Intl. Trans. in Op. Res. xx (2023) 1–22

A Biased-Randomized Iterated Local Search for the Vehicle Routing Problem with Optional Backhauls

Julio C. Londoño[a], Rafael Tordecilla-Madera[b, c, *], Leandro Martins[b] and Angel A. Juan[b]

[a]*Faculty of Engineering, Universidad del Valle, Cali, Colombia*
[b]*IN3 – Computer Science Dept., Universitat Oberta de Catalunya, 08860 Castelldefels, Spain*
[c]*Faculty of Engineering, Universidad de La Sabana, km 7 Autopista norte de Bogota, D.C., Chia, Colombia*
E-mail: julio.londono@correounivalle.edu.co [J. Londono]; rtordecilla@uoc.edu [R. Tordecilla]; leandrocm@uoc.edu [L. Martins]; ajuanp@uoc.edu [A. Juan]

Received DD MMMM YYYY; received in revised form DD MMMM YYYY; accepted DD MMMM YYYY

Abstract

The vehicle routing problem with backhauls integrates decisions on products delivery with decisions on collection of returnable items. In this paper we analyze the scenario in which collection is optional. Both transport costs and penalty costs for not collecting are considered. A mixed-integer linear model is proposed and solved for small instances. A metaheuristic algorithm combining biased randomization techniques with iterated local search is introduced for larger instances. This approach yields cost savings and is competitive compared to other state-of-the-art approaches.

Keywords: Vehicle Routing Problem with Optional Backhauls; Returnable Transport Items; Biased Randomization; Iterated Local Search

@Julio: please add your affiliation below the title. Also, check if you want to add something in the acknowledgments section.

@All: text in black has been already reviewed and enhanced by Angel.

1. Introduction

Reverse logistics and closed-loop supply chains have been topics broadly studied in the literature and, specially, during the last years. Both concepts are related to the return of products or materials from the point of consumption in order to recover value. In particular, Govindan and Soleimani (2017) find that the most addressed issues are re-manufacturing and waste management, while the topic of package recovery

[*]Author to whom all correspondence should be addressed (e-mail: rtordecilla@uoc.edu).

is barely tackled despite its environmental impact (Kroon and Vrijens, 1995). Given these considerations, disposable packages have been replaced by returnable transport items (RTIs) such as reusable pallets, trays, boxes, or any other mean to assemble goods (ISO, 2016). Still, environmental issues are not the only concern regarding RTIs management. According to Glock (2017), RTIs are an important asset for many industries, since they can decrease the selling cost for customers.

Regarding RTIs management, three control strategies have been proposed by Kroon and Vrijens (1995): *(i)* the switch pool system, which is an integrated system between suppliers and customers where each one owns RTIs –the supplier is responsible for the return process in this strategy; *(ii)* systems with return logistics, in which RTIs are property of a central agency –this agency takes care of their collection; and *(iii)* systems without return logistics, in which RTIs are also property of a central agency –here, suppliers rent only the required RTIs and are responsible for the return process. Finally, if some RTIs are not used they can be returned to the agency. These strategies are analyzed by Hellström and Johansson (2010) through a case study, where the goals are to reduce cost of RTI management and transport. A good strategy for RTIs management is the interchange between suppliers and customers (Elia and Gnoni, 2015). However, such interchange can not be done simultaneously because pickups are only possible when deliveries have already been made and the vehicle is empty (Koç and Laporte, 2018). Therefore, the return of RTIs can be done in two forms: *(i)* by using dedicated collection vehicles; and *(ii)* by using vehicles that make first deliveries and then collections. The latter case is known as the vehicle routing problem with backhauls (VRPB) (Toth and Vigo, 1997; Belloso et al., 2017). Being an extension of the vehicle routing problem, the VRPB is also a *NP-hard* problem.

As far as we know, in the existing literature on the VRPB it is always assumed that all customers, both linehaul and bahckhaul, have to be serviced. In this paper, however, we explore the scenario in which backhaul customers might be optional by introducing inventory costs (Figure 1). Thus, for example, if the quantity of RTIs available in a supplier's facility is enough to ensure future deliveries, collection of RTIs in the current period can be reduced to diminish routing cost or speed up the routing process. This might make sense whenever the cost of holding the additional inventory at the customers' facilities –i.e., the cost of holding the RTIs in stock– is lower than the marginal routing cost associated with their collection. In practice, the cost of holding RTIs in stock at the customers' facilities might vary from one period to another, depending on factors such as how many RTIs will be need for the next period distribution or how long have been the RTIs staying at the customers' inventories.

Hence, the main contribution of this paper can be summarized as follows: *(i)* we propose and analyze the vehicle routing problem with optional backhauls (VRPOB), which has not been considered before in the literature; *(ii)* we provide a mathematical model of the problem, which is then solved using exact methods for small-scale instances; *(iii)* we propose a biased-randomized iterated local search to solve larger instances of the problem; and *(iv)* we analyze how the routing solutions evolve as we consider different levels of inventory or penalty cost. Biased-randomization techniques allow for extending traditional metaheuristic frameworks in order to enhance their performance (Juan et al., 2013; Ferone et al., 2018) and facilitate parallelization issues (Juan et al., 2014b).

The remaining of this document is organized as follows. Section 2 reviews the existing literature on the VRPB. Section 3 formulates the problem as a mixed-integer linear programming model. Section 4 analyzes the results obtained after solving the model using an exact method for small-scale instances. Section 5 introduces a metaheuristic algorithm that can be employed for solving large-scale instances. Section 6 compares the results of our algorithm with the ones from the literature and with those obtained

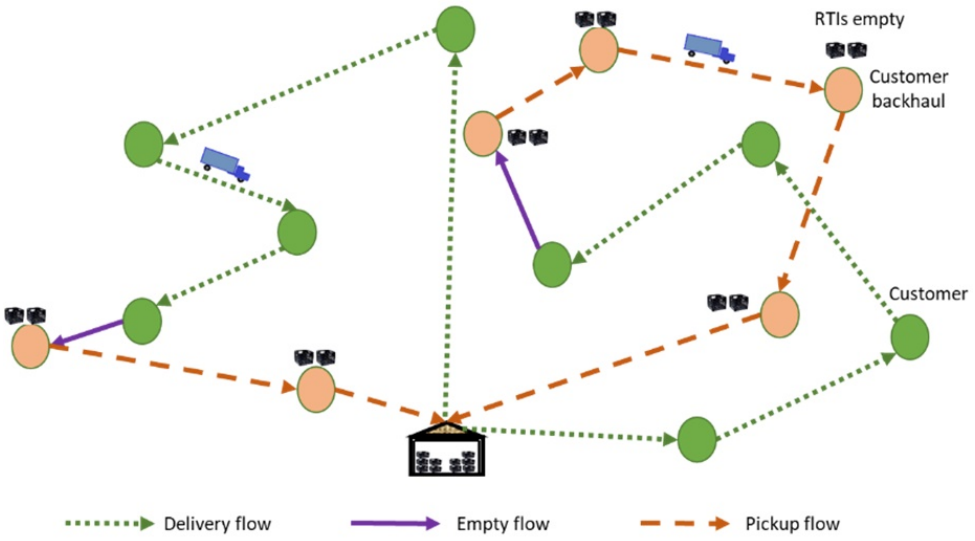

Fig. 1: The vehicle routing problem with optional backhauls.

by the exact method; it also establishes new results for the extended VRPOB. Finally, Section 7 outlines some concluding remarks.

2. Literature Review on the VRPB

Decisions regarding the integration of forward logistics with reverse logistics have been studied systematically along the last three decades. In the following, we review some works related to the vehicle routing problem with backhauls. This version of the vehicle routing problem considers a sequential process in which linehaul customers are serviced first and then bachauls customers are visited for collecting the items that need to be returned to the depot. Some constraints that can lead to this type of sequential process are: *(i)* deliveries have priority over collections; *(ii)* due to time or space limitations, it is not possible to load returned products to a vehicle that still has products to deliver; or *(iii)* cross contamination between products to deliver and returned products must be avoided.

Traditionally, the main goal of the VRPB is to minimize the total distribution and collection cost by taking advantage of the non-used capacity of the vehicles in the return trip. Some initial approaches for the VRPB are presented by Deif and Bodin (1984) and Jacobs-Blecha and Goetschalckx (1992). Both papers present heuristic algorithms to solve the problem efficiently. On the other hand, the VRPB is also modelled as a mixed-integer linear problem by Toth and Vigo (1997). They solve it through a branch-and-cut algorithm for instances between 25 and 68 customers. As pointed out by Koç and Laporte (2018) and Belloso et al. (2019), the VRPB is still offering challenges that need to be solved.

Thus, Wassan (2007) proposes a heuristic that uses reactive tabu search and adaptive memory programming for solving the VRPB. Zachariadis and Kiranoudis (2012) propose the static move descriptor

strategy, which is intended to reduce the computational complexity required to examine neighborhoods with very large solutions. Dominguez et al. (2016) consider two-dimensional loads in a VRPB. Here, a hybrid approach that combines biased randomization with a large neighborhood search metaheuristic is proposed. In general, better solutions –in terms of cost and computational times– were found in comparison with current state-of-the-art heuristics. Belloso et al. (2019) use an iterative method based on local search and a biased-randomization process to solve the heterogeneous-fleet VRPB, obtaining 20 new best-known solutions in a set of 36 instances.

