An efficient two-phase iterative heuristic for Collection-Disassembly problem

M. K. Khakim Habibi [a,*], Olga Battaïa [b], Van-Dat Cung [c,d], Alexandre Dolgui [e]

[a] École Nationale Supérieure des Mines de Saint-Étienne, Lab. LIMOS, CNRS UMR6158, 42023 Saint-Étienne Cedex 2, France
[b] ISAE-SUPAÉRO, 10 Avenue Edouard Belin, 31400 Toulouse, France
[c] Univ. Grenoble Alpes, G-SCOP, F-38000 Grenoble, France
[d] CNRS, G-SCOP, F-38000 Grenoble, France
[e] IMT Atlantique, LS2N, CNRS, La Chantrerie, 4, rue Alfred Kastler - B.P. 20722, F-44307 Nantes Cedex 3, France

A B S T R A C T

Closing the loop in the supply chains is one of the mandatory conditions for more sustainable development. The Collection-Disassembly Problem appears in the reverse part of the closed-loop supply chains. Its aim is to coordinate the activities of collection of end-of-life products from collection centres and their subsequent disassembly. The disassembly step is required for efficient remanufacturing and recycling of returned products. The Collection-Disassembly problem integrates such optimization problems as dynamic lot-sizing and vehicle routing in general cases. In this paper, we develop a Two-Phase Iterative Heuristic to efficiently address large size instances. The numerical tests show that the heuristic provides good solutions under acceptable computational time.

Keywords:
Production-Distribution Problem
Collection
Disassembly
Reverse supply chain
Waste electrical and electronics equipment
Two-phase iterative heuristics

1. Introduction

The implementation of closed-loop supply chains for electrical and electronic equipment (EEE) may have a sustainable impact in several ways. From the economic point of view, the returned products can provide cheaper components and materials resulting also in savings in energy, production and transportation costs. From the environmental point of view, the recovery of materials and components reduces the need for new (virgin) resources and avoids landfill. Recently, the 21st session of the Conference of the Parties (COP 21) 2015 held in Paris was one of the most important milestones to tackle climate change and environmental issues including waste electrical and electronic equipment (WEEE) minimisation. From the social point of view, it was also shown that the creation of new activities in reverse supply chains can be a source for new jobs in logistics and recovery.

A successful collection of EEE from the collection centres and their subsequent disassembly process are both important for the sustainable operations in closed-loop supply chains. By definition, the disassembly process aims to extract the components and subassemblies from End-of-Life (EOL) products (McGovern & Gupta, 2011) in order to satisfy customers' demands for remanufacturing and recycling processes. Referring to forward supply chains context, both collection and disassembly processes are counterparts of distribution and manufacturing processes, respectively. The coordinated management of these activities through Production-Distribution Problem (PDP) was proved to bring economic advantage (Chandra & Fisher, 1994). Similarly, the Collection-Disassembly Problem coordinating the collection and disassembly processes will lead to such an economic advantage as proved in our previous work (Habibi et al., 2017). This problem is especially relevant for Third-Party Reverse Logistics Provider (3PRLP) which is responsible to manage WEEE from its collection point until remanufacturing process or disposal. As an integration of dynamic lot-sizing and vehicle routing problem in general case, the problem studied in this paper is $\mathcal{NP}$-hard. To deal with large size instances, we develop an efficient heuristic method.

The paper is organized as follows. The state-of-the-art is provided in Section 2. The optimisation problems are formalised in Section 3. Section 4 contains the description of the solving method developed. The obtained results are provided and analysed in Section 5. Section 6 gives concluding remarks.

* Corresponding author.
E-mail address: muhammad.habibi@emse.fr (M. K. K. Habibi).

http://dx.doi.org/10.1016/j.cie.2017.06.031

2. Literature review

To the best of our knowledge, the Collection-Disassembly Problem was initially introduced and formulated in Habibi et al. (2017). Fig. 1 depicts the decision levels about the collection of end-of-life products and their disassembly leading to the fulfilment of customer demands in different types of components and subassemblies. In recent years, a lot of scientific attention has been attracted to the disassembly process. Existing researches mainly focus on single level decision such as lot-sizing (Barba-Gutiérrez, Adenso-Díaz, & Gupta, 2008), line balancing (Bentaha, Battaïa, & Dolgui, 2014a, 2014b), sequencing problem (Yeh, 2012), inventory control (Godichaud et al.), RFID application (Ferrer et al., 2011). However, the integration with the collection problem was rarely considered despite the fact that it was shown to improve the overall benefit. The study of Habibi et al. (2017) evaluated the performances of available commercial solvers to tackle the problem instances of different level of difficulty. The authors concluded about the necessity to develop an efficient approximate method to deal with large size instances.

