frac{d(S_u)}{Q} \right\rfloor$ to the bin packing value). The framed capacity inequality*

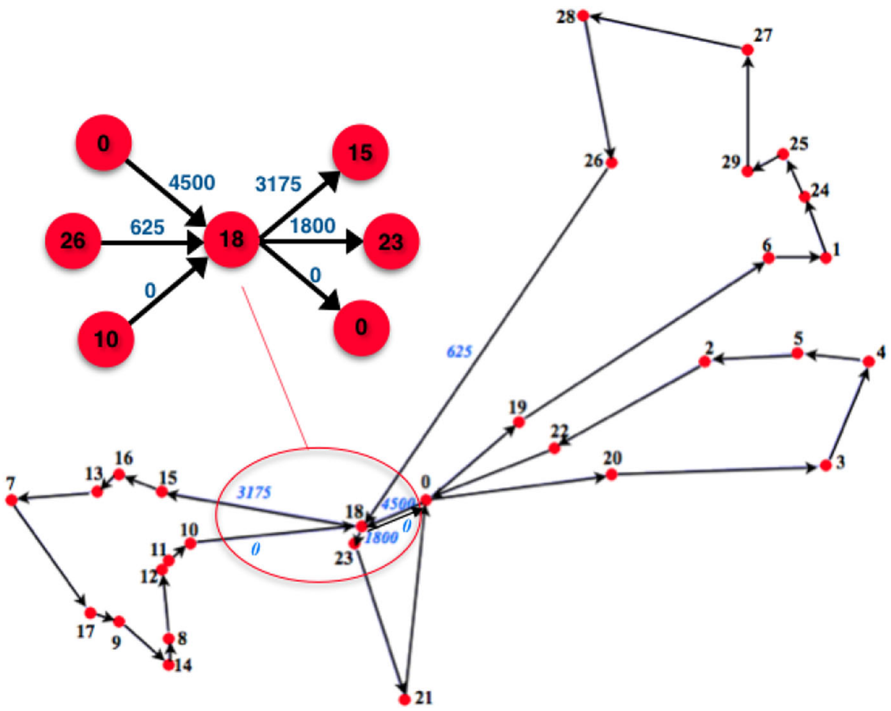


Fig. 1 The optimal solution of R-SDVRP for $eil30$

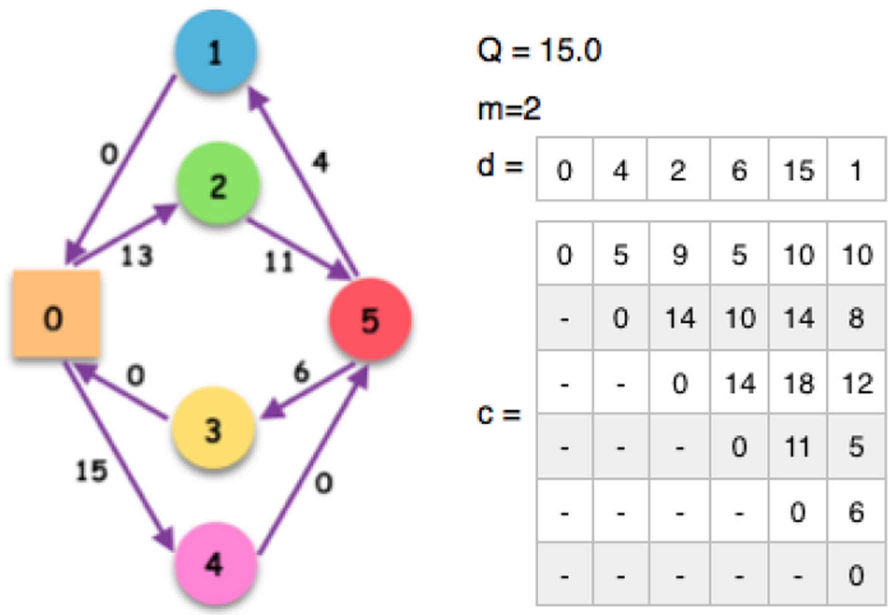
$$x(\delta^-(H)) + \sum_{u=1}^{t} x(\delta^-(S_u)) \geq \sum_{u=1}^{t} \left\lceil \frac{d(S_u)}{Q} \right\rceil + b(S_1, \ldots, S_t) \tag{18}$$

is valid for the feasible set of SDVRP.

Proof If $x(\delta^-(S_u)) = \left\lceil \frac{d(S_u)}{Q} \right\rceil$ for all $u = 1, \ldots, t$, then we need at least $b(S_1, \ldots, S_t)$ vehicles to enter set H to satisfy the demand of $\cup_{u=1}^{t} S_u$. Hence $x(\delta^-(H)) \geq b(S_1, \ldots, S_t)$. Since each split in S_u can reduce the number of required vehicles by at most 1, the result follows. □

Note that, for the CVRP, the bin packing value is computed using all customers in H. In our case, if $b(S_1, \ldots, S_t) \leq \left\lceil \frac{d(H)}{Q} \right\rceil$, then the inequality is dominated by the sum of rounded capacity inequalities $x(\delta^-(H)) \geq \left\lceil \frac{d(H)}{Q} \right\rceil$ and $x(\delta^-(S_u)) \geq \left\lceil \frac{d(S_u)}{Q} \right\rceil$ over all $u = 1, \ldots, t$. If $b(S_1, \ldots, S_t) > \left\lceil \frac{d(H)}{Q} \right\rceil$, considering all customers of H by letting splits for the ones in $H \backslash \cup_{u=1}^{t} S_u$ does not change the result of the bin packing problem since $b(S_1, \ldots, S_t)Q > d(H)$.

The inequalities used by Belenguer et al. (2000) are special cases of inequality (18) with $H = V \backslash \{0\}$ and consequently $x(\delta^-(H)) = m$.



Q = 15.0

m=2

d =

0	4	2	6	15	1

c =

0	5	9	5	10	10
-	0	14	10	14	8
-	-	0	14	18	12
-	-	-	0	11	5
-	-	-	-	0	6
-	-	-	-	-	0

Fig. 2 An optimal solution to R-SDVRP that cannot be cut off by any framed capacity inequality

Next, we show with an example that even if we include all framed capacity inequalities into our relaxed model, the resulting model is still a relaxation and may have optimal solutions that are not feasible for the SDVRP. In other words, there exist optimal R-SDVRP solutions that are not admissible for the SDVRP, yet cannot be eliminated using any framed capacity inequality. Such a solution is depicted in Fig. 2 along with the cost matrix associated with the problem instance. The demands are $d_1 = 4$, $d_2 = 2$, $d_3 = 6$, $d_4 = 15$ and $d_5 = 1$. There are two vehicles, each with a capacity of 15 units. The number on each arc corresponds to its flow value. Notice that a load exchange takes place between the vehicles at node 5. The total cost associated with this solution is 60, while the optimal SDVRP solution has cost 61. Therefore, there does not exist an optimal SDVRP solution using the edges in this R-SDVRP solution.

First note that the bin packing value cannot be larger than 2 for all possible subsets H and $S_1, \ldots, S_t$. If $x(\delta^-(H)) \geq 2$, then as $b(S_1, \ldots, S_t) \leq 2$ and $x(\delta^-(S_u)) \geq \left\lceil \frac{d(S_u)}{Q} \right\rceil$ for $u = 1 \ldots, t$, inequality (18) is satisfied. Now for $x(\delta^-(H)) = 1$, we need $H \subset N\backslash\{0, 5\}$ and $|H| = 1$. Then, $S_1 = H$ or $S_1 = \emptyset$, and accordingly, the bin packing value $b(S_1)$ is 1 or 0 and the inequality is again satisfied. Hence, the framed capacity inequalities fail to omit the solution in this example from the feasible set of the R-SDVRP.

2.6 Rounded capacity and cutset inequalities

To conclude this section, we describe two classes of valid inequalities that are employed for strengthening our relaxation. Let $\mathcal{Y}$ be the feasible set of the R-SDVRP and $S \subseteq N\backslash\{0\}$. The rounded capacity inequality

$$x(\delta^-(S)) \geq \left\lceil \frac{d(S)}{Q} \right\rceil \tag{19}$$

is valid for $\mathcal{Y}$.

