Submitted to *INFORMS Journal on Computing*
manuscript (Please, provide the mansucript number!)

A Unified Decomposition Matheuristic for Assembly, Production and Inventory Routing

Masoud Chitsaz
HEC Montréal and GERAD, 3000 Chemin de la Côte-Sainte-Catherine, Montréal, H3T 2A7 Canada
masoud.chitsaz@hec.ca

Jean-François Cordeau
HEC Montréal and GERAD, 3000 Chemin de la Côte-Sainte-Catherine, Montréal, H3T 2A7 Canada
jean-francois.cordeau@hec.ca

Raf Jans
HEC Montréal and GERAD, 3000 Chemin de la Côte-Sainte-Catherine, Montréal, H3T 2A7 Canada
raf.jans@hec.ca

While the joint optimization of production and outbound distribution decisions in a manufacturing context has been intensively studied in the past decade, the integration of production, inventory and inbound transportation from suppliers has received much less attention despite its practical relevance. This paper aims to fill the gap by introducing a general model for the assembly routing problem (ARP), which consists of simultaneously planning the assembly of a finished product at a plant and the routing of vehicles collecting materials from suppliers to meet the inventory requirements imposed by the production. We formulate the problem as a mixed-integer linear program and we propose a three-phase decomposition matheuristic that relies on the iterative solution of different subproblems. The first phase determines a setup schedule while the second phase optimizes production quantities, supplier visit schedules and shipment quantities. The third phase solves a vehicle routing problem for each period in the planning horizon. The algorithm is flexible and we show how it can also be used to solve two well-known problems related to the ARP: the production routing problem (PRP) and the inventory routing problem (IRP). Using the same parameter setting for all problems and instances, we obtained 818 new best known solutions out of 2,628 standard IRP and PRP test instances. In particular, on large-scale multi-vehicle instances, the new algorithm outperforms specialized state-of-the-art heuristics for these two problems.

Key words: production, inventory, routing, assembly decomposition matheuristic, three-phase iterative heuristic

History:

1. Introduction

The literature on production planning has paid a lot of attention in the past decade to the integration of lot sizing and outbound transportation decisions. The typical supply chain that is considered consists of a plant that delivers final products to several customers. Considering both the production planning at the plant and the outbound delivery to the customers via routes results in what is called the production routing problem (PRP) (Adulyasak et al. 2015). If the production quantities at the plant are assumed to be given and the decisions only relate to the inventory and route planning, the problem is referred to as the inventory routing problem (IRP) (Andersson et al. 2010, Bertazzi et al. 2008, Coelho et al. 2013).

In contrast, only few studies have focused on the integration of production planning with inbound transportation planning. Yet, in a standard supply chain, a plant often uses several different components to assemble a final product. These components are typically produced in other plants or purchased from suppliers. If the assembly plant is responsible for organizing the inbound transportation of the various components, then gains can be achieved by integrating the production planning with the inbound vehicle routing. We refer to this problem as the assembly routing problem (ARP).

The aim of this paper is to introduce a general model for the ARP. We provide a mathematical formulation of the problem which serves as the basis for a decomposition matheuristic that iteratively solves different subproblems. We also explain how the same methodology can solve the related PRP and IRP. Using the same parameter setting for all three problems, this algorithm outperforms existing heuristics on large-scale multi-vehicle instances of the PRP and IRP, obtaining new best known solutions to many standard test instances.

The ARP has many industrial applications in situations where the production plant and several suppliers are owned by the same company, or when the manufacturer is the biggest player in the supply chain and centrally coordinates the inbound logistics decisions. This is a relevant practical problem in several areas. Fleischmann and Meyr (2003) indicate that in the automotive industry the organization that receives the components is usually responsible for the supply transport. Florian et al. (2011) showed that in addition to the direct financial benefits for the supply chain, inbound logistics integration for a German car manufacturer has some further important outcomes such as a reduction in CO_2 emissions.

In an application for the Delco Electronics Division of General Motors (GM), Blumenfeld et al. (1987) found that the overall optimization of the inbound transportation resulted in a 26% (2.9 million dollars per year, in 1987's USD) savings potential. They proposed the use of an approximation method for the routing cost estimation in their studies to reduce the complexity of the problem. Implementing their solution package, GM of Canada reported savings of approximately 157 thousand USD in four months. Danese (2006) presented the case of GlaxoSmithKline (GSK), an international pharmaceutical group that extended the vendor managed inventory (VMI) approach to its suppliers as a response to the highly competitive and regulated market to benefit from the integrated and coordinated planning process.

Other cases where the buyer is responsible for the transportation are incorporated in several Incoterms, which are often used to clearly define the contractual responsibilities of the buyer and seller in international commercial transactions. Several of these terms consider the cases where the buyer is responsible for the transportation costs and risks. The Incoterm EXW (Ex Works) indicates a situation in which the seller makes the goods available, typically at the factory or a warehouse, and the buyer is responsible for the further transportation. In maritime transport, the Incoterms like FOB (Free On Board) for sea transport or inland waterway transport and FCA (Free Carrier) for roll-on/roll-off or container traffic, address the situation where the seller is responsible for the costs and risks up to when the goods are delivered to the ship at the named port of shipment. Then, it is the buyer who is responsible for the costs and risks from that point onwards.

In the retail sector, the concept of factory gate pricing (FGP) has emerged (Whiteoak 1994, Le Blanc et al. 2006, Fernie and Sparks 2014). Under FGP, the supplier no longer delivers the products to the customer but makes them available at its own factory gate (Le Blanc et al. 2006). This requires the customer to plan and synchronize the pickups from the suppliers to reduce the transportation costs as reported by a number of FGP studies. Examples are Le Blanc et al. (2006) for a large Dutch retail distribution company and Potter et al. (2007) for UK retailers. Potter et al. (2007), Whiteoak (1994) and Fernie and Sparks (2014) report success in increasing the product flow while at the same time reducing distance for Tesco, ASDA and Sainsbury's retailers.

To the best of our knowledge, the problem of jointly optimizing production planning and inbound vehicle routing with a finite horizon and discrete planning periods has only

been considered by Hein and Almeder (2016) in a restricted just-in-time (JIT) environment. They studied multiple components and products and the JIT environment assumes that the components which arrive at the plant must be used immediately in production. Therefore, Hein and Almeder (2016) do not consider the possibility of holding inventory of the components at the plant level before production takes place. Furthermore, the holding cost at the suppliers is not considered in their specific study. Consequently, the combined decision making is entirely centered on the plant costs without taking the suppliers' cost into account.

Motivated by the above-mentioned applications and to fill the gap in the literature, we study for the first time the problem of the integrated inbound transportation, production and inventory planning in a finite planning horizon with the standard basic assumptions similar to the IRP and PRP. This is the first contribution of this paper. Second, we present a unified decomposition matheuristic capable of solving not only the ARP, but also the IRP and the PRP. Third, we report the results of extensive computational experiments on more than four thousand instances for these three problems, including standard data sets for the IRP and the PRP. The results indicate that our algorithm outperforms the state-of-the-art heuristics on the large-scale multi-vehicle IRP and PRP instances. Finally, further analyses demonstrate the robust behavior of the algorithm.

The remainder of the paper is organized as follows. A short literature review on the integration of production planning with outbound and inbound transportation will be given in Section 2 in order to better position our problem with respect to the existing literature. Then, the ARP will be defined and expressed mathematically in Section 3. The decomposition matheuristic is described in Section 4. The algorithm implementation, the benchmark algorithms and the results of extensive computational experiments on all data sets are presented in Section 5. Finally, Section 6 concludes the paper.

2. Literature Review

The majority of the research on the integrated production planning and outbound routing problem, which is most commonly referred to as the PRP, considers a finite time horizon with discrete planning periods. The associated models are typically formulated as mixed integer linear programs. Chandra (1993) was the first to address this problem by assuming a fixed cost for the warehouse orders, which in terms of modeling is similar to the

production setup cost. He studied a problem with an uncapacitated order size and an unlimited number of capacitated vehicles. Later, Chandra and Fisher (1994) defined the same multi-commodity version of the problem in a more formal way, this time by considering the production setup costs. Several studies on this problem (Boudia et al. 2007, Boudia and Prins 2009, Bard and Nananukul 2009, 2010, Adulyasak et al. 2014a,b, Absi et al. 2014) considered one capacitated production plant producing a single product for multiple customers with inventory costs and inventory capacities both at the plant and customers. The plant is responsible for fulfilling the deterministic demand of the customers during the planning periods. The production setup cost is considered to be constant over the periods. A limited number of homogeneous and capacitated vehicles is also considered to perform the shipments from the plant to the customers. The multi-commodity version of the problem was studied by Fumero and Vercellis (1999) and Armentano et al. (2011). Lei et al. (2006) is the only study that considered multiple production plants producing one single final product and assumed a heterogeneous fleet of vehicles. The studies of Solyalı et al. (2009) and of Archetti et al. (2011) did not assume a capacity for the production. The state-of-the-art heuristic algorithms for the PRP are the adaptive large neighborhood search (ALNS) of Adulyasak et al. (2014b) and the matheuristic of Absi et al. (2014). For the IRP, the heuristic of Archetti et al. (2012) is the best performing algorithm for single-vehicle instances and the matheuristic of Archetti et al. (2017) is the best algorithm for multi-vehicle instances.

There are some studies that considered the optimization of the inbound transportation and inventory decisions without considering the production planning at the central plant. Inspired by the automotive parts supply chain, Lee et al. (2003), Moin et al. (2011) and Mjirda et al. (2014) studied a multi-period, multi-supplier problem with a single assembly plant in which each supplier provides a distinct part type. Popken (1994) and Berman and Wang (2006) studied a single period (static) multicommodity inbound logistics problem with three sets of nodes: origin nodes or suppliers, a destination node and transshipment terminal nodes. It is considered that the origin-destination commodity flows pass through the paths of the network using at most one terminal node. Therefore, the vehicle routes are not considered explicitly. Qu et al. (1999) and Sindhuchao et al. (2005) considered the joint replenishment of multiple items in an inbound material-collection system for a central warehouse under the assumption of an infinite planning horizon.

The problem of integrating inbound transportation with the production and inventory decisions is also drawing a growing attention. Almost all of the research on this problem, with the exception of the previously mentioned study by Hein and Almeder (2016), considers an infinite planning horizon in a continuous time framework and uses mixed integer nonlinear programming models. These models are designed to tackle longer term strategic planning decisions with a cyclic approach. This problem is referred to in the literature as the economic lot and supply scheduling problem (ELSSP) and was introduced by Liske and Kuhn (2009). The problem considers the case of a single production plant responsible for satisfying the given constant demand rates for all the final products over an infinite time horizon. The suppliers provide components necessary for the production within a predetermined rate. Extending the economic order quantity (EOQ) assumptions, the ELSSP aims at finding synchronized cyclic production and routing patterns. The problem does not take into account the inventory capacities at the plant and suppliers. It does consider the inventory costs for the components as well as the final products at the plant's (inbound and outbound) storage areas, but not at the suppliers. Therefore, it targets the cases where the latter costs are negligible or not a matter of interest for the central plant. Liske and Kuhn (2009) considered a power-of-two policy and later Kuhn and Liske (2011) studied the problem of finding a common production cycle policy. This policy assumes that the production cycle times of all end items are equal. This assumption is relaxed in the study of Kuhn and Liske (2014). However, a common basic period (equal to the lowest common multiple of all production cycle times) for the end items exists and is introduced in their study. Bae et al. (2014) studied the problem under the common cycle and the time-varying lot size assumptions. They considered that each route can contain components only for a single final product. Chen and Sarker (2014) investigated a similar problem in which each supplier is responsible for providing a unique component.

3. Problem Definition and Formulation

We consider a many-to-one assembly system where n suppliers, represented by the set $N_s = \{1, ..., n\}$, each provide a unique component necessary for the production of a final product at the central plant, denoted by node 0. The planning horizon comprises a finite number of discretized time periods, represented by the set $T = \{1, .., l\}$. The component supply, s_{it}, at each supplier $i \in N_s$ in each period $t \in T$ is predetermined over the planning

horizon. The production system has to satisfy the external demand, d_t, for the final product at the plant in each period $t \in T$ without stockouts while respecting the plant's production capacity, which is given by C. Both the suppliers and the plant can hold inventory. Each supplier $i \in N_s$ has a storage capacity L_i for its components. The plant provides a shared storage with capacity L for the components and has a separate outbound storage capacity K for the final product. A fleet of m homogeneous vehicles, each with a capacity of Q, is available to perform shipments from the suppliers to the plant using routes that start and end at the plant. We suppose throughout that the components delivered to the plant in period $t \in T$ can be used for production in the same period.

We assume that one unit of each component is needed to make one unit of the final product. Note that in basic assembly structures, it is possible to define the units of measurement of the components so as to satisfy this assumption without loss of generality (see Pochet and Wolsey 2006, chap. 13). Obviously, the unit components may not have identical sizes. Therefore, we consider that each component has a unit size of b_i. This size will be taken into account in the vehicle capacity and plant storage area for components. We consider a unit production cost u and setup cost f at the plant level. The unit holding costs of h_i and r_i are imposed for the inventory of component i at its supplier and at the plant, respectively. The inventory of the final product incurs a unit holding cost of r_0 at the plant. When a vehicle travels from location i to j it entails a period-independent cost of c_{ij}.

In the ARP, the following decisions should be optimized simultaneously for each period:
1. whether or not to produce the final product at the plant and the quantity to be produced;
2. the quantity to be shipped from the suppliers to the plant, and;
3. which suppliers to visit, in what order and by which vehicle.

