CIRRELT

CIRRELT-2020-22

Comparison of Symmetry Breaking and Input Ordering Techniques for Routing Problems

Maryam Darvish
Leandro C. Coelho
Raf Jans

July 2020

Document de travail également publié par la Faculté des sciences de l'administration de l'Université Laval, sous le numéro FSA-2020-008.

Bureau de Montréal
Université de Montréal
C.P. 6128, succ. Centre-Ville
Montréal (Québec) H3C 3J7
Tél. : 1-514-343-7575
Télécopie : 1-514-343-7121

Bureau de Québec
Université Laval,
2325, rue de la Terrasse
Pavillon Palasis-Prince, local 2415
Québec (Québec) G1V 0A6
Tél. : 1-418-656-2073
Télécopie : 1-418-656-2624

Comparison of Symmetry Breaking and Input Ordering Techniques for Routing Problems

Maryam Darvish[1,2,*], Leandro C. Coelho[1,2,3], Raf Jans[1,4]

[1] Interuniversity Research Centre on Enterprise Networks, Logistics and Transportation (CIRRELT)

[2] Department of Operations and Decision Systems, Université Laval, Québec, Canada

[3] Canada Research Chair in Integrated Logistics, Université Laval, Québec, Canada

[4] Department of Logistics and Operations Management, HEC Montréal, Québec, Canada

Abstract. In this paper, we consider routing problems with identical vehicles. In their standard formulations, decision variables (such as the routing decisions and delivery quantities) often have a vehicle index present. For such formulations, alternative solutions exist since the vehicles are identical, and routes can be assigned to different vehicles without changing the objective function value. The existence of these symmetrical solutions causes duplication in the branch-and-bound tree and leads to long computing time. To date, some symmetry breaking constraints have been proposed to deal with this issue. However, to the best of our knowledge, no direct comparison among them has been performed yet. In this paper, besides comparing these symmetry breaking constraints, we propose new constraints and ways for formulating routing problems. Moreover, in order to better exploit each of the formulations, we propose and test several input ordering techniques. We analyze all these on a multi-vehicle inventory routing problem and present and discuss detailed and extensive computational experiments. Our experiments show that the best method to break symmetry is to give an order to the customers. Even after combining this method with other symmetry breaking constraints, it remains the most dominant one. Our results also demonstrate the interdependence between symmetry breaking and input ordering techniques.

Keywords: vehicle routing, symmetry breaking, input ordering, formulations.

Acknowledgements. This project was partly funded by the Natural Sciences and Engineering Council of Canada (NSERC) under grants 2020-00401, 2019-00094 and 2014-03849. This support is greatly acknowledged. We also thank Calcul Québec for computing resources.

---

* Corresponding author: maryam.darvish@FSA.ulaval.ca

Dépôt légal – Bibliothèque et Archives nationales du Québec
Bibliothèque et Archives Canada, 2020

1. Introduction

The effectiveness of solving combinatorial optimization problems using a branch-and-bound/cut (B&B/C) algorithm relies mainly on the structure of its mathematical formulation. Therefore, the formulation not only needs to be *mathematically correct* but also it has to be *good* (Sherali and Driscoll, 2000). Good formulations are known to encompass two important characteristics: they are tight and free from solution symmetry (Sherali and Smith, 2001). In branching techniques, a relaxed version of the problem is solved iteratively, and the search space is explored (Barnhart et al., 1993). Therefore, by tightening the constraints, fewer subproblems need to be solved (integrality is achieved faster), and by breaking the symmetries visiting equivalent solutions can be avoided. Once applied, both of these techniques result in faster computation and dramatically better performance. Most of the research focus, however, has been on tightening the formulation rather than breaking the symmetry (Sherali and Smith, 2001). A problem is called symmetric if by changing its variables, the structure of the problem does not alter (Margot, 2010). Several research communities have studied various techniques for dealing with symmetric problems, yielding similar approaches (Puget, 2005). These techniques can be applied to variables, values, or both (Walsh, 2006). Additionally, input ordering strategies are proven to be effective in symmetry breaking (Jans and Desrosiers, 2013; Coelho and Laporte, 2014; Aziez et al., 2020).

In this paper, we focus on the symmetry that is present in vehicle routing problems with multiple identical vehicles. The Vehicle Routing Problem (VRP) is one of the most studied problems in combinatorial optimization. In its classical version, originating from a depot, a set of vehicles with a limited capacity distribute a single product to several customers and return to the depot. The demand for each customer must be met while the (routing) cost needs to be minimized. Standard formulations for the VRP, in which customers and routes are assigned to specific vehicles, give rise to many alternative solutions with the same total cost. For any given solution, we can indeed permute the vehicles to obtain an equivalent solution with exactly the same value for the objective function. The alternative optimal solutions cannot be pruned merely based on the obtained dual bound. In the absence of some form of symmetry breaking, each path to an alternative optimal solution has to be explored until a feasible solution is obtained. Depending on the bounds, a number of paths to alternative but equivalent non-optimal solutions might also have to be explored. The presence of symmetry in these problems causes, hence, much duplication in the B&B search, which consequently slows down the solution process. The inherent symmetry makes such problems extremely difficult, if not downright impossible, to be solved to optimality in a reasonable time using integer linear program (ILP) solvers.

An extension of the VRP is the Inventory-Routing Problem (IRP), where the quantities delivered to customers over time are also decision variables. In the IRP, as the name suggests, the goal is to optimize the integration of inventory and routing decisions. To date, several symmetry breaking constraints are proposed in the VRP/IRP literature. However, to the best of our knowledge, no direct comparison among different symmetry breaking constraints has yet been yet performed. This paper aims to provide such comparisons as a guideline for symmetry breaking in the integrated routing problems. In this paper, our approach to break symmetry is to change the a priori formulation. This can be done in two different ways. The first one is to add symmetry breaking constraints (SBC) to the original formulation while the second one is to reformulate the problem using new decision variables so that the new formulation no longer allows symmetric solutions.

The contributions of this paper are as follows. Given the limited number of symmetry breaking constraints presented for multi-vehicle routing problems and the fact that no direct comparison has been yet performed to evaluate the relative effectiveness of these constraints, our first contribution is to propose several new SBCs and compare them in a computational experiment against the already existing ones. Besides comparing several existing symmetry breaking techniques from the literature, we propose new techniques and a new formulation for IRPs. Moreover, in order to better exploit each of the formulations, we propose and test several input ordering techniques. We analyze all these for a multi-vehicle IRP and present and discuss detailed and extensive computational experiments.

The remainder of this paper is organized as follows. In Section 2, we provide a review of the related literature. In Section 3, we present the formal description and mathematical formulation of the problem. We consider several symmetry breaking techniques that are elaborated in Section 4. We present our extensive computational results along with elaborate sensitivity analyses and discussions of the results in Section 5. Finally, conclusions are drawn in Section 6.

2. Literature review

In the last decade, growing attention has been observed in the Mixed Integer Programming (MIP) community on how to handle the symmetry issue. Plastria (2002), Puget (2005), and Margot (2010) provide a detailed overview of several techniques for symmetry breaking.

Notably, the literature on symmetry breaking suggests that the input parameters' order can have a significant effect on computational performance (Jans and Desrosiers, 2013). This has been confirmed for the multi-vehicle routing problem in the experiments of Coelho and Laporte (2014). They test three specific orderings against a random one. The results indicate that ordering the customers based on either the highest demand or on the highest distance yields improved results. Nevertheless, most attempts in the literature focus on applying symmetry breaking techniques.

Generally, one can classify these symmetry breaking approaches into two broad categories. The first category is to change the formulation, so that (some of) the alternative solutions are excluded, and then the new formulation is solved using a standard B&B-based solver. The second one is to exploit an algorithm, so that symmetry is detected and dealt with during the B&B process. In this section, we review research on symmetry breaking techniques first and then draw particular attention to the papers on symmetry breaking for routing problems.

2.1. Problem reformulation

As the name suggests, the main idea here is to reformulate the problem by adding several SBCs. The reformation results in some symmetric solutions to become infeasible (Costa et al., 2013). Therefore, either symmetry breaking inequalities need to be added to the model to reduce the number of possibilities for different solutions or, the original problem has to be reformulated so that it becomes asymmetric.

2.1.1. Symmetry breaking constraints

In many cases, the symmetry inherent in a formulation can be reduced by a priori fixing some variables. Such a variable reduction technique has been applied for many different problems, such as partitioning problems (Caprara, 1998), the problem of scheduling a doubles tennis tournament (Ghoniem and Sherali, 2010), grouping objects in identical clusters (Sherali and Desai, 2005; Denton et al., 2010), job grouping (Jans and Desrosiers, 2013), and the safe set and connected safe set problems (Hosteins, 2020).