Now consider the relaxation

$$f(\delta^-(S)) - f(\delta^+(S)) = d(S), \tag{20}$$

$$0 \leq f_a \leq Qx_a \quad a \in \delta^-(S) \cup \delta^+(S), \tag{21}$$

$$x_a \in \mathbb{Z}_+ \quad a \in \delta^-(S) \cup \delta^+(S). \tag{22}$$

The convex hull of the solutions of the above set is defined by trivial inequalities and the following cutset inequalities (see Atamtürk 2002). Let $A^- \subseteq \delta^-(S)$, $A^+ \subseteq \delta^+(S)$, $\eta = \left\lceil \frac{d(S)}{Q} \right\rceil$ and $r = d(S) - \left\lfloor \frac{d(S)}{Q} \right\rfloor Q$. The cutset inequality is

$$f(\delta^-(S)\backslash A^-) + rx(A^-) + (Q-r)x(A^+) - f(A^+) \geq r\eta \tag{23}$$

and is valid for $\mathcal{Y}$. If $A^- = \delta^-(S)$ and $A^+ = \emptyset$, the cutset inequality reduces to the rounded capacity inequality.

3 New exact methods for the SDVRP

Two novel iterative algorithms are devised for solving the SDVRP to optimality. Essentially, the mechanism behind both algorithms is the same. First, an optimal solution (f^*, x^*) of the R-SDVRP is obtained. If the solution (f^*, x^*) is feasible for SDVRP, then it is also an optimal SDVRP solution. Otherwise, new variables and constraints are added to the formulation R-SDVRP such that when the new variables are projected out, some portion of $\mathcal{Y}$, including the vector (f^*, x^*), is cut off. The relaxation is solved again over a more constrained region. This process continues iteratively until an optimal SDVRP solution is found. The two methods are distinguished by the routines they use for eliminating the solution (f^*, x^*) at every iteration. Before elaborating more on these routines, we describe what we refer to as the *regularity property*.

Definition (*Regularity property*) A feasible solution of R-SDVRP possesses the regularity property, or equivalently, it is called regular, if for any node $i \in N\backslash\{0\}$, the following holds:

$$f^-(i,j) \geq f^+(i,j) \quad \text{for all } j = 1, \ldots, i_n,$$

where i_n is the number of vehicles passing through node i, $f^-(i,j)$ and $f^+(i,j)$ are the amounts of the jth largest incoming and outgoing flows associated with the node i, respectively.

Note that the regularity of an R-SDVRP solution can be established in $O(m^2 \log m)$ time since there can be at most $m-1$ split nodes (see Archetti et al. 2008), and for each one, ordering the incoming and outgoing flow values takes at most $O(m \log m)$

time. For a node i for which $x_{i0} > 1$, f_{i0} should be decomposed into x_{i0} flow values having the potential of satisfying the regularity property which can easily be handled within the same time complexity.

Archetti et al. (2014) prove that if an optimal solution of the R-SDVRP has the regularity property, then it solves SDVRP optimally. This result establishes an equivalence between the regular R-SDVRP solutions and the feasible SDVRP solutions. Given a solution to the R-SDVRP, one can check in polynomial time whether it is regular and thus feasible for the SDVRP. However, deciding on the regularity of an R-SDVRP solution is different from checking whether a given solution x is feasible for the SDVRP, which is shown to be NP-complete by Belenguer et al. (2000).

We can construct an optimal SDVRP solution from an optimal regular solution (f, x) of the R-SDVRP in the following way. Consider the arcs in the corresponding support graph; that is, the arcs $a \in A$ with $x_a \geq 1$. We shall apply depth first search traversals in the support graph in order to construct m viable routes. Start each route with an arc emanating from the depot and perform depth first search making sure at every node among the potential outgoing arcs, the one having the largest flow value less than or equal to the flow value of the used incoming arc is selected. For a node i such that $x_{i0} \geq 2$, such a route extension is not that obvious. Suppose our traversal enters such a node i using arc (j, i). We shall split arc $(i, 0)$ into x_{i0} identical arcs. The route will be completed by choosing one of these arcs with flow value as $\min(f_{ji}, f_{i0})$. Now, take out this constructed route from the support graph, update the demands and the flow values on multiple arcs entering the depot and repeat the same steps for another route. Note that our arc selection preserves regularity and after m steps we construct an optimal solution for the SDVRP. Since the support graph has at most nm arcs and since each arc can be visited at most m times during our traversals, the complexity of this algorithm is $O(nm^2)$.

In the following subsections, the details of the exact solution methods we propose are discussed.

3.1 Patching algorithm

Even though our vehicle-indexed flow formulation is not computationally efficient, it may be reasonable to use vehicle indices, at least for some arcs, to be able to find an optimal SDVRP solution by solving a relaxation. The patching algorithm is based on the idea of locally extending the R-SDVRP formulation with vehicle-indexed variables when needed. More precisely, at each iteration of the algorithm, a node violating the regularity property is identified and vehicle-indexed variables are introduced associated with the arcs incident to this node. These variables allow us to formulate the constraints necessary to enforce the regularity at this node. The steps of the patching algorithm are given below.

Step 0. Initialization: Solve the R-SDVRP, and let $(\bar{f}, \bar{x})$ denote the optimal solution found. Set current solution to $(\bar{f}, \bar{x})$.

Step 1. Check the regularity of the current solution. If it is regular, stop. The current solution is optimal for the SDVRP.

Step 2. Let $\bar{G} = (N, \bar{A})$ represent the support graph corresponding to the current solution; i.e., the graph induced by the arcs (i, j) for which $\bar{x}_{ij} \geq 1$. Update $\bar{G}$ by adding it the arcs (j, i) for all $(i, j) \in \bar{A}$, and solve the exact (vehicle-indexed) SDVRP formulation on the updated graph $\bar{G}$. If a feasible solution exists, stop; it is optimal for the SDVRP.

Step 3. Among the nodes violating the regularity of the current solution, select the first one encountered during the regularity check. Denote this node by i^*. Add vehicle-indexed variables for the arcs in $\delta^-(i^*) \cup \delta^+(i^*)$, and introduce the following set of constraints to the model solved in the previous iteration.

$$g^k(\delta^-(i^*)) - g^k(\delta^+(i^*)) \geq 0 \qquad k \in K, \tag{24}$$

$$y^k(\delta^-(i^*)) - y^k(\delta^+(i^*)) = 0 \qquad k \in K, \tag{25}$$

$$y^k(\delta^-(i^*)) \leq 1 \qquad k \in K, \tag{26}$$

$$\sum_{k \in K} g_a^k = f_a \qquad a \in \delta^-(i^*) \cup \delta^+(i^*), \tag{27}$$

$$\sum_{k \in K} y_a^k = x_a \qquad a \in \delta^-(i^*) \cup \delta^+(i^*), \tag{28}$$

$$g_a^k \leq Q y_a^k \qquad a \in \delta^-(i^*) \cup \delta^+(i^*), k \in K, \tag{29}$$

$$g_a^k \geq 0, y_a^k \in \{0, 1\} \qquad a \in \delta^-(i^*) \cup \delta^+(i^*), k \in K. \tag{30}$$

Constraint (24) forces the regularity property at node i^*, and constraint (25) ensures that vehicle flow is conserved at this node. Constraint (26) prevents node i^* from being visited more than once by the same vehicle. The vehicle-indexed variables g_a^k and y_a^k are linked to the original decision variables f_a and x_a with constraints (27) and (28), respectively. Constraint (29) set the upper bounds on the flows for the arcs in $\delta^-(i^*) \cup \delta^+(i^*)$. Finally, nonnegativity and binary requirements for the new variables are given by (30).

Step 4. Solve the modified model and update the current solution accordingly. Return to step 1.