To model the ARP we define a complete undirected graph $G = (N, E)$, where $N = N_s \cup \{0\}$ is the set of nodes and $E = \{(i, j) : i, j \in N, i < j\}$ is the set of edges. Since we assume a one-to-one relationship between suppliers and components, N_s also represents the set of components and $i = 0$ the final product. For each period $t \in T$, we let the binary variable y_t take value 1 iff production takes place at the plant and we let p_t denote the production quantity. Let I_{it} represent the inventory of component i at supplier $i \in N_s$ at the end of period t. Define F_{it} as the inventory of component $i \in N$ or the final product $i = 0$ at the plant at the end of period t. Let q_{it} indicate the shipment quantity from supplier i to

the plant in period t. The variable x_{ijt} represents the number of times a vehicle traverses the edge $(i,j) \in E$ in period $t \in T$. Since we define the model on an undirected network, x_{ijt} is a binary variable for $i > 0$ and may take values in $\{0,1,2\}$ for $i = 0$. The binary supplier visit variable z_{it} takes value 1 iff a supplier $i \in N_s$ is visited in period t, and the integer variable z_{0t} indicates the number of vehicles dispatched from the plant in period t. Table 1 presents a summary of the notation.

Table 1 ARP notation

Sets:	
N	Set of nodes, indexed by $i \in \{0,...,n\}$, where 0 represents the plant and $N_s = N \setminus \{0\}$ is the set of suppliers. Note that since there is a one-to-one relationship between nodes and items, N also represents the set of components and the final product.
E	Set of edges, $E = \{(i,j) : i,j \in N, i < j\}$.
T	Set of time periods, indexed by $t \in \{1,..,l\}$.
$E(S)$	Set of edges $(i,j) \in E$ such that $i,j \in S$, where $S \subseteq N$ is a given set of nodes.
$\delta(S)$	Set of edges incident to a node set S, $\delta(S) = \{(i,j) \in E : i \in S, j \notin S$ or $i \notin S, j \in S\}$.
Parameters:	
f,u	Fixed setup and unit production costs, respectively.
h_i	Unit holding cost at node $i \in N_s$.
r_i	Unit holding cost of component/final product $i \in N$ at the plant.
c_{ij}	Transportation cost between nodes i and j, $(i,j) \in E$.
C,Q	Production and vehicle capacity, respectively.
s_{it}	Component supply at node $i \in N_s$ in period t.
b_i	Unit size of component $i \in N_s$.
d_t	Demand for the final product in period t.
L_i	Inventory capacity for the components at node $i \in N$.
L	Inventory capacity for the components at the plant.
K	Inventory capacity for the final product (at the plant).
I_{i0}	Initial inventory available at node $i \in N_s$.
F_{i0}	Initial inventory of component/final product $i \in N$ at the plant.
Decision variables:	
p_t	Production quantity in period t at the plant.
y_t	Equals to 1 if there is a setup at the plant in period t, 0, otherwise.
I_{it}	Inventory of component i at node $i \in N_s$ at the end of period t.
F_{it}	Inventory of component/final product $i \in N$ at the plant at the end of period t.
x_{ijt}	Number of times a vehicle traverses the edge $(i,j) \in E$ in period t.
z_{it}	Equals to 1 if node $i \in N_s$ is visited in period t, 0, otherwise.
z_{0t}	Number of vehicles dispatched from the plant in period t.
q_{it}	Quantity shipped from node $i \in N_s$ to the plant in period t.

Using this notation, the ARP can be formulated as the following mixed integer program $(\mathcal{M}_{ARP})$.

$$(\mathcal{M}_{ARP}) \quad \min \sum_{t \in T} \Big(up_t + fy_t + \sum_{i \in N_s} h_i I_{it} + \sum_{i \in N} r_i F_{it} + \sum_{(i,j) \in E} c_{ij} x_{ijt} \Big) \tag{1}$$

s.t.

$$F_{i,t-1} + q_{it} = p_t + F_{it} \quad \forall i \in N_s, \forall t \in T \tag{2}$$

$$F_{0,t-1} + p_t = d_t + F_{0t} \quad \forall t \in T \tag{3}$$

$$I_{i,t-1} + s_{it} = q_{it} + I_{it} \quad \forall i \in N_s, \forall t \in T \tag{4}$$

$$p_t \leq C y_t \quad \forall t \in T \tag{5}$$

$$\sum_{i \in N_s} b_i F_{it} \leq L \quad \forall t \in T \tag{6}$$

$$F_{0t} \leq K \quad \forall t \in T \tag{7}$$

$$I_{it} \leq L_i \quad \forall i \in N_s, \forall t \in T \tag{8}$$

$$z_{0t} \leq m \quad \forall t \in T \tag{9}$$

$$b_i q_{it} \leq Q z_{it} \quad \forall i \in N_s, \forall t \in T \tag{10}$$

$$\sum_{(j,j') \in \delta(i)} x_{jj't} = 2 z_{it} \quad \forall i \in N, \forall t \in T \tag{11}$$

$$Q \sum_{(i,j) \in E(S)} x_{ijt} \leq \sum_{i \in S} (Q z_{it} - b_i q_{it}) \quad \forall S \subseteq N_s, |S| \geq 2, \forall t \in T \tag{12}$$

$$z_{0t} \in \mathbb{Z} \quad \forall t \in T \tag{13}$$

$$F_{0t}, p_t \geq 0, y_t \in \{0,1\} \quad \forall t \in T \tag{14}$$

$$I_{it}, F_{it}, q_{it} \geq 0, z_{it} \in \{0,1\} \quad \forall i \in N_s, \forall t \in T \tag{15}$$

$$x_{ijt} \in \{0,1\} \quad \forall (i,j) \in E : i \neq 0, \forall t \in T \tag{16}$$

$$x_{0it} \in \{0,1,2\} \quad \forall i \in N_s, \forall t \in T. \tag{17}$$

The objective function (1) minimizes the total production, setup and holding costs in addition to the transportation costs. The holding cost includes components inventory at the suppliers and plant as well as the final product inventory at the plant. The inventory flow balance for the components and the final product at the plant is imposed through constraints (2) and (3). Constraints (4) ensure the inventory flow balance at each supplier. Constraints (5) force a setup at the plant for each period in which production takes place. They also impose the production capacity. Constraints (6) and (7) represent the storage capacity for the components and final product at the plant. The storage capacity for the components at each supplier is imposed by constraints (8). Constraints (9) limit the fleet size. Constraints (10) force a vehicle visit whenever components are shipped from a supplier to the plant. The maximum component shipment quantity from each supplier in each period is also limited by the vehicle capacity. Constraints (11) are the degree constraints.

Constraints (12) are the subtour elimination constraints (SECs) and they also impose the vehicle capacity. These constraints are referred to as generalized fractional subtour elimination constraints (GFSEC) (Adulyasak et al. 2014a). It is easy to show that the ARP is NP-hard since the VRP is a special case of it.

4. A Decomposition Matheuristic

In this section we present a unified decomposition matheuristic for the ARP, which can also be applied to the PRP and the IRP. We explain the algorithm in the context of the ARP and its adaptation for the other two problems is explained in Section 5.2 and in Appendix C.

Our algorithm decomposes the $\mathcal{M}_{ARP}$ model into three separate subproblems. The first subproblem, $\mathcal{M}_y$, is a special lot-sizing problem that determines a setup schedule by relying on an approximation of the routing cost (Section 4.1). Considering a given setup schedule, the second subproblem, $\mathcal{M}_z$, chooses the node visits and shipment quantities (Section 4.2). For multi-vehicle instances, a modified model ($\mathcal{M}_z^{\mathcal{R}}$) is employed in this phase to look for possible improvements in node visits and shipments. Finally, the third subproblem solves a series of separate vehicle routing problems (Section 4.3), one for each period t (VRP_t). The solutions of the routing subproblems are then used to update the transportation cost approximation (σ_{it}) in the $\mathcal{M}_z$ model. This procedure is repeated for a number of iterations to reach a local optimum. Then, a local branching scheme is used to change the setup schedule and explore other parts of the feasible solution space, looking for better solutions (Section 4.4). The entire procedure continues until a stopping condition is met (Section 4.5).

Our algorithm shares similarities with the decomposition-based heuristic developed by Absi et al. (2014) for the PRP. However, there are also important differences between the two algorithms. The method of Absi et al. (2014) uses a two-phase approach where in the first phase it the fixes y_t, p_t and q_{it} variables. In our algorithm this is done in two separate phases: it fixes the y_t variables at the end of the first phase, then finds p_t and q_{it} in the second phase. Our method also prevents the same solution to appear twice by adding local branching inequalities (Section 4.4) to cut the current node visit pattern for the next iteration and to cut the current setup schedule in order to diversify the search. Finally, for the diversification, Absi et al. (2014) employ a random transportation cost perturbation mechanism while we change the setup schedule. An overview of our method (CCJ-DH) is presented in Algorithm 1.

Algorithm 1: CCJ-DH

Initialize σ_{it}
repeat
  if first iteration **or** diversification step **then**
    if diversification step **then**
      Cut the current setup schedule from $\mathcal{M}_y$ model
      Reset aggregate fleet capacity for all periods
    end if
    Solve $\mathcal{M}_y \rightarrow y_t$ (and p_t, z_{it}, q_{it})
    Fix y_t variables
  else
    Solve $\mathcal{M}_z \rightarrow p_t, z_{it}, q_{it}$
  end if
  Solve VRP_t subproblems $\rightarrow \sigma_{it}$
  if all VRP_t solutions are feasible **then**
    Update incumbent solution
    if aggregate fleet capacity is reduced in any period **then**
      repeat
        Solve $\mathcal{M}_z^{\mathcal{R}} \rightarrow p_t, z_{it}, q_{it}$
        Solve VRP_t subproblems $\rightarrow \sigma_{it}$
        Update incumbent solution
      until the stopping condition is met
    end if
  else
    Decrease aggregate fleet capacity for the periods with infeasible VRP_t
  end if
  Cut the current node visit pattern from $\mathcal{M}_z$ model
until the stopping condition is met
return incumbent solution

4.1. Phase 1: The $\mathcal{M}_y$ Subproblem

The $\mathcal{M}_y$ subproblem aims to generate a good setup schedule by solving a simplified problem in which we use approximate transportation costs based on the number of vehicles dispatched from the plant. To this end, we update the original objective function (1) with the following:

$$\min \sum_{t \in T} \Big(u p_t + f y_t + \sum_{i \in N_s} h_i I_{it} + \sum_{i \in N} r_i F_{it} + \sigma_{0t} z_{0t} \Big). \tag{18}$$

To estimate the vehicle dispatch cost, σ_{0t}, we consider node visit costs, σ_{it}, and then distribute the total node visit costs $\sum_{i \in N_s} \sigma_{it}$ equally among the available number of vehicles in each period. We discuss the node visit cost approximation in Section 4.2 and its initialization in Section 4.3. With this modification, constraints (11)-(12) become redundant and they are replaced with the following constraints which impose an aggregate fleet capacity:

$$\sum_{i \in N_s} b_i q_{it} \leq Q z_{0t} \quad \forall t \in T. \tag{19}$$

We define the $\mathcal{M}_y$ model with the objective function (18) subject to constraints (2)-(10), (13)-(15) and (19). This model yields a setup schedule in the first iteration and whenever a diversification step is performed. Adding a local branching inequality LBI_y (Section 4.4) prevents the same setup schedule to appear when we solve the model again. As a by-product, the solution to this model specifies the shipment quantity variables, q_{it}. Based on these shipment quantities, we can deduce the corresponding node visit variables. This allows us to skip phase 2 and immediately go to phase 3 in which the VRP_t subproblems (line 13 of Algorithm 1), are solved.

4.2. Phase 2: $\mathcal{M}_z$ and $\mathcal{M}_z^{\mathcal{R}}$ Subproblems

In this second phase the focus is on obtaining proper node visit decisions and shipment quantities. Using the solution found in the first phase, the binary variables y_t are fixed in constraints (5) of the $\mathcal{M}_{ARP}$ model. We approximate the transportation cost in the objective function using the node visit variables z_{it}, which results in the following objective function:

$$\min \sum_{t \in T} \Big(up_t + \sum_{i \in N_s} h_i I_{it} + \sum_{i \in N} r_i F_{it} + \sum_{i \in N_s} \sigma_{it} z_{it} \Big). \tag{20}$$

We assume a cost for each node visit in each period (σ_{it}). Having a complete solution at hand, we approximate the node visit costs for the next iteration as follows: If node i is visited in the current solution, then we set $\sigma_{it} = (c_{i_p i} + c_{i i_s}) - c_{i_p i_s}$, where i_p and i_s are the predecessor and successor of node i in its current route. If node i is currently not served in period t, then we set σ_{it} equal to the cost of the cheapest insertion into an existing route. This is based on the assumption that when a node i is eliminated from its route, an acceptable route can be obtained by connecting the predecessor and successor nodes. Similarly, when inserting node i in a certain period t, an acceptable route can be obtained by the best insertion among all the routes in that period. Hence, σ_{it} can be seen as the (estimated) marginal transportation cost for visiting node i in period t. This marginal cost updating procedure is also used by Absi et al. (2014). We tested two other mechanisms with which the node visit costs can be updated for the next iteration. The first mechanism splits the TSP route cost over its visited nodes proportional to their direct shipment cost at each period. The other mechanism divides the entire transportation cost (VRP_t cost) among the visited nodes of a certain period proportional to their direct shipment cost. In both cases, for each node that is not visited in a particular period of a current solution, the

best insertion cost will be added to the route cost and the share of the node is calculated similarly. The first new mechanism generally gives improved results on the IRP instances compared to the marginal cost updating, but is not the best procedure for the PRP data sets. Therefore, we report results with the marginal cost updating mechanism in Section 5.4, while the comparison of the various updating mechanisms is given in Appendix D.