The VRPB with time windows for visiting customers has also been studied. For instance, Küçükoğlu and Öztürk (2015) propose a hybrid metaheuristic algorithm, which integrates simulated annealing with tabu search and local search. A total of 34 best-known solutions were found for a benchmark set of 45 instances. A heterogeneous fleet is considered by Wu et al. (2016), who solve the VRPB with time windows using an algorithm based on ant colony optimization. Variables such as *vehicle type*, *fleet size*, and *routes* are determined in order to minimize the service total cost. Lin et al. (2017) integrate new operative constraints into the VRPB with time windows, among them: last-in-first-out rules, vehicle capacity depending on the specific order, driving-time limits, etc. To solve this version of the problem, a hybrid approach combining the greedy randomized adaptive search procedure with tabu search is proposed. A two-phase approach that combines tabu search with a multi-start evolutionary strategy is proposed by Reil et al. (2018) to solve a VRPB with time windows. These authors also consider constraints regarding three-dimensional loading and different backhaul strategies.

Despite a large number of works have addressed the VRPB or the inventory routing problem (Juan et al., 2014a; Gruler et al., 2018a,b), both problems have been rarely studied as an integrated problem. For instance, Arab et al. (2018) consider both problems by addressing the inventory routing problem with backhauls. The goal was twofold: to minimize total cost (inventory plus transportation) as well as transportation risk. A heterogeneous fleet, multiple periods, and multiple products are considered. Exact algorithms using the ε-constraint method, and evolutionary algorithms were applied to solve the problem.

None of the aforementioned works consider the option of not collecting items from backhaul customers. This means that some customers might not be visited since the additional inventory costs might be justified by the savings in routing costs. To the best of our knowledge, only Qin et al. (2016) show a similar idea by differentiating inventory holding cost for products and for returned items. However, their problem is not formally a VPRB since backhauls are not considered, i.e., deliveries and pickups can be done simultaneously for the same customer.

3. Problem Formulation

The vehicle routing problem with optional backhauls consists in a set of linehaul and backhaul customers whose demands must be satisfied using a fleet of homogeneous vehicles initially located at a depot. This depot has enough capacity and vehicles as to cover the aggregated customers' demand. Therefore, all linehaul customers will be serviced. The supplier uses returnable transport items for transportation of product units. Hence, after all units have been delivered, empty RTIs from previous deliveries should be collected at some backhaul customers and returned to central depot for future deliveries. Collecting RTIs from backhaul customers is optional, and not doing it might generate savings in transportation

costs, but it might also generate 'penalty' costs due to the marginal inventories that need to be held at the backhaul customers. Thus, in the VRPOP the following decisions need to be made in order to minimize total costs (inventory plus routing): *(i)* to determine which backhaul customers will be visited; *(ii)* to assig customers to routes (and vehicles); and *(iii)* to establish the sequence in which customers should be visited.

Consider a non-directed graph $G = (V, A)$, with $V = \{0\} \cup L \cup B$ represents the set of nodes, being node 0 the depot, $L = \{1, 2, \ldots, n\}$ the set of n linehaul customers, and $B = \{n+1, n+2, \ldots, n+m\}$ the set of m backhaul customers. Likewise, $A = \{(i, j)|i, j \in V, i \neq j\}$ is the set of edges linking each pair of nodes. For each $i \in L$, there is a positive demand $d_i > 0$ representing units of product to be delivered. Similarly, for each $j \in B$, there is a negative demand $d_j < 0$ representing items to be collected. There is a set K of homogeneous vehicles available at the depot, each of them with a capacity $q >> \max\{d_i|i \in V\}$. Travelling an edge from node i to node j has a cost $c_{ij} = c_{ji} > 0$. The following constraints need to be considered:

- Each route begins and ends in the depot.
- Each route must have at least a linehaul customer, i.e.: routes formed just by backhaul customers are not allowed.
- In any route, linehaul customers are serviced before backhaul customers.
- Each linehaul customer must be serviced, but visiting backhaul customers is optional.
- For each route, the quantity of product to deliver and to collect must not exceed the vehicle's capacity.
- Whenever a linehaul customer is visited in a route, all its demand is serviced; similarly, whenever a backhaul customer is visited in a route, all its items are collected.

(Angel says): This model needs to be reviewed with Angel and simplified. Use upper case only for random variables or sets. Why using different variables for demands? why using different parameters for travel costs?

3.1. Parameters

d_i = Demand of customer $i \in V$
c_{ij} = Cost of travel from node $i \in V$ to node $j \in V$
h_i = Penalty cost per unit of RTI not picked up from the customer $i \in B$
q = Capacity of each vehicle
M = A very big number

3.2. Variables

y_{ijk} = Binary variable that indicates whether arc $(i, j) : i \in V, j \in V$ is on the route traveled by vehicle $k \in K$ or not
z_{ik} = Binary variable that indicates whether customer $i \in V \setminus \{0\}$ is visited by vehicle $k \in K$ or not
u_{ik} = Accumulated deliveries or pickups by vehicle $k \in K$ until the customer $i \in V \setminus \{0\}$

3.3. Mathematical model

$$Minimize \sum_{i \in B} h_i * d_i * \left(1 - \sum_{k \in K} z_{ik}\right) + \sum_{k \in K} \sum_{i \in V} \sum_{j \in V, i \neq j} c_{ij} * y_{ijk} \quad (1)$$

s.t.

$$\sum_{k \in K} z_{ik} = 1, \ \forall i \in L \quad (2)$$

$$\sum_{k \in K} z_{ik} \leq 1, \ \forall i \in B \quad (3)$$

$$\sum_{j \in L} y_{0jk} = 1, \ \forall k \in K \quad (4)$$

$$\sum_{i \in V \setminus \{0\}} y_{i0k} = 1, \ \forall k \in K \quad (5)$$

$$\sum_{i \in L \cup \{0\}, i \neq f} y_{ifk} + \sum_{j \in V, j \neq f} y_{fjk} = 2 * z_{fk}, \ \forall f \in L, \forall k \in K \quad (6)$$

$$\sum_{i \in V \setminus \{0\}, i \neq f} y_{ifk} + \sum_{j \in B \cup \{0\}, j \neq f} y_{fjk} = 2 * z_{fk}, \ \forall f \in B, \forall k \in K \quad (7)$$

$$\sum_{i \in L \cup \{0\}, i \neq f} y_{ifk} = \sum_{j \in V, j \neq f} y_{fjk} \ \forall f \in L, \forall k \in K \quad (8)$$

$$\sum_{i \in V \setminus \{0\}, i \neq f} y_{ifk} = \sum_{j \in B \cup \{0\}, j \neq f} y_{fjk}, \ \forall f \in B, \forall k \in K \quad (9)$$

$$\sum_{j \in V, j \neq f} y_{fjk} \leq 1, \ \forall f \in L, \forall k \in K \quad (10)$$

$$\sum_{i \in V \setminus \{0\}, i \neq f} y_{ifk} \leq 1, \ \forall f \in B, \forall k \in K \quad (11)$$

$$\sum_{i \in L \cup \{0\}, j \neq l} y_{ilk} + \sum_{j \in B \cup \{0\}, j \neq b} y_{bjk} \geq 2 * y_{lbk}, \ \forall l \in L, \forall b \in B, \forall k \in K, l \neq b \tag{12}$$

$$\sum_{i \in L} d_i * z_{ik} \leq q, \ \forall k \in K \tag{13}$$

$$\sum_{i \in B} d_i * z_{ik} \leq q, \ \forall k \in K \tag{14}$$

$$u_{ik} + d_j * z_{jk} \leq u_{jk} + M * (1 - y_{ijk}), \ \forall i \in V \setminus \{0\}, \forall j \in V \setminus \{0\}, \forall k \in K, i \neq j \tag{15}$$

$$u_{ik} \leq q, \ \forall i \in L, \forall k \in K \tag{16}$$

$$u_{ik} \leq 2 * q, \ \forall i \in B, \forall k \in K \tag{17}$$

$$d_i * z_{ik} \leq u_{ik}, \ \forall i \in V \setminus \{0\}, \forall k \in K \tag{18}$$

$$\forall \ y_{ijk}, z_{ik} \in \{0, 1\} \tag{19}$$

$$\forall \ u_{ik} \geq 0 \tag{20}$$

(Angel says): use labels and ref for equations, instead of numbers

In this model, Equation 1 minimizes the total penalties and routing costs. Constraint 2 ensures that every linehaul customer is visited by only one vehicle. Constraint 3 ensures that at most only one vehicle visits a backhaul customer. Constraints 4 and 5 ensures that each vehicle leaves and returns to the depot, respectively, only one time. Constraints 6 and 7 ensure that two arcs (one entering and one leaving) are assigned to a customer only if this is served, both in linehaul and backhaul groups. Constraints 8 and 9 ensure that if one vehicle visits a customer, it must leave it. Constraint 10 ensures that after departing from each linehaul customer, a vehicle must either serve another linehaul customer or make an empty trip either to the depot or to a backhaul customer.

