TRANSPORTATION SCIENCE
Vol. 49, No. 4, November 2015, pp. 784–795
ISSN 0041-1655 (print) | ISSN 1526-5447 (online)

http://dx.doi.org/10.1287/trsc.2014.0523

A Two-Phase Iterative Heuristic Approach for the Production Routing Problem

N. Absi
Department of Manufacturing Sciences and Logistics, Ecole des Mines de Saint-Etienne, F-13541 Gardanne, France, absi@emse.fr

C. Archetti
Department of Economics and Management, University of Brescia, Brescia I-25122, Italy, archetti@eco.unibs.it

S. Dauzère-Pérès, D. Feillet
Department of Manufacturing Sciences and Logistics, Ecole des Mines de Saint-Etienne, F-13541 Gardanne, France {dauzere-peres@emse.fr, feillet@emse.fr}

This paper investigates the integrated optimization of production, distribution, and inventory decisions related to supplying multiple retailers from a central production facility. A single-item capacitated lot-sizing problem is defined for optimizing production decisions and inventory management. The optimization of daily distribution is modeled as a traveling salesman problem or a vehicle routing problem depending on the number of vehicles. A two-phase iterative method, from which several heuristics are derived, is proposed that iteratively focuses on lot-sizing and distribution decisions. Computational results show that our best heuristic outperforms existing methods.

Keywords: lot sizing; production; distribution; routing; heuristics
History: Received: January 2013; revisions received: August 2013, October 2013; accepted: October 2013. Published online in *Articles in Advance* July 11, 2014.

1. Introduction

Challenges related to integrating decisions usually taken independently have always drawn the attention of the operations research community. This is the case, for example, when customer demands and inventory levels are taken into account in vehicle routing, leading to the stream of research named inventory routing (see for instance the recent survey of Andersson et al. 2010). Going one step further, the integration of production planning and vehicle routing decisions was introduced in Chandra (1993). The problem of simultaneously optimizing production, distribution, and inventory decisions in a supply chain where retailers are supplied from a central production facility has been called the production routing problem (see Ruokokoski et al. 2010; Adulyasak, Cordeau, and Jans 2014a, b). The production routing problem combines lot-sizing decisions, inventory management, and routing. Given the complexity of the resulting problem, research has been mainly focused on heuristic algorithms (e.g., Chandra 1993, 1994; Boudia, Louly, and Prins 2007, 2009; Bard and Nananukul 2009; Armentano, Shiguemoto, and Løkketangen 2011, and Adulyasak, Cordeau, and Jans 2014b). For a detailed survey on heuristic algorithms, the reader is referred to Adulyasak, Cordeau, and Jans (2014b). Few exact algorithms have been proposed for the production routing problem. The first exact approach was proposed in Fumero and Vercellis (1999) and is based on a Lagrangian relaxation of the problem. A similar relaxation was used later in Solyali and Süral (2009) to solve the problem with the order-up-to level policy, i.e., each time a retailer is visited, the quantity delivered is such that the inventory level reaches the inventory capacity of the retailer. In Ruokokoski et al. (2010) and Bard and Nananukul (2010), the problem with the maximum level policy is studied, i.e., the quantity delivered to each retailer is such that the inventory capacity is not exceeded. In Ruokokoski et al. (2010), the authors propose a branch-and-cut algorithm, while in Bard and Nananukul (2010), a branch-and-price approach is developed. Finally, two branch-and-cut approaches are proposed in Archetti et al. (2011) and Adulyasak, Cordeau, and Jans (2014a). Note that the only exact approaches that solve the problem with multiple vehicles are the ones in Bard and Nananukul (2010) and Adulyasak, Cordeau, and Jans (2014a).

In this paper, we introduce a new heuristic algorithm for the production routing problem with a maximum level policy. We propose an iterative approach where production planning and routing subproblems are solved in sequence. This approach originates from

previous iterative approaches developed in Dauzère-Pérès and Lasserre (1994) for lot sizing and scheduling and in Boudia et al. (2006) for the multi-item production routing problem. In the latter, a classic production planning problem is solved in phase 1, and vehicle routes are determined in phase 2. On the basis of the quantities actually distributed in phase 2, demands are updated and the production planning problem is solved again to start the next iteration.

Here, we consider the single-item production routing problem. In the iterative approach, vehicle capacity constraints are considered in the first phase, called the lot-sizing phase. This phase determines which customers to serve on each specific day. The second phase, called the routing phase, then aims at optimizing routes. We take routing information into account in the first phase by introducing a visiting cost for customers. This cost is updated at each iteration according to the results of the second phase. The idea of first solving a lot-sizing problem with an approximate routing cost was also proposed in Bard and Nananukul (2009), but not in an iterative setting. To prevent a quick convergence to a local optimum, two diversification mechanisms are used.

Following the literature, cases with a single vehicle or multiple vehicles are investigated. In the multiple vehicle case, two variants of the approach are proposed, depending on whether the individual capacity of the vehicles are modeled or not in the lot-sizing phase. Numerical results on benchmark instances show that our simple heuristic outperforms previous heuristics, thus demonstrating that it is not only simple but also competitive.

The paper is organized as follows. The problem is formalized in §2. Our two-phase iterative approach is described in §3. Section 4 is devoted to computational experiments. Conclusions and perspectives are provided in §5.

2. The Production Routing Problem

In this section, we introduce the production routing problem addressed in this paper (hereafter PRP) and present a mixed-integer linear programming formulation. We consider a set $\mathcal{M} = \{1, \ldots, M\}$ of retailers and a single product sold by these retailers along a discrete time horizon $\mathcal{T} = \{1, \ldots, T\}$. The demand of a retailer $i \in \mathcal{M}$ at time period $t \in \mathcal{T}$ is denoted by d_{it}. Retailers are restocked from a common production facility. Products are then kept in retail stores, with an inventory limit U_i and at a unitary holding cost h_i, that depends on the retailer.

The production facility is identified with index 0. A capacity C is assumed on production, and a maximal inventory level U_0 is defined. The unitary inventory cost at the production facility is denoted by h_0. A fixed production setup cost f and a variable cost p, proportional to the number of produced items, are considered.

Table 1 Notation for the Production Routing Problem

$\mathcal{M}$:	Set of retailers
M:	Number of retailers
$\mathcal{T}$:	Set of periods
T:	Number of periods
$\mathcal{V}$:	Set of vehicles
V:	Number of vehicles
Q:	Vehicle capacity
U_i:	Inventory limit at retailer $i \in \mathcal{M}$
U_0:	Inventory limit at production facility
h_i:	Unitary inventory cost at retailer $i \in \mathcal{M}$
h_0:	Unitary inventory cost at production facility
C:	Production capacity per period
f:	Fixed production cost
p:	Variable production cost
d_{it}:	Demand of retailer $i \in \mathcal{M}$ at period $t \in \mathcal{T}$
A:	Pairs of location: $A = \mathcal{M} \cup \{0\} \times \mathcal{M} \cup \{0\}$
c_{ij}:	Travel cost between two locations $(i, j) \in A$

A fleet $\mathcal{V} = \{1, \ldots, V\}$ of homogeneous vehicles with capacity Q is available for the distribution. Travel costs c_{ij} are defined between every pair (i, j) of locations $A = \mathcal{M} \cup \{0\} \times \mathcal{M} \cup \{0\}$. No limit is imposed on the tour duration of vehicles. Split delivery is forbidden: no retailer can be replenished by more than one vehicle at the same period. The notation is summarized in Table 1.