Modifying the objective function, variables x_{ijt} and z_{0t}, and constraints (9) and (11)-(13) are removed from the formulation and it is no longer possible to enforce the vehicle capacity through the SECs (12). However, by adding constraint $\sum_{i\in N_s} b_i q_{it} \leq mQ$ for every period t we can preserve the aggregate fleet capacity. Since split pickups are not allowed, we may not be able to find a feasible VRP solution for a certain period in phase 3 because the different quantities (q_{it}) to be shipped cannot be packed in the available vehicles. Therefore, as in Absi et al. (2014), we use the following constraints to impose a smaller aggregate fleet capacity ($0 \leq \lambda_t \leq 1$):

$$\sum_{i\in N_s} b_i q_{it} \leq \lambda_t mQ \quad \forall t \in T. \tag{21}$$

The $\mathcal{M}_z$ model minimizes the objective function (20) subject to constraints (2)-(8), (10), (14)-(15) and (21). In the single-vehicle case ($m = 1$) and unlimited vehicle case ($m = n$), a modification of the aggregate fleet capacity is not necessary since a feasible VRP solution can always be found in phase 3 for each period. Therefore, lines 18-20 of Algorithm 1 are not executed. When the routing subproblem cannot find a feasible solution for a certain period, we reduce the λ_t for that period (line 24). Next, the $\mathcal{M}_z$ model is solved with the reduced capacity (line 11), and based on this solution the VRP_t subproblems are solved (line 13). If all VRP_t solutions are feasible, we update the incumbent solution. Since we have reduced some λ_t, this yields some unused aggregate fleet capacity. To explore the possible benefits from the unutilized capacity, we solve a modified $\mathcal{M}_z$ model. Let $\mathcal{R}_{kt}$ be the set of suppliers visited by vehicle k in period t. We replace constraints (21) with the following constraints for the periods where $\lambda_t < 1$ in the $\mathcal{M}_z$ model:

$$\sum_{i\in\mathcal{R}_{kt}} b_i q_{it} \leq Q \quad \forall k \in \{1, ..., m\}, \forall t \in T | \lambda_t < 1. \tag{22}$$

Each constraint (22) relates to a vehicle that is used in a period with $\lambda_t < 1$. Then, we fix the node visit variables z_{it} for these periods and obtain the $\mathcal{M}_z^{\mathcal{R}}$ model (line 18 of

Algorithm 1). Using the $\mathcal{M}_z^{\mathcal{R}}$ model the algorithm can directly impose the vehicle capacity for each route, while we avoid the vehicle-indexed formulation which requires many more binary node visit variables for every vehicle as well as continuous quantity variables. This step is repeated until a stopping criteria is met (lines 17-21 of Algorithm 1).

4.3. Phase 3: VRP_t Subproblems

Following the solution of the $\mathcal{M}_y$, $\mathcal{M}_z$ and $\mathcal{M}_z^{\mathcal{R}}$ models, we fix for each time period the node visit decisions z_{it} and the shipment amounts q_{it}. Therefore, we have to solve one VRP for each period. As discussed in the previous section, this routing problem can be infeasible for one or several periods. To solve this subproblem we use the tabu search heuristic of Cordeau et al. (1997), which allows violations of the vehicle capacity constraints through a penalty cost in the objective function. Based on this (possibly infeasible) solution, we update the transportation costs for the next iteration. To reduce the computing time in the tabu search heuristic, we limit the number of available vehicles at each period using the following expression:

$$\bar{m}_t = \min\left(m, \left\lceil \frac{2}{Q} \sum_{i \in N_s | z_{it}=1} b_i \bar{q}_{it} \right\rceil\right) \quad \forall t \in T. \tag{23}$$

This upper bound is valid because in any optimal solution there is at most one vehicle that is exactly half full or less. Otherwise, under the assumption that the triangle inequality holds, it is possible to merge at least two routes (and obtain a new route with a lower cost). Therefore, the case where all vehicles are exactly half full provides an upper bound on the number of vehicles needed in an optimal solution. Moreover, to control the running time of the heuristic, we set the number of tabu search iterations $\iota^{VRP} = \iota^V \sqrt{\bar{m} \sum_{i \in N_s | z_{it}=1} \bar{z}_{it}}$, for every period t, where ι^V is a parameter in our algorithm.

4.4. Inequalities LBI_z and LBI_y

To diversify the search, we rely on two types of inequalities. These inequalities are inspired by the local branching approach of Fischetti and Lodi (2003). The first type of inequality, LBI_z, is specific to the $\mathcal{M}_z$ model and ensures that we do not return to a node visit pattern (and hence solution) we obtained before. The inequality

$$\sum_{i,t|\bar{z}_{it}=1} (1 - z_{it}) + \sum_{i,t|\bar{z}_{it}=0} z_{it} \geq r \tag{24}$$

forces at least r node visit variables to change value compared to the current solution. By varying r we can force different numbers of node visit changes in the next iteration of our algorithm. Our experiments show that if we let $r > 1$ the algorithm reaches a better solution in a shorter time compared to the case of $r = 1$. However, large values of r may remove some good quality solutions. We chose two different values for r. When the algorithm returns a better solution value compared to the previous iteration, we let $r = 1$ to allow the algorithm to search the entire neighborhood of the current solution. In case a worse solution value (compared to the previous iteration) is obtained, we let $r = l$ which forces at least l (i.e. the number of periods) changes in the node visits. One inequality will be added to the $\mathcal{M}_z$ model at each iteration. These inequalities slow down the solution of the $\mathcal{M}_z$ model. When the setup schedule is changed (by means of the diversification mechanism) we remove all the previous LBI_z inequalities and continue adding new ones in future iterations.

The second type of inequality, LBI_y, is specific to the $\mathcal{M}_y$ model and forces the model to obtain a new setup schedule. Therefore, it is used as a means of diversification. The inequality

$$\sum_{t|\bar{y}_t=1} (1 - y_t) + \sum_{t|\bar{y}_t=0} y_t \geq 1 \tag{25}$$

forces at least one of the binary setup schedule variables to change value. One inequality for each diversification procedure will be added to the $\mathcal{M}_y$ model and they will be kept until the end of the algorithm. Adulyasak et al. (2014b) used this latter type of inequality to generate new setup schedules in their ALNS.

4.5. Stopping Conditions

The stopping condition for the overall algorithm (line 27 of Algorithm 1) is a maximum number of iterations, ι^A. To terminate the search for a local optimum within a specific setup schedule and introduce a diversification step, we consider two stopping conditions. The search procedure stops after a maximum number of local search iterations, ι^L, or after a number of iterations without incumbent solution improvement, ι^N. Whenever one of these stopping conditions is met, the algorithm stops the local search, adds the associated LBI_y and solves the $\mathcal{M}_y$ model to find another setup schedule (lines 4-9 of Algorithm 1). We allow the algorithm to use the $\mathcal{M}_z^{\mathcal{R}}$ model only when it has performed at least ι^s iterations. This is to avoid wasting time with the very first solutions. The algorithm also runs the $\mathcal{M}_z^{\mathcal{R}}$ model only for the cases where the current solution obtained from the $\mathcal{M}_z$

model (and subsequent VRP_t subproblems) is close enough to the incumbent solution. More specifically, if the gap is less than g (%) the algorithm starts using the $\mathcal{M}_z^{\mathcal{R}}$ model to fix some vehicle routes as explained in Section 4.2. The $\mathcal{M}_z^{\mathcal{R}}$ model is allowed to be run until a maximum of $\iota^{\mathcal{R}}$ iterations is reached or until at any iteration it fails to return a solution with a gap less than $g^{\mathcal{R}}$ (%) from the incumbent solution. This condition corresponds to line 21 of Algorithm 1. The specific setting for the algorithm parameters and stopping conditions will be presented in the next section.

5. Computational Experiments

We tested our algorithm on three different problems, the IRP, the PRP and the ARP, with a total of 4,068 instances. The IRP data sets were generated by Archetti et al. (2007) for the single-vehicle case and were later adapted to the multi-vehicle case by Coelho and Laporte (2013a) and by Desaulniers et al. (2015). The PRP data sets were introduced by Archetti et al. (2011) and by Boudia et al. (2005). The ARP instances are introduced in Section 5.1. An overview of all the problem data sets is presented in Appendix A.

We considered the same parameter setting when applying our algorithm to all data sets. The maximum number of algorithm iterations ι^A is set to 150 and the number of local search iterations ι^L is set to 60. The maximum number of non-improving iterations ι^N is set to 30. A maximum number of $\mathcal{M}_z^{\mathcal{R}}$ model iterations ($\iota^{\mathcal{R}}$) equal to 15 is considered. The tabu search iteration coefficient (ι^V) is given a value of 300. The values of g and $g^{\mathcal{R}}$ are set to 3% and 0.3%, respectively. We set the initial node visit cost equal to $c_{0i}/2$, where c_{0i} is the cost of the edge between the plant and node i. The details of the parameter setting procedure are explained in Appendix B.

5.1. ARP Test Instances

We used the PRP data sets of Archetti et al. (2011) as a base for developing our ARP data sets. For each test instance, the number of nodes, their position and the distance function as well as the number of time periods and vehicles have been kept identical. Note that the nodes are suppliers in the ARP, but represent customers in the IRP and PRP.

As in Archetti et al. (2011), an unlimited production capacity is considered. The component supply at each node, s_{it}, is considered constant and equal to the amount used for the demand in Archetti et al. (2011). The demand at the central plant, d_t, is set equal to the average amount of all the suppliers' production rates. An integer number is randomly

chosen according to a uniform distribution in the interval [1,10] for each component's size, b_i. Then, to adjust the vehicle capacity (C) we multiplied the values given in Archetti et al. (2011) by a factor of 10. The component inventory capacities at the suppliers (L_i) are considered the same as the retailers' capacities presented in Archetti et al. (2011). An uncapacitated storage for the components is assumed at the plant. We consider a uniformly distributed random integer generated between 2 to 4 times the product demand of a period as the storage limit, K. The unit component holding cost at the suppliers, h_i, is considered the same as in Archetti et al. (2011). The unit component holding cost at the plant, r_i, is assumed to be equal to a uniform random integer over the $[h_i, 2h_i]$ interval. To generate the unit product holding cost, r_0, we select a uniformly distributed random integer over the interval $[\sum_{i\in N_s} r_i, 2\sum_{i\in N_s} r_i]$. The initial inventory of the components at the suppliers, I_{i0}, is considered equal to what Archetti et al. (2011) considered for the customers. The initial inventory of the final product at the plant, F_{00}, is set randomly in the interval from 0 to the demand of two periods ($[0, 2d_t]$). To avoid infeasibility and meet the final product demand, we need to have enough initial component inventory at the plant. Therefore, we consider for each component i the initial inventory F_{i0} equal to $\max\{0, \sum_{t\in T}(d_t - s_{it}) - I_{i0} - F_{00}\}$. Table 2 presents an overview of the ARP instance parameters.

Table 2 ARP test instances

	Set 1	Set 2	Set 3
Nb. of instances (SA‡)	480	480	480
Nb. of components (SA‡): n	14	50	100
Nb. of periods (SA‡): l	6	6	6
Nb. of suppliers (SA‡): n	14	50	100
Nb. of trucks (SA‡): m	1	UL†	UL†
Component supply: s_{it}		SA‡	
Production capacity: C		SA‡	
Demand (final product): d_t		$(\sum_{i=1}^{n} s_{it})/n$	
Item size: b_i		UDRI$^{\dagger\dagger}[1, 10]$	
Vehicle capacity: Q		SA‡ by a factor of 10	
Supplier inventory capacity: L_i		SA‡	
Plant inventory capacity for components: L		UL†	
Plant inventory capacity for final product: K		UDRI$^{\dagger\dagger}[2d_t, 4d_t]$	
Supplier initial inventory: I_{i0}		SA‡	
Plant initial inventory of components: F_{i0}		$\max\{0, \sum_{t\in T}(d_t - s_{it}) - I_{i0} - F_{00}\}$	
Plant initial inventory of final product: F_{00}		UDRI$^{\dagger\dagger}[0, 2d_t]$	
Supplier and plant x,y coordinates		SA‡	
Unit production cost: u		SA‡	
Production setup cost: f		SA‡	
Unit transportation cost		SA‡	
Travel distance		SA‡	
Supplier unit holding cost: h_i		SA‡	
Plant unit component holding cost: r_i		UDRI$^{\dagger\dagger}[h_i, 2h_i]$	
Plant unit final product holding cost: r_0		UDRI$^{\dagger\dagger}[\sum_{i\in N_s} r_i, 2\sum_{i\in N_s} r_i]$	

† UL: Unlimited
‡ SA: The same as Archetti et al. (2011)
†† UDRI: Uniformly Distributed Random Integer

5.2. Algorithm Implementation

Some modules of the algorithm become redundant for some problems or data sets. The main modules of the algorithm are the $\mathcal{M}_y, \mathcal{M}_z$ and $\mathcal{M}_z^{\mathcal{R}}$ models and VRP_t subproblems. The aim of the $\mathcal{M}_y$ model is to find proper setup schedules. Therefore, this module is not applicable in the case of the IRP. The module with which we find node visit schedules, the $\mathcal{M}_z$ model, is relevant and necessary for all data sets and problems. The $\mathcal{M}_z^{\mathcal{R}}$ model is only required for the data sets with a limited number of vehicles ($1 < m < n$). We present the $\mathcal{M}_y$ model for the PRP and $\mathcal{M}_z$ models for the IRP and PRP in Appendix C. We solve the $\mathcal{M}_y$, $\mathcal{M}_z$ and $\mathcal{M}_z^{\mathcal{R}}$ models with CPLEX 12.6. Since all problems take the routing decisions into account, the VRP_t subproblems must be solved in every case.

5.3. Benchmark Algorithms

Since the ARP is a new problem, there is no algorithm to compare with. Consequently, we used CPLEX as a standard base for comparison. For the IRP and PRP, we selected the state-of-the-art algorithms. Some of these are exact algorithms which we included for two reasons: to show the difference in running times and to consider their best found solutions in our comparison.