Constraint 11 ensures that before a vehicle visits a backhaul customer, it must have served either another backhaul customer or a linehaul customer. Constraint 12 ensures a continuous flow using the same vehicle when a linehaul customer is connected to a backhaul customer. Constraints 13 and 14 ensure that the total quantity of product to deliver or pick up does not exceed a single vehicle capacity, both in linehaul and backhaul groups. Based on MTZ model (Miller et al., 1960), constraint 15 eliminates subtours. Constraints 16 and 17 ensure that accumulated deliveries and pickups do not exceed a single vehicle capacity. Constraint 18 ensures that products' or RTIs' demand does not exceed the value of the accumulation variables. Finally, constraints 19 and 20 indicate the variables that are binary and positive, respectively.

4. Solving Small-Scale Instances with Exact Methods

The previous model was implemented in IBM Cplex, which was used to solve instances *eil_22*, *eil_23*, and *eil_30* by Christofides and Eilon (1969). Such instances were adapted for the VRPB by Toth and Vigo (1997) (*TV* instances), using 3 proportions of linehaul (LH) customers: 50%, 66% and 80%. Thus, for instance, in the latter case (80% or 4 out of 5) customers 1, 2, 3, and 4 are LH, while the 5th one is backhaul (BH), and so on. Here, we can assume that the RTIs to be collected were used in previous periods to deliver products to the associated customer.

Firstly, the model is solved as a VRPB in which pickup decisions are integrated with delivery decisions (Figure 1). This is compared with the case in which deliveries and pickups are made independently, i.e., LH customers and BH customers are not serviced in the same route but in different routes. Table 1 shows the associated costs and the convenience of merging the linehaul and backhaul routes since it generates noticeable reductions in cost.

Table 1: Comparison between VRPB and independent deliveries and pickups.

Instances	**LH**	**BH**	**Independent deliveries and pickups**			**VRPB**	**% Cost reduction**
			Deliveries cost	Pickups cost	Total cost	Total cost	
eil22_50	11	10	281	225	506	371	36.4%
eil23_50	11	11	434	348	782	682	14.7%
eil30_50	15	14	328	340	668	501	33.3%

Secondly, four scenarios are considered for the unitary inventory cost (i.e., the penalty cost of not visiting a BH customer), namely: high ($h_i = 0.60$), medium ($h_i = 0.33$), low ($h_i = 0.16$), and very low ($h_i = 0.06$). These values were established after some preliminary tests with the instances. Table 2 shows the results for instances with 22, 23, and 30 nodes. For instances with 22 nodes, all BH customers are visited regardless of the proportion of LH and BH customers. Some BH customers are not visited in other instances, although only if the unitary inventory cost is not high (for instances with 23 nodes), or if it is low or very low (for instances with 30 nodes). These results indicate that the unitary inventory cost might affect the number of non-visited BH customers. As expected, the lower the unitary inventory cost (h_i), the higher the number of non-visited BH. Thus, for example, all BH customers are served in the case of instances with the highest cost ($h_i = 0.60$).

Table 2: Solutions of the exact algorithm for different levels of unitary inventory cost.

Instance	h_i	Total BH demand	LH	BH	Total penalty cost	Total cost	Gap (Cost reduction)	Number of not collected units	% of not collected RTI's	NSBH
eil22_50	0.60	12800	11	10	0.00	371.00	0.00%	0	0.00%	0
eil22_50	0.33				0.00	371.00	0.00%	0	0.00%	0
eil22_50	0.16				0.00	371.00	0.00%	0	0.00%	0
eil22_50	0.06				0.00	371.00	0.00%	0	0.00%	0
eil22_66	0.60	5500	14	7	0.00	366.00	0.00%	0	0.00%	0
eil22_66	0.33				0.00	366.00	0.00%	0	0.00%	0
eil22_66	0.16				0.00	366.00	0.00%	0	0.00%	0
eil22_66	0.06				0.00	366.00	0.00%	0	0.00%	0
eil22_80	0.60	5400	17	4	0.00	375.00	0.00%	0	0.00%	0
eil22_80	0.33				0.00	375.00	0.00%	0	0.00%	0
eil22_80	0.16				0.00	375.00	0.00%	0	0.00%	0
eil22_80	0.06				0.00	375.00	0.00%	0	0.00%	0
eil23_50	0.60	6604	11	11	45.00	682.00	0.00%	75	1.14%	1
eil23_50	0.33				24.75	657.75	-3.56%	75	1.14%	1
eil23_50	0.16				12.00	645.00	-5.43%	75	1.14%	1
eil23_50	0.06				93.24	567.24	-16.83%	1554	23.53%	7
eil23_66	0.60	2080	15	7	0.00	649.00	0.00%	0	0.00%	0
eil23_66	0.33				127.50	601.05	-7.39%	385	18.51%	3
eil23_66	0.16				89.60	533.60	-17.78%	560	26.92%	4
eil23_66	0.06				33.60	477.60	-26.41%	560	26.92%	4
eil23_80	0.60	5050	18	4	0.00	623.00	0.00%	0	0.00%	0
eil23_80	0.33				49.50	608.50	-2.33%	150	2.97%	1
eil23_80	0.16				24.00	583.00	-6.42%	150	2.97%	1
eil23-80	0.06				57.00	542.00	-13.00%	950	18.81%	3
eil30_50	0.60	5825	15	14	0.00	501.00	0.00%	0	0.00%	0
eil30_50	0.33				0.00	501.00	0.00%	0	0.00%	0
eil30_50	0.16				40.00	481.00	-3.99%	250	1.95%	2
eil30_50	0.06				36.00	431.00	-13.97%	600	4.69%	3
eil30_66	0.60	3500	20	9	0.00	537.00	0.00%	0	0.00%	0
eil30_66	0.33				0.00	537.00	0.00%	0	0.00%	0
eil30_66	0.16				16.00	527.00	-1.86%	100	0.78%	1
eil30_66	0.06				6.00	517.00	-3.72%	100	0.78%	1
eil30_80	0.60	1950	24	5	0.00	514.00	0.00%	0	0.00%	0
eil30_80	0.33				0.00	514.00	0.00%	0	0.00%	0
eil30_80	0.16				0.00	514.00	0.00%	0	0.00%	0
eil30_80	0.06				12.00	503.00	-2.14%	200	1.56%	1

In general, not collecting RTIs might yield some savings in total cost when the unitary inventory cost is not high. For example, in instance *eil23_66* the cost for $h_i = 0.60$ (equivalent to 100% of serviced BH customers) is 649. However, when $h_i = 0.33$, the associated inventory cost is 127.50 and the total cost is 601.05 –i.e., equivalent to a reduction of 7.39% in total cost. The behavior is similar for $h_i = 0.16$

and $h_i = 0.06$, both for instances with 23 and 30 nodes. These results show that, if th unitary inventory cost is low, it might pay off not to visit all BH customers.

Figure 2 shows an example of a solution for instance *eil30_50* with $h_i = 0.60$ (high) and $h_i = 0.06$ (very low). All customers are visited in the first case *(a)*, while 3 BH customers are not served in the second case *(b)*. The depot is depicted as node 0. In this example, non-visited customers are those that are more distant from the depot and have the lowest demands (in absolute value). Finally, notice that changes in the number of visited BH customers might also affect the linehaul part of a route.

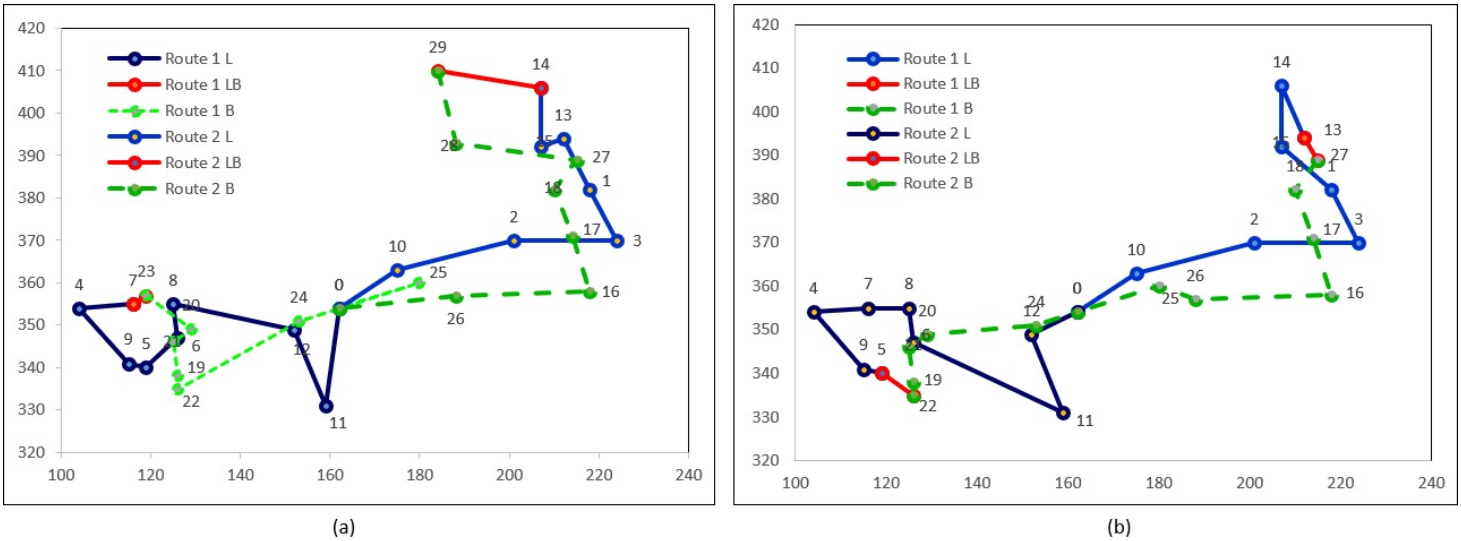

Fig. 2: Optimal solutions for instance *eil30_50* and unitary inventory costs of 0.60 *(a)* and 0.06 *(b)*.