The goal is to simultaneously minimize production, inventory, and routing costs so that the demands of retailers, inventory limits, and production capacity at the production facility and the inventory limit at the retailers are satisfied. The problem can be formulated as a mixed-integer linear program with the following decision variables:

- x_{vit}: Quantity shipped from the production facility to retailer i at period t using vehicle v.
- y_t: Quantity produced at the production facility at period t.
- γ_{vit}: Binary variable that is equal to 1 if $x_{vit} > 0$, and 0 otherwise.
- δ_t: Binary variable that is equal to 1 if $y_t > 0$, and 0 otherwise.
- I_{0t}: Inventory level at the production facility at the end of period t.
- I_{it}: Inventory level at retailer i at the end of period t.
- u_{vijt}: Binary variable that is equal to 1 if vehicle v travels from location i to location j at period t, and 0 otherwise.

The PRP can be formulated as follows:

$$\min \left\{ \sum_{t \in \mathcal{T}} (f\delta_t + py_t + h_0 I_{0t}) + \sum_{t \in \mathcal{T}} \sum_{i \in \mathcal{M}} h_i I_{it} + \sum_{t \in \mathcal{T}} \sum_{v \in \mathcal{V}} \sum_{(i,j) \in A} c_{ij} u_{vijt} \right\} \quad (1)$$

subject to

$$I_{it} = I_{i,t-1} + \sum_{v \in \mathcal{V}} x_{vit} - d_{it} \quad \forall i \in \mathcal{M}, \forall t \in \mathcal{T}, \quad (2)$$

$$I_{0t} = I_{0,t-1} + y_t - \sum_{v\in\mathcal{V}}\sum_{i\in\mathcal{M}} x_{vit}, \quad \forall t\in\mathcal{T}, \tag{3}$$

$$y_t \le C, \quad \forall t\in\mathcal{T}, \tag{4}$$

$$I_{it} \le U_i, \quad \forall i\in\mathcal{M}\cup\{0\}, \forall t\in\mathcal{T}, \tag{5}$$

$$\sum_{i\in\mathcal{M}} x_{vit} \le Q, \quad \forall v\in\mathcal{V}, \forall t\in\mathcal{T}, \tag{6}$$

$$x_{vit} \le \min\left(\sum_{t'=t}^{T} d_{it'}, U_i + d_{it}, Q\right)\gamma_{vit}, \quad \forall v\in\mathcal{V}, \forall i\in\mathcal{M}, \forall t\in\mathcal{T}, \tag{7}$$

$$y_t \le \left(\sum_{i\in\mathcal{M}}\sum_{t'=t}^{T} d_{it'}\right)\delta_t, \quad \forall t\in\mathcal{T}, \tag{8}$$

$$\sum_{v\in\mathcal{V}} \gamma_{vit} \le 1, \quad \forall i\in\mathcal{M}, \forall t\in\mathcal{T}, \tag{9}$$

$$\sum_{j\in\mathcal{M}\cup\{0\}\setminus\{i\}} u_{vijt} = \gamma_{vit}, \quad \forall v\in\mathcal{V}, \forall i\in\mathcal{M}, \forall t\in\mathcal{T}, \tag{10}$$

$$\sum_{j\in\mathcal{M}} u_{v0jt} \le 1, \quad \forall v\in\mathcal{V}, \forall t\in\mathcal{T}, \tag{11}$$

$$\sum_{j\in\mathcal{M}\cup\{0\}\setminus\{i\}} u_{vijt} - \sum_{j\in\mathcal{M}\cup\{0\}\setminus\{i\}} u_{vjit} = 0, \quad \forall v\in\mathcal{V}, \forall i\in\mathcal{M}\cup\{0\}, \forall t\in\mathcal{T}, \tag{12}$$

$$\sum_{i\in S}\sum_{j\in S} u_{vijt} \le |S| - 1, \quad \forall v\in\mathcal{V}, \forall t\in\mathcal{T}, \forall S\subseteq\mathcal{M}, \tag{13}$$

$$I_{0t}, y_t \ge 0, \quad \forall t\in\mathcal{T}, \tag{14}$$

$$I_{it}, x_{it} \ge 0, \quad \forall i\in\mathcal{M}, \forall t\in\mathcal{T}, \tag{15}$$

$$\delta_t \in \{0,1\}, \quad \forall t\in\mathcal{T}, \tag{16}$$

$$\gamma_{vit} \in \{0,1\}, \quad \forall v\in\mathcal{V}, \forall i\in\mathcal{M}, \forall t\in\mathcal{T}, \tag{17}$$

$$u_{vijt} \in \{0,1\}, \quad \forall v\in\mathcal{V}, \forall i,j\in\mathcal{M}\cup\{0\}, \forall t\in\mathcal{T}. \tag{18}$$

The objective function (1) minimizes the sum of production, inventory, and distribution costs. Constraints (2) and (3) are the inventory balance equations at retailers and at the production facility, respectively. Production capacity constraints are expressed using constraints (4). The upper bounds on inventory levels are guaranteed through constraints (5). Constraints (6) define the vehicle capacity. The binary variables indicating when production and distribution occur are linked to the production and distribution variables through constraints (7) and (8). In constraints (7), the shipped quantity for each vehicle, each retailer, and each period is limited by the remaining demand, maximal inventory, and vehicle capacity. In constraints (8), the quantity produced in each period is limited by the total remaining demand. Constraints (9)–(13) are routing constraints. Constraints (9) prevent split deliveries. Constraints (10) force a vehicle to visit a retailer if the corresponding visiting variable is set to 1. Constraints (11) limit each vehicle to perform at most one tour per period. Constraints (12) ensure the flow conservation, and constraints (13) eliminate subtours, which do not include the production facility. Constraints (14)–(18) are variable definitions.

Note that this formulation is analogous to the one proposed in Adulyasak, Cordeau, and Jans (2014b) and the one in Archetti et al. (2011), where there is no constraint on the maximum inventory level at the production plant and no production capacity constraint.

Finding a good, feasible solution to the PRP is challenging because it integrates two well-known and hard combinatorial problems—namely, lot sizing and vehicle routing. One way to deal with such a complex problem is to decompose it into smaller problems of reduced complexity for which optimal or near-optimal solutions can be determined. The key idea behind the methods proposed in this paper is thus to separate the problem into two subproblems that are solved iteratively.

3. A Two-Phase Iterative Approach

We propose a two-phase iterative method (IM) to solve the PRP. The first phase corresponds to solving a lot-sizing problem. In this phase, routing costs are integrated in an approximate way: A visiting cost SC_{vit} is counted each time a delivery is planned for a combination of vehicle, retailer, and period. The solution of the lot-sizing problem establishes when and how much to produce, when to visit retailers, and how much to deliver at each visit. Vehicle capacity constraints and maximum inventory constraints are considered in this phase. The routing decisions are taken in the second phase. Decision variables γ_{vit} are fixed from phase 1; thus one knows the set of retailers to be visited by each vehicle at each period. A traveling salesman problem (TSP) is solved for each vehicle v and each period t. The solution of this second phase is used to update the visiting costs for the next iteration of the first phase. The procedure is repeated until a given number of iterations is reached or the solution is not improved for a given number of iterations.