On the IRP data sets, we compare our algorithm with eight different algorithms: 1) The branch-and-cut (BC) of Archetti et al. (2007) (ABLS-BC) applied to SV-I1 instances; 2) The BC of Coelho and Laporte (2013b) (CL-BC) applied to all the single- and multi-vehicle IRP instances of Archetti et al. (2007), the single-vehicle IRP instances of Archetti et al. (2012) and the MV-I2 instances of Archetti et al. (2012) with $n = 50$ and 100, and $m = 2$ and 3; 3) The matheuristic of Archetti et al. (2017) (ABS-H) applied to all the multi-vehicle instances; 4) The BC of Avella et al. (2014) (ABW-BC) for the multi-vehicle instances of Archetti et al. (2007); 5) The branch-and-price-and-cut algorithm of Desaulniers et al. (2015) (DRC-BPC) for the instances in the MV-I1 data set; 6) The heuristics of Archetti et al. (2012) (ABHS-H) for both of the single-vehicle IRP data sets; 7) The ALNS of Coelho et al. (2012) (CCL-ALNS) for the SV-I1 instances; and 8) The ALNS of Adulyasak et al. (2014a) (ACJ-ALNS-1000) adapted for the PRP and applied to the the MV-I1 instances with $m = 2$ and 3.

On the PRP data sets, we compare our algorithm with ten algorithms presented in six separate papers: 1) The BC of Archetti et al. (2011) (ABPS-BC) applied to the instances in the SV-A1 data set; 2) The heuristic of Archetti et al. (2011) (ABPS-H) applied to all

the instances in Archetti et al. (2011); 3-5) The memetic algorithm of Boudia and Prins (2009) (BP-MA), the reactive tabu search of Bard and Nananukul (2009) (BN-TS) and the tabu search with path relinking of Armentano et al. (2011) (ASL-TS) applied to all the instances of Boudia et al. (2005); 6) The ALNS of Adulyasak et al. (2014b) with 500 iterations (ACJ-ALNS-500) applied to all PRP instances and with 1000 iterations (ACJ-ALNS-1000) applied to all PRP data sets except MV-B3; 7-10) The heuristics of Absi et al. (2014) including AADF-MS and AADF-DMS applied to the single-vehicle instances of SV-A1, and AADF-VRP and AADF-MTSP applied to the multi-vehicle data sets (MV-A2, MV-A3, MV-B1, MV-B2 and MV-B3). Absi et al. (2014) did not report the results of the first two algorithms (AADF-MS and AADF-DMS) on the multi-vehicle data sets. Although the third and fourth algorithms (AADF-VRP and AADF-MTSP) benefit from VRP and TSP subproblems, respectively (and hence are capable of solving single-vehicle instances), Absi et al. (2014) did not report the result of applying them to SV-A1. ACJ-ALNS-1000 is the only algorithm that is applied to instances from both IRP and PRP data sets but in two separate papers (Adulyasak et al. 2014a,b).

Table 3 provides the list of benchmark algorithms, their running platform and solver version. Since some of the algorithms for the IRP solved only a subset of the instances, we provide more details in Table 4 on the number of instances each algorithm was applied to.

5.4. Computational Results for the IRP and PRP Data Sets

Tables 5, 6 and 7 present the computational results and comparison between our algorithm, CCJ-DH, and the benchmark algorithms. Table 5 presents the average gap of different algorithms applied to the IRP and PRP data sets. The percentage gap for each solution to each instance is calculated with respect to the previous best known solution so far (not including CCJ-DH). Then, for each class and number of vehicles (m) of a data set, the average gaps of different algorithms are calculated. The number of best solutions found by different algorithms is presented in Table 6. Since for some small instances it is possible that CCJ-DH finds the same previous best found solution, we also present the number of new best solutions (NBS) in the last column of this table. The average running times (in seconds) of the different algorithms are shown in Table 7.

For the SV-I1 data set the exact BC algorithms (ABLS-BC and CL-BC) solved all the instances to optimality. ABHS-H and CCL-ALNS are the state-of-the-art heuristics on this data set. They were able to find 122 and 71 optimal solutions, respectively, and

Table 3 Benchmark algorithms, the running platforms and standard MILP solver for the IRP and PRP data sets

Prob	Reference	Name	Sol	CPU	Solver
IRP	Archetti et al. (2007)	ABLS-BC	E	Intel Pentium IV 2.8 GHz	CPLEX 9.0
	Coelho and Laporte (2013b)	CL-BC	E	Xeon 2.66 GHz	CPLEX 12.3
	Archetti et al. (2017)	ABS-H	HM	Xeon W3680, 3.33 GHz	CPLEX 12.5
	Avella et al. (2014)	ABW-BC	E	Intel Core i7-2600, 3.40 GHz	Xpress 7.6
	Desaulniers et al. (2015)	DRC-BPC	E	Intel Core i7-2600 3.4 GHz	CPLEX 12.2
	Archetti et al. (2012)	ABHS-H	H	Intel Dual Core 1.86 GHz	CPLEX 10.1
	Coelho et al. (2012)	CCL-ALNS	M	Intel T7700, 2.4 GHz	-
	Adulyasak et al. (2014a)	ACJ-ALNS-1000	M	2.10 GHz, 2 GB RAM	CPLEX 12.3
PRP	Archetti et al. (2011)	ABPS-BC	E	AMD Athlon 64, 2.89 GHz	CPLEX 10.1
		ABPS-H	H	Intel Core 2, 2.40 GHz	-
	Boudia and Prins (2009)	BP-MA	M	2.30 GHz PC	-
	Bard and Nananukul (2009)	BN-TS	M	2.53 GHz PC	-
	Armentano et al. (2011)	ASL-TS	M	Intel Pentium IV 2.8 GHz	-
	Adulyasak et al. (2014b)	ACJ-ALNS-500	M	2.10 GHz Duo CPU PC	CPLEX 12.2
		ACJ-ALNS-1000	M	2.10 GHz Duo CPU PC	CPLEX 12.2
	Absi et al. (2014)	AADF-MS	H	Xeon 2.67 GHz PC	CPLEX 12.1
		AADF-DMS	H	Xeon 2.67 GHz PC	CPLEX 12.1
		AADF-VRP	H	Xeon 2.67 GHz PC	CPLEX 12.1
		AADF-MTSP	H	Xeon 2.67 GHz PC	CPLEX 12.1
Both	This paper	CCJ-DH	H	Xeon X5675 3.07 GHz	CPLEX 12.6

Note. E: Exact, M: Metaheuristic, H: Heuristic, Prob: Problem, Sol: Solution approach

finished with small gaps. Our algorithm was able to find 25 of the optimal solutions in a relatively small average running time of 39 seconds. Table 7 shows that these state-of-the-art heuristics used computing times more than 11 times longer compared to CCJ-DH. However, the average gap of our algorithm on this data set is 3.08%, which is high compared to the gap of the other algorithms. For the MV-I1 data set, the state-of-the-art heuristic algorithm is ABS-H. It was applied to all the instances in this set and obtained 263 best solutions with average gaps ranging from 0.21% to 1.5%. ACJ-ALNS-1000 found 26 BUBs in total with gaps of more than 6.5%. CCJ-DH obtained solutions with an average gap of approximately 4% and found 70 best solutions among which it was successful to obtain 35 new best solutions. The computing times of ABS-H are more than 10 to 22 times longer compared to CCJ-DH for different values of m.

For the SV-I2 data set, results are available for the CL-BC and ABHS-H algorithms. The first algorithm (which is a BC) spent on average more than 64,000 seconds to solve the instances in the set and obtained 30 BUBs. This algorithm has an average gap of more than 10.9%. The ABHS-H heuristic spent an average computing time of 3,630 seconds and obtained 31 BUBs. The UBs obtained by this algorithm are generally of high quality, resulting in an average gap of 0.27%. CCJ-DH spent about 2,500 seconds on average for the instances in this set and ended up with an average gap of around 5%.

Table 4 Number of instances each benchmark algorithm is applied to for the IRP and PRP data sets

Name of the Algorithm					ABLS-BC	CL-BC	ABS-H	ABW-BC	DRC-BPC	ABHS-H	CCL-ALNS	ABPS-BC	ABPS-H	BP-MA	BN-TS	ASL-TS	ACJ-ALNS-500	ACJ-ALNS-1000	AADF-MS	AADF-DMS	AADF-VRP	AADF-MTSP	CCJ-DH
Prob	Set	m	Class	Size	E	E	HM	E	E	H	M	E	H	M	M	M	M	M	H	H	H	H	H
IRP	SV-I1	1	-	160	160	160	-	-	-	160	160	-	-	-	-	-	-	-	-	-	-	-	160
	MV-I1	2	-	160	-	160	160	50	158	-	-	-	-	-	-	-	-	150	-	-	-	-	160
		3	-	160	-	160	160	50	160	-	-	-	-	-	-	-	-	150	-	-	-	-	160
		4	-	160	-	160	160	50	160	-	-	-	-	-	-	-	-	-	-	-	-	-	160
		5	-	158	-	158	158	49	158	-	-	-	-	-	-	-	-	-	-	-	-	-	158
	SV-I2	1	-	60	-	60	-	-	-	60	-	-	-	-	-	-	-	-	-	-	-	-	60
	MV-I2	2	-	60	-	40	60	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	60
		3	-	60	-	40	60	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	60
		4	-	60	-	-	60	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	60
		5	-	60	-	-	60	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	60
PRP	SV-A1	1	1	120	-	-	-	-	-	-	-	120	120	-	-	-	120	120	120	120	-	-	120
		1	2	120	-	-	-	-	-	-	-	120	120	-	-	-	120	120	120	120	-	-	120
		1	3	120	-	-	-	-	-	-	-	120	120	-	-	-	120	120	120	120	-	-	120
		1	4	120	-	-	-	-	-	-	-	116	120	-	-	-	120	120	120	120	-	-	120
	MV-A2	UL	1	120	-	-	-	-	-	-	-	-	120	-	-	-	120	120	-	-	120	120	120
		UL	2	120	-	-	-	-	-	-	-	-	120	-	-	-	120	120	-	-	120	120	120
		UL	3	120	-	-	-	-	-	-	-	-	120	-	-	-	120	120	-	-	120	120	120
		UL	4	120	-	-	-	-	-	-	-	-	120	-	-	-	120	120	-	-	120	120	120
	MV-A3	UL	1	120	-	-	-	-	-	-	-	-	120	-	-	-	120	120	-	-	120	120	120
		UL	2	120	-	-	-	-	-	-	-	-	120	-	-	-	120	120	-	-	120	120	120
		UL	3	120	-	-	-	-	-	-	-	-	120	-	-	-	120	120	-	-	120	120	120
		UL	4	120	-	-	-	-	-	-	-	-	120	-	-	-	120	120	-	-	120	120	120
	MV-B1	5	-	30	-	-	-	-	-	-	-	-	-	30	30	30	30	30	-	-	30	30	30
	MV-B2	9	-	30	-	-	-	-	-	-	-	-	-	30	30	30	30	30	-	-	30	30	30
	MV-B3	13	-	30	-	-	-	-	-	-	-	-	-	30	30	30	30	-	-	-	30	30	30
Total				2,628	160	938	878	199	636	220	160	476	1,440	90	90	90	1,530	1,800	480	480	1,050	1,050	2,628

Note. E: Exact, M: Metaheuristic, H: Heuristic, Prob: Problem, UL: Unlimited

For the MV-I2 data set there are two algorithms to compare with: CL-BC and ABS-H. Since the size of the instances and the number of available vehicles are larger compared to the MV-I1 data set, the CL-BC algorithm was not able to solve the instances with $m = 4$ and 5 and $n = 200$. This algorithm left average gaps of more than 64% and 110% for the instances with $m = 2$ and 3, respectively and found only 8 BUBs (among the instances with $m = 2$) while spending 86,400 seconds on every instance in the set. ABS-H was also successful on this data set by finding 59 BUBs. CCJ-DH outperformed the two existing approaches on this data set, finding 173 new best solutions and obtaining average gaps between -0.48% and -3.96% while spending less time compared to ABS-H. The larger the number of nodes and the number of vehicles, the better the results obtained by CCJ-DH compared to ABS-H. This is an interesting result since ABS-H is a specialized algorithm for the multi-vehicle IRPs.