5. A Biased-Randomized Iterated Local Search

(Angel says:) I would like review with Rafael the explanation of the algorithm.

The exact solution is only possible for small TV instances, i.e., up to 30 customers. Executions of larger quantities take long times since this problem is NP-Hard. Therefore, a metaheuristic algorithm based on a Biased-Randomized version of the Clarke & Wright Savings Heuristic (BR-CWS) and Iterated Local Search (ILS) is proposed (Figure 3 and Algorithm 1). Firstly, an initial solution is obtained through the BR-CWS (Algorithm 2 and line 2 in Algorithm 1). The savings list is created for each edge not connected with the depot, however, since each BH customer has a penalty cost for not being served, the traditional way to calculate such savings ($s_{ij} = c_{i0} + c_{0j} - c_{ij}$) is replaced with the expression in Equation (21) (Panadero et al., 2018) only for those edges whose both nodes are backhaul.

$$s_{ij}^{'} = \alpha * s_{ij} + (1 - \alpha) * (h_i + h_j), \ \forall i \in B, \forall j \in B \tag{21}$$

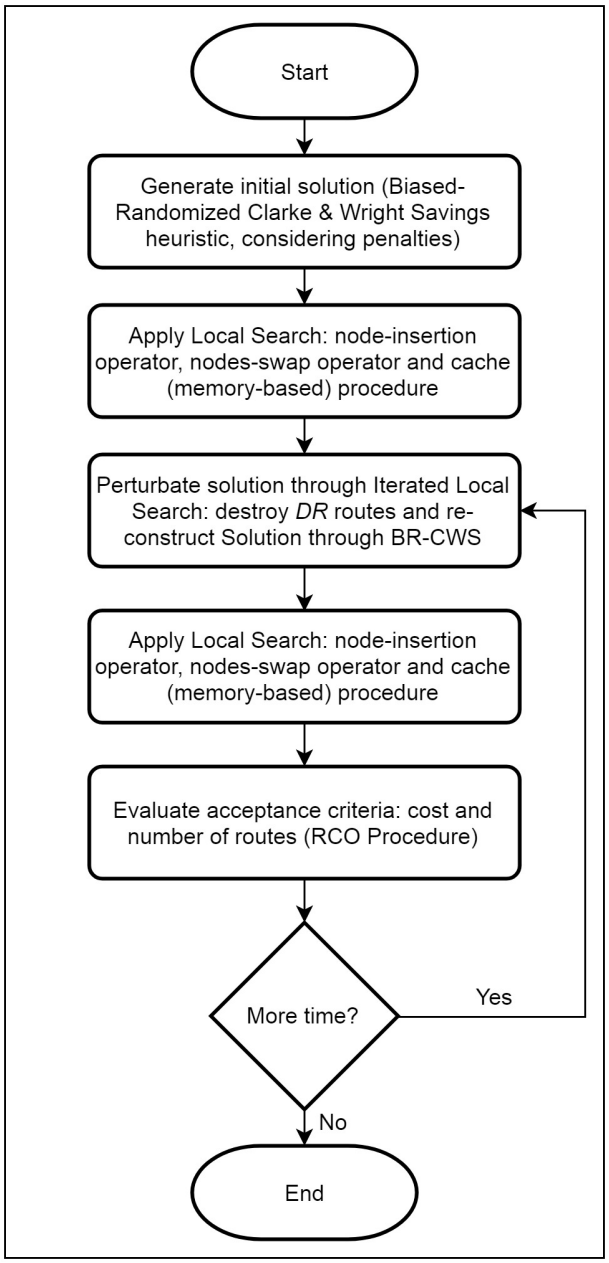

Fig. 3: Flow chart for our approach

Where $0 \leq \alpha \leq 1$. Panadero et al. (2018) affirm that its concrete value only depends on "the heterogeneity of the customers in terms of rewards", i.e, the higher the heterogeneity, the closer to 0 α should be. The case in which $\alpha = 1$ corresponds to the traditional case in CWS, where penalty costs are not considered. In the case in which $\alpha = 0$, savings (and therefore, transport costs) are not considered and

Algorithm 1 Procedure VRPOB(inputParameters, α, β, λ, r)

1: *// Generate initial solution applying Biased-Randomization*
2: newSolution ← penalizedBRCWS(inputParameters, α, β, λ)
3: *// Improve initial solution using Local Search*
4: newSolution ← improveSolutionUsingOperatorsAndHashMap(inputParameters, newSolution)
5: *// Apply ILS*
6: **repeat**
7: pendingNodes ← destroyRoutes(r, newSolution)
8: newSolution ← penalizedBRCWS(inputParameters, α, β, pendingNodes)
9: newSolution ← improveSolutionUsingOperatorsAndHashMap(inputParameters, newSolution)
10: bestSolution ← acceptanceCriteria(bestSolution, newSolution)
11: **until** time reaches the limit
12: **return** bestSolution

only penalty costs are relevant. Intermediate values for α assign more or less weight to transport and penalty costs.

Once computed, these savings values are adapted according to Deif and Bodin (1984) in order to address properly the backhaul nodes. This procedure is described in Algorithm 3. The idea is to delay the selection of interface edges, so that both LH and BH routes are "complete" before being merged. This delay is done by subtracting p from the previously computed savings. The value of p is computed by $p = max_s * \lambda$, in which max_s is the value of the maximum savings that can be attained and λ is a penalty coefficient ranging between 0 and 1. In our case, λ is uniformly chosen between 0.05 and 0.20 (Deif and Bodin, 1984).

After a dummy solution is obtained, the selection of each edge that will be part of the solution route is done through Biased-Randomization (BR) (Algorithm 2) (Juan et al., 2010). Here, a probability is assigned to each edge in the savings list. Then, a geometric distribution is employed in a Monte Carlo simulation in which only one parameter (β) must be fine-tuned. This parameter can be interpreted as the probability of choosing the edge with the highest savings. In our algorithm, the value of β is selected uniform randomly between 0.01 and 0.5. Once selected, the merging conditions are checked (line 11). The two corresponding routes of an edge e can be merged if: *(i)* its nodes are adjacent to the depot, *(ii)* its nodes belong to different routes, and *(iii)* the vehicle capacity constraint is met. If both routes belong to the same cluster, i.e., to either the LH group or the BH group (line 12), the merge is done. Otherwise (line 14), this union is allowed only if total transport cost (line 20) is less than the total penalty cost for not picking up (line 21), i.e., if the total transport cost for BH nodes is high, the algorithm chooses not to collect RTI's in that route. In this case, the union is discarded, and the BH route is penalized. At the end of each iterative step, the selected savings edge is removed from the list (line 29), and the process is repeated until no more options are available (line 3).

This procedure yields an initial solution that can be improved. Our main acceptance criterion is the total cost given by adding transport costs and penalty costs. The search for better solutions initiates with the implementation of Local Search procedures (LS) based on the use of: an inter-routes node-insertion operator, an inter-routes nodes-swap operator, and a cache (memory-based) procedure (hash map) (Algorithm 1, line 4). Since along the whole algorithm execution a high number of iterations are

Algorithm 2 Procedure penalizedBRCWS(inputParameters, α, β, λ)

```
1: savingsList ← penalizeSavingsList(inputParameters, α, λ)
2: savingsList ← sortDescSavingsList(savingsList)
3: while savingsList is not empty do
4:    Randomly select position pos ∈ {1, ..., |savingsList|} according to distribution Geom(β)
5:    e ← selectEdgeFromList(pos, savingsList)
6:    e_s ← getSavingsValue(e)
7:    iNode ← getOrigin(e)
8:    jNode ← getEnd(e)
9:    iR ← getEvolvingRouteOfNode(iNode)
10:   jR ← getEvolvingRouteOfNode(jNode)
11:   if route-merging conditions are met then
12:      if iR and jR are of the same type then
13:         newSolution ← mergeRoutes(newSolution, e)
14:      else
15:         lhRoute ← findLHRoute(iR, jR)
16:         bhRoute ← findBHRoute(iR, jR)
17:         lhRCost ← getCost(lhRoute)
18:         bhRCost ← getCost(bhRoute)
19:         bhPenCost ← getPenalizedCost(bhRoute)
20:         totalTransportCost ← lhRCost + bhRCost - e_s
21:         totalPenaltyCost ← lhRCost + bhPenCost
22:         if totalTransportCost < totalPenaltyCost then
23:            newSolution ← mergeRoutes(newSolution, e)
24:         else
25:            newSolution ← removeBHRoute(newSolution)
26:         end if
27:      end if
28:   end if
29:   removeEdge (savingsList, pos)
30: end while
31: return newSolution
```

carried out, such cache procedure stores in memory the best found route for a given group of nodes. That is, if in an iteration a route connecting some nodes is found (current route) but, in any previous step a route with a better acceptance criterion for the same group of nodes had already been found, the current route is discarded.