When the iterative procedure stops, a diversification mechanism is performed and the whole scheme is repeated until a stopping criterion is met. The general scheme of the IM can be found in Algorithm 1, where *sol* stores the best solution found so far. Details on the two phases and on the diversification mechanism are given in subsequent §§3.1–3.3, respectively.

Algorithm 1 (General scheme of the two-phase iterative approach.)

1: $sol \leftarrow \varnothing$
2: Initialize SC_{vit} for each $i\in\mathcal{M}$, $v\in\mathcal{V}$, and $t\in\mathcal{T}$
3: **repeat**

4: **repeat**
5: Solve the *lot-sizing problem* (SC_{vit}) and get γ_{vit} for each $i \in \mathcal{M}$, $v \in \mathcal{V}$, and $t \in \mathcal{T}$
6: Solve the *routing problem* (γ_{vit})
7: Update *sol* if necessary
8: Update SC_{vit}
9: **until** a stopping criterion is met
10: Diversify
11: **until** a stopping criterion is met

Visiting costs SC_{vit} play a central role in this approach because they create a connection between the first and second phases. The value of SC_{vit} is initially set to $c_{0i} + c_{i0}$. This forces the solution of the lot-sizing phase to serve less frequently the retailers that are far from the production plant and, thus, for which the corresponding transportation cost is high. However, the initial value of SC_{vit} does not take into account the clustering of retailers: There is no measure of the proximity of retailers visited at a certain period, so retailers that are far from each other may be clustered together and served at the same period. This of course negatively impacts the transportation cost. Thus, in subsequent iterations, the values of SC_{vit} are updated using the information provided by the solution of the routing phase so that solutions of the lot-sizing phase are driven to better solutions in terms of retailer clustering.

Allocating retailers to vehicles in phase 1 can be questionable, except for instances with a single vehicle. Indeed, one could prefer to simply select which retailers to visit at each period during phase 1 and to determine the clustering of retailers and routes during phase 2. This is the focus of the alternative approach described in §3.5.

3.1. The Lot-Sizing Phase

This section describes the mathematical model used in the first phase of IM. The cost of visiting retailer i at period t with vehicle v is represented by SC_{vit}. As already mentioned, the lot-sizing phase decides when and how much to produce, when to visit retailers, and how much to deliver. The objective is to minimize production and inventory costs as well as costs related to inserting retailers into vehicle routes. The lot-sizing model (LSM) for the first IM phase is

$$\text{(LSM)} \quad \min\Big\{ \sum_{t\in\mathcal{T}} (f\delta_t + py_t + h_0 I_{0t}) + \sum_{i\in\mathcal{M}} \sum_{t\in\mathcal{T}} h_i I_{it} + \sum_{v\in\mathcal{V}} \sum_{i\in\mathcal{M}} \sum_{t\in\mathcal{T}} SC_{vit}\gamma_{vit} \Big\}$$

subject to (2)–(9), (14)–(17).

Note that all decisions related to quantities and assignment to vehicles, particularly the quantities that have to be delivered to retailers at every period by a given vehicle, are taken in the model (LSM). These quantities satisfy the vehicle capacity constraints (6); thus, the second phase only needs to determine how to route the retailers served by each vehicle.

This LSM model is NP-hard, mainly because the classic single-item capacitated lot-sizing problem is NP-hard (Florian, Lenstra, and Rinnoy Kan 1980; Bitran and Yanasse 1982). If we consider the special case without vehicle capacity and production capacity constraints, without inventory constraints at the facility and retailers, and where inventory costs of the facility are set to a sufficiently large value to force the model to transport the entire quantity produced each time production is started, we obtain the classic joint-replenishment problem (Arkin, Joneja, and Roundy 1989), which is still NP-hard. The joint-replenishment problem considers the production of items that need major and minor setup costs. In our case, major setup costs correspond to production setup costs, and minor setup costs to the cost of visiting a retailer at a given period.

We solve the model (LSM) using a standard mathematical programming solver. Although it is NP-hard, good solutions can be obtained relatively efficiently.

Note that we also tested the facility location formulation for this problem. Generally, this formulation provides a strong lower bound for capacitated lot-sizing problems. Contrary to the previous formulation, the production variables are defined on the basis of the production period, the delivery period, and the consumption period. This formulation has more production variables ($O(T^3)$ versus $O(T)$) but generally provides better lower bounds. However, in our case, computational times increased significantly.

3.2. The Routing Phase

In this section, we describe the routing phase of IM. It consists of computing vehicle routes at each period. Phase 1 allocates retailers to vehicles and takes vehicle capacities into account, so the problem to solve for each vehicle and each period is a standard TSP.

At each iteration, many TSPs (up to $V \times T$) need to be solved. Furthermore, in view of the size of the benchmark instances, one can expect state-of-the-art heuristics to find excellent solutions quickly. For these reasons, we use the LKH implementation (Helsgaun 2012) of the Lin-Kernighan heuristic (Lin and Kernighan 1973) for this phase.

3.3. Update of Visiting Costs

For all $v \in \mathcal{V}$ and $t \in \mathcal{T}$, let S_{vt} be the set of retailers $i \in \mathcal{M}$ for which $\gamma_{vit} = 1$. Let r_{vt} be the route of vehicle v at period t, obtained from phase 2. Route r_{vt} is, hopefully, an optimal or near-optimal solution of the TSP defined on set $S_{vt} \cup \{0\}$.

For $v \in \mathcal{V}$, $t \in \mathcal{T}$, and $i \in S_{vt}$, let us denote i^- the predecessor of vertex i in route r_{vt} and i^+ its successor.

For $v \in \mathcal{V}$, $t \in \mathcal{T}$, and $i \notin S_{vt}$, let Δ_{vit} be the cheapest insertion cost for inserting i in route r_{vt}.

The procedure for updating visiting costs SC_{vit} is described in Algorithm 2.

Algorithm 2 (Update of visiting costs.)
1: **for all** $t \in \mathcal{T}$ and $v \in \mathcal{V}$ **do**
2: **for all** $i \in \mathcal{M}$ **do**
3: **if** $i \in S_{vt}$ **then**
4: $SC_{vit} \leftarrow c_{i^-i} + c_{ii^+} - c_{i^-i^+}$
5: **else**
6: $SC_{vit} \leftarrow \Delta_{vit}$
7: **end if**
8: **end for**
9: **end for**

Whether $i \in S_{vt}$ (line 4) or not (line 6), SC_{vit} approximates the difference in the total routing cost if the next execution of phase 1 only modifies the status of vertex i for vehicle v at period t. As noted earlier, updating the parameters SC_{vit} is crucial to improving the clustering of retailers for the subsequent iterations. The rule we adopt has two main advantages. First, it is simple. Second, it captures, for each retailer, the complementarity with the other retailers served in the same route (in this case, if SC_{vit} is large, the retailer will probably be removed from the route at the next iteration of the lot-sizing phase) and to the retailers served in different routes (in this case, if SC_{vit} is small, the retailer will probably be inserted in the route at the next iteration of the lot-sizing phase). Also, this updating rule for SC_{vit} helps in clustering retailers that are close, but it does not fix any grouping of retailers, and this helps diversification between successive solutions of the lot-sizing phase.

3.4. Diversification Mechanisms

We propose two mechanisms to diversify the search: a multistart procedure and a second procedure called an update diversification mechanism. Different variants of the IM algorithm are defined depending on which of these two mechanisms are applied and which parameter is used for the stopping criterion. These variants are detailed in §4. In all cases, the diversification mechanism reinitializes visiting costs SC_{vit} and restarts the iterative procedure.