Another popular approach for adding SBCs is to impose a hierarchy. The symmetry can be caused by permuting the identical objects. Therefore, to break symmetry, one can impose an increasing or decreasing order according to some specific rules. There are several applications for this technique, such as in the area of telecommunication network design (Sherali et al., 2000), process scheduling problems (Mouret et al., 2011), temporal bin packing problem (De Cauwer et al., 2016), an integrated process configuration, lot-sizing, and scheduling problem (Martínez et al., 2019), stochastic edge partition problem (Taşkın et al., 2009), blockmodelling (Proll, 2007), minimizing the total treatment time in cancer radiotherapy (Wake et al., 2009), a combined lot sizing and scheduling problem (Kim et al., 2010), among others. In some papers, a lexicographic order is considered (e.g., (Jans, 2009; Liberti and Ostrowski, 2014; Bendotti et al., 2020)).

In most studies, It should be noted that a combination of variable reduction and order imposing is considered, e.g., in Hosteins (2020) and Sherali and Desai (2005); Vo-Thanh et al. (2018).

Another technique that has been proposed in the literature to deal with symmetry is objective perturbation (Ghoniem and Sherali, 2011). This technique is used in conjunction with hierarchical symmetry breaking constraints, a priori added to the formulation.

For the routing problem, Coelho and Laporte (2013a) propose symmetry breaking constraints for the multi-vehicle IRP, which impose a hierarchy on the vehicles. Adulyasak et al. (2014) use some other symmetry breaking constraints imposing a lexicographic ordering, as also done in Jans and Desrosiers (2010).

2.1.2. Asymmetric representatives formulation

The asymmetric representatives formulation (ARF) is another technique for breaking the symmetry. As the name suggests, instead of using the original formulation, a new formulation free from symmetry is proposed. The ARF is first introduced by Campêlo et al. (2008) for the node coloring problem. Melo and Ribeiro (2015) use the ARF for the freight consolidation and containerization problem, and Jans and Desrosiers (2013) apply it to the job grouping problem. In Braga et al. (2017), it is proposed to solve a minimum chromatic violation problem. In order to find an optimal orthogonal blocking pattern for an orthogonal design, Vo-Thanh et al. (2018) also use the ARF. Jans and Desrosiers (2010) indicate that the ARF idea can also be applied to multi-vehicle routing problems, and it has provided excellent results in its application to the multi-pickup and delivery problem with time windows (Aziez et al., 2020).

2.2. Algorithmic symmetry breaking

Another stream of research focuses on detecting and dealing with the symmetry issue in the solution algorithm. Modern MIP solvers are already equipped with these strategies. Although not well documented, techniques such as orbital fixing (Pfetsch and Rehn, 2019) are used to solve NP-hard problems. As this is not the focus of our approach, we only provide some brief references.

Margot (2002) proposes algorithms for isomorphism pruning and variable fixing, which can be used when the symmetry group is (partially) known and part of the input. Isomorphism pruning is also used to solve the football pool problem (Linderoth et al., 2009).

An alternative method is the orbital branching method proposed by Ostrowski et al. (2011). Research related to this latter approach includes the works on orbital branching (Ostrowski et al., 2011, 2015), orbitopal fixing (Kaibel et al., 2007), orbital independence (Dias and Liberti, 2019), and an isomorphism pruning algorithm and variable setting procedures using orbits of the symmetry group (Margot, 2003), and subtree splitting strategy (Fidalgo et al., 2018).

Interested readers are referred to Pfetsch and Rehn (2019) for a computational performance comparison of several of these algorithms.

2.3. Positioning of this paper

Currently, it is not clear which of the proposed approaches is the best. Therefore, the ultimate goal of this paper is to compare the effectiveness of various SBCs and reformulations for multi-vehicle routing problems in a comprehensive computational experiment. We test the symmetry breaking constraints used in Coelho and Laporte (2013a), and Adulyasak et al. (2014), as well as the reformulation, suggested in Jans and Desrosiers (2010). Besides, we propose various other new symmetry breaking constraints. In a first experiment, nine different SBCs and all their combinations are tested on the data set from Archetti et al. (2007). Recent research (Jans and Desrosiers, 2010, 2013; Coelho and Laporte, 2013a) indicates that the input parameters' order can impact the computational performance of symmetry breaking constraints. We assess, in the second set of computational experiments, the impact of several input ordering strategies. In addition to the three strategies proposed by Coelho and Laporte (2014), we develop and test several new strategies.

3. Description of the Problem and Mathematical Formulation

We consider a multi-vehicle IRP with a single product and symmetric routing costs, as described by Coelho and Laporte (2013a). We define an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{0, \dots, n\}$ is the vertex set and $\mathcal{E} = \{(i,j) : i,j \in \mathcal{V}, i < j\}$ is the edge set. The supplier is indicated by vertex 0, while the n customers are represented by the set of remaining vertices $\mathcal{V}' = \mathcal{V} \setminus \{0\}$. The problem is defined over a planning horizon with length p, and $\mathcal{T}$ is the set of all periods. For each customer i and period t, the demand d_i^t is known. In this problem, we consider two different types of costs. First, a routing cost c_{ij} is incurred if a vehicle travels on the edge $(i,j) \in \mathcal{E}$. Second, an inventory holding cost h_i has to be paid for each unit of product that remains in inventory at the end of a period either at the plant ($i = 0$), or at one of the customers ($i \in \mathcal{V}'$). The quantity of inventory held at each customer i is limited by C_i. For each period $t \in \mathcal{T}$, r^t represents the amount of the product newly made available at the supplier (e.g., through predetermined production or delivery). This amount is a known parameter. No backlogging is allowed, and we assume that the supplier has sufficient inventory to meet the demand of all customers in each period. There might be some initial inventory available at the beginning of the planning horizon, either at customers or at the supplier. This initial inventory level is represented by the parameter I_i^0 ($i \in \mathcal{V}$). We further assume zero leadtimes, i.e., a demand in period t can be satisfied by the quantity r^t. A set $\mathcal{K} = \{1, \dots, K\}$ of identical vehicles is available to perform the routes from the suppliers to a subset of customers. Each vehicle k has a capacity Q, and can perform at most one route per period.

Moreover, we define the following decision variables. The routing variables x_{ij}^{kt} indicate the number of times edge (i,j) is used by vehicle k in period t. The binary variables y_i^{kt} are equal to one if and only if vertex i is visited by vehicle k in period t. The variables I_i^t represent the inventory level at vertex $i \in \mathcal{V}$ at the end of period $t \in \mathcal{T}$. Finally, variables q_i^{kt} are the quantity delivered by vehicle k to customer i in period t. The problem can then be formulated as follows (Coelho and Laporte, 2013a):

$$\text{minimize} \sum_{i \in \mathcal{V}} \sum_{t \in \mathcal{T}} h_i I_i^t + \sum_{(i,j) \in \mathcal{E}} \sum_{k \in \mathcal{K}} \sum_{t \in \mathcal{T}} c_{ij} x_{ij}^{kt}, \tag{1}$$

subject to

$$I_0^t = I_0^{t-1} + r^t - \sum_{k \in \mathcal{K}} \sum_{i \in \mathcal{V}'} q_i^{kt} \quad t \in \mathcal{T} \tag{2}$$

$$I_i^t = I_i^{t-1} + \sum_{k \in \mathcal{K}} q_i^{kt} - d_i^t \quad i \in \mathcal{V}' \quad t \in \mathcal{T} \tag{3}$$

$$I_i^{t-1} + \sum_{k \in \mathcal{K}} q_i^{kt} \leq C_i \quad i \in \mathcal{V}' \quad t \in \mathcal{T} \tag{4}$$

$$q_i^{kt} \leq C_i y_i^{kt} \quad i \in \mathcal{V}' \quad k \in \mathcal{K} \quad t \in \mathcal{T} \tag{5}$$

$$\sum_{i \in \mathcal{V}'} q_i^{kt} \leq Q y_0^{kt} \quad k \in \mathcal{K} \quad t \in \mathcal{T} \tag{6}$$

$$\sum_{j \in \mathcal{V}, i<j} x_{ij}^{kt} + \sum_{j \in \mathcal{V}, j<i} x_{ji}^{kt} = 2 y_i^{kt} \quad i \in \mathcal{V} \quad k \in \mathcal{K} \quad t \in \mathcal{T} \tag{7}$$

$$\sum_{i \in \mathcal{S}} \sum_{j \in \mathcal{S}, i<j} x_{ij}^{kt} \leq \sum_{i \in \mathcal{S}} y_i^{kt} - y_m^{kt} \quad \mathcal{S} \subseteq \mathcal{V}' \quad k \in \mathcal{K} \quad t \in \mathcal{T} \quad m \in \mathcal{S} \tag{8}$$

$$\sum_{k \in \mathcal{K}} y_i^{kt} \leq 1 \quad i \in \mathcal{V}' \quad t \in \mathcal{T} \tag{9}$$

$$I_i^t, q_j^{kt} \geq 0 \quad i \in \mathcal{V} \quad j \in \mathcal{V}' \quad k \in \mathcal{K} \quad t \in \mathcal{T} \tag{10}$$

$$x_{0i}^{kt} \in \{0, 1, 2\} \quad i \in \mathcal{V}' \quad k \in \mathcal{K} \quad t \in \mathcal{T} \tag{11}$$

$$x_{ij}^{kt} \in \{0,1\} \quad i,j \in \mathcal{V}' \quad k \in \mathcal{K} \quad t \in \mathcal{T} \tag{12}$$

$$y_{i}^{kt} \in \{0,1\} \quad i \in \mathcal{V} \quad k \in \mathcal{K} \quad t \in \mathcal{T}. \tag{13}$$

The objective function (1) minimizes the inventory and routing costs. Constraints (2) and (3) are the demand balance equations at the supplier and the customers, respectively. Constraints (4) impose that the inventory level just after delivery cannot be higher than the maximum allowed inventory level at each customer. Constraints (5) impose that the quantity delivered to a customer by a vehicle is zero unless the customer is visited by the vehicle. Constraints (6) impose the vehicle capacity limit and ensure that the y_0^{kt} variable is one if vehicle k makes any delivery in period t. Further, we have traditional degree constraints (7) and the subtour elimination constraints (8). No split deliveries are allowed, as imposed by constraint (9). Constraints (10)−(13) enforce the appropriate integrality and non-negativity conditions on the variables. This problem is NP-hard since the vehicle routing problem is a special subcase (Laporte, 2009).