Next, the ILS is implemented (Lourenço et al., 2010). The perturbation process consists in destroying at least two routes (Algorithm 1, line 7) and reconstructing them. Such destruction implies that, temporarily, m nodes do not belong to any route. Observe that $m < n$, where n is the total number of nodes in the instance. This implies that a smaller problem must be solved and, therefore, a faster solution will be attained. A destruction ratio (r) is defined here, in order to establish a maximum number of routes

Algorithm 3 Procedure penalizeSavingsList(inputParameters, α, λ)

1: *// Create the savings list*
2: savingsList $\leftarrow$ createSavingsList(inputParameters, α)
3: $max_s \leftarrow$ obtainMaximumSavings(savingsList)
4: **for** each edge in savingsList **do**
5: *// An interface edge connects a LH node with a BH node*
6: **if** edge is interface **then**
7: Randomly select $\lambda \in \{0.05, 0.2\}$
8: $p \leftarrow max_s * \lambda$
9: updateSavings(edge, p)
10: **end if**
11: **end for**
12: **return** savingsList

to be destroyed (DR). In general, $0 < r < 1$, but in our algorithm r is generated as a random number between 0.1 and 0.5. On the one hand, a destruction ratio greater than 0.5 would destroy most of the constructed routes and the process would lose time-efficiency. On the other hand, a destruction ratio less than 0.1 would destroy very few routes and the effect of the perturbation would be almost null. DR is calculated by Equation (22) and is defined as the maximum between 2 and the multiplication of r and the total quantity of routes in the solution. The objective here is to avoid the destruction of only one route.

$$DR = \max\{2,\ r * \#Routes\} \tag{22}$$

Once the destruction process is finished, the reconstruction is made from scratch for the m nodes, by using again the BR-CWS and the LS (Algorithm 1, lines 8-9. Algorithm 2 is implemented again only for the m nodes). Observe that, until now, the constraint regarding the number of vehicles has not been considered, since the BR-CWS relaxes it implicitly. This can lead to temporarily-allowed infeasible solutions in our algorithm execution, which let explore new regions of the solution space. Nevertheless, final solutions must be feasible, and a Recursive Corrective Operator (RCO) is executed at the end of the algorithm to attain it (Belloso et al., 2017). The use of this procedure implies that the *cost* is not the only acceptance criterion, but also the *number of routes* (Algorithm 1, line 10), i.e., the quantity of routes in the solution must equal the number of vehicles.

6. Computational Results and Discussion

The *TV* set of 33 instances introduced by Toth and Vigo (1997) has been solved in order to evaluate the performance of our proposed methodology. These instances are different in the number of linehaul and backhaul nodes, vehicle capacities, and demands. Also, the previous proportions for LH customers were considered here, i.e.: 50%, 66%, and 80%. Initially, α is not taken into account in order to compare our algorithm's results with those by Toth and Vigo (1997) and Belloso et al. (2017). Table 3 shows the comparison between our best solutions (OBS) and the best-known solutions (BKS) in the literature. Since

previous papers do not consider that parameter, the unitary inventory cost (h_i) is assumed to be virtually infinite here in order to make an adequate comparison. In 78.8% of the instances the gap between OBS and the BKS is 0.0%. Actually, OBS is always better than the BKS provided by Toth and Vigo (1997) except for the instance *eil33_50*, in which the gap is 1.5%. For the rest of the instances, Belloso et al. (2017) achieve better solutions than us, although the gap never exceeds 0.9%.

The previous gap needs to be further reduced

These results show that our algorithm obtains competitive solutions when compared with state-of-the-art approaches in the base scenario –where h_i and α are not considered. Now that this point has been made, we can use the algorithm for analyzing new scenarios where collection of items becomes optional and will be made only if it is cost worthy.

In order to calibrate the value of the parameter α, several tests were performed. After these tests, we set the parameter $\alpha \in \{0.2, 0.5, 0.8, 1.0\}$. Regarding the unitary inventory costs, we set $h_i \in \{0.66, 0.33, 0.16, 0.06\}$. In this preliminary stage, only small instances with optimal solutions were considered. Thus, for each combination of instances, unitary inventory cost, and α, a total of 5 runs were performed (each run using a different seed for the pseudo-random number generator). The performance of the metaheuristic was measured as the percentage gap between OBS and the BKS –optimal solution in this case. Figure 4 provides the average gap obtained by the metaheuristic when combined with each α and h_i. As we can see, $\alpha = 0.8$ provides better solutions, on the average, for all values of h_i.

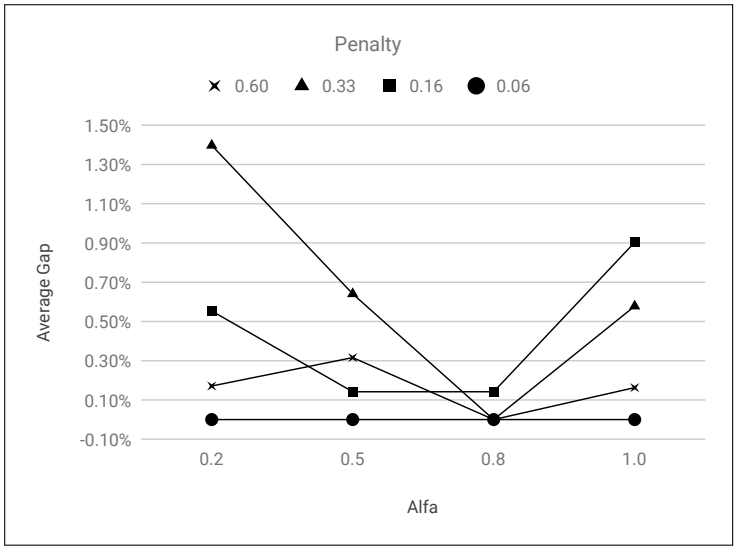

Fig. 4: Average gap obtained by the metaheuristic when combined with each α and penalty values.

Due to the satisfactory performance of our biased-randomized approach in small-sized instances, the metaheuristic was also tested in the larger ones. Thus, for each of the 33 instances, a total of 10 runs were performed. The stopping criterion was fixed to 25, 75, and 300 seconds for instances up to 50 nodes, up to 100 nodes, and with over 100 nodes, respectively. Our algorithms have been coded in Java and a standard PC with an Intel Core i7 CPU at 2.7 GHz and 16 GB RAM has been employed to run all

Table 3: Comparison between our algorithm and previous works.

Instance	LH	BH	Q	K	Toth and Vigo (1997) (BKS in 1997)	Belloso et al. (2017) (BKS in 2017)	OBS ($h_i = \infty$)	Gap OBS-Belloso
eil22_50	11	10	6000	3	371.0	371.0	371.0	0.0%
eil22_66	14	7	6000	3	366.0	366.0	366.0	0.0%
eil22_80	17	4	6000	3	375.0	375.0	375.0	0.0%
eil23_50	11	11	4500	2	682.0	682.0	682.0	0.0%
eil23_66	15	7	4500	2	649.0	649.0	649.0	0.0%
eil23_80	18	4	4500	2	623.0	623.0	623.0	0.0%
eil30_50	15	14	4500	2	501.0	501.0	501.0	0.0%
eil30_66	20	9	4500	3	537.0	537.0	537.0	0.0%
eil30_80	24	5	4500	3	514.0	514.0	514.0	0.0%
eil33_50	16	16	8000	3	738.0	738.0	749.0	1.5%
eil33_66	22	10	8000	3	750.0	750.0	750.0	0.0%
eil33_80	26	6	8000	3	736.0	736.0	736.0	0.0%
eil51_50	25	25	160	3	559.0	559.0	559.0	0.0%
eil51_66	34	16	160	4	548.0	548.0	548.0	0.0%
eil51_80	40	10	160	4	565.0	565.0	565.0	0.0%
eilA76_50	37	38	140	6	739.0	739.0	739.0	0.0%
eilA76_66	50	25	140	7	768.0	768.0	768.0	0.0%
eilA76_80	60	15	140	8	781.0	781.0	781.0	0.0%
eilB76_50	37	38	100	8	801.0	801.0	801.0	0.0%
eilB76_66	50	25	100	10	873.0	873.0	873.0	0.0%
eilB76_80	60	15	100	12	919.0	919.0	919.0	0.0%
eilC76_50	37	38	180	5	713.0	713.0	713.0	0.0%
eilC76_66	50	25	180	6	734.0	734.0	734.0	0.0%
eilC76_80	60	15	180	7	733.0	733.0	733.0	0.0%
eilD76_50	37	38	220	4	690.0	690.0	690.0	0.0%
eilD76_66	50	25	220	5	715.0	715.0	715.0	0.0%
eilD76_80	60	15	220	6	703.0	694.0	695.0	0.1%
eilA101_50	50	50	200	4	843.0	831.0	832.0	0.1%
eilA101_66	67	33	200	6	846.0	846.0	846.0	0.0%
eilA101_80	80	20	200	6	916.0	856.0	857.0	0.1%
eilB101_50	50	50	112	7		923.0	925.0	0.2%
eilB101_66	67	33	112	9		982.0	987.0	0.5%
eilB101_80	80	20	112	11		1008.0	1009.0	0.1%
Average								0.1%

tests.