The multistart procedure restarts the iterative process with randomly generated values for SC_{vit}. For each $v \in \mathcal{V}$, $i \in \mathcal{M}$, and $t \in \mathcal{T}$, a random number ρ_{vit} is drawn in range [0.5, 1.5] and the visiting cost SC_{vit} is set to $\rho_{vit} \times (c_{0i} + c_{i0})$.

The update diversification mechanism modifies costs SC_{vit} according to the best known solution. The goal is to help the heuristic moving to parts of the solution space that were not explored recently. Using the following rule, SC_{vit} is updated: For all retailers and according to the best known solution, multiply SC_{vit} by the number of retailers that are served at period t plus one. (Note that one is solely added to always avoid multiplying by zero.) This mechanism favors the ejection of retailers from periods where the number of visits is high to periods where the number of visits is low.

3.5. Variant with Aggregated Capacity Constraints

In cases with more than one vehicle ($V \geq 2$), we investigate a variant of IM where decisions on the allocation of retailers to vehicles are transferred to the routing phase. We call this variant IM-VRP. From now on, when $V \geq 2$, the iterative method described previously (IM) will be called IM-MultiTSP to better emphasize the difference between IM-MultiTSP and IM-VRP.

In IM-VRP, we replace visiting costs SC_{vit} with visiting costs SC_{it} that evaluate the cost induced by the visit of retailer i at period t. Apart from this change, and subsequent adaptations of the different steps of the algorithm, the approach is not modified. We now detail how the different steps are impacted.

As in IM-MultiTSP, the lot-sizing phase decides when and how much to produce, when to visit retailers, and how much to deliver. However, there is no assignment of retailers to vehicles. The lot-sizing model for the first phase of IM-VRP is

$$\min \left\{ \sum_{t\in\mathcal{T}} (f\delta_t + py_t + h_0 I_{0t}) + \sum_{i\in\mathcal{M}} \sum_{t\in\mathcal{T}} h_i I_{it} + \sum_{i\in\mathcal{M}} \sum_{t\in\mathcal{T}} SC_{it}\gamma_{it} \right\} \tag{19}$$

subject to

$$I_{it} = I_{i,t-1} + x_{it} - d_{it}, \quad \forall i \in \mathcal{M}, \forall t \in \mathcal{T}, \tag{20}$$

$$I_{0t} = I_{0,t-1} + y_t - \sum_{i\in\mathcal{M}} x_{it}, \quad \forall t \in \mathcal{T}, \tag{21}$$

$$y_t \leq C, \quad \forall t \in \mathcal{T}, \tag{22}$$

$$I_{it} \leq U_i, \quad \forall i \in \mathcal{M} \cup \{0\}, \forall t \in \mathcal{T}, \tag{23}$$

$$\sum_{i\in\mathcal{M}} x_{it} \leq \lambda_t VQ, \quad \forall t \in \mathcal{T}, \tag{24}$$

$$x_{it} \leq \min\left(\sum_{t'=t}^{T} d_{it'}, U_i + d_{it}, Q \right) \gamma_{it}, \quad \forall i \in \mathcal{M}, \forall t \in \mathcal{T}, \tag{25}$$

$$y_t \leq \left(\sum_{i\in\mathcal{M}} \sum_{t'=t}^{T} d_{it'} \right) \delta_t, \quad \forall t \in \mathcal{T}, \tag{26}$$

$$I_{0t}, y_t \geq 0, \quad \forall t \in \mathcal{T}, \tag{27}$$

$$I_{it}, x_{it} \geq 0, \quad \forall i \in \mathcal{M}, \forall t \in \mathcal{T}, \tag{28}$$

$$\delta_t \in \{0,1\}, \quad \forall t \in \mathcal{T}, \tag{29}$$

$$\gamma_{it} \in \{0,1\}, \quad \forall v \in \mathcal{V}, \forall i \in \mathcal{M}, \forall t \in \mathcal{T}. \tag{30}$$

Compared with the model (LSM), vehicle index v is removed from the x and γ variables. All decisions

related to quantities that are delivered to retailers at every period are still taken in this phase. Constraints (24) imply that these quantities satisfy the total capacity of all available vehicles at a given period. In these constraints, λ_t is a scaling parameter initially set to 1 and then modified according to rules that will be explained later. Again, lot-sizing problem (19)–(30) is NP-hard and is solved with a standard mathematical programming solver.

The IM-VRP routing phase consists of solving a vehicle routing problem (VRP) at each period. Indeed, quantities required by the retailers are known from phase 1. To limit computing time, the VRP is solved with a heuristic approach. We use the VRPH package (Groër, Golden, and Wasil 2010), with parameters set for using a combination of the tabu search and the record-to-record travel algorithm.

Constraints (24) only impose that the total demand is not more than the available capacity of the fleet, so it is possible that the algorithm does not find any feasible solution or even that no feasible solution exists. Indeed, because split delivery is forbidden, it might be possible that the different items to be transported cannot be packed in the vehicles. Parameter λ_t is introduced to deal with this difficulty. Initially, λ_t is set to 1 for each period $t \in \mathcal{T}$. When the VRPH heuristic does not find any feasible solution for a period t, λ_t is decreased by a parameter ϵ: $\lambda_t \leftarrow \lambda_t - \epsilon$. On the contrary, when VRPH finds a feasible solution and when this solution does not use all of the V vehicles, λ_t is increased, unless it is equal to 1: $\lambda_t \leftarrow \min(\lambda_t + \epsilon, 1)$. Finally, diversification resets all λ_t to 1.

Visiting costs are updated as described in Algorithm 3. In this algorithm, we denote S_t as the set of retailers served at period t ($t \in \mathcal{T}$). The remaining notation is the same as in §3.3.

Algorithm 3 (Update of visiting costs.)
1: **for all** $t \in \mathcal{T}$ **do**
2: **for all** $i \in \mathcal{M}$ **do**
3: **if** $i \in S_t$ **then**
4: $SC_{it} \leftarrow c_{i^-i} + c_{ii^+} - c_{i^-i^+}$
5: **else**
6: $SC_{it} = \min_{v \in \mathcal{V}} \Delta_{vit}$
7: **end if**
8: **end for**
9: **end for**

Finally, the update diversification mechanism is as follows: For all retailers and according to the best known solution, multiply SC_{it} by the number of retailers that are served at period t plus one.

4. Computational Experiments

Our algorithms are implemented in C++ using Microsoft Visual Studio 2008. We use the callable library IBM ILOG CPLEX 12.1 with the default settings to solve the lot-sizing problems. The tests are performed on an Intel Xeon CPU 2.67 GHz PC with 3.48 GB RAM using the Windows 7 operating system. CPU times are given in seconds.

4.1. Test Instances

We report computational tests on the 3×480 instances proposed in Archetti et al. (2011) and 3×30 instances proposed in Boudia, Louly, and Prins (2007). The instances in Archetti et al. (2011) are classified in three sets (A1, A2, and A3) and are characterized by six time periods and 14, 50, and 100 retailers, respectively, with constant demand, no production capacity, and no plant inventory capacity, but with initial inventory at the retailers. The set of instances A1 has a single capacitated vehicle, and sets A2 and A3 have an unlimited number of capacitated vehicles.