4. Symmetry Breaking and Input Ordering Techniques

As discussed before, an efficient way to break the existing symmetry in the VRPs is to add symmetry breaking valid inequalities to the standard formulation presented in Section 3. In Section 4.1, we introduce the general symmetry breaking techniques used in the VRP literature and some new constraints and formulation. Then, in Section 4.2, we present and discuss several input ordering techniques.

4.1. Symmetry Breaking Constraints

4.1.1. Vehicle Constraints (VC)

$$y_0^{kt} \leq y_0^{k-1,t} \quad k \in \mathcal{K}\backslash\{1\} \quad t \in \mathcal{T}. \tag{14}$$

Constraints (14) assure that vehicle k is used in a specific period t only if vehicle $k-1$ is used as well (Coelho and Laporte, 2014; Adulyasak et al., 2014).

4.1.2. Variable Reduction (VR)

$$\sum_{k>i} y_i^{kt} = 0 \quad i \in \mathcal{V}' \quad t \in \mathcal{T}. \tag{15}$$

Using constraints (15), each customer (if visited) is always assigned to a vehicle with an index lower than or equal to its own index. Such logic has been used for other problems such as grouping jobs on identical machines (Jans and Desrosiers, 2013).

4.1.3. Hierarchical constraints Type 1 (HC1)

Coelho and Laporte (2014) used the following constraints in addition to constraints (14), to impose a hierarchical order on the assignment of customers to vehicles:

$$y_i^{kt} \le \sum_{j=1}^{i-1} y_j^{k-1,t} \quad i \in \mathcal{V}' \quad k \in \mathcal{K} \backslash \{1\} \quad t \in \mathcal{T}. \tag{16}$$

Inspired by Fischetti et al. (1995), constraints (16) ensure that if customer i is served by vehicle k, then at least one other customer with a smaller index is served by vehicle $k-1$. Similar constraints are also used by Albareda-Sambola et al. (2011) for a capacity and distance constrained plant location problem.

4.1.4. Hierarchical constraints Type 2 (HC2)

In this method in addition to constraints (14), we have the following constraints.

$$y_i^{kt} \le \sum_{j=1}^{i-1} y_j^{lt} \quad k \in \mathcal{K} \backslash \{1\} \quad l \in \{1, 2, \ldots, k-1\} \quad i \in \{k, k+1, \ldots, n\} \quad t \in \mathcal{T}. \tag{17}$$

These constraints impose that if customer i is served by vehicle k, then each vehicle with an index smaller than k, must visit a customer with an index lower than i.

4.1.5. Hierarchical constraints Type 3 (HC3)

In addition to constraints (14) and (16), we have the following constraints.

$$(k-1)y_i^{kt} \leq \sum_{j=1}^{i-1} \sum_{l=1}^{k-1} y_j^{lt} \quad i \in \mathcal{V}' \backslash \{1\} \quad k \in \mathcal{K} \backslash \{1\} \quad t \in \mathcal{T}. \tag{18}$$

Using these constraints, customers with lower indices always have a priority on vehicles also with lower indices

4.1.6. Ordering by routing cost (COS)

As the name suggests, this set of constraints breaks the symmetry by ordering the routes based on their total transportation costs (Adulyasak et al., 2014).

$$\sum_{(i,j)\in\mathcal{E}} c_{ij} x_{ij}^{k-1,t} \geq \sum_{(i,j)\in\mathcal{E}} c_{ij} x_{ij}^{kt} \quad k \in \mathcal{K} \backslash \{1\} \quad t \in \mathcal{T}. \tag{19}$$

4.1.7. Ordering by the quantity delivered per route (QUA)

Another alternative to assigning routes to dispatched vehicles is to order them by their total quantity delivered (Adulyasak et al., 2014) as presented below.

$$\sum_{i\in\mathcal{V}'} q_i^{k-1,t} \geq \sum_{i\in\mathcal{V}'} q_i^{kt} \quad k \in \mathcal{K} \backslash \{1\} \quad t \in \mathcal{T}. \tag{20}$$

4.1.8. Ordering by the number of customers per route (CUS)

The routes can also be ordered based on the number of customers they are serving.

$$\sum_{i\in\mathcal{V}'} y_i^{k-1,t} \geq \sum_{i\in\mathcal{V}'} y_i^{kt} \quad k \in \mathcal{K} \backslash \{1\} \quad t \in \mathcal{T}. \tag{21}$$

4.1.9. Lexicographic ordering (LEX)

The lexicographic ordering constraints with the use of power of two is originally presented in Jans (2009) for a production planning problem with parallel machines and by Adulyasak et al.

(2014) for the IRP.

$$\sum_{i\in\mathcal{V}} 2^{n-i} y_i^{k-1,t} \geq \sum_{i\in\mathcal{V}} 2^{n-i} y_i^{kt} \quad k \in \mathcal{K}\backslash\{1\} \quad t \in \mathcal{T}. \tag{22}$$

In Table 1, we provide a summary on the SBCs used in the literature.

Table 1: The symmetry breaking constraints used in selected papers

Author (Year)	Instance set	SBC	Solution Algorithm
Alkaabneh et al. (2020)	Alkaabneh et al. (2020)	VC	Benders decomposition
Coelho and Laporte (2013a)	Archetti et al. (2007)	VC-HC1	Branch and Cut
Coelho and Laporte (2013b)	Coelho and Laporte (2013b)	VC-HC1	Branch and Cut
Adulyasak et al. (2014)	Archetti et al. (2007) - Adapted	VC- COS-QUA- LEX	Branch and Cut + ALNS
Archetti et al. (2017)	Archetti et al. (2007)	VC-LEX	Tabu search based matheuristic
Archetti et al. (2014)	Adulyasak et al. (2014)	VR-VC-LEX	Branch and Cut
Lmariouh et al. (2017)	Lmariouh et al. (2017)	VC-HC1	Branch and Cut
Coelho and Laporte (2015)	Archetti et al. (2007)	VC-HC1	Branch and Cut
Larrain et al. (2017)	Larrain et al. (2017)	VC-HC1	Variable MIP Neighborhood Descent
Rodríguez-Martín et al. (2019)	Rodríguez-Martín et al. (2019)	VR	Branch and Cut

4.1.10. Asymmetric Representatives Formulation (ARF)

In addition to the previously mentioned SBCs, we introduce a new formulation for the problem in this section. The formulation is based on the idea of the Asymmetric Representatives Formulation.