Using $\alpha = 0.8$, each instance was run 10 times for each h_i value. Tables 4 and 6 show the results obtained by our algorithm. Notice that the value of h_i is higher for Table 6. A new fine-tuning process was carried out to establish their values, namely: high ($h_i = 12.40$), medium ($h_i = 6.20$), low ($h_i = 3.10$), and very low ($h_i = 1.24$). Re-adjusting the values of h_i for these instances is necessary since they have different levels of demand as compared to the smaller ones. For each instance and h_i, the following data is provided: the BKS, OBS, non-serviced BH customers, CPU time (in seconds), and the percentage gap.

The results obtained for small instances (22, 23, and 30 nodes) are the same as the BKS ones (Table 2). The most relevant results are obtained when the number of non-serviced BH customers is greater than zero. Here, 44 solutions show this effect. In 39 out of these (88.6%), the gap is negative, meaning that by allowing collections to be optinal it is possible to generate solutions with a lower total cost than when all BH customers need to be visited.

There is not a general relation between the non-serviced BH customers and the gap. However, in those 5 out of 44 cases in which a gap is positive, there are only 1 or 2 non-serviced BH customer. That is, a higher quantity of non-serviced BH customers usually yields to savings in total cost. This does not mean that the RTIs will never be collected, but just that they can be picked up at another period of the planning horizon. Likewise, BH customers that are visited during the current period might be skipped at the next one.

Table 4: Found solutions for instances with small demands and penalty costs.

Instance	***BKS***	$h_i = 0.66$				$h_i = 0.33$				$h_i = 0.16$				$h_i = 0.06$			
		OBS	**NSBH**	**Time**	**GAP**	**OBS**	**NSBH**	**Time**	**GAP**	**OBS**	**NSBH**	**Time**	**GAP**	**OBS**	**NSBH**	**Time**	**GAP**
eil22_50	371	371.0	0	0.2	0.0%	371.0	0	0.2	0.0%	371.0	0	0.2	0.0%	371.0	0	0.0	0.0%
eil22_66	366	366.0	0	0.2	0.0%	366.0	0	0.2	0.0%	366.0	0	0.1	0.0%	366.0	0	0.0	0.0%
eil22_80	375	375.0	0	1.3	0.0%	375.0	0	0.1	0.0%	375.0	0	0.2	0.0%	375.0	0	0.0	0.0%
eil23_50	682	682.0	0	3.0	0.0%	657.8	1	1.6	-3.6%	645.4	1	16.0	-5.4%	567.2	7	0.4	-16.8%
eil23_66	649	649.0	0	5.8	0.0%	601.1	1	12.4	-7.4%	536.4	4	1.3	-17.3%	477.6	4	0.1	-26.4%
eil23_80	623	623.0	0	0.0	0.0%	608.5	1	0.1	-2.3%	583.8	1	0.0	-6.3%	542.0	3	6.2	-13.0%
eil30_50	501	501.0	0	0.3	0.0%	501.0	0	0.1	0.0%	482.3	2	0.0	-3.7%	431.0	3	0.3	-14.0%
eil30_66	537	537.0	0	0.2	0.0%	537.0	0	0.0	0.0%	527.5	1	0.1	-1.8%	517.0	1	0.0	-3.7%
eil30_80	514	514.0	0	0.7	0.0%	514.0	0	0.1	0.0%	514.0	0	0.2	0.0%	503.0	1	0.0	-2.1%
eil33_50	738	749.0	0	1.8	1.5%	749.0	0	0.8	1.5%	749.0	0	3.8	1.5%	749.0	0	0.3	1.5%
eil33_66	750	750.0	0	2.9	0.0%	750.0	0	0.1	0.0%	750.0	0	0.1	0.0%	750.0	0	0.0	0.0%
eil33_80	736	736.0	0	0.3	0.0%	736.0	0	2.4	0.0%	736.0	0	5.4	0.0%	736.0	0	0.2	0.0%

Table 5: Found solutions for instances with small demands and penalty costs.

Instance	***BKS***	$h_i = 0.66$				$h_i = 0.33$				$h_i = 0.16$				$h_i = 0.06$			
		OBS	**NSBH**	**Time**	**GAP**	**OBS**	**NSBH**	**Time**	**GAP**	**OBS**	**NSBH**	**Time**	**GAP**	**OBS**	**NSBH**	**Time**	**GAP**
eil22_50	371.0	371.0	0	0.3	0.0%	371.0	0	0.3	0.0%	371.0	0	0.1	0.0%	371.0	0	0.0	0.0%
eil22_66	366.0	366.0	0	1.3	0.0%	366.0	0	0.2	0.0%	366.0	0	0.1	0.0%	366.0	0	0.0	0.0%
eil22_80	375.0	375.0	0	0.4	0.0%	375.0	0	0.1	0.0%	375.0	0	0.0	0.0%	375.0	0	0.0	0.0%
eil23_50	682.0	682.0	0	1.3	0.0%	657.8	1	0.7	-3.5%	657.0	1	9.2	-3.7%	567.2	7	0.7	-16.8%
eil23_66	649.0	649.0	0	2.5	0.0%	615.8	1	1.1	-5.1%	533.6	4	0.7	-17.8%	477.6	4	0.0	-26.4%
eil23_80	623.0	623.0	0	0.0	0.0%	608.5	1	0.1	-2.3%	583.0	1	0.0	-6.4%	542.0	3	0.3	-13.0%
eil30_50	501.0	501.0	0	0.1	0.0%	501.0	0	0.2	0.0%	481.0	2	0.4	-4.0%	431.0	3	0.3	-14.0%
eil30_66	537.0	537.0	0	0.0	0.0%	537.0	0	0.0	0.0%	527.0	1	0.0	-1.9%	517.0	1	0.0	-3.7%
eil30_80	514.0	514.0	0	0.5	0.0%	514.0	0	0.2	0.0%	514.0	0	0.1	0.0%	503.0	1	0.0	-2.1%
eil33_50	738.0	749.0	0	0.7	1.5%	749.0	0	0.1	1.5%	749.0	0	0.9	1.5%	749.0	0	0.2	1.5%
eil33_66	750.0	750.0	0	1.2	0.0%	750.0	0	0.7	0.0%	750.0	0	0.1	0.0%	750.0	0	0.0	0.0%
eil33_80	736.0	736.0	0	7.9	0.0%	736.0	0	0.7	0.0%	736.0	0	0.5	0.0%	736.0	0	0.1	0.0%
Average			0.0	1.4	0.1%		0.3	0.4	-0.8%		0.8	1.0	-2.7%		1.6	0.1	-6.2%

The unitary inventory cost is also a parameter that has an influence on cost savings. On the one hand, high values of h_i lead to the collection of most RTIs. In this case, we obtain a gap of 0.0% for most of the instances, especially those in which the number of customers is low. On the other hand, only in a few

Table 6: Found solutions for instances with large demands and penalty costs

Instance	*BKS*	$h_i = 12.40$				$h_i = 6.20$				$h_i = 3.10$				$h_i = 1.24$			
		OBS	**NSBH**	**Time**	**GAP**	**OBS**	**NSBH**	**Time**	**GAP**	**OBS**	**NSBH**	**Time**	**GAP**	**OBS**	**NSBH**	**Time**	**GAP**
eil51_50	559.0	559.0	0	2.1	0.0%	559.0	0	16.0	0.0%	559.0	0	0.7	0.0%	538.6	3	60.0	-3.6%
eil51_66	548.0	548.0	0	36.2	0.0%	548.0	0	18.7	0.0%	548.0	0	0.4	0.0%	541.4	3	1.9	-1.2%
eil51_80	565.0	565.0	0	20.7	0.0%	572.0	0	0.6	1.2%	548.7	1	3.1	-2.9%	535.4	1	0.5	-5.2%
eilA76_50	739.0	739.0	0	5.6	0.0%	739.0	0	6.5	0.0%	736.1	1	1.3	-0.4%	734.2	1	1.1	-0.6%
eilA76_66	768.0	771.0	0	30.5	0.4%	768.0	0	17.6	0.0%	767.1	1	24.6	-0.1%	765.2	1	66.6	-0.4%
eilA76_80	781.0	792.0	0	44.6	1.4%	783.0	0	34.7	0.3%	781.0	0	26.5	0.0%	775.4	1	48.2	-0.7%
eilB76_50	801.0	801.0	0	23.4	0.0%	801.0	0	22.8	0.0%	808.1	1	40.6	0.9%	796.2	1	3.1	-0.6%
eilB76_66	873.0	873.0	0	21.5	0.0%	873.0	0	73.3	0.0%	872.1	1	19.1	-0.1%	870.2	1	6.9	-0.3%
eilB76_80	919.0	920.0	0	26.1	0.1%	919.0	0	63.4	0.0%	919.0	0	17.3	0.0%	913.4	1	4.3	-0.6%
eilC76_50	713.0	713.0	0	10.0	0.0%	713.0	0	33.6	0.0%	713.0	0	3.8	0.0%	711.2	1	1.6	-0.3%
eilC76_66	734.0	734.0	0	62.2	0.0%	741.0	0	62.6	1.0%	734.1	1	34.3	0.0%	731.2	1	8.2	-0.4%
eilC76_80	733.0	743.0	0	16.3	1.4%	735.0	0	52.2	0.3%	736.0	0	4.2	0.4%	728.4	2	10.2	-0.6%
eilD76_50	690.0	690.0	0	17.2	0.0%	690.0	0	8.0	0.0%	690.0	0	1.2	0.0%	689.2	1	2.1	-0.1%
eilD76_66	715.0	720.0	0	15.5	0.7%	715.0	0	41.1	0.0%	715.0	0	2.5	0.0%	715.0	0	2.3	0.0%
eilD76_80	694.0	704.0	0	48.2	1.4%	696.0	0	25.7	0.3%	699.0	0	7.6	0.7%	688.2	3	28.0	-0.8%
eilA101_50	831.0	844.0	0	84.2	1.6%	844.0	0	232.7	1.6%	838.4	2	291.1	0.9%	793.8	10	268.6	-4.5%
eilA101_66	846.0	859.0	0	189.9	1.5%	846.0	0	293.9	0.0%	846.0	0	13.8	0.0%	828.8	6	44.7	-2.0%
eilA101_80	856.0	871.0	0	203.6	1.8%	865.0	0	59.3	1.1%	872.0	0	168.8	1.9%	827.6	6	231.9	-3.3%
eilB101_50	923.0	925.0	0	275.3	0.2%	923.2	1	140.5	0.0%	921.8	4	123.0	-0.1%	884.8	10	167.4	-4.1%
eilB101_66	982.0	1001.0	0	90.3	1.9%	997.0	0	236.4	1.5%	995.0	0	275.4	1.3%	973.2	4	184.5	-0.9%
eilB101_80	1008.0	1025.0	0	38.2	1.7%	1014.0	0	218.7	0.6%	1015.8	2	261.8	0.8%	984.0	6	240.3	-2.4%