The patching algorithm guarantees convergence to an optimal solution of the SDVRP by fixing the regularity violation for at least one node from one iteration to another. Adding vehicle-indexed variables and regularity-related restrictions for a node makes it possible to distinguish between different vehicles visiting the node and prevents load exchanges. Although the R-SDVRP grows in terms of the number of variables and constraints with the increasing number of iterations, as our computational results in Sect. 4 will attest to, this algorithm is capable of reaching the optimum much faster compared to the vehicle-indexed model, which can be seen by comparing the results in Table 1 to those that are provided in Tables 2 and 3.

3.2 Node-split algorithm

The patching algorithm adds vehicle-indexed variables for all vehicles at a node violating regularity. In most practical cases, the demand of a node is split among two or three

Table 2 Results for the instances taking single iteration for the SDVRP

Instance	Number of nodes	Number of vehicles	Best-known upper bound	Lower bound	Upper bound	Gap (%)	Time (s)
eil22	22	4	375	375	375	0	3.07
eil23	23	3	569	569	569	0	1.56
eil33	33	4	835	835	835	0	19.09
eil51	51	5	521	521	521	0	264.98
eilA76	76	10	818	777.42	–	–	7200
eilB76	76	14	1002	941.25	–	–	7200
eilC76	76	8	733	709.14	–	–	7200
eilD76	76	7	681	657.46	–	–	7200
S51D1	51	3	458	458.00	458	0	21.68
S51D2	51	9	703	682.01	–	–	7200
S51D3	51	15	943	911.64	945	3.53	7200
S51D4	51	27	1551	1504.67	1555	3.24	7200
S51D5	51	23	1328	1297.37	1329	2.38	7200
S51D6	51	41	2163	2093.05	**2153**	2.78	7200
S76D1	76	4	592	592	592	0	1728.26
S76D2	76	15	1081	1011.45	–	–	7200
S76D3	76	23	1419	1349.64	–	–	7200
S76D4	76	37	2071	1979.51	–	–	7200
SD1	9	6	228	228	228	0	0.03
SD2	17	12	708	708	708	0	0.38
SD3	17	12	432	432	432	0	0.11
SD4	25	18	630	630	630	0	0.44
SD5	33	24	1392	1392	1392	0	6137.18
SD6	33	24	832	832	832	0	4.32
SD7	41	30	3640	3484.12	–	–	7200
SD8	49	36	5068	4790.15	–	–	7200
SD9	49	36	2046	2005.48	2046	1.98	7200
SD10	65	48	2688	2620.33	2696	2.81	7200
p01-110	51	3	458	458	458	0	21.97
p01-1030	51	11	753	722.38	755	4.32	7200
p01-1050	51	16	998	969.97	998	2.81	7200
p01-1090	51	26	1481	1440.76	**1480**	2.65	7200
p01-3070	51	26	1473	1433.04	1478	3.04	7200
p01-7090	51	41	2212	2075.84	**2142**	3.09	7200
p02-110	76	5	612	599.56	–	–	7200
p02-1030	76	16	1157	1044.54	–	–	7200
p02-1050	76	24	–	1433.98	–	–	7200
p02-1090	76	40	–	2212.47	–	–	7200
p02-3070	76	39	–	2133.99	–	–	7200
p02-7090	76	61	–	3103.35	**3205**	3.17	7200

Table 3 Results for the instances taking multiple iterations for the SDVRP

Instance	Number of nodes	Number of vehicles	Best-known upper bound	Patching algorithm				Number of iterations
				Lower bound	Upper bound	Gap (%)	Time (s)	
eil30	30	3	510	510	510	0	30.58	3
r1	30	4	–	708	708	0	105.41	2
r2	36	3	–	398	398	0	857.07	4
r3	36	4	–	421	421	0	698.62	2
r4	41	3	–	410	410	0	1033.69	3
r5	48	3	–	37,025	–	–	7200	3
Instance	**Number of nodes**	**Number of vehicles**	**Best-known upper bound**	**Node-split algorithm**				**Number of iterations**
				Lower bound	Upper bound	Gap (%)	Time (s)	
eil30	30	3	510	510	510	0	183.55	4
r1	30	4	–	708	708	0	27.22	2
r2	36	3	–	398	398	0	465.49	5
r3	36	4	–	421	421	0	115.08	2
r4	41	3	–	410	410	0	382.43	2
r5	48	3	–	37,234	37,234	0	3686.20	4

vehicles. Hence, by patching, we may use unnecessary variables and constraints. The node-split method provides a way to make a distinction between the vehicles visiting a certain node without using vehicle-indexed variables. It is similar to the patching algorithm in the following respects: (1) the R-SDVRP is solved at the initialization step, (2) an extended version of the R-SDVRP obtained by adding new variables and constraints is solved at each iteration, and (3) regularity violations are detected and eliminated iteratively until an optimal SDVRP solution is obtained. However, it differs from the patching algorithm in terms of the approach adopted for enforcing the regularity property at a violating node.

The idea of the node-split algorithm is to create duplicates of the nodes violating regularity and to constrain the net incoming flow to each such node and every one of its duplicates to take nonnegative values. Duplicating a certain node provides means to decompose the flow carried on the incoming arcs of the original node and the flow carried on its outgoing arcs into distinct vehicles. Note that the network associated with the original problem is enlarged every time a node is duplicated because both the number of nodes and the number of arcs increase. Hence, after a number of iterations, a regular solution is found on an extended network, for which the corresponding solution on the original network can be obtained simply by merging each node with its duplicates (if there is any).

We present the generic version of the model solved at each iteration of the node-split algorithm below along with some additional notation. Suppose that $N' = \cup_{i \in N \setminus \{0\}} N_i$, where N_i represents the set of nodes containing node $i \in N$ and its duplicates. Let $A' = \{(k,l) : \exists (i,j) \in A, k \in N_i \text{ and } l \in N_j\} \cup \{(0,i) \cup (i,0) : i \in N'\}$ and $\bar{c}_{kl} = c_{ij}$ if $k \in N_i$ and $l \in N_j$. Similarly, let $\bar{c}_{0k} = c_{0i}$ and $\bar{c}_{k0} = c_{i0}$ if $k \in N_i$. Assume that N_i is ordered so that a node $j \in N_i$ is denoted by (i,l), where l is the order of node j in the set N_i. Also, define:

$$v_{i,l} = \begin{cases} 1 & \text{if node } (i,l) \in N' \text{ is visited,} \\ 0 & \text{otherwise.} \end{cases}$$

(Node-split model)

$$\min \sum_{a \in A'} \bar{c}_a x_a \tag{31}$$

$$\text{s.t.} \sum_{j \in N_i} \left(f(\delta^-(j)) - f(\delta^+(j)) \right) = d_i \qquad i \in N \setminus \{0\}, \tag{32}$$

$$x(\delta^+(0)) = m, \tag{33}$$

$$x(\delta^-(j)) - x(\delta^+(j)) = 0 \qquad j \in N', \tag{34}$$

$$f(\delta^-(i,l)) - f(\delta^+(i,l)) \geq 0 \qquad (i,l) \in N' : |N_i| \geq 2, \tag{35}$$

$$x(\delta^-(i,l)) = v_{i,l} \qquad (i,l) \in N' : |N_i| \geq 2, l \neq |N_i|, \tag{36}$$

$$x(\delta^-(i,|N_i|)) \leq (m - |N_i| + 1) v_{i,|N_i|} \qquad i \in N \setminus \{0\} : |N_i| \geq 2, \tag{37}$$

$$v_{i,l} \geq v_{i,l+1} \qquad (i,l) \in N' : |N_i| \geq 2, l \neq |N_i|, \tag{38}$$

$$0 \leq f_a \leq Q x_a \qquad a \in A', \tag{39}$$

$$v_{i,l} \in \{0, 1\} \qquad (i, l) \in N', \quad (40)$$

$$x_a \in \{0, 1\} \qquad a \in A' \backslash \delta^-(0), \quad (41)$$

$$x_a \in \mathbb{Z}_+ \qquad a \in \delta^-(0). \quad (42)$$

The objective of the node-split model is to minimize the total transportation cost. Constraint (32) guarantees that the demand of each customer is completely satisfied. Exactly m vehicles depart from the depot due to (33), and constraint (34) ensures that the vehicle flow is conserved everywhere. For the customers having at least one duplicate; i.e., the nodes that have caused regularity violation at a previous iteration, constraint set (35) intends to enforce regularity property at these customers together with constraints (36). More specifically, for every violating customer i, inequality (35) imposes nonnegativity restrictions on the net incoming flow to every node in N_i and equality (36) prevents more than one visit to all but the last node in N_i. In this way, only the last node in N_i can cause regularity violation during the succeeding iterations if $N_i < m$, which would be eliminated later by adding more duplicates as necessary. Eventually, the regularity is established at a violating customer by (35) and (36) after creating at most $m - 1$ duplicates. The number of visits v to every duplicate node is determined by the inequalities (37) and (38). In particular, these constraints ensure that multiple entries are allowed only for the last duplicate of a particular node and that duplicate nodes are visited in the increasing order; i.e., if lth duplicate of a node is visited, then all the preceding duplicates must have been visited once. Lower and upper bounds on the arc flows are imposed by (39). Finally, constraints (40)–(42) are integrality and binary restrictions on the variables.