For the SV-A1 data set, there are five algorithms available in the benchmark set that were applied to all the instances: ABPS-BC, ABPS-H, ACJ-ALNS with 500 and 1000 iter-

Table 5 Average gaps by different algorithms applied to IRP and PRP data sets (%)

Name of the Algorithm					ABLS-BC	CL-BC	ABS-H	ABW-BC	DRC-BPC	ABHS-H	CCL-ALNS	ABPS-BC	ABPS-H	BP-MA	BN-TS	ASL-TS	ACJ-ALNS-500	ACJ-ALNS-1000	AADF-MS	AADF-DMS	AADF-VRP	AADF-MTSP	CCJ-DH
Prob	Set	m	Class	Size	E	E	HM	E	E	H	M	E	H	M	M	M	M	M	H	H	H	H	H
IRP	SV-I1	1	-	160	**0**	**0**	-	-	-	0.05	0.46	-	-	-	-	-	-	-	-	-	-	-	3.08
	MV-I1	2	-	160	-	**0**	0.21	0.41	19.95	-	-	-	-	-	-	-	-	6.57	-	-	-	-	4.27
		3	-	160	-	**0.27**	0.67	0.82	18.41	-	-	-	-	-	-	-	-	6.75	-	-	-	-	3.8
		4	-	160	-	5.32	1.34	**1.17**	17.05	-	-	-	-	-	-	-	-	-	-	-	-	-	4.11
		5	-	158	-	10.53	1.5	**1.27**	14.85	-	-	-	-	-	-	-	-	-	-	-	-	-	4.24
	SV-I2	1	-	60	-	10.91	-	-	-	**0.27**	-	-	-	-	-	-	-	-	-	-	-	-	5.04
	MV-I2	2	-	60	-	61.32	0.12	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	**-0.48**
		3	-	60	-	106.28	0	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	**-2.31**
		4	-	60	-	-	0	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	**-3.58**
		5	-	60	-	-	0	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	**-3.96**
PRP	SV-A1	1	1	120	-	-	-	-	-	-	-	**0**	2.21	-	-	-	1.73	1.7	0.09	0.13	-	-	0.47
		1	2	120	-	-	-	-	-	-	-	**0**	0.3	-	-	-	0.36	0.36	0.01	0.02	-	-	0.08
		1	3	120	-	-	-	-	-	-	-	**0**	3.65	-	-	-	9.1	8.43	0.57	0.72	-	-	2.2
		1	4	120	-	-	-	-	-	-	-	**0**	0.9	-	-	-	0.96	0.96	0.04	0.07	-	-	0.25
	MV-A2	UL	1	120	-	-	-	-	-	-	-	-	1.95	-	-	-	1.01	1	-	-	0.04	0.95	**-0.04**
		UL	2	120	-	-	-	-	-	-	-	-	0.35	-	-	-	0.14	0.14	-	-	0.02	0.07	**0**
		UL	3	120	-	-	-	-	-	-	-	-	2.76	-	-	-	3.01	2.78	-	-	0.26	1.99	**-0.39**
		UL	4	120	-	-	-	-	-	-	-	-	1.19	-	-	-	0.14	0.13	-	-	0.04	0.4	**-0.06**
	MV-A3	UL	1	120	-	-	-	-	-	-	-	-	2.12	-	-	-	0.89	0.84	-	-	**0.06**	1.7	0.10
		UL	2	120	-	-	-	-	-	-	-	-	0.33	-	-	-	0.3	0.29	-	-	0.19	0.27	**0.18**
		UL	3	120	-	-	-	-	-	-	-	-	2.65	-	-	-	2.81	2.63	-	-	**0.23**	2.79	0.6
		UL	4	120	-	-	-	-	-	-	-	-	1.22	-	-	-	0.28	0.26	-	-	0.18	0.64	**0**
	MV-B1	5	-	30	-	-	-	-	-	-	-	-	-	13.63	6.83	4.53	0.34	0.23	-	-	0.84	0.6	**0.03**
	MV-B2	9	-	30	-	-	-	-	-	-	-	-	-	12.58	12.22	8.06	0.36	0.35	-	-	**0.24**	0.71	0.48
	MV-B3	13	-	30	-	-	-	-	-	-	-	-	-	15.84	19.69	10.05	1.39	-	-	-	0.14	1.71	**-0.21**

Note. The best average gap at each row is presented in the bold font.

ations, AADF-MS and AADF-DMS. Among the heuristic and metaheuristic algorithms, the specialized algorithms of AADF-MS and AADF-DMS are the best performing ones, but also spent the largest computing times. Their times are comparable to or larger than those of the exact algorithm of ABPS-BC. ABPS-H and ACJ-ALNS (with 500 and 1000 iterations) are the only benchmark algorithms that were applied to all three data sets of Archetti et al. (2011). While ABPS-H generally obtained better results than ACJ-ALNS for SV-A1 with almost negligible computing times, both are outperformed by CCJ-DH in terms of the number of BUBs and average gaps.

There are five sophisticated heuristic or metaheuristic algorithms available for the MV-A2 and MV-A3 data sets. Due to the size of the instances ($n = 50$ and 100, $l = 6$), no exact algorithm has yet been applied to these sets. The results presented in Tables 5 and 6 show that our algorithm outperforms all the other algorithms on these two data sets both in total number of BUBs and average gaps, except for AADF-VRP which obtains better gaps on two subclasses in MV-A3, and more BUBs in one subclass of MV-A3.

Six different algorithms were tested on the data sets of Boudia et al. (2005): BP-MA, BN-TS, ASL-TS, ACJ-ALNS with 500 and 1000 iterations, AADF-VRP and AADF-MTSP.

Table 6 Number of best solutions found by different algorithms applied to IRP and PRP data sets

Name of the Algorithm					ABLS-BC	CL-BC	ABS-H	ABW-BC	DRC-BPC	ABHS-H	CCL-ALNS	ABPS-BC	ABPS-H	BP-MA	BN-TS	ASL-TS	ACJ-ALNS-500	ACJ-ALNS-1000	AADF-MS	AADF-DMS	AADF-VRP	AADF-MTSP	CCJ-DH	
Prob	Set	m	Class	Size	E	E	HM	E	E	H	M	E	H	M	M	M	M	M	H	H	H	H	H	NBS
IRP	SV-I1	1	-	160	**160**	**160**	-	-	-	122	71	-	-	-	-	-	-	-	-	-	-	-	25	0
	MV-I1	2	-	160	-	**158**	92	22	84	-	-	-	-	-	-	-	-	14	-	-	-	-	12	0
		3	-	160	-	**142**	75	2	91	-	-	-	-	-	-	-	-	12	-	-	-	-	17	6
		4	-	160	-	**107**	52	5	95	-	-	-	-	-	-	-	-	-	-	-	-	-	19	14
		5	-	158	-	78	44	6	**101**	-	-	-	-	-	-	-	-	-	-	-	-	-	22	15
	SV-I2	1	-	60	-	30	-	-	-	**31**	-	-	-	-	-	-	-	-	-	-	-	-	0	0
	MV-I2	2	-	60	-	8	17	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	**35**	35
		3	-	60	-	0	20	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	**40**	40
		4	-	60	-	-	14	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	**46**	46
		5	-	60	-	-	8	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	**52**	52
PRP	SV-A1	1	1	120	-	-	-	-	-	-	-	**119**	6	-	-	-	1	1	78	68	-	-	10	0
		1	2	120	-	-	-	-	-	-	-	**120**	5	-	-	-	0	0	78	65	-	-	10	0
		1	3	120	-	-	-	-	-	-	-	**120**	2	-	-	-	0	0	52	44	-	-	4	0
		1	4	120	-	-	-	-	-	-	-	**116**	14	-	-	-	1	1	83	73	-	-	20	0
	MV-A2	UL	1	120	-	-	-	-	-	-	-	-	1	-	-	-	0	0	-	-	28	2	**89**	89
		UL	2	120	-	-	-	-	-	-	-	-	4	-	-	-	2	2	-	-	25	22	**67**	67
		UL	3	120	-	-	-	-	-	-	-	-	2	-	-	-	0	2	-	-	12	6	**98**	98
		UL	4	120	-	-	-	-	-	-	-	-	0	-	-	-	0	1	-	-	25	9	**85**	85
	MV-A3	UL	1	120	-	-	-	-	-	-	-	-	10	-	-	-	0	0	-	-	50	0	**60**	60
		UL	2	120	-	-	-	-	-	-	-	-	24	-	-	-	0	0	-	-	27	15	**54**	54
		UL	3	120	-	-	-	-	-	-	-	-	15	-	-	-	1	1	-	-	**55**	2	47	47
		UL	4	120	-	-	-	-	-	-	-	-	16	-	-	-	0	0	-	-	29	3	**72**	72
	MV-B1	5	-	30	-	-	-	-	-	-	-	-	-	0	0	1	5	7	-	-	1	7	**14**	14
	MV-B2	9	-	30	-	-	-	-	-	-	-	-	-	0	0	0	5	6	-	-	**15**	5	4	4
	MV-B3	13	-	30	-	-	-	-	-	-	-	-	-	0	0	0	0	-	-	-	7	3	**20**	20
Total				2,628	160	683	322	35	371	153	71	475	99	0	0	1	15	47	291	250	274	74	922	818

Note. The largest number of obtained BUBs at each row is presented in the bold font.

On MV-B1, CCJ-DH is the best performing algorithm. AADF-VRP is the best performing algorithm on the MV-B2 data set. Also, ACJ-ALNS with both 500 and 1000 iterations performs better than CCJ-DH both in terms of the number of BUBs and the average gap on the MV-B2 data set. While AADF-MTSP found one more BUB compared to CCJ-DH, our algorithm returned a lower average gap. CCJ-DH also obtained the best results on the MV-B3 set compared to other algorithms, especially in terms of the average gap. Overall, CCJ-DH and AADF-VRP are the best performing algorithms (non dominated ones) on these three data sets. The average gap of CCJ-DH on all the 90 instances in these data sets is 0.1%, performing better than AADF-VRP with an overall average gap of 0.40%. Moreover, CCJ-DH found more BUBs compared to AADF-VRP on these three data sets.

On all IRP and PRP data sets with 2,628 instances, CCJ-DH was able to find 922 BUBs out of which 818 are new best solutions. Our algorithm shows consistent performance especially on the large-scale multi-vehicle instances of both IRP and PRP. For this family of

Table 7 Average running time of different algorithms applied to IRP and PRP data sets (seconds)

Name of the Algorithm					ABLS-BC	CL-BC	ABS-H	ABW-BC	DRC-BPC	ABHS-H	CCL-ALNS	ABPS-BC	ABPS-H	BP-MA	BN-TS	ASL-TS	ACJ-ALNS-500	ACJ-ALNS-1000	AADF-MS	AADF-DMS	AADF-VRP	AADF-MTSP	CCJ-DH
Prob	Set	m	Class	Size	E	E	HM	E	E	H	M	E	H	M	M	M	M	M	H	H	H	H	H
IRP	SV-I1	1	-	160	616	19	-	-	-	459	498	-	-	-	-	-	-	-	-	-	-	-	39
	MV-I1	2	-	160	-	4,099	1,259	853	4,121	-	-	-	-	-	-	-	-	34	-	-	-	-	65
		3	-	160	-	15,319	1,585	1,083	4,124	-	-	-	-	-	-	-	-	39	-	-	-	-	71
		4	-	160	-	23,884	800	1,125	3,862	-	-	-	-	-	-	-	-	-	-	-	-	-	77
		5	-	158	-	28,244	914	1,139	3,680	-	-	-	-	-	-	-	-	-	-	-	-	-	66
	SV-I2	1	-	60	-	64,509	-	-	-	3,630	-	-	-	-	-	-	-	-	-	-	-	-	2,583
	MV-I2	2	-	60	-	86,400	4,066	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	2,590
		3	-	60	-	86,400	4,540	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	2,503
		4	-	60	-	-	4,257	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	3,632
		5	-	60	-	-	4,418	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	3,115
PRP	SV-A1	1	1	120	-	-	-	-	-	-	-	445	†	-	-	-	5	9	251	243	-	-	14
		1	2	120	-	-	-	-	-	-	-	11	†	-	-	-	5	9	214	210	-	-	14
		1	3	120	-	-	-	-	-	-	-	81	†	-	-	-	5	9	237	233	-	-	13
		1	4	120	-	-	-	-	-	-	-	527	†	-	-	-	5	9	217	218	-	-	12
	MV-A2	UL	1	120	-	-	-	-	-	-	-	-	11	-	-	-	29	50	-	-	23	315	329
		UL	2	120	-	-	-	-	-	-	-	-	12	-	-	-	28	50	-	-	23	288	273
		UL	3	120	-	-	-	-	-	-	-	-	9	-	-	-	24	43	-	-	26	335	253
		UL	4	120	-	-	-	-	-	-	-	-	11	-	-	-	26	44	-	-	26	330	272
	MV-A3	UL	1	120	-	-	-	-	-	-	-	-	188	-	-	-	136	249	-	-	86	514	1,313
		UL	2	120	-	-	-	-	-	-	-	-	217	-	-	-	125	221	-	-	76	497	1,063
		UL	3	120	-	-	-	-	-	-	-	-	168	-	-	-	107	191	-	-	75	509	1,017
		UL	4	120	-	-	-	-	-	-	-	-	181	-	-	-	108	189	-	-	87	507	1,057
	MV-B1	5	-	30	-	-	-	-	-	-	-	-	-	173	331	317	298	481	-	-	551	1,653	978
	MV-B2	9	-	30	-	-	-	-	-	-	-	-	-	1,108	976	1,148	1,405	1,570	-	-	2,054	9,483	5,441
	MV-B3	13	-	30	-	-	-	-	-	-	-	-	-	4,098	2,492	3,926	5,794	-	-	-	4,197	19,270	13,693

† The computing times are negligible.

instances, CCJ-DH was successful to obtain improved solutions compared to the previous BUBs found in the literature by the specialized algorithms. Among the 1,290 large-scale multi-vehicle instances of IRP and PRP data sets (240 instances of MV-I2, 960 instances of MV-A2 and MV-A3 and 90 instances of MV-B1, MV-B2 and MV-B3), CCJ-DH found 783 new best solutions. The algorithm also finished with the best or one of the best average gaps among the other benchmark algorithms. Moreover, CCJ-DH is the only algorithm that was applied to all the IRP and PRP data sets. ACJ-ALNS-1000 is the only other algorithm that has been applied to both the IRP and PRP problems. This metaheuristic was developed specifically for the PRP (Adulyasak et al. 2014b) and was next applied to a limited set of multi-vehicle IRP instances (Adulyasak et al. 2014a). The results in Table 5 indicate that CCJ-DH obtains improved gaps compared to ACJ-ALSN-1000 in all the tested classes, except for MV-B2.

An interesting observation can be made from Table 7: heuristic and metaheuristic algorithms (ABS-H, ABHS-H and CCL-ALNS) applied to the IRP data sets of Archetti et al. (2007) (SV-I1 and MV-I1) with at most 30 nodes and 6 periods or 50 nodes and 3 periods used more computing time than the heuristic and metaheuristic algorithms (ABPS-H,

ACJ-ALNS-500, ACJ-ALNS-1000, AADF-VRP and AADF-MTSP) applied to the MV-A2 PRP data set of Archetti et al. (2011) with 50 nodes and 6 periods. This is while the PRP is generally a harder problem to solve. The only exception is ACJ-ALNS-1000 which is applied to both problems and has relatively comparable running times for both problems. Almost the same happens for the heuristic and metaheuristic algorithms (ABS-H and ABHS-H) applied to the SV-I2 and MV-I2 IRP data sets that have 50 to 200 nodes and 6 periods compared to the heuristic and metaheuristic algorithms (BP-MA, BN-TS, ASL-TS, ACJ-ALNS-500, ACJ-ALNS-1000 and AADF-VRP) applied to the much larger and harder PRP data sets of Boudia et al. (2005) (MV-B1, MV-B2 and MV-B3) that have 50 to 200 nodes and 20 periods. The exception here is AADF-MTSP which has higher running times compared to the other algorithms applied to the Boudia et al. (2005) data sets. The imbalance between the running times of the different algorithms made it difficult to find a unique parameter setting for CCJ-DH on all problems and data sets.