cases a very low h_i does not yield savings, which shows the advantage of using our approach for such values of h_i.

7. Conclusions

Setting delivery and collection routing plans represents a challenging problem that has to be addressed on a daily basis in many supply chains. In this paper, we analyze a variant of the vehicle routing problem with linehaul and backhaul customers in which visiting the latter is an optional action that might be skipped. If so, savings in transportation cost can be achieved, although additional inventory costs might be incurred.

To deal with this *NP-hard* combinatorial optimization problem, a mixed-integer linear programming model is proposed and solved for a set of traditional instances. Here, instances up to 30 nodes can be solved using exact methods. For larger instances, a biased-randomized iterated local search is proposed. After showing that our approach is competitive with state-of-the-art methods for solving the traditional vehicle routing problem with backhauls, it is adapted to consider that collection of items might be optional subject to a penalty (inventory holding) cost. Our analysis shows that it is possible to obtain better aggregated (routing plus inventory) costs in scenarios where collection is not mandatory.

Several lines are identified for future research: *(i)* our approach could be extended to consider the multi-period and / or multi-depot cases; and *(ii)* customers' demands or (time-based) transportation costs might be considered as random variables.

Table 7: Found solutions for instances with large demands and penalty costs

Instance	*BKS*	$h_i = 12.40$				$h_i = 6.20$				$h_i = 3.10$				$h_i = 1.24$			
		OBS	**NSBH**	**Time**	**GAP**	**OBS**	**NSBH**	**Time**	**GAP**	**OBS**	**NSBH**	**Time**	**GAP**	**OBS**	**NSBH**	**Time**	**GAP**
eil51_50	559.0	559.0	0	1.6	0.0%	559.0	0	1.2	0.0%	559.0	0	0.8	0.0%	541.2	4	72.3	-3.2%
eil51_66	548.0	548.0	0	4.2	0.0%	548.0	0	0.7	0.0%	548.0	0	0.2	0.0%	542.4	2	0.3	-1.0%
eil51_80	565.0	565.0	0	9.0	0.0%	565.0	0	2.1	0.0%	548.7	1	2.7	-2.9%	535.7	1	1.0	-5.2%
eilA76_50	739.0	739.0	0	2.0	0.0%	739.0	0	1.2	0.0%	736.1	1	1.9	-0.4%	734.2	1	1.6	-0.6%
eilA76_66	768.0	771.0	0	44.8	0.4%	771.0	0	30.5	0.4%	767.1	1	70.7	-0.1%	765.2	1	4.6	-0.4%
eilA76_80	781.0	792.0	0	72.8	1.4%	781.0	0	17.3	0.0%	781.0	0	56.4	0.0%	775.7	1	55.8	-0.7%
eilB76_50	801.0	801.0	0	11.8	0.0%	801.0	0	2.1	0.0%	798.1	1	6.6	-0.4%	796.2	1	3.9	-0.6%
eilB76_66	873.0	873.0	0	56.6	0.0%	873.0	0	21.4	0.0%	872.1	1	11.2	-0.1%	870.2	1	2.1	-0.3%
eilB76_80	919.0	920.0	0	60.2	0.1%	919.0	0	37.0	0.0%	919.0	0	5.4	0.0%	913.7	1	19.7	-0.6%
eilC76_50	713.0	713.0	0	6.6	0.0%	713.0	0	9.7	0.0%	713.0	0	5.8	0.0%	711.2	1	0.7	-0.3%
eilC76_66	734.0	736.0	0	68.1	0.3%	737.0	0	74.3	0.4%	733.1	1	33.0	-0.1%	731.2	1	1.6	-0.4%
eilC76_80	733.0	739.0	0	73.7	0.8%	736.0	0	73.0	0.4%	738.0	0	24.2	0.7%	729.1	2	25.2	-0.5%
eilD76_50	690.0	690.0	0	23.7	0.0%	690.0	0	15.2	0.0%	690.0	0	1.4	0.0%	689.2	1	1.1	-0.1%
eilD76_66	715.0	717.0	0	16.4	0.3%	715.0	0	12.9	0.0%	715.0	0	1.7	0.0%	715.0	0	7.0	0.0%
eilD76_80	694.0	699.0	0	29.3	0.7%	701.0	0	64.0	1.0%	695.0	0	13.5	0.1%	693.1	2	17.9	-0.1%
eilA101_50	831.0	845.0	0	282.5	1.7%	834.2	1	172.9	0.4%	838.4	2	27.9	0.9%	797.8	11	282.5	-4.0%
eilA101_66	846.0	848.0	0	246.6	0.2%	846.0	0	38.6	0.0%	846.0	0	7.8	0.0%	830.0	6	32.7	-1.9%
eilA101_80	856.0	872.0	0	140.0	1.9%	871.0	0	260.0	1.8%	860.0	0	222.4	0.5%	838.7	7	79.2	-2.0%
eilB101_50	923.0	925.0	0	108.3	0.2%	926.2	1	43.0	0.3%	921.8	4	76.2	-0.1%	879.8	8	84.9	-4.7%
eilB101_66	982.0	999.0	0	286.0	1.7%	992.0	0	119.4	1.0%	984.3	1	138.7	0.2%	963.8	4	256.7	-1.9%
eilB101_80	1008.0	1008.0	0	268.0	0.0%	1011.0	0	238.6	0.3%	1012.8	2	64.5	0.5%	986.0	6	278.2	-2.2%
Average			0.0	86.3	0.5%		0.1	58.8	0.3%		0.7	36.8	-0.1%		3.0	58.5	-1.5%

Acknowledgments

This work has been partially supported by COLCIENCIAS - Colombia, the School of Industrial Engineering of Universidad del Valle, the IoF2020, the AGAUR (2018-LLAV-00017), and the Erasmus+ Program (2018-1-ES01-KA103-049767). We also acknowledge the support of the doctoral programs at the Universitat Oberta de Catalunya and the Universidad de La Sabana.

References

Agra, A., Cerveira, A., Requejo, C., 2016. Lagrangian relaxation bounds for a production-inventory-routing problem. In *International Workshop on Machine Learning, Optimization and Big Data*, Springer, pp. 236–245.

Andersson, H., Hoff, A., Christiansen, M., Hasle, G., Løkketangen, A., 2010. Industrial aspects and literature survey: Combined inventory management and routing. *Computers & Operations Research* 37, 9, 1515–1536.

Anily, S., Federgruen, A., 1990. One warehouse multiple retailer systems with vehicle routing costs. *Management Science* 36, 1, 92–114.

Arab, R., Ghaderi, S., Tavakkoli-Moghaddam, R., 2018. Bi-objective inventory routing problem with backhauls under transportation risks: two meta-heuristics. *Transportation Letters* pp. 1–17.

Archetti, C., Speranza, M.G., 2016. The inventory routing problem: The value of integration. *International Transactions in Operational Research* 23, 3, 393–407.

Belloso, J., Juan, A.A., Faulin, J., 2019. An iterative biased-randomized heuristic for the fleet size and mix vehicle-routing problem with backhauls. *International Transactions in Operational Research* 26, 1, 289–301.

Belloso, J., Juan, A.A., Martinez, E., Faulin, J., 2017. A biased-randomized metaheuristic for the vehicle routing problem with clustered and mixed backhauls. *Networks* 69, 3, 241–255.

Chien, T.W., Balakrishnan, A., Wong, R.T., 1989. An integrated inventory allocation and vehicle routing problem. *Transportation science* 23, 2, 67–76.

Christofides, N., Eilon, S., 1969. An algorithm for the vehicle-dispatching problem. *Journal of the Operational Research Society* 20, 3, 309–318.

Coelho, L.C., Cordeau, J.F., Laporte, G., 2013. Thirty years of inventory routing. *Transportation Science* 48, 1, 1–19.