The node-split algorithm follows the same steps as the patching algorithm except Step 3. In this step of the node-split algorithm, among the nodes violating the regularity of the current solution, we select the first one encountered during the regularity check. We denote this node by i, create a duplicate i' of node i, and update N' by setting $N_i = N_i \cup \{i'\}$ and A' by establishing the arcs between i' and the nodes in $(N' \cup \{0\}) \backslash N_i$. We redefine the node-split model over the enlarged sets N' and A' and then proceed to the next step.

Convergence to an optimal solution of the SDVRP is guaranteed by the node-split algorithm since the regularity violation is eliminated for a given node after $m - 1$ iterations in the worst case. More precisely, if all of the m vehicles visit a certain node, there will be at most $m-1$ copies of the node after $m-1$ iterations, and constraints (33) will force regularity for all copies and thus for the original node. Essentially, creating $m - 1$ duplicates of a node in this algorithm is analogous to adding vehicle-indexed variables in the patching algorithm. Even if the number of iterations required to reach an optimum is higher compared to the patching algorithm, the node-split algorithm usually works faster as will be apparent through our computational results.

4 Computational study

We implemented our algorithms in Java using the mixed integer linear programming solver CPLEX 12.6 and performed a computational study on a 64-bit machine with

Intel Xeon E5-2630 v2 processor at 2.60 GHz and 96 GB of RAM. The experiments were conducted on a total of 46 problem instances including benchmark instances proposed by Belenguer et al. (2000), Archetti et al. (2006), Chen et al. (2007), and a new set of randomly generated instances. In each of these instances, the number of vehicles is equal to the minimum number of vehicles required to serve the total demand, i.e., $|K| = \left\lceil \frac{d(N\backslash\{0\})}{Q} \right\rceil$. We attempted to solve the problems up to 75 customers with rounded costs. We check triangle inequality and set $c_{ik} = c_{ij} + c_{jk}$ for $(i, k) \in A$ with $c_{ik} > c_{ij} + c_{jk}$. Parallel processing is employed in our study with 8 threads or 24 threads depending on the problem size. For the instances containing less than 50 customers, we use 8 threads, while for larger problems, the processing is performed on 24 threads. Based on the results of preliminary computational tests, flow cover, flow path and the mixed integer rounding cuts are switched off.

The R-SDVRP is strengthened by adding rounded capacity inequalities and cutset inequalities at the root node of the branch-and-bound tree. Starting with a fractional solution obtained by solving the linear relaxation of the R-SDVRP, we separate the rounded capacity inequalities employing a heuristic procedure known as the connected component heuristic in the CVRP literature (see Ralphs et al. 2003 for details). Consider the support graph $\bar{G}$ associated with a given fractional solution $\bar{x}$. First, we find the connected components of $\bar{G}$ excluding the depot node. We denote these components by $S_1, \ldots, S_t$, and for every $u = 1, \ldots, t$ we check whether S_u violates the rounded capacity inequality (19). If no violation is detected, we try to identify a node $i \in S_u$ for which

$$\left\lceil \frac{d(S_u\backslash\{i\})}{Q} \right\rceil = \left\lceil \frac{d(S_u)}{Q} \right\rceil$$

and

$$x(\delta^-(S_u\backslash\{i\})) < x(\delta^-(S_u)),$$

remove node i from the set S_u and check for violation for the new set $S_u\backslash\{i\}$. If the new set still does not violate (19), we repeat the same steps until either a violated rounded capacity inequality is detected, or no node whose removal would induce a violated rounded capacity inequality exists. For the cutset inequality, separation can be performed by checking violation for subsets $A^- = \{a \in \delta^-(S) : f_a \geq rx_a\}$ and $A^+ = \{a \in \delta^+(S) : (Q - r)x_a - f_a < 0\}$ given a fractional solution (f, x) and a set $S \subseteq N\backslash\{0\}$. We apply this separation procedure for the sets S with $|S| = 1$ only; i.e., we check violation for subsets $A^- = \{a \in \delta^-(i) : f_a \geq rx_a\}$ and $A^+ = \{a \in \delta^+(i) : (Q - r)x_a - f_a < 0\}$ for every $i \in N\backslash\{0\}$. A violated rounded capacity or cutset inequality is introduced to the model if its violation is at least 10%, and the search is terminated when the improvement in the objective function value cannot exceed 5% in the last two iterations. Additionally, the variables x_a are restricted to take 0–1 values for the arcs $a \in A\backslash\delta^-(0)$ by Proposition 2.1.

For each problem instance, the time limit is set to 2 h after violated rounded capacity cuts and cutset inequalities are separated at the root node of the search tree. If an optimal solution to the R-SDVRP cannot be found within 2 h, we investigate the existence of a feasible SDVRP solution on the support graph associated with the incumbent solution $(\bar{f}, \bar{x})$, which is induced by the arcs (i, j) such that $\bar{x}_{ij} \geq 1$ or $\bar{x}_{ji} \geq 1$, by employing our vehicle-indexed formulation, for which the time limit is an additional 30 min. Under the above settings, our results regarding the patching and the node-split algorithms are summarized in Tables 2 and 3.

For the majority of the instances in the literature, either the optimal solution of the R-SDVRP is also feasible for the SDVRP; that is, the R-SDVRP yields an optimal SDVRP solution, or the R-SDVRP cannot be solved within the time limit of 2 h. In fact, there is only one instance, namely eil30, for which our algorithms perform more than a single iteration. We note that the results in Archetti et al. (2014) show that the undirected formulation without flow variables performs better in solving most of these instances. Our aim here is not to compare different formulations with and without flow variables but to test whether the idea of extending the formulation iteratively can be useful in solving the problem. We use the formulation with flow variables and only change the way we extend the formulation in applying different methods. We need instances that are solved after several iterations to be able to compare the performances of the patching and the node-split algorithms and to see if there is a gain in extending the formulation iteratively. To this end, we introduce five new instances to the literature (available at ozbaygin.bilkent.edu.tr), namely $r1$ through $r5$, with the number of nodes ranging between 30 and 48, and the number of vehicles is three or four.

Among these instances, $r1$ is completely random. For the remaining ones, the coordinates were taken from the existing CVRP instances, while the demands are randomly generated according to three different scenarios; that is, between $[0.01Q, 0.1Q]$, $[0.01Q, 0.15Q]$, or $[0.01Q, 0.2Q]$, and the demand of one customer is increased by $Q/2$ to enhance the possibility of having at least one split customer.

The results regarding the instances for which an optimal R-SDVRP solution cannot be obtained at the end of 2 h as well as the instances for which the optimal solution of our relaxation yields an admissible SDVRP solution are provided in Table 2. Both the patching and the node-split algorithms solve the R-SDVRP in their first iteration; hence, the two algorithms yield the same results for these instances. For the remaining instances, we give the results in Table 3.

We can solve 19 instances to optimality, 13 of which take a single iteration to solve because either the optimal R-SDVRP solution satisfies the regularity property, or an alternative regular solution of the same cost exists. The solution times and iterations performed by both algorithms are provided in Table 3 for the remaining instances. Accordingly, the node-split algorithm converges to an optimal solution faster than the patching algorithm in five of the six instances.