The instances in Boudia, Louly, and Prins (2007) are larger than the instances in Archetti et al. (2011). They are classified in three sets (B1, B2, and B3) and are characterized by 20 time periods and 50, 100, and 200 retailers, respectively, with a time-varying demand, production capacity, plant and retailers inventory capacity, initial inventory at the plant, and a limited number of vehicles.

Table 2 summarizes the characteristics of the instances, where "C" stands for constant, "V" for varying, and "∞" for unlimited.

Each set of 480 instances in Archetti et al. (2011) is divided into four classes with different parameter settings. The first class corresponds to the standard instances. The second class is characterized by high production unit costs and the third class by large transportation costs. The last class has no retailer inventory costs. Each class is composed of 24 groups of five instances. More details are given in Table 3.

4.2. Algorithms and Implementation

For the sets of instances (A1, A2, and A3), we compare our methods with the local search heuristic (H)

Table 2 Overview of the Benchmark Instances

Instance set	A1	A2	A3	B1	B2	B3
No. of instances	480	480	480	30	30	30
No. of periods	6	6	6	20	20	20
No. of retailers	14	50	100	50	100	200
No. of trucks	1	∞	∞	5	9	13
Demand	C	C	C	V	V	V
Production capacity	∞	∞	∞	C	C	C
Plant inventory capacity	∞	∞	∞	C	C	C
Retailer inventory capacity	C	C	C	C	C	C
Initial inventory at plant	0	0	0	V	V	V
Initial inventory at retailers	V	V	V	0	0	0
Vehicle capacity	C	C	C	C	C	C

Note. V—Varying, C—Constant, ∞—Unlimited. (Table from Adulyasak, Cordeau, and Jans 2014b).

Table 3 Descriptions of Instances Classes of Archetti et al. (2011)

Class type	Descriptions
Class I 1–24	Standard instances
Class II 25–48	High production unit cost, ×10
Class III 49–72	Large transportation costs, ×5
Class IV 73–96	No retailer inventory costs

Source. Table from Adulyasak, Cordeau, and Jans (2014b).

proposed in Archetti et al. (2011) and the adaptive large neighborhood search (ALNS) proposed in Adulyasak, Cordeau, and Jans (2014b) with the tunings that provide the best results (1,000 iterations). We also compare our heuristics with the optimal solutions obtained with the branch-and-cut (BC) method proposed in Archetti et al. (2011). The BC method is only designed for the set of instances A1 and provides optimal solutions for all instances. The results of Archetti et al. (2011) for algorithm H are obtained on a 2.40 GHz CPU PC, while the ones of ALNS in Adulyasak, Cordeau, and Jans (2014b) are obtained on a 2.10 GHz CPU PC.

For the sets of instances (B1, B2, and B3), we compare our methods with four metaheuristics: a memetic algorithm (MA) proposed in Boudia, Louly, and Prins (2009), a reactive tabu search (RTS) proposed in Bard and Nananukul (2009), a tabu search with path relinking (TSPR) proposed in Armentano, Shiguemoto, and Løkketangen (2011), and an adaptive large neighborhood search (ALNS) proposed in Adulyasak, Cordeau, and Jans (2014b) with the tunings that provide the best results (1,000 iterations for B1 and B2 and 500 iterations for B3). The results of ALNS, MA, RTS, and TSPR are obtained on a 2.10 GHz CPU PC, a 2.30 GHz CPU PC, a 2.53 GHz CPU PC, and a 2.80 GHz CPU PC, respectively.

For both series of instances, the variety of hardware complicates computing time comparisons. However, we can observe that all of the computers used in these experiments are of approximately equivalent efficiency.

4.2.1. Single-Vehicle Instances (A1). Because the first set of instances (A1) has only one vehicle, the IM algorithm is only considered. We tested different variants of IM. In all cases, in addition to the stopping criteria cited below, the method stops after a total of 100 two-phase iterations. These variants are

- IM: Iterative method without diversification (the method stops after 10 iterations without improvement);
- IM-DIV: IM completed with the update diversification mechanism (invoked when IM stops);
- IM-MS: IM in a multistart scheme, with 100 restarts (each with up to 100 two-phase iterations); and
- IM-DIV-MS: IM-DIV in a multistart scheme, with 20 restarts (each with up to 100 two-phase iterations).

4.2.2. Multiple Vehicle Instances (A2 and A3). For the multiple vehicle instances (A2 and A3), we compared IM-MultiTSP and IM-VRP to H and ALNS. Because no limit is assumed on the number of vehicles, a maximum of M vehicles is declared in the LSM for method IM-MultiTSP. For method IM-VRP, constraint (24) is omitted. The update diversification mechanism is called after five iterations of the IM without improvements. Both heuristics are stopped after a total of 20 iterations. The first phases of our algorithms are solved with a time limit of 10 seconds for IM-MultiTSP and 5 seconds for IM-VRP.

4.2.3. Multiple Vehicle Instances with Production Capacity Constraint (B1, B2, and B3). For the instances with production capacity constraints (B1, B2, and B3), we compared IM-MultiTSP and IM-VRP to GRAPS, MA, RTS, TSPR, and ALNS. The update diversification mechanism is called after five iterations of the IM without improvements. Iterations where phase 2 does not find a feasible solution are not counted. Both heuristics are stopped after a total of 100 iterations. The first phases of our algorithms are solved with a time limit of 50, 100, and 200 seconds for B1, B2, and B3, respectively, when using IM-MultiTSP and 100 seconds for the three sets when using IM-VRP. Because of the cost structure, the algorithm is initialized with visiting costs SC_{vit} or SC_{it} equal to 0, in order to guide the search towards well-optimized production plans.

4.3. Computational Results on Single-Vehicle Instances (A1)

On set A1, the BC algorithm proposed in Archetti et al. (2011) provides optimal solutions for 467 out of the 480 instances. Since then, the authors were able to solve the remaining 13 instances. They sent us the new optimal values. Table 4 summarizes the gaps with respect to the optimal solution for the 480 instances. Results are presented per class of instances and for each heuristic.

Table 5 provides average CPU times for the different methods. Table 6 summarizes the number of optimal solutions found by each heuristic and for each class of instances.

Tables 4 and 5 show that our iterative approach is competitive with respect to previous heuristic

Table 4 Average Gaps with Respect to the Optimal Solutions for Instances of Set A1

Classes	IM (%)	IM-DIV (%)	IM-MS (%)	IM-DIV-MS (%)	H (%)	ALNS (%)
Class I	1.17	0.90	0.09	0.13	2.33	1.70
Class II	0.17	0.12	0.01	0.02	0.31	0.36
Class III	5.63	3.87	0.57	0.72	3.71	8.43
Class IV	0.43	0.27	0.02	0.04	1.02	0.93
All	1.85	1.29	0.17	0.23	1.87	2.85

Table 5 Average CPU Times in Seconds for Instances of Set A1

Classes	IM	IM-DIV	IM-MS	IM-DIV-MS	H	ALNS
Class I	1.9	12.4	251.0	242.8	—	9.2
Class II	1.7	10.6	214.2	210.3	—	8.9
Class III	1.8	11.3	237.2	233.2	—	7.6
Class IV	1.7	10.7	216.9	217.8	—	8.7
All	1.8	11.3	229.8	226.0	—	8.6

approaches proposed in the literature. IM has the smallest CPU time of our different variants (less than 2 seconds). It can be compared with H, which has a negligible CPU time. Note that IM provides better average gaps than ALNS and equivalent average gaps to H over all classes of instances. If the update diversification mechanism is allowed, the average CPU time is slightly larger than the one of ALNS. However, IM-DIV provides better solutions than ALNS and H.