The customers that are served in the same route are grouped into clusters. The main difference with the traditional formulation is that the smallest customer identifies a cluster in it. Let variables v_i^{kt} be equal to one if customer i belongs to cluster k in period t, i.e., customer i belongs to the cluster in which customer k is the smallest indexed customer. If the variable v_i^{kt} equals 1, this means that cluster k will be used in period t, and hence, all the customers that are in this cluster will be visited by the same vehicle in period t. Furthermore, let variables x_{ij}^{kt} be equal to one if arc (i, j) is used in cluster k in period t, i.e., arc (i, j) belongs to the cluster in which customer k is the smallest indexed customer. Variables q_i^{kt} represent the quantity delivered to customer i belonging to cluster $k \in \mathcal{V}'$ in period t. Variables I remain unchanged.

$$\sum_{i\in\mathcal{V}}\sum_{j\in\mathcal{V},j>i}\sum_{k\in\mathcal{V}'}\sum_{t\in\mathcal{H}} c_{ij}x_{ij}^{kt} + \sum_{i\in\mathcal{V}}\sum_{t\in\mathcal{H}} h_i I_i^t \tag{23}$$

subject to (2)–(4) and to:

$$q_i^{kt} \leq C_i v_i^{kt} \quad i,k \in \mathcal{V}' \quad t \in \mathcal{T} \tag{24}$$

$$\sum_{i\in\mathcal{V}'} q_i^{kt} \leq Q v_k^{kt} \quad k \in \mathcal{V}' \quad t \in \mathcal{T} \tag{25}$$

$$\sum_{j\in\mathcal{V}'} x_{0j}^{kt} = 2v_k^{kt} \quad k \in \mathcal{V}' \quad t \in \mathcal{T} \tag{26}$$

$$\sum_{j\in\mathcal{V},j<i} x_{ji}^{kt} + \sum_{j\in\mathcal{V},j>i} x_{ij}^{kt} = 2v_i^{kt} \quad i,k \in \mathcal{V}' \quad t \in \mathcal{T} \tag{27}$$

$$\sum_{k\in\mathcal{V}'} v_i^{kt} \leq 1 \quad i \in \mathcal{V}' \quad t \in \mathcal{T} \tag{28}$$

$$\sum_{k\in\mathcal{V}'} v_k^{kt} \leq K \quad t \in \mathcal{T} \tag{29}$$

$$v_i^{kt} = 0 \quad i,k \in \mathcal{V}', k > i, \quad t \in \mathcal{T} \tag{30}$$

$$q_i^{kt} = 0 \quad i,k \in \mathcal{V}', k > i, \quad t \in \mathcal{T} \tag{31}$$

$$x_{ij}^{kt} = 0 \quad i,j,k \in \mathcal{V}', k,j > i, \quad t \in \mathcal{T} \tag{32}$$

$$x_{0j}^{kt} = 0 \quad j, k \in \mathcal{V}', k > j \quad t \in \mathcal{T} \tag{33}$$

$$x_{ij}^{kt} \leq 1 \quad (i, j) \in \mathcal{E}, k \in \mathcal{V}', \quad t \in \mathcal{T} \tag{34}$$

$$v_i^{kt} - v_k^{kt} \leq 0 \quad i, k \in \mathcal{V}', \quad t \in \mathcal{T} \tag{35}$$

Constraints (24) indicate that the amount delivered to any visited customer has to respect its capacity. Constraints (25) ensure that the capacity of the vehicle is respected. The vehicles leave from the depot and return to it after vising a customer, as shown by constraints (26). Constraints (27) are the equivalent of the degree constraints. By constraints (28), each customer can be assigned only to one cluster, where the number of clusters needs to be at most equal to the total number of available vehicles, as in constraints (29). No visit and no delivery can take place for all customers with an index less than the vehicle index (constraints (30) –(33)). Constraints (34) show that each arc (i, j) can belong to only one cluster. Customer i always belongs to a cluster in which customer k is the smallest indexed customer; (35) guarantees this.

4.2. Input ordering techniques

The impact of input ordering for different classes of SBCs is analyzed first by Jans and Desrosiers (2013) for a job grouping problem. Later Coelho and Laporte (2014) analyze the impact of three input orderings for a multi-vehicle IRP. They propose ordering customers based on the following three criteria: 1) highest demand, 2) smallest distance, and 3) the largest distance. They conclude that the highest demand and the largest distance orderings have the most significant positive impact on the total CPU time. Continuing this line of research, here we propose the following input ordering criteria for the customers: 1) largest distance, 2) highest demand, 3) farthest from the last inserted (starting with the largest distance), 4) farthest from all inserted (starting with the largest distance), 5) highest sum of normalized demand

and normalized distance, 6) highest of either normalized demand and normalized distance, 7) highest product of normalized demand and normalized distance.

The normalized demand for a customer is calculated as the demand for that customer divided by the maximum demand of all customers. Demand is calculated as the total demand over the whole planning horizon. The normalized distance for a customer is calculated as the distance from the depot to that customer divided by the maximum distance between the depot and a customer.

Furthermore, we propose and test the inverse ordering criteria as 1) smallest distance, 2) lowest demand, 3) closest to the last inserted (starting with the smallest distance), 4) closest to all inserted (starting with the smallest distance), 5) lowest sum of normalized demand and normalized distance, 6) lowest of either normalized demand and normalized distance, 7) lowest product of normalized demand and normalized distance.

5. Computational Experiments

We have implemented a branch-and-cut algorithm capable of solving the formulation presented in Section 3 using CPLEX 12.8 and IBM Concert Technology in C++. All experiments are conducted on an Intel Core i7 processor running at 3.4 GHz with 64 GB of RAM installed with the Ubuntu Linux operating system. The maximum execution time is 3,600 seconds.

Concerning the B&C algorithm, all the formulation variables are explicitly handled by the algorithm, but not all subtour elimination constraints (8). These are not explicitly included in the initial subproblem but are dynamically generated as cuts. These formulations can then be solved by B&C as follows. At a generic node of the search tree, a linear program with relaxed integrality constraints is solved, a search for violated constraints (8) is performed, and violated valid inequalities are added to the current program which is reoptimized. This process is reiterated until a feasible or dominated solution has been reached, or until no more cuts can be added. At this point, branching on a fractional variable occurs.

We conduct all our experiments on the classical instances introduced by Archetti et al. (2007). For years, these widely used instances have been utilized as the testbed to compare different IRP algorithms. These benchmarks instances are identified by the number of customers (ranging from five to 50), periods (either three or six), and inventory costs (high versus low). For each combination, five instances are generated randomly, which results in a total of 160 instances. In our experiments, we solve each instance with either two or three vehicles. Moreover, we modify the customer order in each instance, based on the 14 ordering criteria presented in Section 4.2. We also include the initial random input order from the benchmark instances in our experiments.

5.1. Results from the standard formulation

To begin our analysis, we compare the average results from running all instances over all input ordering techniques. As presented in Table 2, we examine the addition of VR, VC, and both of them (indicated as $VCVR$) to the standard formulation (SF), which is the model presented in Section 3 solved by B&C.

Note that CPLEX 12.8 already includes some automatic symmetry breaking. However, its documentation does not give any further information on what exactly is done. In the default setting, CPLEX chooses the best level of symmetry breaking automatically. The symmetry breaking techniques can also be turned off or set to a specific level. Previous research on a job grouping problem (Jans and Desrosiers, 2013) shows that setting the CPLEX symmetry breaking to the highest level does not significantly improve CPU time, whereas turning them off leads to a substantial increase in CPU time for the standard symmetric formulation. For formulations that explicitly incorporate symmetry breaking constraints, different settings did not have any significant effect. Therefore, we choose to use the default symmetry breaking setting, and the results presented in the tables for SF are obtained using CPLEX in its default setting.

Results in Table 2 show that adding either variable reduction (VR) or vehicle constraints (VC)

reduces the average obtained gap (*Gap*), the average time (*Time*), the total number of unsolved instances (*#Unsolved*), and increases the total number of optimal solutions (*#Opt*) compared to the standard formulation. While VR is more effective than VC, the results show that adding both of these constraints to SF provides the best results. As shown in Table 2, by using both of these constraint sets, all 2,400 instances run with two vehicles yield a feasible solution, and 1,664 optimal cases are found; with three vehicles the number of unsolved cases reduces to 18 (from 257), and a total of 960 cases with an optimal solution are obtained.

Table 2: Results of the standard formulation

	2 vehicles						3 vehicles					
	Average				Sum		Average				Sum	
	UB	LB	Gap	Time	#Opt	#Unsolved	UB	LB	Gap	Time	#Opt	#Unsolved
SF	8012.20	7668.87	3.33	1803	1353	3	8624.90	7828.34	13.56	2987	478	257
VC	7997.73	7694.64	2.86	1654	1466	5	8603.37	7869.32	12.45	2924	532	269
VR	7949.77	7745.06	1.87	1398	1626	1	8993.34	8092.76	8.31	2552	845	22
$VCVR$	7950.21	7753.60	1.79	1336	1664	0	8988.68	8108.05	8.03	2405	960	18

5.2. The impact of symmetry breaking constraints

In Section 4, we proposed several techniques for breaking the symmetry in a mathematical formulation, and we now analyze their effectiveness. First, we study each technique separately, and then we investigate the combined effects of different techniques.

- **Individual constraint effect**

  The SBCs presented in Section 4 are compared, and the results are summarized in Table 3. These results present averages over all the input ordering techniques.

  The best techniques seem to be $HC1$, $HC2$, and $HC3$, while the worst results are obtained with COS, QUA, and CUS.

  For the cases with two and three vehicles, among the three best-performing methods, $HC1$ performs slightly better than $HC3$ in terms of the average gap, time reduction, and the number of optimal solutions found. QUA and CUS techniques seem not to be as efficient as the others for both cases with two or three vehicles. In terms of the number of unsolved instances, the performance of COS is similar to CUS, but for the cases with three vehicles, COS has more difficulty in finding a feasible solution for instances. The lexicographic ordering (LEX) is somewhere in between, as its performance decreases in instances with many nodes.