We obtain an upper bound for 12 problem instances, and we are able to improve the best-known upper bound in the literature for four of the instances that are highlighted bold in Table 2. In fact, regarding the instance *p02-7090*, we report an upper bound for the first time in the literature. In general, once the optimal R-SDVRP solution is attained, iterating for an optimal solution of the SDVRP with our patching or node-split algorithms can be effectively done. In particular, as Table 3 also depicts, this time is much lower for the node-split algorithm. However, as Table 2 clearly points out, solving even the relaxed form of the SDVRP could be quite challenging.

Finally, we also provide the best-known upper bounds as well as the upper and lower bounds for each instance using nonrounded costs in Table 4. These values are obtained with the node-split algorithm. Overall, we observe that the results are similar to those in the rounded cost case.

5 Extensions

In this section, we introduce two new extensions of the SDVRP: (1) SDVRP with at most r splits and (2) SDVRP with open routes (SDOVRP). To the best of our knowledge, no results have been presented previously regarding these extensions, both of which can be modeled by slightly modifying our flow-based formulations.

5.1 SDVRP with at most *r* splits

Even though delivery splitting has a potential for cost savings, customers might not be willing to receive several separate deliveries due to handling inefficiencies in practice. In the SDVRP with at most r splits, split deliveries are allowed, but the demand of any customer may be covered by at most r vehicles where $1 < r < m$. Notice that when $r = 1$, the problem reduces to the CVRP, and when $r = m$, it becomes the SDVRP. There are some studies in the literature that impose a restriction on minimum delivery amounts for the vehicles visiting a customer. However, we are not aware of any work in which the number of splits is limited. A mathematical model for SDVRP with at most r splits is readily available by adding the following constraints to SDVRP model (1)–(9)

$$\sum_{k \in K} y^k(\delta^-(i)) \leq r \quad i \in N \backslash \{0\}.$$

Similarly, introducing the restriction

$$x(\delta^-(i)) \leq r \quad i \in N \backslash \{0\} \tag{43}$$

to the R-SDVRP model (10)–(16) provides a relaxation to SDVRP with at most r splits.

Regarding the solution approach, the patching algorithm can be implemented directly when constraint set (43) is added to the R-SDVRP, and the node-split algorithm can be employed by replacing (37) with the following set of constraints:

$$x(\delta^-(i, |N_i|)) \leq (r - |N_i| + 1) v_{i,|N_i|} \qquad i \in N \backslash \{0\} : |N_i| \geq 2 \tag{44}$$

since fulfilling the demand of a customer with at most r vehicles means that the customer can have no more than r duplicates at any iteration of the algorithm.

Here we consider the case $r = 2$ and provide the results of our computational experiments for the SDVRP with at most two splits. Notice that when $r = 2$ the node-split model can be simplified further, because in this case, we do not need the

Table 4 Results for the SDVRP instances with nonrounded costs

Instance	Number of nodes	Number of vehicles	Best-known upper bound	Lower bound	Upper bound	Gap (%)	Time (s)
eil22	22	4	375.28	375.28	375.28	0	3.59
eil23	23	3	568.56	568.56	568.56	0	0.81
eil30	30	3	512.72	512.72	512.72	0	465.09
eil33	33	4	837.06	837.06	837.06	0	600.90
eil51	51	5	524.61	524.61	524.61	0	890.85
eilA76	76	10	823.89	783.58	–	–	7200
eilB76	76	14	1009.04	949.56	–	–	7200
eilC76	76	8	738.67	713.34	–	–	7200
eilD76	76	7	687.60	663.44	–	–	7200
S51D1	51	3	459.50	459.50	459.50	0	17.22
S51D2	51	9	708.42	679.81	–	–	7200
S51D3	51	15	947.97	909.75	951.08	4.54	7200
S51D4	51	27	1560.88	1506.14	1569.08	4.18	7200
S51D5	51	23	1333.67	1302.63	1335.98	2.56	7200
S51D6	51	41	2169.10	2101.62	2183.02	3.87	7200
S76D1	76	4	598.94	598.94	598.94	0	4453.60
S76D2	76	15	1087.99	1023.28	–	–	7200
S76D3	76	23	1427.81	1362.89	–	–	7200
S76D4	76	37	2079.76	1994.38	–	–	7200
SD1	9	6	228.28	228.28	228.28	0	0.03
SD2	17	12	708.28	708.28	708.28	0	0.74
SD3	17	12	430.58	430.58	430.58	0	0.20
SD4	25	18	631.05	631.05	631.05	0	0.25
SD5	33	24	1390.57	1390.57	1390.57	0	2330.64
SD6	33	24	831.24	831.24	831.24	0	4.46
SD7	41	30	3640	3557.53	3640.00	2.32	7200
SD8	49	36	5068.28	4798.36	5068.28	5.63	7200
SD9	49	36	2044.20	1998.32	2044.20	2.30	7200
SD10	65	48	2684.88	2622.56	2684.88	2.38	7200
p01-110	51	3	459.50	459.50	459.50	0	18.44
p01-1030	51	11	756.71	730.04	756.71	3.65	7200
p01-1050	51	16	1005.75	980.92	1005.75	2.53	7200
p01-1090	51	26	1487.41	1457.01	1488.76	2.18	7200
p01-3070	51	26	1481.71	1439.81	1481.76	2.91	7200
p01-7090	51	41	2162.58	2093.48	2159.81	3.17	7200
p02-110	76	5	617.85	607.11	–	–	7200
p02-1030	76	16	1122.91	1050.71	–	–	7200
p02-1050	76	24	1509.79	1441.15	–	–	7200
p02-1090	76	40	2372.22	2224.98	–	–	7200

Table 4 continued

Instance	Number of nodes	Number of vehicles	Best-known upper bound	Lower bound	Upper bound	Gap (%)	Time (s)
p02-3070	76	39	2235.61	2147.40	–	–	7200
p02-7090	76	61	3259.36	3131.52	3240.92	3.49	7200
r1	30	4	711.50	711.50	711.50	0	20.07
r2	36	3	399.04	399.04	399.04	0	894.00
r3	36	4	419.79	419.79	419.79	0	46.86
r4	41	3	410.81	410.81	410.81	0	2144.34
r5	48	3	37,232.93	37,232.93	37,232.93	0	6017.46

variable v, and regularity violation at a node can be eliminated at once by creating a single duplicate of the node (unlike the SDVRP, which may take $m-1$ iterations to establish regularity at a node in the worst case). More precisely, the node-split model reduces to the following:

$$
\begin{aligned}
\min \ & \sum_{a\in A'} \bar{c}_a x_a && \\
\text{s.t.} \ & \sum_{j\in N_i} \big(f(\delta^-(j)) - f(\delta^+(j))\big) = d_i && i \in N\backslash\{0\}, \\
& f(\delta^-(i,l)) - f(\delta^+(i,l)) \geq 0 && (i,l) \in N' : |N_i| = 2, \\
& x(\delta^+(0)) = m, && \\
& x(\delta^-(j)) - x(\delta^+(j)) = 0 && j \in N', \\
& x(\delta^-(i,1)) = 1 && i \in N\backslash\{0\} : |N_i| = 2, \\
& x(\delta^-(i,2)) \leq 1 && i \in N\backslash\{0\} : |N_i| = 2, \\
& 0 \leq f_a \leq Q x_a && a \in A', \\
& x_a \in \{0,1\} && a \in A'\backslash\delta^- n(0), \\
& x_a \in \mathbb{Z}_+ && a \in \delta^-(0).
\end{aligned}
$$

Since our node-split algorithm proved more effective than the patching algorithm for the SDVRP, we attempted to solve at most two splits version using only the node-split algorithm. Tables 5 and 6 indicate our results. In this case, we can solve 18 instances optimally and obtain an upper bound for 16 instances. Different from our results for the SDVRP, we cannot reach an optimal solution for the instance $r5$.