5.5. Computational Results for the ARP Data Sets

On the ARP data sets, we compare our algorithm against a truncated branch-and-cut search using CPLEX with a time limit of 7,200 seconds. The results are shown in Table 8. Due to the size of the instances and memory limit of 96 GB, it was only possible to obtain the CPLEX results for the SV-C1 data set with $n = 14$ (see Table 9). CPLEX was able to solve most of the instances within the time limit. Also, it found most of the best upper bounds (BUB) for this data set. The gaps shown in this table are calculated based on the BUBs. Our algorithm was able to find 59 BUBs and in 5 instances it outperformed CPLEX. Moreover, on average our algorithm was able to obtain results with a reasonable gap of 0.7% in less than 25 seconds compared to an average CPU time of 527 seconds for CPLEX.

Table 8 Average gaps of CCJ-DH compared to CPLEX UBs on the SV-C1 ARP data set (%)

						CPLEX			CCJ-DH			
Set	m	l	n	Class	Size	CPU[†] (sec)	Nb. BUB	Gap (%)	CPU (sec)	Nb. BUB	NBS	Gap (%)
SV-C1	1	6	14	1	120	678.6	119	0.0	23.2	20	1	0.40
				2	120	414.9	119	0.0	23.2	10	1	0.32
				3	120	946.3	117	0.0	20.8	14	3	1.48
				4	120	66.4	120	0.0	28.0	15	0	0.53
Total/*Average*					480	*526.53*	475	*0.0*	*23.82*	59	5	*0.68*

[†] Maximum 7,200 seconds.

Table 9 Average lower bound gaps compared to the best LBs on the ARP data sets (%)

						CPLEX			LB1			LB2			CCJ-DH	
Set	m	l	n	Class	Size	CPU (sec)	Nb. BLB	Diff.† in LB (%)	CPU (sec)	Nb. BLB	Diff.† in LB (%)	CPU (sec)	Nb. BLB	Diff.† in LB (%)	CPU (sec)	Gap‡ (%)
SV-C1	1	6	14	1	120	678.6	120	0.0	0.3	0	2.6	2.1	0	4.3	23.2	0.40
				2	120	414.9	120	0.0	0.3	0	2.7	2.2	0	4.3	23.2	0.32
				3	120	946.3	120	0.0	0.3	0	12.1	3.3	0	19.0	20.8	1.48
				4	120	66.4	120	0.0	0.2	0	2.2	1.1	0	4.2	28.0	0.53
MV-C2	UL	6	50	1	120	-	-	-	7.8	120	0	1,245.7	0	1.0	473.8	3.22
				2	120	-	-	-	7.8	120	0	1,329.1	0	1.1	466.8	3.21
				3	120	-	-	-	9.3	120	0	2,196.3	0	3.7	344.4	12.89
				4	120	-	-	-	6.5	120	0	68.6	0	1.5	792.0	2.44
MV-C3	UL	6	100	1	120	-	-	-	63.8	120	0	5,313.5	0	0.8	2,056.0	3.39
				2	120	-	-	-	65.5	120	0	5,201.4	0	0.8	2,012.8	3.43
				3	120	-	-	-	61.1	120	0	7,186.6	0	2.9	1,516.5	15.10
				4	120	-	-	-	61.1	120	0	709.8	0	1.1	3,542.6	2.86

† Difference.
‡ The gap between the solution (UB) obtained by CCJ-DH and the best LB obtained by other methods.

To obtain a better assessment of the performance of the algorithm on the larger data sets (MV-C2 and MV-C3 with $n = 50$ and $n = 100$, respectively) we computed two more lower bounds for each instance by using $\mathcal{M}_{ARP}$ and either relaxing the GFSEC and keeping the integrality on all variables (LB1) or replacing the GFSEC by the Miller–Tucker–Zemlin (MTZ) SECs and then relaxing the integrality on the x variables (LB2). The results are presented in Table 9. The difference in LB for each method is calculated by comparing the obtained LB to the best lower bound obtained (BLB). The new lower bounds (LB) obtained by these two methods are not as good as the ones found by CPLEX for the small data set (SV-C1). In all cases, the LB1 bounds are better than the LB2's and it takes less time to find them. The last column in this table presents the average gaps in each class of instances for the solution obtained by our algorithm (UB) compared to the BLB for each instances. For SV-C1, the LBs obtained by the relaxations (LB1 and LB2) for the third class of instances in this data set have higher gaps compared to the other classes due to the higher relative transportation costs of this class. For MV-C2 and MV-C3, we observe a similar pattern, where the average gaps of the best upper bound are around 3% for classes 1, 2 and 3, but this gaps increases up to 15% for the third class.

We further discuss the behavior of our algorithm in Appendix D. All instances, detailed solutions and results can be found at `http://chairelogistique.hec.ca/en/scientific-data/`.

6. Summary and Conclusion

This study aimed at filling the gap in the literature by introducing a MILP model for the integrated production, inventory and inbound routing problem. Although some similari-

ties between the PRP and ARP exist, fundamental differences arise in the nature of the problem and in the modeling such as the presence of inventory at the plant level both for the final product and the components. We presented a compact formulation for the ARP ($\mathcal{M}_{ARP}$) and developed many test instances for this problem as well as an efficient heuristic algorithm. In short computing times CCJ-DH performs very well on the small ARP instances compared to the solutions obtained by a general-purpose MILP solver (CPLEX). We further tested this algorithm on other problems of the same nature where the routing decisions are integrated with inventory management (and production planning): the IRP and the PRP. We considered standard data sets from the literature which have been used by various researchers in more than one paper. These data sets include 2,628 instances ranging from small to very large-scale ones. Many state-of-the-art algorithms that were applied to one or several of these data sets are considered and the results are compared in detail with our algorithm. One of the most important contributions of this paper is the design of a unified algorithm that can be applied to different data sets of different problems (ARP, PRP and IRP) with the same parameter setting. Our algorithm presents acceptable results on the small data sets and outperforms specialized state-of-the-art algorithms for the large-scale multi-vehicle instances. We also outperform the only other algorithm that has been applied to both the IRP and PRP problems. We believe this shows the robustness of our decomposition approach and of the algorithm.

Acknowledgments

The authors thank professors Nabil Absi for making the solutions of Absi et al. (2014) available to us and Claudia Archetti for providing the results of Archetti et al. (2017). Computations were performed on the Calcul Québec infrastructure. This work was partly supported by the Canadian Natural Sciences and Engineering Research Council under grants 2014-03849 and 2014-04959. This support is gratefully acknowledged.

References

Absi, N., Archetti, C., Dauzère-Pérès, S. and Feillet, D. (2014). A two-phase iterative heuristic approach for the production routing problem, *Transportation Science* **49**(4): 784–795.

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

Adulyasak, Y., Cordeau, J.-F. and Jans, R. (2014b). Optimization-based adaptive large neighborhood search for the production routing problem, *Transportation Science* **48**(1): 20–45.

Adulyasak, Y., Cordeau, J.-F. and Jans, R. (2015). The production routing problem: A review of formulations and solution algorithms, *Computers & Operations Research* **55**: 141–152.

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

Archetti, C., Bertazzi, L., Hertz, A. and Speranza, M. G. (2012). A hybrid heuristic for an inventory routing problem, *INFORMS Journal on Computing* **24**(1): 101–116.

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

Archetti, C., Bertazzi, L., Paletta, G. and Speranza, M. G. (2011). Analysis of the maximum level policy in a production-distribution system, *Computers & Operations Research* **38**(12): 1731–1746.

Archetti, C., Boland, N. and Speranza, M. (2017). A matheuristic for the multi-vehicle inventory routing problem, *INFORMS Journal on Computing* (Forthcoming).

Armentano, V. A., Shiguemoto, A. and Løkketangen, A. (2011). Tabu search with path relinking for an integrated production–distribution problem, *Computers & Operations Research* **38**(8): 1199–1209.

Avella, P., Boccia, M., Wolsey, L. et al. (2014). Single-period cutting planes for inventory routing problems, *Technical report*, Universita del Sannio.

Bae, H., Moon, I. and Yun, W. (2014). Economic lot and supply scheduling problem: a time-varying lot sizes approach, *International Journal of Production Research* **52**(8): 2422–2435.

Bard, J. F. and Nananukul, N. (2009). The integrated production–inventory–distribution–routing problem, *Journal of Scheduling* **12**(3): 257–280.

Bard, J. F. and Nananukul, N. (2010). A branch-and-price algorithm for an integrated production and inventory routing problem, *Computers & Operations Research* **37**(12): 2202–2217.

Berman, O. and Wang, Q. (2006). Inbound logistic planning: minimizing transportation and inventory cost, *Transportation Science* **40**(3): 287–299.

Bertazzi, L., Savelsbergh, M. and Speranza, M. G. (2008). Inventory routing, *in* B. L. Golden, S. Raghavan and E. A. Wasil (eds), *The Vehicle Routing Problem: Latest Advances and New Challenges*, Springer Science & Business Media, pp. 49–72.

Blumenfeld, D. E., Burns, L. D., Daganzo, C. F., Frick, M. C. and Hall, R. W. (1987). Reducing logistics costs at general motors, *Interfaces* **17**(1): 26–47.

Boudia, M., Louly, M. A. O. and Prins, C. (2005). Combined optimization of production and distribution, *CD-ROM Proceedings of the international conference on industrial engineering and systems management, IESM*, Vol. 5.

Boudia, M., Louly, M. A. O. and Prins, C. (2007). A reactive grasp and path relinking for a combined production–distribution problem, *Computers & Operations Research* **34**(11): 3402–3419.

Boudia, M. and Prins, C. (2009). A memetic algorithm with dynamic population management for an integrated production–distribution problem, *European Journal of Operational Research* **195**(3): 703–715.

Chandra, P. (1993). A dynamic distribution model with warehouse and customer replenishment requirements, *Journal of the Operational Research Society* **44**(7): 681–692.

Chandra, P. and Fisher, M. L. (1994). Coordination of production and distribution planning, *European Journal of Operational Research* **72**(3): 503–517.

Chen, Z. and Sarker, B. R. (2014). An integrated optimal inventory lot-sizing and vehicle-routing model for a multisupplier single-assembler system with JIT delivery, *International Journal of Production Research* **52**(17): 5086–5114.

Coelho, L. C., Cordeau, J.-F. and Laporte, G. (2012). The inventory-routing problem with transshipment, *Computers & Operations Research* **39**(11): 2537–2548.

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

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

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

Cordeau, J.-F., Gendreau, M. and Laporte, G. (1997). A tabu search heuristic for periodic and multi-depot vehicle routing problems, *Networks* **30**(2): 105–119.

Danese, P. (2006). The extended vmi for coordinating the whole supply network, *Journal of Manufacturing Technology Management* **17**(7): 888–907.

Desaulniers, G., Rakke, J. G. and Coelho, L. C. (2015). A branch-price-and-cut algorithm for the inventory-routing problem, *Transportation Science* **50**(3): 1060–1076.

Fernie, J. and Sparks, L. (2014). *Logistics and Retail Management: Emerging Issues and New Challenges in the Retail Supply Chain*, Kogan Page Publishers.

Fischetti, M. and Lodi, A. (2003). Local branching, *Mathematical Programming* **98**(1-3): 23–47.

Fleischmann, B. and Meyr, H. (2003). Planning hierarchy, modeling and advanced planning systems, *Handbooks in Operations Research and Management Science* **11**: 455–523.

Florian, M., Kemper, J., Sihn, W. and Hellingrath, B. (2011). Concept of transport-oriented scheduling for reduction of inbound logistics traffic in the automotive industries, *CIRP Journal of Manufacturing Science and Technology* **4**(3): 252–257.

Fumero, F. and Vercellis, C. (1999). Synchronized development of production, inventory, and distribution schedules, *Transportation Science* **33**(3): 330–340.

Hein, F. and Almeder, C. (2016). Quantitative insights into the integrated supply vehicle routing and production planning problem, *International Journal of Production Economics* **177**: 66–76.

Kuhn, H. and Liske, T. (2011). Simultaneous supply and production planning, *International Journal of Production Research* **49**(13): 3795–3813.

Kuhn, H. and Liske, T. (2014). An exact algorithm for solving the economic lot and supply scheduling problem using a power-of-two policy, *Computers & Operations Research* **51**: 30–40.

Le Blanc, H. M., Cruijssen, F., Fleuren, H. A. and De Koster, M. (2006). Factory gate pricing: An analysis of the dutch retail distribution, *European Journal of Operational Research* **174**(3): 1950–1967.

Lee, C.-G., Bozer, Y. A. and White III, C. (2003). A heuristic approach and properties of optimal solutions to the dynamic inventory routing problem, *Technical report*, University of Toronto, Ontario, Canada.

Lei, L., Liu, S., Ruszczynski, A. and Park, S. (2006). On the integrated production, inventory, and distribution routing problem, *IIE Transactions* **38**(11): 955–970.

Liske, T. and Kuhn, H. (2009). The economic lot and supply scheduling problem under a power-of-two policy, *Operations Research Proceedings 2008*, Springer Science & Business Media, pp. 215–220.

Mjirda, A., Jarboui, B., Macedo, R., Hanafi, S. and Mladenović, N. (2014). A two phase variable neighborhood search for the multi-product inventory routing problem, *Computers & Operations Research* **52**: 291–299.

Moin, N. H., Salhi, S. and Aziz, N. (2011). An efficient hybrid genetic algorithm for the multi-product multi-period inventory routing problem, *International Journal of Production Economics* **133**(1): 334–343.

Pochet, Y. and Wolsey, L. A. (2006). *Production Planning by Mixed Integer Programming*, Springer Science & Business Media.