To almost close the gap with optimal solutions, more computing time is needed, especially for instances of class III. The parameters of IM-MS and IM-DIV-MS were set to attain computing times comparable to BC on average ($\approx$ 200 seconds, see Archetti et al. 2011). The average gaps provided by IM-MS and IM-DIV-MS are almost zero ($\approx$ 0.2%), with similar results for both methods. The BC method performs slightly better on average and has the advantage of proving optimality, but it has two important inconveniences: its highly varying and unpredictable computing times (with some instances for which no solution is found within two hours) and inability to tackle larger instances.

Consistent with previous methods, the iterative approach is less effective for instances of class III (instances with large transportation costs). IM, IM-DIV, H, and ALNS all show average gaps greater than 3% on this class. Gaps for IM-MS and IM-DIV-MS are still larger than for other classes, but more reasonable (inferior to 1%).

Finally, one can observe that a relatively small number of optimal solutions are found, even with large computing times. It shows that, while the iterative method is generally able to find near-optimal solutions, it has difficulties reaching the optimal solution.

Table 6 Number of Optimal Solutions for Instances of Set A1

Classes	IM	IM-DIV	IM-MS	IM-DIV-MS	H	ALNS
Class I	6	10	81	69	0	1
Class II	4	15	81	68	0	0
Class III	4	9	54	46	0	0
Class IV	23	32	91	78	0	1
Total	37	66	307	261	0	2

Table 7 Average Gaps with Respect to the Best Known Solutions for Instances of Set A2

Classes	IM-VRP (%)	IM-MultiTSP (%)	H (%)	ALNS (%)
Class I	0.04	0.93	1.89	0.98
Class II	0.02	0.07	0.35	0.14
Class III	0.26	1.92	2.66	2.66
Class IV	0.04	0.40	1.17	0.13
All	0.09	0.83	1.52	0.98

Table 8 Average CPU Times in Seconds for Instances of Set A2

Classes	IM-VRP	IM-MultiTSP	H	ALNS
Class I	25.6	338.5	11.3	50.2
Class II	21.7	235.7	12.4	49.5
Class III	22.6	317.9	9.5	42.7
Class IV	27.7	375.8	10.9	44.1
All	24.4	317.0	11.0	46.6

4.4. Computational Results on Multiple Vehicle Instances (A2 and A3)

In this section, we summarize all experiments on the sets of instances A2 and A3. We first compare IM-VRP, IM-MultiTSP, H, and ALNS. Tables 7 and 10 show average gaps with respect to the best known solution for each heuristic and for each class of instances of A2 and A3, respectively. Tables 8 and 11 show average CPU times for each heuristic and for each class of instances of A2 and A3, respectively. Finally, Tables 9 and 12 report the number of best solutions found by each heuristic for each class of instances of A2 and A3, respectively. Detailed results are provided in the online appendix (available as supplemental material at http://dx.doi.org/10.1287/trsc.2014.0523).

Considering Tables 7–9, we can notice that IM-VRP outperforms the other three heuristics. Average gaps to best known solutions are always small, and 353 out of 480 best known solutions are found, compared with 74 for IM-MultiTSP, 46 for ALNS, and 9 for H.

IM-MultiTSP provides gaps of the same quality as ALNS, but at the expense of long CPU times. IM-VRP is much quicker and exhibits computing times comparable to those of ALNS (actually divided by two but with a slightly better processor).

The bad and good computing times of methods IM-MultiTSP and IM-VRP, respectively, are two opposite consequences of the unlimited size of the vehicle fleet. In IM-MultiTSP, it implies one capacity constraint for

Table 9 Number of Best Solutions Found for Instances of Set A2

Classes	IM-VRP	IM-MultiTSP	H	ALNS
Class I	107	3	1	9
Class II	71	35	6	10
Class III	85	21	2	12
Class IV	90	15	0	15
Total	353	74	9	46

Table 10 Average Gaps with Respect to the Best Known Solutions for Instances of Set A3

Classes	IM-VRP (%)	IM-MultiTSP (%)	H (%)	ALNS (%)
Class I	0.06	1.66	2.06	0.82
Class II	0.19	0.27	0.32	0.29
Class III	0.23	2.70	2.55	2.53
Class IV	0.18	0.63	1.19	0.26
All	0.16	1.31	1.53	0.98

Table 11 Average CPU Times in Seconds for Instances of Set A3

Classes	IM-VRP	IM-MultiTSP	H	ALNS
Class I	85.5	514.2	188.0	228.5
Class II	76.1	497.4	216.7	217.6
Class III	75.1	509.3	167.8	197.8
Class IV	86.6	507.0	181.1	206.0
All	80.8	507.0	188.4	212.5

each potential vehicle, that is one per retailer (see §4.2), which drastically complicates the resolution. In IM-VRP, no vehicle capacity constraint is needed and the situation where phase 2 does not find a feasible solution cannot occur.

Again, IM-VRP has more difficulties with instances of class III, but these difficulties are far less pronounced than for other heuristics. However, it is difficult to draw clear conclusions from these statistics because gaps/number of best solutions are not against optimal solutions but against the best solutions found by the four heuristics.

Results for instances of set A3 (Tables 10–12) are similar to those observed on set A2. The tables illustrate again how IM-VRP outperforms the three other heuristics: 366 best known solutions are found with IM-VRP, against 28 for IM-MultiTSP, 68 for H, and 18 for ALNS. Thus, computing times are largely in favor of IM-VRP.

On these instances, however, ALNS now clearly outperforms IM-MultiTSP: with smaller computing times and smaller gaps obtained. IM-MultiTSP only dominates ALNS on class II (but with longer computing times) where, strangely, ALNS misses all of the best solutions.

Table 12 Number of Best Solutions Found for Instances of Set A3

Classes	IM-VRP	IM-MultiTSP	H	ALNS
Class I	105	0	11	4
Class II	75	21	24	0
Class III	100	3	16	1
Class IV	86	4	17	13
Total	366	28	68	18

Table 13 Average Gaps with Respect to the Best Known Solutions for the Sets of Instances B1, B2, and B3

Set of instances	IM-VRP (%)	IM-MultiTSP (%)	ALNS (%)	MA (%)	RTS (%)	TSPR (%)
B1	0.84	0.60	0.23	13.63	6.83	4.53
B2	0.24	0.71	0.35	12.58	12.22	8.06
B3	0.14	1.54	1.39	15.84	19.69	10.05
All	0.41	0.95	0.66	14.02	12.91	7.55

Table 14 Average CPU Times in Seconds for the Sets of Instances B1, B2, and B3

Set of instances	IM-VRP	IM-MultiTSP	ALNS	MA	RTS	TSPR
B1	550	1,653	481	173	331	317
B2	2,054	9,483	1,570	1,108	976	1,148
B3	4,179	19,270	5,794	4,098	2,492	3,926

4.5. Computational Results on Multivehicle Instances with Production Capacity Constraints (B1, B2, and B3)

In this section, we summarize all experiments on the sets of instances B1, B2, and B3. We compare IM-VRP, IM-MultiTSP, ALNS, MA, RTS, and TSPR. Table 13 shows average gaps with respect to the best known solution for each heuristic and for each set of instances. Table 14 shows average CPU times for each heuristic and for each set of instances. Finally, Table 15 reports the number of best solutions found by each heuristic for each set of instances. Detailed results are provided in the online appendix.