Another way to tackle the problem with at most two splits is to solve the R-SDVRP without adding constraint (43), and create duplicates of the customers receiving more than two separate deliveries in addition to those violating the regularity of the solution. We also tried to solve the SDVRP with at most two splits in this manner. The results are demonstrated in Tables 7 and 8. We can reach an optimum for the instance $r5$ in addition to 17 of the instances that can be solved optimally in the presence of (43),

Table 5 Results for the instances taking single iteration for the SDVRP with at most two splits

Instance	Number of nodes	Number of vehicles	Lower bound	Upper bound	Gap (%)	Time (s)
eil22	22	4	375	375	0	4.93
eil23	23	3	569	569	0	1.53
eil33	33	4	835	835	0	60.55
eil51	51	5	521	521	0	676.38
eilA76	76	10	775.91	828	6.29	7200
eilB76	76	14	940.62	1015	7.33	7200
eilC76	76	8	708	–	–	7200
eilD76	76	7	657.01	684	3.95	7200
S51D1	51	3	458	458	0	18.94
S51D2	51	9	677.53	–	–	7200
S51D3	51	15	908.62	944	3.75	7200
S51D4	51	27	1504.19	–	–	7200
S51D5	51	23	1293.61	1329	2.66	7200
S51D6	51	41	2088.57	2206	5.32	7200
S76D1	76	4	592	592	0	1351.40
S76D2	76	15	1019.85	–	–	7200
S76D3	76	23	1349.70	–	–	7200
S76D4	76	37	1989.93	–	–	7200
SD1	9	6	228	228	0	0.02
SD2	17	12	708	708	0	1.60
SD3	17	12	432	432	0	0.28
SD4	25	18	630	630	0	0.27
SD5	33	24	1392	1392	0	10.66
SD6	33	24	832	832	0	3.39
SD7	41	30	3606.23	3640	0.93	7200
SD8	49	36	4875.15	5068	3.81	7200
SD9	49	36	2007.52	2046	1.88	7200
SD10	65	48	2612.83	2688	2.80	7200
p01-110	51	3	458	458	0	22.94
p01-1030	51	11	726.02	755	3.84	7200
p01-1050	51	16	967.69	–	–	7200
p01-1090	51	26	1445.06	1483	2.56	7200
p01-3070	51	26	1440.55	1479	2.60	7200
p01-7090	51	41	2077.65	2166	4.08	7200
p02-110	76	5	600.76	–	–	7200
p02-1030	76	16	1043.23	–	–	7200
p02-1050	76	24	1434.84	–	–	7200
p02-1090	76	40	2210.41	–	–	7200
p02-3070	76	39	2134.39	–	–	7200
p02-7090	76	61	3108.36	3343	7.02	7200

Table 6 Results for the instances taking multiple iterations for the SDVRP with at most two splits

Instance	Number of nodes	Number of vehicles	Node-split algorithm				Number of iterations
			Lower bound	Upper bound	Gap (%)	Time (s)	
eil30	30	3	510	510	0	42.20	3
r1	30	4	708	708	0	22.43	2
r2	36	3	398	398	0	327.93	4
r3	36	4	421	421	0	151.58	2
r4	41	3	410	410	0	234.69	2
r5	48	3	37,105	37,234	0.34	7200	4

while we can obtain an upper bound only for the instance $eilD76$. Observe that in general, when the number of vehicles is large and the instance cannot be solved to optimality, an upper bound cannot be obtained because the solution found at the end of the 2-h time limit usually contains customers that are visited by at least three vehicles. Besides, even though some instances with large number of vehicles can be solved optimally, finding an optimal solution takes many iterations without (43), yielding longer computational times. Therefore, relaxing constraint (43) makes it harder to terminate with an optimal or a feasible solution to the SDVRP with at most two splits for the instances containing large number of vehicles. When the number of vehicles is small, not imposing restriction (43) usually improves the solution times if the optimal SDVRP solution is also feasible to the at most two splits version. Nonetheless, if the number of iterations performed to reach an optimum increases due to the relaxation of (43), solution times may get worse.

5.2 SDVRP with open routes

Another extension we present is the SDVRP with open routes (SDOVRP), which is essentially the SDVRP where vehicles are not required to return to the depot upon completing their service, or they may return by visiting the customers on their route in the reverse order. The notion of open routes is mentioned for the first time by Schrage (1981), but the open vehicle routing problem (OVRP) did not receive much attention until the formal introduction of the problem by Sariklis and Powell (2000). Hence, it is relatively new compared to the SDVRP and the majority of the research effort on this problem seems to focus on heuristic methods (see Sariklis and Powell 2000; Tarantilis and Kiranoudis 2002; Brandão 2004; Fu et al. 2005 for some examples). One exact solution approach for the problem is the branch-and-cut algorithm due to Letchford et al. (2007). For a review of the OVRP algorithms, the reader is referred to Li et al. (2007). Several variants of the OVRP have been studied so far, including capacitated OVRP, the OVRP with time windows and the OVRP with fuzzy demands. Also, there are studies involving split deliveries and open routes under the same framework as in Ceselli et al. (2009b) and Wang et al. (2014), but the former is the part of a rich VRP, and the latter is within the context of a location-routing problem. To the best of

Table 7 Results for the instances taking single iteration for the SDVRP with at most two splits when (43) is relaxed

Instance	Number of nodes	Number of vehicles	Lower bound	Upper bound	Gap (%)	Time (s)
eil22	22	4	375	375	0	2.90
eil23	23	3	569	569	0	1.50
eil33	33	4	835	835	0	18.81
eil51	51	5	521	521	0	266.69
eilA76	76	10	777.40	–	–	7200
eilB76	76	14	941.24	–	–	7200
eilC76	76	8	709.16	–	–	7200
eilD76	76	7	657.46	684	3.88	7200
S51D1	51	3	458	458	0	21.45
S51D2	51	9	681.97	–	–	7200
S51D3	51	15	911.59	–	–	7200
S51D4	51	27	1504.59	–	–	7200
S51D5	51	23	1297.38	–	–	7200
S51D6	51	41	2092.83	–	–	7200
S76D1	76	4	592	592	0	1789.73
S76D2	76	15	1011.41	–	–	7200
S76D3	76	23	1349.60	–	–	7200
S76D4	76	37	1979.51	–	–	7200
SD7	41	30	3483.04	–	–	7200
SD8	49	36	4790.31	–	–	7200
SD9	49	36	2005.46	–	–	7200
SD10	65	48	2620.32	–	–	7200
p01-110	51	3	458	458	0	21.91
p01-1030	51	11	722.39	–	–	7200
p01-1050	51	16	969.95	–	–	7200
p01-1090	51	26	1440.63	–	–	7200
p01-3070	51	26	1432.99	–	–	7200
p01-7090	51	41	2075.79	–	–	7200
p02-110	76	5	599.38	–	–	7200
p02-1030	76	16	1044.37	–	–	7200
p02-1050	76	24	1433.67	–	–	7200
p02-1090	76	40	2212.56	–	–	7200
p02-3070	76	39	2134.03	–	–	7200
p02-7090	76	61	3103.14	–	–	7200

our knowledge, the only study incorporating the open route structure into the classical SDVRP is due to Song and Liu (2013), who present a tabu search heuristic for the problem. However, we are not aware of any published work in which an exact solution algorithm is proposed for the SDOVRP.