Popken, D. A. (1994). An algorithm for the multiattribute, multicommodity flow problem with freight consolidation and inventory costs, *Operations Research* **42**(2): 274–286.

Potter, A., Mason, R. and Lalwani, C. (2007). Analysis of factory gate pricing in the uk grocery supply chain, *International Journal of Retail & Distribution Management* **35**(10): 821–834.

Qu, W. W., Bookbinder, J. H. and Iyogun, P. (1999). An integrated inventory–transportation system with modified periodic policy for multiple products, *European Journal of Operational Research* **115**(2): 254–269.

Sindhuchao, S., Romeijn, H. E., Akçali, E. and Boondiskulchok, R. (2005). An integrated inventory-routing system for multi-item joint replenishment with limited vehicle capacity, *Journal of Global Optimization* **32**(1): 93–118.

Solyalı, O., Süral, H., Neogy, S., Das, A. and Bapat, R. (2009). A relaxation based solution approach for the inventory control and vehicle routing problem in vendor managed systems, *Modeling, Computation and Optimization, World Scientific, Singapore* pp. 171–189.

Whiteoak, P. (1994). The realities of quick response in the grocery sector: a supplier viewpoint, *International Journal of Physical Distribution & Logistics Management* **24**(10): 33–39.

Appendix A: Overview of Problem Data Sets

We tested our algorithm on three different problems: the IRP, the PRP and the ARP. Each problem has its own set of different instances. IRP instances include four data sets. The first set (SV-I1) was provided by Archetti et al. (2007) and contains a total of 160 single-vehicle IRP instances. This data set includes instances with 5 to 50 nodes and 3 periods (100 instances) and instances with 5 to 30 nodes and 6 periods (60 instances). Coelho and Laporte (2013a) and Desaulniers et al. (2015) further adapted the SV-I1 data set and presented new instances (second data set) by dividing the fleet capacity equally between the number of vehicles. They considered $m = 2, 3, 4$ and 5 vehicles for each SV-I1 instance and made four new multi-vehicle IRP instances (MV-I1). Dividing the vehicle capacity by 5 made two of the instances infeasible. Therefore, instead of 640 they came up with a total number of 638 instances in this set. The third IRP data set (SV-I2) includes bigger single-vehicle instances presented by Archetti et al. (2012). This data set includes 60 instances with 6 periods and 50, 100 and 200 nodes (20 instances for each). Similar to the second data set, Coelho and Laporte (2013b) adapted the SV-IRP instances of the third set and developed the fourth multi-vehicle IRP data set (MV-I2) which includes 240 instance. Therefore, the total number of IRP instances in the four data sets that we considered in this study is 1,098.

PRP instances include six data sets. Archetti et al. (2011) and Boudia et al. (2005) each introduced three data sets. Each set of Archetti et al. (2011) has 480 instances including four classes of randomly generated problem instances. The first set provides single-vehicle PRP instances (SV-A1). The other two sets include multi-vehicle instances (MV-A2 and MV-A3). Sets SV-A1, MV-A2 and MV-A3 have 14, 50, 100 customers, respectively, each with 6 periods. Each of these sets has four classes of instances. The first class includes the normal or standard instances. The second class contains high production unit and setup cost instances. The third class represents the case with high transportation costs (by multiplying the customer coordinates of the first class by a factor of 5). Finally, the fourth class includes instances with no retailer inventory costs. Boudia et al. (2005) presented 30 test instances in each of their three sets: sets MV-B1, MV-B2 and MV-B3 which have 50, 100 and 200 customers, respectively, each with 20 periods. Accordingly, the total number of instances in these six PRP data sets is 1,530.

ARP instances include three data sets (SV-C1, MV-C2 and MV-C3) with a total number of 1,440 instances and are adapted from the PRP data sets of Archetti et al. (2011) for this new problem. Consequently, we are solving a total of 4,068 instances of the IRP, PRP and ARP problems. An overview of the main characteristics of these instances is presented in Table 10.

Appendix B: Parameter Setting

We chose a random but varied subset of instances from the entire test bed of problems to calibrate the parameters of the algorithm. For the IRP data sets of Archetti et al. (2007) and Archetti et al. (2012), with a total number of 1,098 instances, we randomly chose two instances for each combination of the fleet size (m), period (l) and inventory cost level (h) which resulted in a total of 60 instances: 40 instances from Archetti et al. (2007) and 20 instances from Archetti et al. (2012). From the PRP data sets of Archetti et al. (2011), we randomly took four instances from each class of instances, resulting in 16 instances from each data set and a total of 48 instances. From the PRP data sets of Boudia et al. (2005), we randomly chose four instances

Table 10 Overview of the benchmark data sets for the IRP, PRP and ARP

Problem	Reference	Set name	Size	l	n	m	d_i	C	L_0	L_i	I_0	I_i	Q
IRP	Archetti et al. (2007)	SV-I1	100	3	5 to 50	1	C	-	UL	C	V	V	C
			60	6	5 to 50	1	C	-	UL	C	V	V	C
		MV-I1	100	3	5 to 50	2	C	-	UL	C	V	V	C
			60	6	5 to 50	2	C	-	UL	C	V	V	C
			100	3	5 to 50	3	C	-	UL	C	V	V	C
			60	6	5 to 50	3	C	-	UL	C	V	V	C
			100	3	5 to 50	4	C	-	UL	C	V	V	C
			60	6	5 to 50	4	C	-	UL	C	V	V	C
			100	3	5 to 50	5	C	-	UL	C	V	V	C
			58	6	5 to 50	5	C	-	UL	C	V	V	C
	Archetti et al. (2012)	SV-I2	60	6	50 to 200	1	C	-	UL	C	V	V	C
		MV-I2	60	6	50 to 200	2	C	-	UL	C	V	V	C
			60	6	50 to 200	3	C	-	UL	C	V	V	C
			60	6	50 to 200	4	C	-	UL	C	V	V	C
			60	6	50 to 200	5	C	-	UL	C	V	V	C
PRP	Archetti et al. (2011)	SV-A1	480	6	14	1	C	UL	UL	C	0	V	C
		MV-A2	480	6	50	UL	C	UL	UL	C	0	V	C
		MV-A3	480	6	100	UL	C	UL	UL	C	0	V	C
	Boudia et al. (2005)	MV-B1	30	20	50	5	V	C	C	C	V	0	C
		MV-B2	30	20	100	9	V	C	C	C	V	0	C
		MV-B3	30	20	200	13	V	C	C	C	V	0	C
ARP	This paper	SV-C1	480	6	14	1	C	UL	UL	C	V	V	C
		MV-C2	480	6	50	UL	C	UL	UL	C	V	V	C
		MV-C3	480	6	100	UL	C	UL	UL	C	V	V	C
		Total	4,068										

Note. C: Constant/Capacitated, MV: multi-vehicle, SV: single-vehicle, UL: Unlimited, V: Varying.

from MV-B1 and MV-B2, resulting in 8 instances. No instances from the MV-B3 data set were chosen since they require long computing times. From the ARP data sets, we randomly selected four instances from each class of instances, resulting in 48 instances. Therefore, we performed the parameter setting experiments on 164 instances.

The most important algorithmic parameters to set are the maximum number of iterations for the algorithm, ι^A, and the tabu search iterations coefficient, ι^V, for the solution of the VRP_t subproblems. The rest of the parameters are the maximum number of local optimum iterations, ι^L, the maximum number of iterations without incumbent solution improvement, ι^N, the maximum number of $\mathcal{M}_z^{\mathcal{R}}$ model iterations, $\iota^{\mathcal{R}}$, the right-hand-side of the LBI_z inequalities, r, the reduction in the aggregate fleet capacity in constraints (21), $1-\lambda_t$, the gap between the solution obtained using the $\mathcal{M}_z$ model and the incumbent solution, g, and the gap between the solution obtained using the $\mathcal{M}_z^{\mathcal{R}}$ model and incumbent solution, $g^{\mathcal{R}}$.

We performed an extensive study on the parameter setting and arrived at the values mentioned in Table 11. Then, we designed a sensitivity analysis to make sure that the selected values are the right choice for our algorithm. Obviously, when ι^A and ι^V increase we obtain better results (see Table 12). But since the same parameter setting is used for all the problems and data sets, we have an implicit limit on the number of iterations in order to spend an acceptable computing time compared to other benchmark algorithms. Our observation indicates that the algorithm has an acceptable performance with small changes in ι^s, $\iota^{\mathcal{R}}$, g and $g^{\mathcal{R}}$ while the current setting for these four parameters helps us to reduce the necessary computing time. Also,

we noticed that the best $1-\lambda_t$ varies among different IRP and PRP data sets. The ranges of the sensitivity analyses on the parameter values are presented in the last column of Table 11.

Table 11 Parameter setting for the algorithm applied to all problems and data sets

Par	Description	Selected value	Other values for the sensitivity analysis
ι^A	Max Nb. of algorithm I-	150	50, 100, 200
ι^V	Tabu search I- coefficient	300	100, 200, 400, 500
ι^L	Max Nb. of local optimum I-	60 ($0.4*\iota^A$)	45 ($0.3*\iota^A$), 75 ($0.5*\iota^A$)
ι^N	Max Nb. of non-improving I-	30 ($0.2*\iota^A$)	15 ($0.1*\iota^A$), 45 ($0.3*\iota^A$)
ι^s	Nb. of I- before $\mathcal{M}_z^{\mathcal{R}}$ model can be used	5	4, 6
$\iota^{\mathcal{R}}$	Max Nb. of $\mathcal{M}_z^{\mathcal{R}}$ model I-	15	3, 5, 7, 10, 20
$1-\lambda_t$	Aggregate fleet capacity reduction amount†	$2/n$	$1/n$, $3/n$, $4/n$
g	Gap of $\mathcal{S}$ obtained using $\mathcal{M}_z$ model and $\mathcal{S}^*$	3%	2%, 4%
$g^{\mathcal{R}}$	Gap of $\mathcal{S}$ obtained using $\mathcal{M}_z^{\mathcal{R}}$ model and $\mathcal{S}^*$	0.3%	0.2%, 0.5%, 0.7%

Note. Max: Maximum, I-: Iterations, Par: Parameter, $\mathcal{S}$: Current solution, $\mathcal{S}^*$: Incumbent solution
† Up to a maximum of 25%

We used the following CPLEX setting for all problems and data sets to solve the $\mathcal{M}_y, \mathcal{M}_z$ and $\mathcal{M}_z^{\mathcal{R}}$ models. We used CPLEX with one thread in all of our experiments. We disabled all the CPLEX MIP cuts except *FlowCovers* and *Gomory*. We set the *AdvInd* parameter to zero to prevent CPLEX from spending time to recover the previous iteration's search tree with its built-in heuristic. The rest of the CPLEX settings follow the strategy of getting quality upper bounds faster rather than closing the optimality gap when solving the $\mathcal{M}_y, \mathcal{M}_z$ and $\mathcal{M}_z^{\mathcal{R}}$ models. We set *Dive* to 2 (probing dive for the MIP dive strategy), *OrderType* to 1 (to use decreasing costs for the MIP priority order generation in the search tree), *CoeffReduce* to 1 (to reduce only to integral coefficients when the coefficient reduction is used by CPLEX), *DGradient* to 4 (steepest-edge pricing with unit initial norms for the dual simplex pricing algorithm) and *MIP Emphasis* to 1 (to emphasize feasibility over optimality in the search tree). These allow us to terminate CPLEX sooner and execute more iterations. We set a maximum time limit of $\max(5, n/5)$ seconds for CPLEX.

Appendix C: Subproblems for the PRP and IRP

In PRP and IRP, the set of nodes $N_s = \{1, ..., n\}$, indexed by $i \in N_s$, represents the customers, $i = 0$ represents the plant and $N = N_s \cup \{0\}$ is the set of all nodes. Let K_i denote the storage capacity and F_{it} represent the inventory of the product (at the end of period t) at customer $i \in N_s$ and at the plant for $i = 0$. The rest of the parameters and variables have a similar definition as in the ARP. The $\mathcal{M}_y$ model for the PRP is defined as follows:

$$\min \sum_{t \in T} \left(up_t + fy_t + \sum_{i \in N} h_i F_{it} + \sigma_{0t} z_{0t} \right) \tag{26}$$

s.t.

$$F_{0,t-1} + p_t = \sum_{i \in N_s} q_{it} + F_{0t} \quad \forall t \in T \tag{27}$$

$$F_{i,t-1} + q_{it} = d_{it} + F_{it} \quad \forall i \in N_s, \forall t \in T \tag{28}$$

$$p_t \leq min\{C, \sum_{i \in N_s} d_{itl}\} y_t \quad \forall t \in T \tag{29}$$

$$q_{it} \leq min\{K_i, Q, d_{itl}\} z_{it} \quad \forall i \in N_s, \forall t \in T \tag{30}$$

$$\sum_{i \in N_s} q_{it} \leq Q z_{0t} \quad \forall t \in T \tag{31}$$

$$z_{0t} \leq m \quad \forall t \in T \tag{32}$$

$$F_{it} \leq K_i \quad \forall i \in N \tag{33}$$

$$p_t \geq 0, y_t \in \{0,1\}, z_{0t} \in \mathbb{Z} \quad \forall t \in T \tag{34}$$

$$F_{it} \geq 0 \quad \forall i \in N \tag{35}$$

$$q_{it} \geq 0, z_{it} \in \{0,1\} \quad \forall i \in N_s, \forall t \in T. \tag{36}$$

The objective function (26) minimizes the total production, setup, and inventory costs both at the plant and customers together with the vehicle dispatch cost. Constraints (27) and (28) ensure the inventory flow at the plant and at the customers, respectively. Constraints (29) and (30) force setup costs at the plant and vehicle visits to the customers, respectively. They also impose limits on production and shipment quantities. Constraints (31) are equivalent to constraints (19) for the ARP. Constraints (32) and (33) enforce the fleet size and storage limits at the plant and customers. The $\mathcal{M}_z$ model for the PRP is to minimize the following objective function:

$$\min \sum_{t \in T} \left(up_t + f y_t + \sum_{i \in N} h_i F_{it} + \sum_{i \in N_s} \sigma_{it} z_{it} \right), \tag{37}$$

subject to constraints (27)-(30), (32)-(36) and (38):

$$\sum_{i \in N_s} q_{it} \leq \lambda_t m Q \quad \forall t \in T. \tag{38}$$

Constraints (38) are the equivalent of constraints (21) for the PRP. Having the binary variables y_t fixed from the solution of the $\mathcal{M}_y$ model, they become constants in constraints (29) for the $\mathcal{M}_z$ model. In the $\mathcal{M}_z$ model for IRP, p_t is a parameter that makes the constraints (29) not applicable. The objective function of the $\mathcal{M}_z$ model will then be:

$$\min \sum_{t \in T} \sum_{i \in N} (h_i F_{it} + \sigma_{it} z_{it}). \tag{39}$$

To comply with the replenishment process timing assumption in Archetti et al. (2007) and Archetti et al. (2011), Adulyasak et al. (2014a) suggested constraints (40) that should be added to the $\mathcal{M}_z$ model for IRP. Constraints (41) are equivalent to the original assumption (Archetti et al. 2007, 2011): $F_{i,t-1} + q_{it} \leq K_i$ and can be obtained by replacing the LHS from constraints (28). The reason for this modification is to impose them as bounds on the inventory variables rather than adding constraints to the model.

$$F_{0t} \geq p_t \quad \forall t \in T \tag{40}$$

$$F_{it} \leq K_i - d_{it} \quad \forall t \in T \tag{41}$$

Moreover, the fixed cost $\sum_{i \in N} h_i F_{i0}$ has to be added to the final solution value for the IRP instances, since Archetti et al. (2007) and Archetti et al. (2011) consider the inventory costs at the beginning of the period starting from period zero.