Since there is no such method available for the Collection-Disassembly Problem, we conduct our literature analysis on the similar works in PDP for forward supply chain context. Essentially, PDP is a problem focusing on both production and distribution aspects as depicted by Fig. 2. It integrates the production decision (dynamic lot-sizing problem) and vehicle routing throughout a planning horizon in operational level decision. It is widely concerned by those who works on Vendor Managed Inventory and Distribution (VMI/D) such as Kellogg Company and Frito-Lay's North America (Brown, Keegan, Vigus, & Wood, 2001; Çetinkaya, Üster, Easwaran, & Keskin, 2009).

The first PDP formulation was proposed by Chandra and Fisher (1994). Since then, an alternative formulation was proposed in Archetti, Bertazzi, Paletta, and Speranza (2011) emphasizing the use of so called the order-up to level (OU) and the maximum level (ML) policies. The OU policy implies that the quantity of products shipped to the customer at the maximum level of its inventory capacity. Whereas the ML policy imposes that the quantity shipped is such that the inventory level is not greater than its capacity. Another formulation using Miller-Tucker-Zemlin subtour elimination constraints from Desrochers and Laporte (1991) was studied in Boudia, Louly, and Prins (2007) and Boudia and Prins (2009). PDP under uncertainty was considered in Adulyasak, Cordeau, and Jans (2015) and solved using stochastic programming. A multi-objective PDP considering multi-vehicle, carbon footprint and time windows was proposed in Kumar et al. (2015).

The current trends of researches in PDP is to propose more efficient solving methods for available data sets of Archetti et al. (2011) and Boudia et al. (2007) and Boudia and Prins (2009) such as exact methods (Amorim, Belo-Filho, Toledo, Almeder, & Almada-Lobo, 2013), Branch & Price (Bard & Nananukul, 2010), Branch & Cut (Archetti et al., 2011; Adulyasak, Cordeau, & Jans, 2014), Mathematical Programming-based Heuristics (Archetti et al., 2011), Lagrangian Relaxation (Fumero & Vercellis, 1999), Decomposition Heuristics (Bertazzi, Paletta, & Speranza, 2005; Chandra & Fisher, 1994; Chen, Hsueh, & Chang, 2009; Çetinkaya et al., 2009) and L-Shaped (Benders) Decomposition (Adulyasak et al., 2015). Some (meta) heuristics were also proposed such as Tabu Search (Shiguemoto & Armentano, 2010), Genetic Algorithm (Buer, Woodruff, & Olson, 1999), Greedy Randomized Adaptive Search Procedure (Boudia et al., 2007), Memetic Algorithm (Boudia & Prins, 2009), Ant Colony Optimization (Calvete, Galé, & Oliveros, 2011), Adaptive Large Neighbourhood Search (Adulyasak et al.) and Two-Phase Iterative Heuristics (Absi, Archetti, Dauzère-Pérès, & Feillet, 2014). To the best of our knowledge, Two-Phase Iterative Heuristics provide the best solutions for all available instances. Readers are suggested to read (Adulyasak, Cordeau, & Jans, 2015; Díaz-Madroñero et al., 2015) for further review in PDP.

The analysis of the literature suggested to use the solving method proposed in (Absi et al., 2014) as a basis for an efficient approximate method for Collection-Disassembly Problem. Before the presentation of a new method developed, a detailed problem formulation is given in the next section.

3. Problem definition

A single disassembly site having a capacitated inventory is responsible for gathering a single type of EOL products available at dispersed collection centres. A vehicle with fixed capacity is available for collecting the products under full truck load policy.