Table 8 Results for the instances taking multiple iterations for the SDVRP with at most two splits when (43) is relaxed

Instance	Number of nodes	Number of vehicles	Node-split algorithm				Number of iterations
			Lower bound	Upper bound	Gap (%)	Time (s)	
eil30	30	3	510	510	0	31.12	3
SD1	9	6	228	228	0	0.14	3
SD2	17	12	708	708	0	21.69	11
SD3	17	12	432	432	0	0.21	2
SD4	25	18	630	630	0	3.20	3
SD5	33	24	1392	–	–	7200	3
SD6	33	24	832	832	0	1572.21	15
r1	30	4	708	708	0	24.52	2
r2	36	3	398	398	0	360.81	5
r3	36	4	421	421	0	88.26	2
r4	41	3	410	410	0	850.77	3
r5	48	3	37,234	37,234	0	2322.83	4

Our vehicle-indexed formulation can be adapted to the SDOVRP by simply omitting the cost terms associated with the arcs returning to the depot in the objective function; that is, the objective function of the SDOVRP is expressed as

$$\sum_{a\in A\setminus\delta^-(0)} \sum_{k\in K} c_a y_a^k.$$

In the exact same way, we can modify the objective function of the R-SDVRP as

$$\sum_{a\in A\setminus\delta^-(0)} c_a x_a$$

and employ our algorithms to solve the SDOVRP. It is important to note here that Proposition 2.1 does not remain valid, because reversing the direction of a route can change the total transportation cost in an open route setting. However, we can still restrict the variables x_a to take binary values for $a \in A\setminus(\delta^-(0)\cup\delta^+(0))$ as the feasible region associated with the problem does not change; i.e., we only modify the objective function. Since $x_a \in \mathbb{Z}_+$ for $a \in \delta^-(0)\cup\delta^+(0)$, the procedure for checking the regularity of a solution is adapted to handle the cases breaking symmetry. Both the patching and the node-split algorithms are used for solving the SDOVRP, and the results we obtain are reported in Tables 9 and 10. In this case, we can find an optimal solution for 24 instances, while we obtain an upper bound for 9 instances. Similar to the results obtained for the SDVRP, the node-split algorithm yields more favorable solution times when our algorithms perform multiple iterations. Overall, the results

Table 9 Results for the instances taking single iteration for the SDOVRP

Instance	Number of nodes	Number of vehicles	Lower bound	Upper bound	Gap (%)	Time (s)
eil22	22	4	252	252	0	0.55
eil23	23	3	426	426	0	1.52
eil33	33	4	511	511	0	8.33
eil51	51	5	413	413	0	60.55
eilA76	76	10	542.18	–	–	7200
eilB76	76	14	592.99	–	–	7200
eilC76	76	8	532	532	0	4015.37
eilD76	76	7	520	520	0	262.22
S51D1	51	3	405	405	0	24.63
S51D2	51	9	449.18	–	–	7200
S51D3	51	15	526.32	541	2.71	7200
S51D4	51	27	798.70	–	–	7200
S51D5	51	23	698.18	–	–	7200
S51D6	51	41	1083.54	–	–	7200
S76D1	76	4	515	515	0	141.94
S76D2	76	15	617.80	–	–	7200
S76D3	76	23	742.03	–	–	7200
S76D4	76	37	1040.08	–	–	7200
SD1	9	6	128	128	0	0.03
SD2	17	12	368	368	0	0.25
SD3	17	12	232	232	0	0.06
SD4	25	18	330	330	0	2.38
SD5	33	24	712	712	0	2.18
SD6	33	24	432	432	0	1.64
SD7	41	30	1820	1820	0	12.12
SD8	49	36	2548	2548	0	8.23
SD9	49	36	1050	1050	0	8.92
SD10	65	48	1377.90	1392	1.01	7200
p01-110	51	3	405	405	0	19.91
p01-1030	51	11	466.41	474	1.60	7200
p01-1050	51	16	588.29	–	–	7200
p01-1090	51	26	770.42	791	2.60	7200
p01-3070	51	26	764.53	786	2.73	7200
p01-7090	51	41	1069.77	1100	2.75	7200
p02-110	76	5	513	513	0	92.20
p02-1030	76	16	625.15	–	–	7200
p02-1050	76	24	781.50	809	3.40	7200
p02-1090	76	40	1153.18	–	–	7200

Table 9 continued

Instance	Number of nodes	Number of vehicles	Lower bound	Upper bound	Gap (%)	Time (s)
p02-3070	76	39	1115.61	–	–	7200
p02-7090	76	61	1580.73	1634	3.26	7200
r2	36	3	332	332	0	12.83
r3	36	4	334	334	0	7.10
r4	41	3	349	349	0	15.19
r5	48	3	30787	30787	0	12

Table 10 Results for the instances taking multiple iterations for the SDOVRP

Instance	Number of nodes	Number of vehicles	Patching algorithm				Number of iterations
			Lower bound	Upper bound	Gap (%)	Time (s)	
eil30	30	3	375	375	0	329.84	3
r1	30	4	506	507	0.19	7200	5
Instance	Number of nodes	Number of vehicles	Node-split algorithm				Number of iterations
			Lower bound	Upper bound	Gap (%)	Time (s)	
eil30	30	3	375	375	0	145.65	5
r1	30	4	506	507	0.19	7200	5

indicate that the problem becomes easier to handle when the depot return requirement is relaxed.

6 Discussion on algorithmic performance

Computational experiments revealed that the main difficulty we face is in solving the R-SDVRP. This is a mixed integer program that our algorithms extend iteratively and try to solve optimally at each iteration. Notice that applying patching or node-split procedures does not require waiting until optimality is achieved. Therefore, we also considered using a continuous relaxation of the R-SDVRP and solving the problem in one branch-and-cut tree. We implemented this by using the lazy constraint callback feature of CPLEX. However, we observed that it did not speed up the algorithm. We believe that there are mainly two reasons behind this. First, using control callbacks disables dynamic search and activates traditional MIP search, which can be significantly slower than dynamic search. Second, a larger number of variables and constraints are added to the initial relaxation when the integrality constraints are relaxed. Table 11

Table 11 Comparison: Using R-SDVRP versus its continuous relaxation

Instance	Solution time (s)	
	R-SDVRP	Continuous rel. of R-SDVRP
eil22	3.07	7.11
eil23	1.56	3.42
eil30	30.58	3600
eil33	19.09	1863.18
eil51	264.98	3600

Table 12 Cuts active versus inactive

Instance	Solution time (s)		Relative reduction (%)
	Cuts active	Cuts disabled	
eil22	3.16	3.07	2.85
eil23	0.53	1.56	–
eil30	557.51	30.58	94.51
eil33	20.03	19.09	4.69
eil51	420.64	264.98	37.01
r1	218.61	105.41	51.78
r2	590.46	857.07	–
r3	255.28	698.62	–
r4	2022.27	1033.69	48.88
r5	3526.03	3686.20	–

provides a comparison between the two approaches using the patching algorithm for five small-to-medium sized instances, namely eil22–eil51 with a 1-h time limit.

As a result, we put our effort in solving the aggregated integer model more efficiently. To this end, we proposed and tested several enhancement ideas. Accordingly, we observed that the solution procedure can be accelerated by restricting a subset of the integer variables, i.e., the arc design variables associated with the outgoing arcs of the depot, to take binary values. Moreover, the lower bounds of the R-SDVRP can be strengthened by separating cutset inequalities and rounded capacity inequalities at the root node of the search tree. It should be noted, however, that after a certain number of cutting plane iterations, the improvement achieved by adding more of these inequalities is marginal. Therefore, we used a stopping criterion for the cutting plane phase to overcome the tailing-off effect. We also performed experiments by switching off some default cuts used by CPLEX, and we found out that flow cover, flow path and MIR cuts usually slow down the solution of the R-SDVRP as demonstrated in Table 12.

We were able to achieve significant performance enhancements by employing the ideas outlined above. Moreover, we considered other means of speeding up our solution procedure, such as the use of framed capacity inequalities as cutting planes, or applying

Benders decomposition to the R-SDVRP, but concluded that these are not helpful based on initial experiments. Note that the results reported in Tables 2–10 adopt all the enhancements that proved useful during the preliminary computations.