Appendix D: Further Analysis of the Algorithm

We further analyzed the algorithm performance for different numbers of iterations. In addition to 150 iterations that we fixed for CCJ-DH for all problems and data sets, we let it run for $\iota^A = \{50, 100, 200, 300\}$. The average gap (%), computing time (seconds) and number of BUBs obtained are presented in Table 12. The best performance of CCJ-DH can be observed on the MV-I2 data set especially when the number of vehicles, m, increases. CCJ-DJ is successful to find average gaps less than 0 or in other words it outperforms the state-of-the-art algorithm (ABS-H) on this data set after 100 iterations for $m = 2$ and 50 iterations for $m = 3, 4$ and 5. On the MV-A2 data set the algorithm with 100 iterations performs almost the same or better than all the previous benchmark algorithms. Within 50 iterations, the algorithm finds 167 BUBs (53, 30, 29 and 55 for the first to fourth class, respectively) on the MV-A3 data set, returns acceptable average gaps on the MV-B1 and MV-B2 data sets and outperforms the previously best found solutions on the MV-B3. CCJ-DH is able to find more than a thousand BUBs after 300 iterations on all data sets.

Table 12 Average gap (%), CPU time and number of BUBs obtained by different number of CCJ-DH iterations

					Gap (%)					Time (seconds)					Number of BUBs				
Prob	Set	m	Class	Size	50	100	150	200	300	50	100	150	200	300	50	100	150	200	300
IRP	SV-I1	1	-	160	3.93	3.43	3.08	2.71	2.43	12	25	39	54	94	22	24	25	25	26
	MV-I1	2	-	160	5.24	4.49	4.27	4.04	3.85	20	42	65	94	149	10	12	12	12	12
		3	-	160	4.94	4.03	3.8	3.62	3.25	23	46	71	93	155	13	17	17	18	21
		4	-	160	4.98	4.44	4.11	3.91	3.63	22	50	77	102	161	17	18	19	20	20
		5	-	158	5.5	4.69	4.24	3.94	3.67	21	42	66	91	151	17	22	22	24	27
	SV-I2	1	-	60	6.47	5.33	5.04	4.78	4.47	914	1,723	2,583	3,608	5,742	0	0	0	0	0
	MV-I2	2	-	60	0.36	-0.17	-0.48	-0.72	-1.04	899	1,673	2,590	3,303	5,419	31	34	35	35	37
		3	-	60	-1.45	-1.94	-2.31	-2.46	-2.74	829	1,560	2,503	3,271	5,021	38	39	40	41	43
		4	-	60	-3.04	-3.4	-3.58	-3.74	-3.98	1,152	2,152	3,632	4,684	7,879	46	46	46	46	49
		5	-	60	-3.22	-3.68	-3.96	-4.09	-4.16	1,102	1,886	3,115	4,236	6,644	49	50	52	52	52
PRP	SV-A1	1	1	120	0.61	0.49	0.47	0.4	0.39	5	9	14	16	25	6	9	10	11	12
		1	2	120	0.1	0.09	0.08	0.07	0.07	4	9	14	14	23	8	9	10	10	10
		1	3	120	2.88	2.57	2.2	2.05	1.97	4	8	13	13	23	3	3	4	4	5
		1	4	120	0.39	0.32	0.25	0.24	0.17	4	8	12	12	21	13	15	20	20	21
	MV-A2	UL	1	120	0.01	-0.02	-0.04	-0.06	-0.08	108	216	329	371	530	78	86	89	93	97
		UL	2	120	0.02	0.01	0	0	-0.01	90	180	273	301	436	43	58	67	70	72
		UL	3	120	0.16	-0.22	-0.39	-0.46	-0.56	85	167	253	279	406	71	90	98	100	100
		UL	4	120	-0.04	-0.06	-0.06	-0.07	-0.08	92	181	272	311	427	78	85	85	89	93
	MV-A3	UL	1	120	0.2	0.13	0.1	0.09	0.07	418	1,011	1,313	1,919	2,891	53	61	60	67	69
		UL	2	120	0.2	0.18	0.18	0.18	0.17	327	812	1,063	1,507	2,153	30	46	54	57	60
		UL	3	120	1.18	0.76	0.6	0.45	0.36	303	749	1,017	1,375	2,042	29	41	47	52	57
		UL	4	120	0.02	0.01	0	0	-0.01	313	808	1,057	1,531	2,142	55	64	72	73	76
	MV-B1	5	-	30	0.24	0.1	0.03	-0.07	-0.12	409	748	978	1,218	1,882	10	14	14	17	19
	MV-B2	9	-	30	0.69	0.56	0.48	0.49	0.4	1,950	4,087	5,441	6,413	9,623	1	1	4	3	6
	MV-B3	13	-	30	-0.08	-0.18	-0.21	-0.23	-0.26	4,325	9,459	13,693	15,625	23,357	16	17	20	21	21
Total				2628											737	861	922	960	1,005

As explained in Section 4.2, in addition to the marginal cost updating, two other mechanisms are proposed and tested to assess their effect on the quality of the results: TSP route cost share and VRP route cost share. Furthermore, we examined the effect of starting with other initial values for the node visit costs, σ_{it}. Table 13 presents the results for different cost update mechanisms and initial node visit costs on all IRP and PRP data sets. The results suggest that the marginal cost update mechanism generally performs better for the

PRP instances. The TSP route cost update procedure results in a (slightly) improved performance (better average gap) for the IRP instances. The VRP route cost update mechanism leads substantial bigger average gaps while it still is competitive on the large-scale multi-vehicle MV-I2 instances compared to the previous state-of-the-art heuristics. The algorithm shows a stable behavior on different initial values: $c_{0i}/2$, c_{0i} and $2c_{0i}$. Note that different node visit values may result in different production setup schedules for the PRP, however the algorithm is capable of finding quality solutions. Therefore, as can be seen in the table, we can expect that the overall best obtained solution for each instance (as reported in column OB) should be much better than any single one of them.

Table 13 Analysis of the different cost update mechanisms and initial node visit costs

					Gap (%)						Time (seconds)					Number of BUBs					
Prob	Set	m	Class	Size	.5†M‡	.5†T‡	.5†V‡	1†M‡	2†M‡	$OB^{\dagger\dagger}$	.5†M‡	.5†T‡	.5†V‡	1†M‡	2†M‡	.5†M‡	.5†T‡	.5†V‡	1†M‡	2†M‡	$OB^{\dagger\dagger}$
IRP	SV-I1	1	-	160	3.08	**2.03**	4.88	3	2.94	1.52	39	16	58	39	39	25	**27**	15	23	24	33
	MV-I1	2	-	160	4.27	**3.44**	6.43	4.37	4.41	2.59	65	23	72	69	64	12	**14**	7	11	11	22
		3	-	160	3.8	**3.09**	6.97	3.76	3.86	2.07	71	25	70	70	70	**17**	**17**	4	16	16	31
		4	-	160	4.11	**3.23**	7.15	4.25	4.25	2.37	77	25	76	73	73	19	**23**	10	18	17	28
		5	-	158	4.24	**3.29**	7.15	4.25	4.39	2.46	66	23	81	69	66	22	**27**	12	22	23	41
	SV-I2	1	-	60	5.04	**4.55**	4.72	5.47	5.12	3.74	2583	5114	4948	2877	2799	0	0	0	0	0	0
	MV-I2	2	-	60	-0.48	**-2.29**	-0.28	0.16	-0.41	-1.5	2590	3875	4322	2723	2688	35	**38**	37	37	36	39
		3	-	60	-2.31	-1.34	-0.95	-2.19	**-2.45**	-3.13	2503	3216	3865	2631	2464	40	37	36	42	**44**	47
		4	-	60	-3.58	-3.65	-2.37	-3.64	**-3.97**	-4.69	3632	3796	4387	3162	3014	46	**49**	41	45	47	53
		5	-	60	-3.96	**-4.42**	-2.58	-3.86	-4.05	-5.02	3115	3029	3807	3281	4036	52	**55**	46	54	52	58
PRP	SV-A1	1	1	120	**0.47**	0.53	1.04	0.5	0.49	0.29	14	20	12	14	14	**10**	2	1	5	7	11
		1	2	120	**0.08**	0.09	0.17	**0.08**	**0.08**	0.04	14	17	10	14	13	10	3	1	**11**	9	13
		1	3	120	**2.2**	2.53	3.07	2.21	2.29	1.24	13	21	9	13	13	4	2	2	**5**	4	6
		1	4	120	**0.25**	0.35	0.59	0.27	0.27	0.12	12	19	13	12	12	**20**	8	2	18	19	28
	MV-A2	UL	1	120	**-0.04**	-0.03	0.18	-0.03	-0.03	-0.18	329	274	258	336	331	89	**92**	65	91	87	108
		UL	2	120	**0**	0.03	0.08	**0**	**0**	-0.02	273	245	210	273	274	**67**	46	17	71	66	88
		UL	3	120	-0.39	**-0.75**	0.37	-0.5	-0.4	-0.99	253	243	196	261	257	98	**104**	68	100	93	118
		UL	4	120	**-0.06**	-0.04	0.05	**-0.06**	**-0.06**	-0.1	272	262	263	276	274	85	64	45	**86**	**86**	95
	MV-A3	UL	1	120	0.1	0.3	0.37	0.09	**0.07**	0.02	1313	1093	1073	1618	1569	60	48	42	60	64	75
		UL	2	120	**0.18**	0.2	0.22	**0.18**	**0.18**	0.16	1063	1016	951	1275	1241	54	32	18	57	52	72
		UL	3	120	0.6	0.59	0.95	**0.5**	0.57	-0.07	1017	925	817	1148	1143	47	50	36	43	44	68
		UL	4	120	0	0.03	0.06	**-0.01**	0	-0.01	1057	1140	1141	1249	1237	72	49	40	68	67	76
	MV-B1	5	-	30	**0.03**	0.89	1.13	0.11	0.23	-0.17	978	2167	2151	1048	1081	**14**	2	0	13	**14**	22
	MV-B2	9	-	30	**0.48**	1.43	1.67	0.5	0.64	0.33	5441	6391	6656	5850	5358	**4**	0	0	2	**4**	7
	MV-B3	13	-	30	**-0.21**	1.96	2.12	-0.15	-0.18	-0.27	13693	14397	14878	12288	11305	20	0	0	17	**22**	24
Total				2628												922	789	545	915	908	1163

† Initial values for node visit cost, σ_{it}. .5: $c_{0i}/2$, 1: c_{0i}, 2: $2c_{0i}$.
‡ Cost update mechanisms. M: Marginal, T: TSP route cost share, V: VRP route cost share.
†† Overall best obtained solution for each instance is taken into account.

Finally, we examined the effect of the $\mathcal{M}_z^{\mathcal{R}}$ model implementation for the multi-vehicle instances with $m < n$. Table 14 shows the results with and without implementing this model. Our observation is that the algorithm faces more infeasible VRP_t subproblems for these very hard instances. Therefore, the $\mathcal{M}_z^{\mathcal{R}}$ model implementation is crucial to obtain quality solutions for them.

Table 14 Average gap (%), CPU time and number of BUBs with and without implementing $\mathcal{M}_z^{\mathcal{R}}$ model

				Gap (%)		Time (seconds)		Number of BUBs	
Prob	Set	m	Size	With	Without	With	Without	With	Without
IRP	MV-I1	2	160	4.27	4.8	65	61	12	6
		3	160	3.8	4.66	71	61	17	6
		4	160	4.11	4.7	77	60	19	11
		5	158	4.24	4.72	66	58	22	12
	MV-I2	2	60	-0.48	-0.48	2,590	2,639	35	35
		3	60	-2.31	-2.16	2,503	2,394	40	40
		4	60	-3.58	-3.44	3,632	2,269	46	45
		5	60	-3.96	-3.82	3,115	2,250	52	52
PRP	MV-B1	5	30	0.03	1.55	978	701	14	0
	MV-B2	9	30	0.48	1.44	5,441	3,281	4	0
	MV-B3	13	30	-0.21	0.42	13,693	9,079	20	4
Total			968					281	211