The structure of product is known: it contains several components with known and deterministic quantity. The disassembly process releases all the components from a product. The capacity

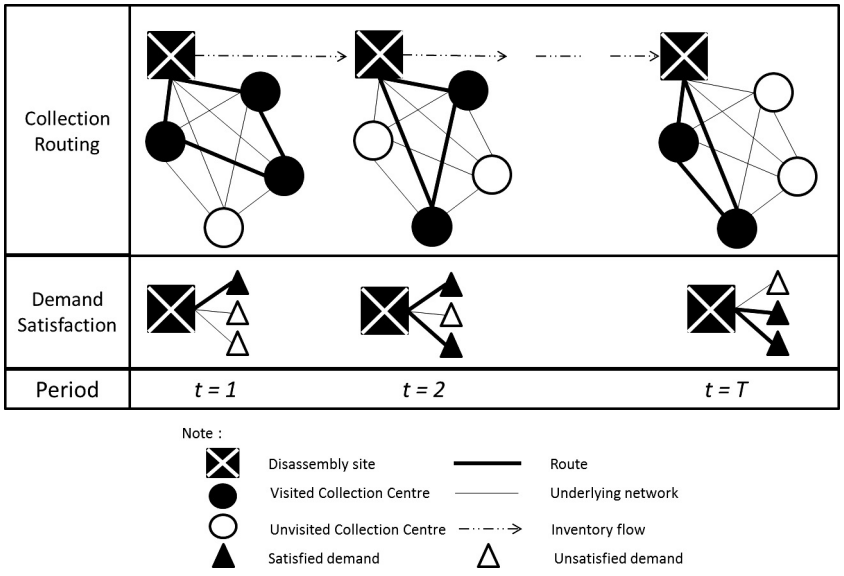

Fig. 1. Representations of our problem in Habibi et al. (2017).

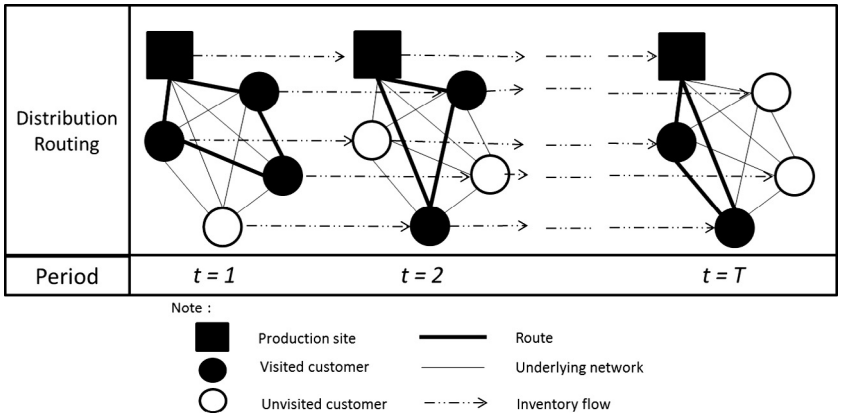

Fig. 2. Representations of production-distribution problem (Adulyasak et al., 2015).

of the disassembly line corresponds to its cycle time. The penalty cost is occurred due to unmet demand of components. There is no salvage value or disposal cost for any leftover components. The optimization problem can be formulated as follows:

Parameter

$\mathcal{A}$	set of component indexed by $a = \{1,2,\ldots,A\}$
$\mathcal{N}$	set of nodes indexed by $i,j = \{1,2,\ldots,N\}$
$\mathcal{N}_c$	set of collection centres indexed by $i,j = \{2,\ldots,N\}$
$\mathcal{T}$	planning horizon indexed by $t = \{1,2,\ldots,T\}$
n_a	amount of component a in product
S_{it}	amount of products available at collection centre i at period t
q_{at}	demand of component a at period t
Q	vehicle capacity
$InvCap$	inventory capacity
$DisCap$	disassembly capacity imposed from its cycle time
CF	fixed vehicle dispatch cost
c_{ij}	mileage cost from node i to j
CD	unit disassembly cost
CH	unit holding cost
CP_a	unit penalty cost of component a.

Decision variables

x_{ijt}	$\begin{cases} 1 & \text{if } j \text{ is visited after } i \text{ directly at period } t \\ 0 & \text{otherwise.} \end{cases}$
y_{it}	vehicle load after visiting i at period t
I_t	product inventory at period t
P_t	number of products disassembled at period t
SO_{at}	unsatisfied demand of component a at period t.