7 Conclusion

The SDVRP is considered in this study. A vehicle-indexed arc flow formulation is proposed for the problem as well as a relaxed model (R-SDVRP) obtained from this flow formulation. A new property regarding the optimal SDVRP solutions is derived, which guarantees the existence of an optimal SDVRP solution in which any arc emanating from the depot is traversed at most once. We devise two novel exact solution algorithms based on the idea of iteratively extending the relaxation by means of variables and constraints until finding a solution satisfying the regularity property. Additionally, we introduce two extensions of the SDVRP, namely the SDVRP with at most r splits, and the SDVRP with open routes (SDOVRP). We adapt our relaxation and algorithms to tackle these extensions. Computational experiments are performed on 46 problem instances in total, 41 of which are benchmark instances from the literature, and five of which are randomly generated new instances. Results are reported regarding the SDVRP, SDVRP with open routes and SDVRP with at most r splits for the case of $r = 2$. Accordingly, we can remark that our algorithms effectively iterate until an optimal SDVRP solution is found as long as the R-SDVRP can be solved quickly. Nevertheless, solving the R-SDVRP is a difficult task, especially for large-sized problem instances. It is important to recognize, however, that both the patching and the node-split methods can be adopted when solving (especially symmetric) problems other than the SDVRP, such as the multi-depot VRP, inventory routing, crew scheduling and unit commitment. We believe that exploring the iterative extension idea further on different problems can yield efficient optimization algorithms and thus can be a worthwhile contribution in the future.

Acknowledgements The research of Gizem Ozbaygin is supported by the Scientific and Technological Research Council of Turkey and the research of Hande Yaman is supported by the Turkish Academy of Sciences.

References

Aleman RE, Hill RR (2010) A tabu search with vocabulary building approach for the vehicle routing problem with split demands. Int J Metaheuristics 1(1):55–80

Aleman RE, Zhang X, Hill RR (2010) An adaptive memory algorithm for the split delivery vehicle routing problem. J Heuristics 16(3):441–473

Archetti C, Speranza MG (2012) Vehicle routing problems with split deliveries. Int Trans Oper Res 19(1–2):3–22

Archetti C, Speranza MG, Hertz A (2006) A tabu search algorithm for the split delivery vehicle routing problem. Transp Sci 40(1):64–73

Archetti C, Speranza MG, Savelsbergh MWP (2008) An optimization-based heuristic for the split delivery vehicle routing problem. Transp Sci 42(1):22–31

Archetti C, Bianchessi N, Speranza MG (2011) A column generation approach for the split delivery vehicle routing problem. Networks 58(4):241–254

Archetti C, Bianchessi N, Speranza MG (2014) Branch-and-cut algorithms for the split delivery vehicle routing problem. Eur J Oper Res 238(3):685–698

Atamtürk A (2002) On capacitated network design cut-set polyhedra. Math Program 92(3):425–437

Belenguer JM, Martinez MC, Mota E (2000) A lower bound for the split delivery vehicle routing problem. Oper Res 48(5):801–810

Berbotto L, García S, Nogales FJ (2014) A randomized granular tabu search heuristic for the split delivery vehicle routing problem. Ann Oper Res 222(1):153–173

Boudia M, Prins C, Reghioui M (2007) An effective memetic algorithm with population management for the split delivery vehicle routing problem. In: Hybrid metaheuristics. Springer, pp 16–30

Brandão J (2004) A tabu search algorithm for the open vehicle routing problem. Eur J Oper Res 157(3):552–564

Ceselli A, Righini G, Salani M (2009a) Column generation for the split delivery vehicle routing problem. Technical report, University of Milan-DTI-Note del Polo

Ceselli A, Righini G, Salani M (2009b) A column generation algorithm for a rich vehicle-routing problem. Transp Sci 43(1):56–69

Chen S, Golden B, Wasil E (2007) The split delivery vehicle routing problem: applications, algorithms, test problems, and computational results. Networks 49(4):318–329

Chen Q, Li K, Liu Z (2014) Model and algorithm for an unpaired pickup and delivery vehicle routing problem with split loads. Transpn Rese E Logist Transpo Revi 69:218–235

Desaulniers G (2010) Branch-and-price-and-cut for the split-delivery vehicle routing problem with time windows. Oper Res 58(1):179–192

Dror M, Trudeau P (1989) Savings by split delivery routing. Transp Sci 23(2):141–145

Dror M, Trudeau P (1990) Split delivery routing. Nav Res Logist 37(3):383–402

Dror M, Laporte G, Trudeau P (1994) Vehicle routing with split deliveries. Discrete Appl Math 50(3):239–254

Fu Z, Eglese R, Li LY (2005) A new tabu search heuristic for the open vehicle routing problem. J Oper Res Soc 56(3):267–274

Gouveia L (1995) A result on projection for the vehicle routing problem. Eur J Oper Res 85(3):610–624

Gulczynski D, Golden B, Wasil E (2010) The split delivery vehicle routing problem with minimum delivery amounts. Transp Res E Logist Transpo Rev 46(5):612–626

Jin M, Liu K, Bowden RO (2007) A two-stage algorithm with valid inequalities for the split delivery vehicle routing problem. Int J Prod Econ 105(1):228–242

Jin M, Liu K, Eksioglu B (2008) A column generation approach for the split delivery vehicle routing problem. Oper Res Lett 36(2):265–270

Khmelev A, Kochetov Y (2015) A hybrid local search for the split delivery vehicle routing problem. Int J Artif Intell 13(1):147–164

Lee C, Epelman MA, White CC, Bozer YA (2006) A shortest path approach to the multiple-vehicle routing problem with split pick-ups. Transp Res B Methodol 40(4):265–284

Letchford AN, Salazar-González JJ (2006) Projection results for vehicle routing. Math Program 105(2–3):251–274

Letchford AN, Lysgaard J, Eglese RW (2007) A branch-and-cut algorithm for the capacitated open vehicle routing problem. J Oper Res Soc 58(12):1642–1651

Li F, Golden B, Wasil E (2007) The open vehicle routing problem: algorithms, large-scale test problems, and computational results. Comput Oper Res 34(10):2918–2930

Moreno L, Aragão MP, Uchoa E (2010) Improved lower bounds for the split delivery vehicle routing problem. Oper Res Lett 38(4):302–306

Mota E, Campos V, Corberán Á (2007) A new metaheuristic for the vehicle routing problem with split demands. In: Evolutionary computation in combinatorial optimization. Springer, pp 121–129

Naddef D, Rinaldi G (2002) Branch-and-cut algorithms for the capacitated VRP. In: Toth P, Vigo D (eds) The vehicle routing problem, SIAM monographs on discrete mathematics and applications. SIAM, Philadelphia, PA, USA, pp 53–81

Ozdamar L, Demir O (2012) A hierarchical clustering and routing procedure for large scale disaster relief logistics planning. Transp Res E Logist Transp Revi 48(3):591–602

Ralphs TK, Kopman L, Pulleyblank WR, Trotter LE (2003) On the capacitated vehicle routing problem. Math Program 94(2–3):343–359

Sahin M, Cavuslar G, Oncan T, Sahin G, Tuzun D (2013) Aksu. An efficient heuristic for the multi-vehicle one-to-one pickup and delivery problem with split loads. Transp Res C Emerg Technol 27:169–188

Sariklis D, Powell S (2000) A heuristic method for the open vehicle routing problem. J Oper Res Soc 51(5):564–573

Schrage L (1981) Formulation and structure of more complex/realistic routing and scheduling problems. Networks 11(2):229–232

Sierksma G, Tijssen GA (1998) Routing helicopters for crew exchanges on off-shore locations. Ann Oper Res 76:261–286

Silva MM, Subramanian A, Ochi LS (2015) An iterated local search heuristic for the split delivery vehicle routing problem. Comput Oper Res 53:234–249

Song Q, Liu L (2013) The application of tabu search algorithm on split delivery open vehicle routing problem. BioTechnology 8(8):1088–1094

Tarantilis CD, Kiranoudis CT (2002) Distribution of fresh meat. J Food Eng 51(1):85–91

Wang H, Du L, Ma S (2014) Multi-objective open location-routing model with split delivery for optimized relief distribution in post-earthquake. Transp Res E Logist Transp Rev 69:160–179

Wilck JH IV, Cavalier TM (2012a) A genetic algorithm for the split delivery vehicle routing problem. Am J Oper Res 2:207–216

Wilck JH IV, Cavalier TM (2012b) A construction heuristic for the split delivery vehicle routing problem. Am J Oper Res 2:153–162

Yi W, Kumar A (2007) Ant colony optimization for disaster relief operations. Transp Res E Logist Transp Rev 43(6):660–672