Again these tables show that IM-VRP outperforms the four other state-of-the-art heuristics (ALNS, MA, RTS, and TSPR). With regard to the quality of the solutions (Table 13), only ALNS is competitive. IM-VRP slightly outperforms it, with slightly better computing times.

IM-MultiTSP is slow. Besides multiple vehicle capacity constraints, production capacity constraints complicate the solution of the lot-sizing model. Surprisingly, IM-MultiTSP finds better solutions than IM-VRP on set B1. When the size of the instances increases (100 retailers in B2, 200 in B3), the effectiveness of IM-MultiTSP tumbles. In this situation indeed, the time required by the lot-sizing model to obtain good solutions dramatically increases.

Table 15 Number of Best Solutions Found for the Sets of Instances B1, B2, and B3

Set of instances	IM-VRP	IM-MultiTSP	ALNS	MA	RTS	TSPR
B1	2	11	16	0	0	1
B2	17	6	7	0	0	0
B3	25	4	1	0	0	0
All	44	21	24	0	0	1

In view of the results obtained on set B1, one can guess that IM-VRP is still relatively far from optimal solutions. Probably, the mechanism introduced to manage aggregated vehicle capacities is too simplistic to address the issue of packing demands in vehicle routes, which is an NP-hard problem.

4.6. Further Discussions on the Algorithm Behavior and Convergence

In this section, we analyze further the behavior of our algorithms. For set A1, Table 5 shows that the iterative method stops quickly (around 15 iterations) for small instances if diversification is not used. Table 4 instead exhibits the importance of diversification to find high-quality solutions. We investigate these issues for other sets of instances.

Figures 1 and 2 show the evolution of the solution value over time for IM-MultiTSP and IM-VRP, respectively. The analysis is done on four instances with 100 retailers (A3-13-3, A3-17-4, B2-12, and B2-18). These instances were selected because they summarized the general behavior of our algorithms. To provide a broader view on the behavior of the algorithms, the stopping criterion was increased to 100 iterations. In the figures, solution values are given through gaps to the best solution found by the algorithm during the solution process. Each dot in the figures corresponds to a (feasible) solution. Unfeasible solutions (instances B2-12 and B2-18 with method IM-VRP) are not reported. Scales (times and gaps) are different for every figure.

Figures 1 and 2 show similar behaviors. In all cases, the following pattern is repeated; from a bad starting solution, the methods quickly converge to good solutions, then slightly oscillate around these solutions for some iterations (sometimes leading to small improvements), before jumping (with diversification) to a new bad-quality solution. Note that the gaps are small in Figure 2(d). It explains the different appearance of this picture: on this example indeed, all solutions obtained after diversification were unfeasible and rejected; when feasibility was recovered, after some iterations, visiting costs had already converged so as to provide good quality solutions.

One can also notice from these figures that best solutions can be found at any iteration. IM-MultiTSP is however slower to converge and has more difficulties in finding good solutions quickly. Actually, Figure 1 shows that the convergence of the first iterations can be rather slow. For subsequent iterations, convergence is much quicker. This can be explained by the fact that the evaluation of numerous visiting costs SC_{vit} has to be improved, which may take several iterations. When these evaluations are obtained, they are not completely destroyed by diversification and obtaining good SC_{vit} necessitates fewer iterations.

For instance sets B1–B3 and for method IM-VRP, the situation where no feasible solution is found during phase 2 is common. On instance B2-12, feasible solutions are only obtained for 34 iterations (out of 100); on instance B2-18, it happened 41 times. Analyses on other instances show similar behaviors.