Deif, I., Bodin, L., 1984. Extension of the clarke and wright algorithm for solving the vehicle routing problem with backhauling. In *Proceedings of the Babson conference on software uses in transportation and logistics management*, Babson Park, MA, pp. 75–96.

Deng, S., Li, Y., Guo, H., Liu, B., 2016. Solving a closed-loop location-inventory-routing problem with mixed quality defects returns in e-commerce by hybrid ant colony optimization algorithm. *Discrete Dynamics in Nature and Society* 2016.

Dominguez, O., Guimarans, D., Juan, A.A., de la Nuez, I., 2016. A biased-randomised large neighbourhood search for the two-dimensional vehicle routing problem with backhauls. *European Journal of Operational Research* 255, 2, 442–462.

Elia, V., Gnoni, M.G., 2015. Designing an effective closed loop system for pallet management. *International Journal of Production Economics* 170, 730–740.

Ferone, D., Gruler, A., Festa, P., Juan, A.A., 2018. Enhancing and extending the classical grasp framework with biased randomisation and simulation. *Journal of the Operational Research Society* pp. 1–14.

Gholamian, M., Heydari, M., 2017. An inventory model with metric approach in location-routing-inventory problem. *Advances in Production Engineering & Management* 12, 2, 115.

Ghorbani, A., Jokar, M.R.A., 2016. A hybrid imperialist competitive-simulated annealing algorithm for a multisource multi-product location-routing-inventory problem. *Computers & Industrial Engineering* 101, 116–127.

Glock, C.H., 2017. Decision support models for managing returnable transport items in supply chains: A systematic literature review. *International Journal of Production Economics* 183, 561–569.

Govindan, K., Soleimani, H., 2017. A review of reverse logistics and closed-loop supply chains: a journal of cleaner production focus. *Journal of Cleaner Production* 142, 371–384.

Gruler, A., Panadero, J., de Armas, J., Pérez, J.A.M., Juan, A.A., 2018a. Combining variable neighborhood search with simulation for the inventory routing problem with stochastic demands and stock-outs. *Computers & Industrial Engineering* 123, 278–288.

Gruler, A., Panadero, J., de Armas, J., Pérez, J.A.M., Juan, A.A., 2018b. A variable neighborhood search simheuristic for the multiperiod inventory routing problem with stochastic demands. *International Transactions in Operational Research*

Hellström, D., Johansson, O., 2010. The impact of control strategies on the management of returnable transport items. *Transportation Research Part E: Logistics and Transportation Review* 46, 6, 1128–1139.

Hiassat, A., Diabat, A., Rahwan, I., 2017. A genetic algorithm approach for location-inventory-routing problem with perishable products. *Journal of manufacturing systems* 42, 93–103.

Iassinovskaia, G., Limbourg, S., Riane, F., 2017. The inventory-routing problem of returnable transport items with time windows and simultaneous pickup and delivery in closed-loop supply chains. *International Journal of Production Economics* 183, 570–582.

ISO, 2016. ISO/IEC 19762:2016. Information technology – Automatic identification and data capture (AIDC) techniques – Harmonized vocabulary.

Jacobs-Blecha, C., Goetschalckx, M., 1992. The vehicle routing problem with backhauls: properties and solution algorithms. *National Transportation Research Board* 13.

Juan, A.A., Faulin, J., Ferrer, A., Lourenço, H.R., Barrios, B., 2013. Mirha: multi-start biased randomization of heuristics with adaptive local search for solving non-smooth routing problems. *Top* 21, 1, 109–132.

Juan, A.A., Faulin, J., Ruiz, R., Barrios, B., Caballé, S., 2010. The sr-gcws hybrid algorithm for solving the capacitated vehicle routing problem. *Applied Soft Computing* 10, 1, 215–224.

Juan, A.A., Grasman, S.E., Caceres-Cruz, J., Bektaş, T., 2014a. A simheuristic algorithm for the single-period stochastic inventory-routing problem with stock-outs. *Simulation Modelling Practice and Theory* 46, 40–52.

Juan, A.A., Lourenço, H.R., Mateo, M., Luo, R., Castella, Q., 2014b. Using iterated local search for solving the flow-shop problem: Parallelization, parametrization, and randomization issues. *International Transactions in Operational Research* 21, 1, 103–126.

Koç, Ç., Laporte, G., 2018. Vehicle routing with backhauls: Review and research perspectives. *Computers & Operations Research* 91, 79–91.

Kroon, L., Vrijens, G., 1995. Returnable containers: an example of reverse logistics. *International Journal of Physical Distribution & Logistics Management* 25, 2, 56–68.

Küçükoğlu, İ., Öztürk, N., 2015. An advanced hybrid meta-heuristic algorithm for the vehicle routing problem with backhauls and time windows. *Computers & Industrial Engineering* 86, 60–68.

Kyee, D.L.T., Moin, N.H., 2016. A two-phase iterative procedure for the production, inventory and distribution routing problem. In *AIP Conference Proceedings*, Vol. 1750, AIP Publishing, p. 030032.

Lee, Y., Chan, C.K., Langevin, A., Lee, H.W.J., 2016. Integrated inventory-transportation model by synchronizing delivery and production cycles. *Transportation Research Part E: Logistics and Transportation Review* 91, 68–89.

Li, K., 2016. A new discrete particle swarm optimization for location inventory routing problem in cold logistics. *Revista de la Facultad de Ingeniería* 31, 5.

Li, Y., Chu, F., Chu, C., Zhou, W., Zhu, Z., 2016. Integrated production inventory routing planning with time windows for perishable food. In *Intelligent Transportation Systems (ITSC), 2016 IEEE 19th International Conference on*, IEEE, pp. 2651–2656.

Lin, S., Bard, J.F., Jarrah, A.I., Zhang, X., Novoa, L.J., 2017. Route design for last-in, first-out deliveries with backhauling. *Transportation Research Part C: Emerging Technologies* 76, 90–117.

Lourenço, H.R., Martin, O.C., Stützle, T., 2010. Iterated local search: Framework and applications. In *Handbook of metaheuristics*. Springer, pp. 363–397.

Malladi, K.T., Sowlati, T., 2018. Sustainability aspects in inventory routing problem: A review of new trends in the literature. *Journal of Cleaner Production*

Miller, C.E., Tucker, A.W., Zemlin, R.A., 1960. Integer programming formulation of traveling salesman problems. *Journal of the ACM (JACM)* 7, 4, 326–329.

Mostafa, N., Eltawil, A., 2015. A mixed-integer programming model for the production-inventory-distribution-routing problem. In *Industrial Engineering and Engineering Management (IEEM), 2015 IEEE International Conference on*, IEEE, pp. 310–314.

Panadero, J., Freixes, A., Mozos, J.M., Juan, A.A., 2018. Agile optimization for routing unmanned aerial vehicles under uncertainty. In *XVIII Spanish Conference on Artificial Intelligence*, pp. 635–640.

Qin, L., Yang, P., Miao, L., 2016. Model and algorithm for inventory/routing decision in a delivery and pickup system. *Journal of Computational and Theoretical Nanoscience* 13, 1, 558–566.

Reil, S., Bortfeldt, A., Mönch, L., 2018. Heuristics for vehicle routing problems with backhauls, time windows, and 3d loading constraints. *European Journal of Operational Research* 266, 3, 877–894.

Shariff, S.S.R., Omar, M., Moin, N.H., 2016. Location routing inventory problem with transshipment points using p-center. In *Industrial Engineering, Management Science and Application (ICIMSA), 2016 International Conference on*, IEEE, pp. 1–5.

Soysal, M., 2016. Closed-loop inventory routing problem for returnable transport items. *Transportation Research Part D: Transport and Environment* 48, 31–45.

Toth, P., Vigo, D., 1997. An exact algorithm for the vehicle routing problem with backhauls. *Transportation science* 31, 4, 372–385.

Vahdani, B., Niaki, S., Aslanzade, S., 2017. Production-inventory-routing coordination with capacity and time window constraints for perishable products: Heuristic and meta-heuristic algorithms. *Journal of Cleaner Production* 161, 598–618.

Wassan, N., 2007. Reactive tabu adaptive memory programming search for the vehicle routing problem with backhauls. *Journal of the Operational Research Society* 58, 12, 1630–1641.

Wu, W., Tian, Y., Jin, T., 2016. A label based ant colony algorithm for heterogeneous vehicle routing with mixed backhaul. *Applied Soft Computing* 47, 224–234.

Yingfei, Z., Shuxia, Z., Xiaojing, C., Fang, L., 2011. Application of modified shapley value in gains allocation of closed-loop supply chain under third-party reclaim. *Energy Procedia* 5, 980–984.

Zachariadis, E.E., Kiranoudis, C.T., 2012. An effective local search approach for the vehicle routing problem with backhauls. *Expert Systems with Applications* 39, 3, 3174–3184.

Zhalechian, M., Tavakkoli-Moghaddam, R., Zahiri, B., Mohammadi, M., 2016. Sustainable design of a closed-loop location-routing-inventory supply chain network under mixed uncertainty. *Transportation Research Part E: Logistics and Transportation Review* 89, 182–214.

Zhao, J., Ke, G.Y., 2017. Incorporating inventory risks in location-routing models for explosive waste management. *Interna-*

tional Journal of Production Economics 193, 123–136.