Table 3: Results for single symmetry breaking constraints

	2 vehicles						3 vehicles					
	Average				Sum		Average				Sum	
	UB	LB	Gap	Time	#Opt	#Unsolved	UB	LB	Gap	Time	#Opt	#Unsolved
$HC1$	7938.69	7766.77	1.55	1262	1720	0	9000.39	8129.95	7.74	2365	955	9
$HC2$	7969.68	7723.86	2.27	1519	1561	0	8864.26	7934.04	11.45	2741	674	159
$HC3$	7940.69	7766.68	1.58	1268	1717	0	9023.07	8124.87	7.96	2371	970	14
COS	8028.00	7703.92	3.11	1600	1522	17	7849.64	7880.39	11.64	2726	646	582
QUA	8011.96	7686.87	3.13	1677	1462	14	8261.71	7857.65	12.87	2925	535	396
CUS	8027.91	7696.30	3.16	1660	1479	17	8388.33	7877.85	12.59	2783	629	393
LEX	7972.62	7737.82	2.31	1379	1644	5	9031.25	8080.52	8.93	2472	878	102

dark gray: worst results – light gray: best results

- **Constraints combined effect**

Since the $HC1$ has shown the best results so far, we focus on its impact when combining it with other constraints. Table 4 summarizes the results. Once again, these results are obtained averaging over all the input ordering techniques. As constraints (16) are written for each $i \in \mathcal{V}'$, by increasing the number of customers, we add more constraints to the model. This might increase the burden on the model. Therefore, we also examine the effect of adding these constraints only for the first half of the customers ($half$) or only the first quarter of the customer list ($quarter$). At the same time, we avoid the large coefficients used in the constraints for the LEX. The results obtained with $HC1$ are used as benchmarks in Table 4 and shown in bold characters.

Table 4: Combined $HC1$ results

	2 vehicles						3 vehicles					
	Average				Sum		Average				Sum	
	UB	LB	Gap	Time	#Opt	#Unsolved	UB	LB	Gap	Time	#Opt	#Unsolved
$HC1$	**7938.69**	**7766.77**	**1.55**	**1262**	**1720**	**0**	**9000.39**	**8129.95**	**7.74**	**2365**	**955**	**9**
$HC1\_half$	7940.55	7763.46	1.60	1281	1699	0	8994.33	8127.30	7.75	2398	968	15
$HC1\_quarter$	7945.97	7759.74	1.71	1334	1679	0	9001.06	8104.22	8.17	2559	865	17
$HC1\_VR$	7939.48	7765.73	1.58	1271	1716	0	9001.47	8127.55	7.76	2370	951	10
$HC1\_half\_VR$	7938.38	7765.07	1.56	1272	1704	0	8984.41	8130.37	7.62	2362	974	16
$HC1\_quarter\_VR$	7940.19	7765.00	1.59	1286	1703	0	8984.75	8127.69	7.70	2343	999	13
$HC1\_LEX$	7937.91	7765.33	1.56	1293	1711	0	9023.35	8124.72	7.97	2375	944	8
$HC1\_LEX\_half$	7944.45	7760.83	1.68	1305	1703	0	8995.45	8124.21	7.89	2387	957	18
$HC1\_LEX\_quarter$	7943.43	7764.73	1.60	1280	1710	0	9002.01	8126.99	7.67	2402	959	15
$HC1\_LEX\_VR$	7939.36	7764.90	1.57	1298	1704	0	9021.14	8124.34	7.93	2374	946	10
$HC1\_half\_LEX\_half\_VR$	7952.33	7755.00	1.82	1353	1683	0	9014.37	8111.08	8.26	2407	939	28
$HC1\_quarter\_LEX\_quarter\_VR$	7945.53	7763.05	1.64	1286	1706	0	9000.58	8124.12	7.72	2378	961	19

The results presented in Table 4 show that for instances solved with two vehicles, all combinations can obtain a feasible solution (no unsolved cases). Moreover, $HC1$ remains the best method with respect to the number of optimal solutions obtained and the gap. For the cases with three vehicles, however, the situation is different. It is the $HC1\_quarter\_VR$ method that solves 999 cases to optimality, but the lowest number of unsolved cases is obtained by combing $HC1$ with LEX.

- **ARF** The ARF formulation proves several optimal solutions but leaves many instances without any feasible solutions as well. In Table 5, we compare the results of the standard formulation (SF) and the ARF. The table shows that the ARF obtains fewer optimal solutions in cases with two vehicles, and many more cases remain unsolved. For cases with three vehicles, ARF finds more optimal solutions and has more unsolved instances. As before, the presented results are the averages over all the input ordering techniques.

Table 5: Comparison of the results obtained with ARF versus SF

	2 vehicles		3 vehicles	
	#Opt	#Unsolved	#Opt	#Unsolved
SF	1353	3	478	257
ARF	852	752	678	877

In order to analyze the gap and the CPU time of ARF, we compare ARF with $HC1$, only on the instances for which ARF obtains a feasible solution. These results are averaged over solutions obtained with all input ordering techniques. $HC1$ is selected as it has been proven to be the best technique so far. Table 6 provides an overview of the results. From this table, we observe that $HC1$ clearly outperforms the ARF.

Table 6: Comparison between ARF and $HC1$, only on instances for which ARF provides a solution

	2 vehicles						3 vehicles					
	Average				Sum		Average				Sum	
	UB	LB	Gap	Time	#Opt	#Unsolved	UB	LB	Gap	Time	#Opt	#Unsolved
$HC1$	6774.53	6692.37	0.76	784.65	1378	0	7393.20	6933.50	4.05	1707.79	906	0
ARF	7006.18	6578.40	4.90	1978.07	852	0	7605.76	6896.79	7.53	2149.60	678	0

Tables 7 and 8 provide more details on the performance of the ARF. The results presented in these two tables are obtained for the original instances of Archetti et al. (2007) in which the input ordering is random. For each combination of the number of periods (H), Inventory (low versus high costs), and the number of customers (n), the average gap (in %), and time (in seconds), over five instances, are obtained. Out of these five instances, the percentages of optimal solutions obtained (%Opt) and unsolved cases (%Unsolved) are also shown in the tables. For cases with two vehicles, the SF solves all instances of the instance sets. However, the performance of ARF very much depends on the size of the instance, and particularly the number of customers. The results show that for instances with six periods and 25 customers or fewer, ARF outperforms SF. With three vehicles, we observe that ARF outperforms SF for instances with both three and six periods and up to 25 customers.

5.3. Impact of input ordering

In our second set of experiments, we study the impact of the input ordering on the performance of the algorithm. We consider several input orderings and enumerate them as follows: 1) random, 2) largest distance, 3) highest demand, 4) farthest from the last inserted (starting with the largest distance), 5) farthest from all inserted (starting with the largest distance), 6) highest sum of normalized demand and normalized distance, 7) highest of either normalized demand and normalized distance, 8) highest product of normalized demand and normalized distance, 9) smallest distance, 10) lowest demand, 11) closest to the last inserted (starting with the smallest distance), 12) closest to all inserted (starting with the smallest distance), 13) lowest sum of normalized demand and normalized distance, 14) lowest of either normalized demand and normalized distance, 15) lowest product of normalized demand and normalized distance.

Note that pairs 2–8 are the opposite of 9–15. Random input ordering is, in fact, the order presented in the benchmark instances of Archetti et al. (2007).

Table 9 shows the total number of optimal solutions obtained for the cases with two and three vehicles. For each symmetry breaking technique (presented in columns), we identify the best ordering technique (in light gray) and the worst one (in dark gray). For example, with respect to the total number of optimal solutions obtained, for SF the best input ordering is order 8. *highest product of normalized demand and normalized distance* ordering and the worst ones are order 9. *closest to all inserted (starting with the largest distance)*, 12. *lowest of either normalized demand*, 14. *lowest of either normalized demand and normalized distance*, and 15. *lowest product of normalized demand and normalized distance*, techniques. As shown in Table 9, although we cannot identify one globally best or worst input ordering technique, we can observe that some of them generally work better with certain symmetry breaking constraints.