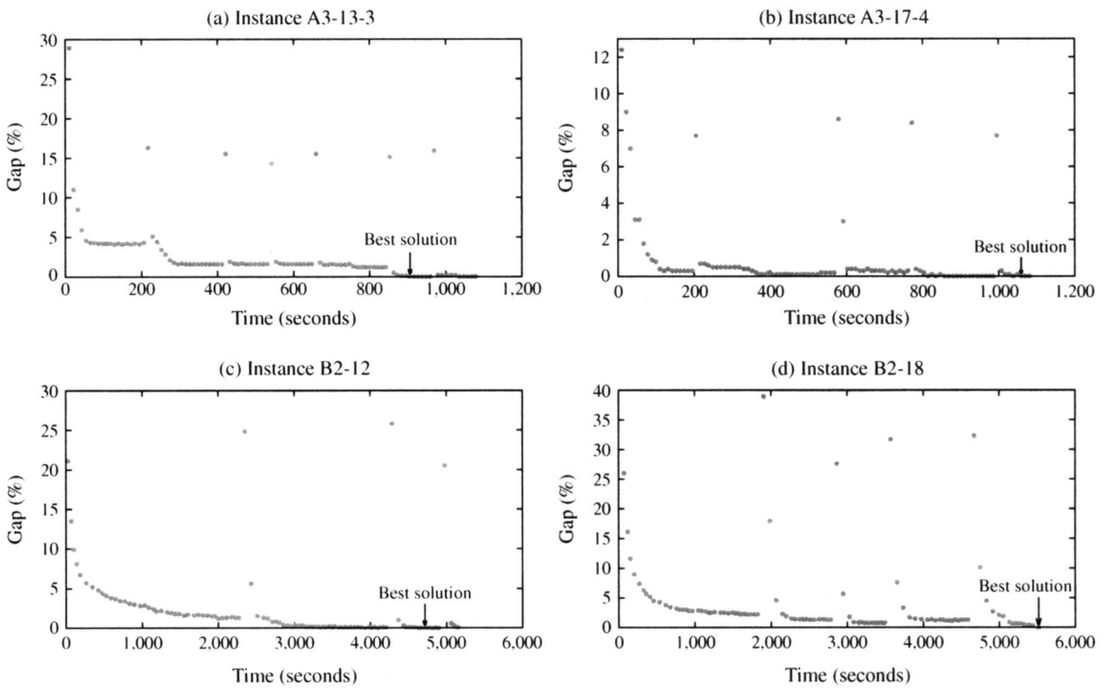

Figure 1 **Behavior of IM-MultiTSP**

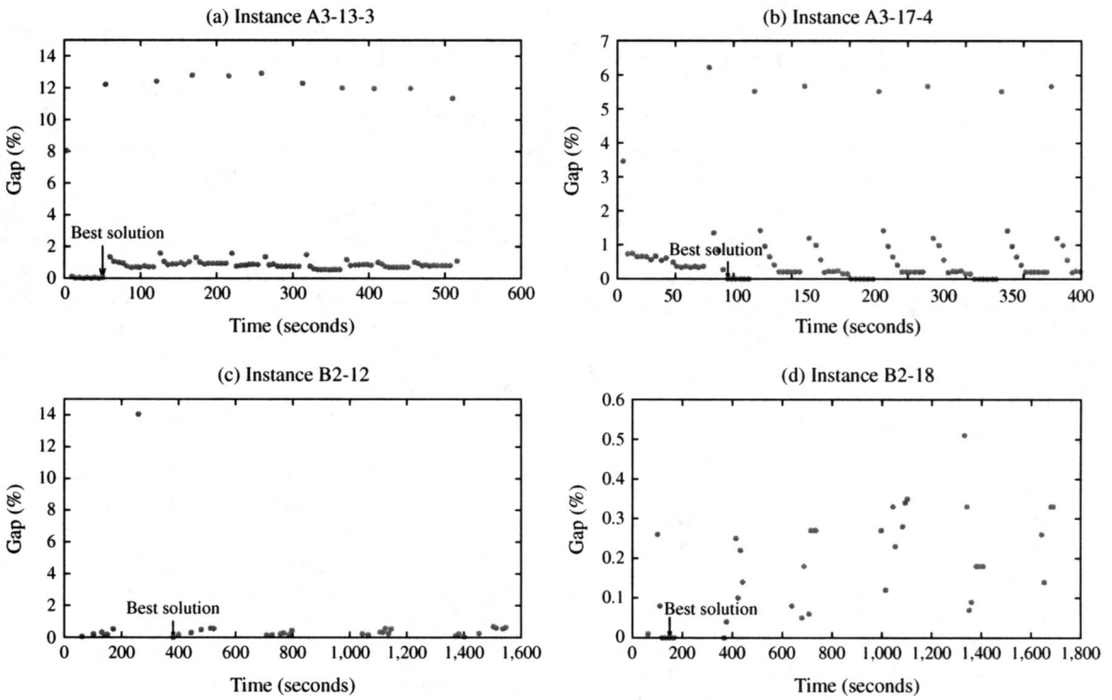

Figure 2 **Behavior of IM-VRP**

5. Conclusions and Perspectives

In this paper, we propose a two-phase iterative scheme to solve the PRP. In the first phase, a lot-sizing problem is solved with approximate routing costs. The first phase decides when and how much to produce at each period, when to visit retailers, and how much to deliver at each visit. The second phase takes routing decisions into account. We developed two variants of the approach. In the first variant (IM-MultiTSP), retailers are assigned to vehicles in the first phase, and TSPs are solved for each vehicle and period in the second phase. The second variant (IM-VRP) does not explicitly consider vehicles in the first phase; the assignment of retailers to vehicles is carried out in the second phase by solving a VRP.

Experimental results show that, despite its simplicity, IM-VRP outperforms all existing methods in the literature. In addition, its computing times are lower than those of existing methods. IM-MultiTSP often provides comparable results with respect to existing methods but is time consuming because of the additional difficulty induced by vehicle capacity constraints in the lot-sizing model.

One of the main perspectives is to develop a fast and effective heuristic for the lot-sizing phase. Then, many more iterations would be possible and more elaborated diversification schemes could be developed. Probably, additional intensification tools (e.g., local search) would also be necessary to close the gap to optimal solutions.

Even if the studied problem has constant costs and capacity constraints, our iterative method can be easily adapted when considering time-varying costs and capacity constraints. This can be done by slightly modifying the lot-sizing model of the first phase.

The multi-item version of the production routing problem is also an interesting perspective because several additional aspects should be considered (multiple production and inventory capacity constraints, multiple transportation modes, etc.).

Supplemental Material

Supplemental material to this paper is available at http://dx.doi.org/10.1287/trsc.2014.0523.

Acknowledgments

The authors wish to thank the two anonymous referees who helped them improve a first version of the paper. The authors also wish to thank Oguz Solyali from Middle East Technical University (Northern Cyprus Campus) for his remarks that permitted us to correct the tables in this paper.

References

Adulyasak Y, Cordeau J-F, Jans R (2014a) Formulations and branch-and-cut algorithms for multivehicle production and inventory routing problems. *INFORMS J. Comput.* 26:103–120.

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

Andersson H, Hoff A, Christiansen M, Hasle G, Løkketangen A (2010) Industrial aspects and literature survey: Combined inventory management and routing. *Comput. Oper. Res.* 37:1515–1536.

Archetti C, Bertazzi L, Paletta G, Speranza MG (2011) Analysis of the maximum level policy in a production-distribution system. *Comput. Oper. Res.* 38:1731–1746.

Arkin E, Joneja D, Roundy R (1989) Computational complexity of uncapacitated multi-echelon production planning problems. *Oper. Res. Lett.* 8:61–66.

Armentano VA, Shiguemoto AL, Løkketangen A (2011) Tabu search with path relinking for an integrated production-distribution system. *Comput. Oper. Res.* 38:1199–1209.

Bard JF, Nananukul N (2009) Heuristics for a multiperiod inventory routing problem with production decision. *Comput. Indust. Engrg.* 57:713–723.

Bard JF, Nananukul N (2009) The integrated production-inventory-distribution-routing problem. *J. Scheduling* 12:257–280.

Bard JF, Nananukul N (2010) A branch-and-price algorithm for an integrated production and inventory routing problem. *Comput. Oper. Res.* 37:2202–2217.

Bitran G, Yanasse HH (1982) Computational complexity of the capacitated lot size problem. *Management Sci.* 28:1174–1186.

Boudia M, Louly MAO, Prins C (2007) A reactive GRASP and path relinking for a combined production distribution problem. *Comput. Oper. Res.* 34:3402–3419.

Boudia M, Louly MAO, Prins C (2009) A memetic algorithm with dynamic population management for an integrated production-distribution problem. *Eur. J. Oper. Res.* 195:703–715.

Boudia M, Dauzère-Pérès S, Prins C, Louly MAO (2006) Integrated optimization of production and distribution for several products. *2006 Internat. Conf. Service Systems Service Management* (IEEE, Troyes, France), 272–277.

Chandra P (1993) A dynamic distribution model with warehouse and customer replenishment requirements. *J. Oper. Res. Soc.* 44:681–692.

Chandra P (1994) Coordination of production and distribution planning. *Eur. J. Oper. Res.* 72:503–517.

Dauzère-Pérès S, Lasserre J-B (1994) Integration of lotsizing and scheduling decisions in a job-shop. *Eur. J. Oper. Res.* 75:413–426.

Florian M, Lenstra JK, Rinnoy Kan AHG (1980) Deterministic production planning: Algorithms and complexity. *Management Sci.* 26:669–679.

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

Groër C, Golden B, Wasil E (2010) A library of local search heuristics for the vehicle routing problem. *Math. Programming Computation* 2:79–101.

Helsgaun K (2012) LKH. http://www.akira.ruc.dk/~keld/research/LKH/.

Lin S, Kernighan BW (1973) An effective heuristic algorithm for the traveling-salesman problem. *Oper. Res.* 21:498–516.

Ruokokoski M, Solyali O, Cordeau J-F, Jans R, Süral H (2010) Efficient formulations and a branch-and-cut algorithm for a production-routing problem. *Les Cahiers du GERAD*, G-2010-66, Montreal.

Solyali O, Süral H (2009) A relaxation based solution approach for the inventory control and vehicle routing problem in vendor managed systems. Neogy SK, Das AK, Bapat RB, eds. *Modeling, Computation and Optimization* (World Scientific, London), 171–189.

CORRECTION

In this article, "A Two-Phase Iterative Heuristic Approach for the Production Routing Problem" by N. Absi, C. Archetti, S. Dauzère-Pérès, and D. Feillet (first published in Articles in Advance, July 11, 2014, *Transportation Science*, DOI:10.1287/trsc.2014.0523), Tables 4, 6, 13, 14, and 15 have been updated.