Table 7: Comparision of ARF and SF for the random input ordering and with 2 vehicles

			ARF				SF			
H	Inventory	n	Gap	%Opt	%Unsolved	Time	Gap	%Opt	%Unsolved	Time
3	high	5	0.00	100.00	0.00	1	0.00	100.00	0.00	2
		10	0.00	100.00	0.00	18	0.00	100.00	0.00	9
		15	0.00	100.00	0.00	178	0.00	100.00	0.00	21
		20	1.77	60.00	0.00	2202	0.15	80.00	0.00	1037
		25	1.83	40.00	0.00	3236	0.00	100.00	0.00	880
		30	5.09	0.00	20.00	3602	0.00	100.00	0.00	817
		35	5.44	0.00	60.00	3605	0.00	100.00	0.00	344
		40	Unk	0.00	100.00	3610	0.81	60.00	0.00	2093
		45	Unk	0.00	100.00	3623	0.99	60.00	0.00	2461
		50	Unk	0.00	100.00	3656	2.94	0.00	0.00	3600
3	low	5	0.00	100.00	0.00	1	0.00	100.00	0.00	0
		10	0.00	100.00	0.00	26	0.00	100.00	0.00	8
		15	0.00	100.00	0.00	315	0.00	100.00	0.00	44
		20	3.01	40.00	0.00	2298	1.31	80.00	0.00	786
		25	13.08	0.00	0.00	3601	0.52	60.00	0.00	1484
		30	17.60	0.00	20.00	3602	0.00	100.00	0.00	779
		35	8.47	0.00	40.00	3606	0.00	100.00	0.00	332
		40	17.73	0.00	60.00	3612	3.89	60.00	0.00	2052
		45	Unk	0.00	100.00	3643	8.40	60.00	0.00	2022
		50	Unk	0.00	100.00	3675	10.95	20.00	0.00	3368
6	high	5	0.00	100.00	0.00	5	0.00	100.00	0.00	28
		10	0.00	100.00	0.00	606	1.27	40.00	0.00	2914
		15	0.93	40.00	0.00	2857	2.77	0.00	0.00	3601
		20	3.00	0.00	0.00	3600	5.90	0.00	0.00	3600
		25	5.45	0.00	20.00	3601	6.07	0.00	0.00	3600
		30	19.16	0.00	80.00	3602	9.37	0.00	0.00	3600
6	low	5	0.00	100.00	0.00	9	0.00	100.00	0.00	90
		10	0.00	100.00	0.00	1573	3.30	40.00	0.00	2857
		15	1.33	40.00	0.00	3432	8.45	0.00	0.00	3600
		20	7.81	0.00	0.00	3601	12.37	0.00	0.00	3601
		25	10.08	0.00	0.00	3601	13.95	0.00	0.00	3600
		30	Unk	0.00	100.00	3603	21.46	0.00	0.00	3601

Table 8: Comparision of *ARF* and *SF* for the random input ordering and with 3 vehicles

			ARF				*SF*			
H	Inventory	n	Gap	%Opt	%Unsolved	Time	Gap	%Opt	%Unsolved	Time
3	high	5	0.00	100.00	0.00	2	0.00	100.00	0.00	7
		10	0.00	100.00	0.00	39	0.00	100.00	0.00	402
		15	0.00	100.00	0.00	795	1.69	60.00	0.00	1563
		20	2.64	40.00	0.00	2546	5.39	40.00	0.00	2381
		25	5.69	0.00	0.00	3601	9.71	0.00	0.00	3600
		30	7.71	0.00	40.00	3601	6.56	20.00	0.00	3406
		35	4.21	0.00	60.00	3604	4.20	0.00	0.00	3600
		40	Unk	0.00	100.00	3612	8.58	0.00	0.00	3600
		45	Unk	0.00	100.00	3618	7.79	0.00	0.00	3600
		50	Unk	0.00	100.00	3640	10.88	0.00	0.00	3600
3	low	5	0.00	100.00	0.00	1	0.00	100.00	0.00	13
		10	0.00	100.00	0.00	55	0.00	100.00	0.00	731
		15	0.00	100.00	0.00	604	2.42	60.00	0.00	1963
		20	7.38	40.00	0.00	2706	17.61	20.00	0.00	2903
		25	20.29	0.00	0.00	3601	22.95	0.00	0.00	3600
		30	19.70	0.00	20.00	3603	16.80	0.00	0.00	3600
		35	19.60	0.00	60.00	3607	17.82	0.00	0.00	3600
		40	21.04	0.00	80.00	3608	40.38	0.00	0.00	3600
		45	Unk	0.00	100.00	3627	25.57	0.00	20.00	3600
		50	Unk	0.00	100.00	3625	33.76	0.00	20.00	3600
6	high	5	0.00	100.00	0.00	35	3.99	0.00	0.00	3600
		10	0.73	60.00	0.00	2143	11.50	0.00	0.00	3602
		15	1.73	0.00	0.00	3600	15.89	0.00	0.00	3600
		20	5.84	0.00	0.00	3601	17.44	0.00	20.00	3600
		25	7.11	0.00	0.00	3601	17.43	0.00	40.00	3600
		30	Unk	0.00	100.00	3604	Unk	0.00	100.00	3600
6	low	5	0.00	100.00	0.00	45	6.03	0.00	0.00	3601
		10	1.27	20.00	0.00	2917	18.08	0.00	0.00	3601
		15	3.56	0.00	0.00	3600	26.63	0.00	0.00	3601
		20	9.40	0.00	0.00	3600	34.03	0.00	0.00	3600
		25	17.48	0.00	20.00	3602	37.64	0.00	40.00	3600
		30	19.96	0.00	80.00	3602	Unk	0.00	100.00	3600

Table 9: Number of optimal solutions obtained

	2 vehicles									3 vehicles									Performance	
	Constraints																		Total	
Ordering	SF	VR	$HC1$	$HC2$	$HC3$	COS	QUA	CUS	LEX	SF	VR	$HC1$	$HC2$	$HC3$	COS	QUA	CUS	LEX	Best	Worst
1	93	112	**119**	101	118	101	93	102	114	30	55	69	43	67	43	32	41	63	3	3
2	91	109	113	102	115	100	96	97	105	33	59	68	43	71	43	35	43	64	0	1
3	91	112	**119**	102	118	105	102	98	111	34	62	70	51	71	45	39	42	60	4	0
4	91	112	**119**	109	115	102	98	97	116	32	66	69	47	70	43	35	41	67	3	1
5	89	109	118	108	117	102	98	97	113	33	62	71	47	70	41	36	42	65	0	0
6	89	109	118	111	117	101	96	98	112	30	63	72	52	74	43	36	41	69	1	1
7	90	112	118	105	118	99	98	96	110	35	66	70	50	73	40	40	42	66	4	2
8	95	108	117	112	116	101	100	101	112	31	67	73	54	**75**	44	35	41	65	6	1
9	88	104	111	98	110	98	98	98	105	32	47	54	39	53	42	38	41	51	0	6
10	91	110	114	107	116	98	96	98	112	32	57	56	48	63	45	40	41	54	1	2
11	91	104	109	101	112	95	97	96	106	31	45	54	39	55	41	32	43	50	0	3
12	88	107	112	100	111	107	95	97	108	29	48	58	41	54	43	32	44	51	2	3
13	90	108	113	100	112	107	96	101	104	32	52	60	41	61	44	32	42	52	1	1
14	88	101	108	103	109	104	97	104	106	32	47	57	38	55	43	34	42	50	1	6
15	88	109	112	102	113	105	102	99	107	32	49	54	41	58	46	39	43	51	2	2
Total	1353	1626	1720	1561	1717	1522	1462	1479	1644	478	845	955	674	970	646	535	629	878		

For each column: dark gray: worst results – light gray: best results

The overall best result with respect to the number of optimal solutions (combined for two and three vehicles) is obtained by $HC3$ in combination with ordering 6, 7 and 8. All lead to 191 optimal solutions. Furthermore, $HC1$ in combination with ordering 6 and 8 comes very close with 190 optimal solutions. All these combinations have none or only one unsolved instance. Other formulations, in combination with their own best ordering, do not obtain the same results. LEX obtains a total of 183 optimal solutions with input ordering 4.

Table 10: Number of unsolved instances

	2 vehicles									3 vehicles									Performance	
	Constraints																		Total	
Ordering	*SF*	*VR*	*HC1*	*HC2*	*HC3*	*COS*	*QUA*	*CUS*	*LEX*	*SF*	*VR*	*HC1*	*HC2*	*HC3*	*COS*	*QUA*	*CUS*	*LEX*	Best	Worst
1	0	0	0	0	0	0	4	0	0	17	1	0	12	1	34	35	26	6	1	1
2	0	0	0	0	0	2	2	0	1	19	0	0	11	0	34	32	17	4	3	1
3	0	0	0	0	0	0	0	2	0	21	0	1	11	0	35	24	31	4	2	0
4	0	0	0	0	0	3	1	2	1	19	0	0	7	1	42	24	27	4	2	1
5	2	0	0	0	0	0	1	1	0	14	2	0	5	0	42	25	27	6	0	1
6	0	0	0	0	0	2	1	2	0	11	0	0	6	1	40	26	29	6	1	0
7	0	0	0	0	0	1	1	1	0	18	0	0	8	0	44	23	27	5	0	1
8	0	0	0	0	0	1	0	1	0	17	0	0	2	1	35	22	25	4	2	0
9	0	1	0	0	0	1	1	0	1	17	5	3	12	1	40	27	28	6	0	3
10	0	0	0	0	0	1	0	1	0	13	1	0	13	0	40	24	28	5	0	0
11	1	0	0	0	0	0	1	3	1	18	3	2	15	1	44	26	31	14	0	4
12	0	0	0	0	0	4	1	0	0	22	3	0	11	2	37	36	27	11	0	3
13	0	0	0	0	0	1	1	1	0	19	0	0	13	1	36	32	23	8	0	0
14	0	0	0	0	0	1	0	0	1	19	7	2	17	4	40	26	27	12	0	4
15	0	0	0	0	0	0	0	3	0	13	0	1	16	1	39	14	20	7	1	1
Total	3	1	0	0	0	17	14	17	5	257	22	9	159	14	582	396	393	102		

For each column: dark gray: worst results – light gray: best results, if not zero

For the number of unsolved instances, Table 10 provides a summary of the results obtained with each SBC and the input ordering technique. In this table, we have highlighted, if not zero, the best (in light gray) cases and the worst (in dark gray) cases.

The last two columns in both Tables 9 and 10 provide a general performance summary for each input ordering technique. The columns *Best* and *Worst* count the number of symmetry breaking constraints yielding the best or worst results using that input ordering technique. It should be noted that the *Best* non-zero results are reported in Table 10. For example, concerning the total number of optimal solutions, the random ordering technique is the best one to be used with three symmetry breaking constraints, and it is also the worst for a total of three other constraints.

5.4. Categorizing the results based on the inventory costs

As the problem at hand contains an objective function with both transportation and inventory costs, the classical instances are divided into two general groups of high versus low inventory cost levels. In this section, we examine each symmetry breaking constraint and input ordering in these two general sub-classes of the benchmark instances. The idea is that some of the symmetry breaking techniques or input ordering methods might work better with specific classes of instances. When the inventory cost is low, more effort is needed to optimize the problem, as the VRP part becomes highly relevant. This can be observed in Table 11, which summarizes the overall results for selected symmetry breaking techniques. These results are the averages over all input orderings. For cases where the number of unsolved instances is almost equal, the average gap is a good indicator of how difficult each sub-class of instances is. For example, comparing the average gap of the instances with two vehicles solved with methods $HC1$, $HC2$, or $HC3$ shows that instances with high inventory costs are easier to be solved. Once more, $HC1$ and $HC3$ seem to be the best formulations for different classes.

Table 11: Comparison between the average results obtained for high and low inventory cost instances

	2 vehicles								3 vehicles							
	High inventory cost				Low inventory cost				High inventory cost				Low inventory cost			
	Average		Sum		Average		Sum		Average		Sum		Average		Sum	
	Gap	Time	#Opt	#Unsolved	Gap	Time	#Opt	#Unsolved	Gap	Time	#Opt	#Unsolved	Gap	Time	#Opt	#Unsolved
SF	1.96	1803	677	2	4.70	1802	676	1	8.19	2967	246	129	18.93	3007	232	128
VR	1.15	1392	812	0	2.59	1404	814	1	5.22	2530	431	10	11.41	2575	414	12
$HC1$	0.89	1268	862	0	2.22	1256	858	0	4.88	2359	481	2	10.61	2371	474	7
$HC2$	1.30	1531	780	0	3.23	1508	781	0	7.23	2718	345	70	15.74	2764	329	89
$HC3$	0.90	1270	862	0	2.25	1266	855	0	5.07	2363	487	4	10.87	2379	483	10
COS	1.83	1596	763	5	4.39	1603	759	12	7.08	2700	332	282	16.29	2750	314	300
QUA	1.80	1676	736	4	4.47	1678	726	10	7.90	2892	285	194	17.87	2958	250	202
CUS	1.91	1664	733	6	4.41	1656	746	11	7.95	2746	324	187	17.33	2821	305	206
LEX	1.29	1375	824	3	3.32	1384	820	2	5.71	2462	444	39	12.22	2482	434	63
	1.45	1508	7049	20	3.51	1506	7033	37	6.58	2637	3375	917	14.59	2679	3235	1017

For each column: dark gray: worst results – light gray: best results

For the input ordering techniques, we compare the number of optimal solutions and unsolved cases using $HC1$ technique and ARF in high and low inventory cost instances. As Table 12 shows, no significant difference between these two classes of instances can be identified. The number of solved instances and the optimal solutions obtained are similar for high versus low category for cases with two or three vehicles. $HC1$ provides consistently superior performance compared to the ARF.

Table 12: Input ordering for $HC1$ and ARF to compare high and low inventory cost instances

	$HC1$								ARF							
	2 Vehicles				3 Vehicles				2 Vehicles				3 Vehicles			
	#Opt		#Unsolved		#Opt		#Unsolved		#Opt		#Unsolved		#Opt		#Unsolved	
Ordering	high	low	high	low	high	low	high	low	high	low	high	low	high	low	high	low
1	59	60	0	0	35	34	0	0	32	29	24	21	25	23	25	23
2	58	55	0	0	33	35	0	0	30	30	21	20	23	24	27	28
3	59	60	0	0	35	35	0	1	32	32	25	24	24	23	27	29
4	61	58	0	0	36	33	0	0	30	29	22	23	24	23	31	29
5	60	58	0	0	36	35	0	0	32	33	25	23	24	23	29	30
6	58	60	0	0	36	36	0	0	31	28	22	24	25	24	24	25
7	58	60	0	0	35	35	0	0	32	31	22	21	24	25	28	29
8	59	58	0	0	36	37	0	0	31	30	23	22	25	26	23	21
9	55	56	0	0	28	26	1	2	25	26	29	26	20	21	35	31
10	58	56	0	0	28	28	0	0	28	28	25	25	23	23	32	30
11	55	54	0	0	28	26	1	1	24	24	27	28	20	20	33	32
12	56	56	0	0	29	29	0	0	26	26	31	29	20	21	31	34
13	57	56	0	0	31	29	0	0	25	26	29	28	21	21	30	33
14	54	54	0	0	28	29	0	2	24	26	25	29	21	20	33	34
15	55	57	0	0	27	27	0	1	26	26	28	31	22	20	32	29
Total	862	858	0	0	481	474	2	7	428	424	378	374	341	337	440	437

6. Conclusions

The integrated routing problems are well studied in the literature, and the IRP is one of the most popular integrated problems. As a variant of the VRP, the IRP with identical vehicles is also prone to the symmetry issue caused by the vehicle index present in most formulations. In this paper, we have first provided a comprehensive list of the symmetry breaking constraints present in the literature. We have also proposed a reformulation for the problem and introduced several other symmetry breaking techniques. We have assessed the performance of the new formulation, each symmetry breaking constraint individually and in combination with other ones.

Moreover, we have evaluated several input ordering techniques. The main constraints used in the IRP literature to break the symmetry caused by identical vehicles are vehicle constraints (VC) and hierarchical constraints type 1 ($HC1$). Our extensive computational experiments show that over all input ordering techniques, the use of $HC1$ leads to the best results. Even after combining this method with all other methods, $HC1$ remains the dominant technique to break the symmetry. Despite all its success in other applications, the new ARF formulation for the IRP does not lead to good results, especially for big size instances. Finally, our investigation on the combined effects of the input orderings and the SBCs reveals that although these two factors are dependent on one another, the highest product of normalized demand and normalized distance provides the highest number of optimal solutions. This method gives the worst results only if used in combination with SBC ordering by the number of customers per route (CUS). With respect to the number of unsolved instances, again the highest product of normalized demand and normalized distance ordering and the lowest demand ordering methods provide the best results for most of the SBCs. Finally, comparing the low versus high inventory costs, as expected with high inventory costs, we are able to solve slightly more cases to optimality and have fewer cases with unsolved status.

References

Y. Adulyasak, J.-F. Cordeau, and R. Jans. Formulations and branch-and-cut algorithms for multi-vehicle production and inventory routing problems. *INFORMS Journal on Computing*, 26(1):103–120, 2014.

M. Albareda-Sambola, E. Fernández, and G. Laporte. A computational comparison of several models for the exact solution of the capacity and distance constrained plant location problem. *Computers & Operations Research*, 38(8):1109–1116, 2011.

F. Alkaabneh, A. Diabat, and H. O. Gao. Benders decomposition for the inventory vehicle routing problem with perishable products and environmental costs. *Computers & Operations Research*, 113:104751, 2020.

C. Archetti, L. Bertazzi, G. Laporte, and M. G. Speranza. A branch-and-cut algorithm for a vendor-managed inventory-routing problem. *Transportation Science*, 41(3):382–391, 2007.

C. Archetti, N. Bianchessi, S. Irnich, and M. G. Speranza. Formulations for an inventory routing problem. *International Transactions in Operational Research*, 21(3):353–374, 2014.

C. Archetti, N. Boland, and M. G. Speranza. A matheuristic for the multivehicle inventory routing problem. *INFORMS Journal on Computing*, 29(3):377–387, 2017.

I. Aziez, J.-F. Côté, and L. C. Coelho. Exact algorithms for the multi-pickup and delivery problem with time windows. *European Journal of Operational Research*, 284(3):906–919, 2020.

C. Barnhart, E. L. Johnson, G. L. Nemhauser, G. Sigismondi, and P. Vance. Formulating a mixed integer programming problem to improve solvability. *Operations Research*, 41(6): 1013–1019, 1993.

P. Bendotti, P. Fouilhoux, and C. Rottner. Symmetry-breaking inequalities for ILP with structured sub-symmetry. *Mathematical Programming*, 2020.

M. Braga, D. Delle Donne, M. Escalante, J. Marenco, M. E. Ugarte, and M. C. Varaldo. The minimum chromatic violation problem: a polyhedral study. *Electronic Notes in Discrete Mathematics*, 62:309–314, 2017.

M. Campêlo, V. A. Campos, and R. C. Corrêa. On the asymmetric representatives formulation for the vertex coloring problem. *Discrete Applied Mathematics*, 156(7):1097–1111, 2008.

A. Caprara. Properties of some ILP formulations of a class of partitioning problems. *Discrete Applied Mathematics*, 87(1-3):11–23, 1998.

L. C. Coelho and G. Laporte. The exact solution of several classes of inventory-routing problems. *Computers & Operations Research*, 40(2):558–565, 2013a.

L. C. Coelho and G. Laporte. A branch-and-cut algorithm for the multi-product multi-vehicle inventory-routing problem. *International Journal of Production Research*, 51(23–24):7156–7169, 2013b.

L. C. Coelho and G. Laporte. Improved solutions for inventory-routing problems through valid inequalities and input ordering. *International Journal of Production Economics*, 155(1): 391–397, 2014.

L. C. Coelho and G. Laporte. An optimised target-level inventory replenishment policy for vendor-managed inventory systems. *International Journal of Production Research*, 53(12): 3651–3660, 2015.

A. Costa, P. Hansen, and L. Liberti. On the impact of symmetry-breaking constraints on spatial branch-and-bound for circle packing in a square. *Discrete Applied Mathematics*, 161(1-2): 96–106, 2013.

M. De Cauwer, D. Mehta, and B. O'Sullivan. The temporal bin packing problem: an application to workload management in data centres. In *2016 IEEE 28th International Conference on Tools with Artificial Intelligence (ICTAI)*, pages 157–164. IEEE, 2016.

B. T. Denton, A. J. Miller, H. J. Balasubramanian, and T. R. Huschka. Optimal allocation of surgery blocks to operating rooms under uncertainty. *Operations Research*, 58(4-part-1): 802–816, 2010.

G. Dias and L. Liberti. Exploiting symmetries in mathematical programming via orbital independence. *Annals of Operations Research*, forthcoming, 2019.

F. Fidalgo, D. S. Gonçalves, C. Lavor, L. Liberti, and A. Mucherino. A symmetry-based splitting strategy for discretizable distance geometry problems. *Journal of Global Optimization*, 71(4):717–733, 2018.

M. Fischetti, J. J. Salazar-González, and P. Toth. Experiments with a multi-commodity formulation for the symmetric capacitated vehicle routing problem. In *Proceedings of the 3rd Meeting of the EURO Working Group on Transportation*, pages 169–173, Barcelona, Spain, 1995.

A. Ghoniem and H. D. Sherali. Models and algorithms for the scheduling of a doubles tennis training tournament. *Journal of the Operational Research Society*, 61(5):723–731, 2010.

A. Ghoniem and H. D. Sherali. Defeating symmetry in combinatorial optimization via objective perturbations and hierarchical constraints. *IIE Transactions*, 43(8):575–588, 2011.

P. Hosteins. A compact mixed integer linear formulation for safe set problems. *Optimization Letters*, forthcoming, 2020.

R. Jans. Solving lot-sizing problems on parallel identical machines using symmetry-breaking constraints. *INFORMS Journal on Computing*, 21(1):123–136, 2009.

R. Jans and J. Desrosiers. Binary clustering problems: Symmetric, asymmetric and decomposition formulations. Technical Report G-2010-44, GERAD, Montreal, Canada, 2010.

R. Jans and J. Desrosiers. Efficient symmetry breaking formulations for the job grouping problem. *Computers & Operations Research*, 40(4):1132–1142, 2013.

V. Kaibel, M. Peinhardt, and M. E. Pfetsch. Orbitopal fixing. In *Integer Programming and Combinatorial Optimization*, pages 74–88. Springer, 2007.

S.-i. Kim, J. Han, Y. Lee, and E. Park. Decomposition based heuristic algorithm for lot-sizing and scheduling problem treating time horizon as a continuum. *Computers & Operations Research*, 37(2):302–314, 2010.

G. Laporte. Fifty years of vehicle routing. *Transportation Science*, 43(4):408–416, 2009.

H. Larrain, L. C. Coelho, and A. Cataldo. A variable MIP neighborhood descent algorithm for managing inventory and distribution of cash in automated teller machines. *Computers & Operations Research*, 85:22–31, 2017.

L. Liberti and J. Ostrowski. Stabilizer-based symmetry breaking constraints for mathematical programs. *Journal of Global Optimization*, 60(2):183–194, 2014.

J. Linderoth, F. Margot, and G. Thain. Improving bounds on the football pool problem by integer programming and high-throughput computing. *INFORMS Journal on Computing*, 21(3):445–457, 2009.

J. Lmariouh, L. C. Coelho, N. Elhachemi, G. Laporte, A. Jamali, and D. Bouami. Solving a vendor-managed inventory routing problem arising in the distribution of bottled water in Morocco. *European Journal of Industrial Engineering*, 11(2):168–184, 2017.

F. Margot. Pruning by isomorphism in branch-and-cut. *Mathematical Programming*, 94(1): 71–90, 2002.

F. Margot. Exploiting orbits in symmetric ILP. *Mathematical Programming*, 98(1-3):3–21, 2003.

F. Margot. Symmetry in integer linear programming. In *50 Years of Integer Programming 1958-2008*, pages 647–686. Springer, 2010.

K. P. Martínez, Y. Adulyasak, R. Jans, R. Morabito, and E. A. V. Toso. An exact optimization approach for an integrated process configuration, lot-sizing, and scheduling problem. *Computers & Operations Research*, 103:310–323, 2019.

R. A. Melo and C. C. Ribeiro. Improved solutions for the freight consolidation and containerization problem using aggregation and symmetry breaking. *Computers & Industrial Engineering*, 85:402–413, 2015.

S. Mouret, I. E. Grossmann, and P. Pestiaux. Time representations and mathematical models for process scheduling problems. *Computers & Chemical Engineering*, 35(6):1038–1063, 2011.

J. Ostrowski, J. Linderoth, F. Rossi, and S. Smriglio. Orbital branching. *Mathematical Programming*, 126(1):147–178, 2011.

J. Ostrowski, M. F. Anjos, and A. Vannelli. Modified orbital branching for structured symmetry with an application to unit commitment. *Mathematical Programming*, 150(1):99–129, 2015.

M. E. Pfetsch and T. Rehn. A computational comparison of symmetry handling methods for mixed integer programs. *Mathematical Programming Computation*, 11(1):37–93, 2019.

F. Plastria. Formulating logical implications in combinatorial optimisation. *European Journal of Operational Research*, 140(2):338–353, 2002.

L. Proll. ILP approaches to the blockmodel problem. *European Journal of Operational Research*, 177(2):840–850, 2007.

J.-F. Puget. Automatic detection of variable and value symmetries. In P. van Beek, editor, *Principles and Practice of Constraint Programming - CP 2005: 11th International Conference, CP 2005, Sitges, Spain, October 1-5, 2005. Proceedings*. Springer, Berlin, Heidelberg, 2005.

I. Rodríguez-Martín, J.-J. Salazar-González, and H. Yaman. The periodic vehicle routing problem with driver consistency. *European Journal of Operational Research*, 273(2):575–584, 2019.

H. D. Sherali and J. Desai. A global optimization RLT-based approach for solving the hard clustering problem. *Journal of Global Optimization*, 32(2):281–306, 2005.

H. D. Sherali and P. J. Driscoll. Evolution and state-of-the-art in integer programming. *Journal of Computational and Applied Mathematics*, 124(1-2):319–340, 2000.

H. D. Sherali and J. C. Smith. Improving discrete model representations via symmetry considerations. *Management Science*, 47(10):1396–1407, 2001.

H. D. Sherali, J. C. Smith, and Y. Lee. Enhanced model representations for an intra-ring synchronous optical network design problem allowing demand splitting. *INFORMS Journal on Computing*, 12(4):284–298, 2000.

Z. C. Taşkın, J. C. Smith, S. Ahmed, and A. J. Schaefer. Cutting plane algorithms for solving a stochastic edge-partition problem. *Discrete Optimization*, 6(4):420–435, 2009.

N. Vo-Thanh, R. Jans, E. D. Schoen, and P. Goos. Symmetry breaking in mixed integer linear programming formulations for blocking two-level orthogonal experimental designs. *Computers & Operations Research*, 97:96–110, 2018.

G. M. G. H. Wake, N. Boland, and L. S. Jennings. Mixed integer programming approaches to exact minimization of total treatment time in cancer radiotherapy using multileaf collimators. *Computers & Operations Research*, 36(3):795–810, 2009.

T. Walsh. General symmetry breaking constraints. In *International Conference on Principles and Practice of Constraint Programming*, pages 650–664. Springer, 2006.