Integer Linear Programming (ILP) model

$$\textbf{Min} \sum_{t\in\mathcal{T}}\left(\sum_{j\in\mathcal{N}_c} CF\cdot x_{1jt} + \sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{N}} c_{ij}\cdot x_{ijt} + CH\cdot I_t + CD\cdot P_t + \sum_{a\in\mathcal{A}} CP_a\cdot SO_{at}\right) \tag{1}$$

Subject to :

$$\sum_{j\in N, i\neq j} x_{ijt} \leqslant 1 \quad \forall i\in\mathcal{N}_c, \quad \forall t\in\mathcal{T} \tag{2}$$

$$\sum_{i\in N, i\neq v} x_{ivt} = \sum_{j\in N\, j\neq v} x_{vjt} \quad \forall v\in\mathcal{N}, \quad \forall t\in\mathcal{T} \tag{3}$$

$$y_{it} + (Q - S_{it})\cdot x_{1it} \leqslant Q \quad \forall i\in\mathcal{N}_c, \quad \forall t\in\mathcal{T} \tag{4}$$

$$y_{it} - y_{jt} + Q\cdot x_{ijt} + (Q - S_{jt} - S_{it})\cdot x_{jit} \leqslant Q - S_{jt} \quad i\neq j,$$
$$\forall i,j\in\mathcal{N}_c, \quad \forall t\in\mathcal{T} \tag{5}$$

$$I_t = I_{t-1} + \sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{N}, i\neq j} S_{it}\cdot x_{ijt} - P_t \quad \forall t\in\mathcal{T} \tag{6}$$

$$n_a\cdot P_t + SO_{at} \geqslant q_{at} \quad \forall a\in\mathcal{A}, \quad \forall t\in\mathcal{T} \tag{7}$$

$$\sum_{j\in\mathcal{N}, i\neq j} S_{it}\cdot x_{ijt} \leqslant y_{it} \leqslant \sum_{j\in\mathcal{N}, i\neq j} Q\cdot x_{ijt} \quad \forall i\in\mathcal{N}, \quad \forall t\in\mathcal{T} \tag{8}$$

$$I_t \leqslant InvCap \quad \forall t\in\mathcal{T} \tag{9}$$

$$P_t \leqslant DisCap \quad \forall t\in\mathcal{T} \tag{10}$$

$$x_{ijt}\in\{0,1\} \quad \forall i,j\in\mathcal{N}, \quad \forall t\in\mathcal{T} \tag{11}$$

$$y_{it}, SO_{at}, I_t, P_t \in \mathbb{Z}^+ \quad \forall i\in\mathcal{N}, \quad \forall a\in\mathcal{A}, \quad \forall t\in\mathcal{T}. \tag{12}$$

The objective function (1) minimises the total cost consisting of total fixed vehicle dispatch cost, total mileage cost, total holding cost, total disassembly cost and total penalty cost.

Constraints (2) impose that each collection centre is visited at most once for each period. Constraints (3) ensure that the vehicle has to leave collection centre once it is visited. Constraints (4) ensure the charge of vehicle after visiting the first node. Constraints (5) prevent any subtour occurrence. The balance of inventory level at the disassembly site is assured by constraints (6). Constraints (7) impose that the demands need to be fulfilled, otherwise penalty will incur. Constraints (8)–(10) are the bound of the decision variables. Constraints (11) and (12) are domain constraints for decision variables.

Contraints (5) is able to prevent any subtour appearance. As depicted in Fig. 3, there is a subtour involving three nodes: 2, 3 and 4. Following the contraints, the arcs of (2,3), (3,4) and (4,2) imposes $y_{2t} - y_{3t} \leqslant -S_{3t}$, $y_{3t} - y_{4t} \leqslant -S_{4t}$ and $y_{4t} - y_{2t} \leqslant -S_{2t}$, respectively. The sum of the three inequalities leads to $S_{2t} + S_{3t} + S_{4t} \leqslant 0$ which is impossible unless S_{2t}, S_{3t} and S_{4t} are zero. If their values are zero, no vehicle visits the three nodes.

4. Two-Phase Iterative Heuristic for collection-disassembly problem

The developed method employs some ideas from Absi et al. (2014) where a Two-Phase Iterative Heuristic method was devel-

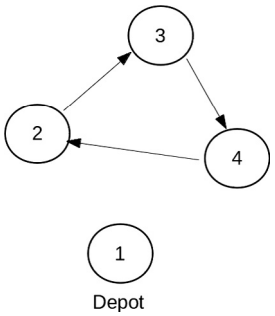

Fig. 3. Subtour of three nodes.

oped to solve PDP. The two phases correspond to the decomposition of the problem into lot-sizing and distribution decisions. The first phase uses approximate visiting costs which are updated throughout the algorithm. This phase corresponds to lot-sizing problem containing the decisions of when and how many products to produce, when to visit retailers and how many products to deliver. Consequently, the first phase provides a set of retailers visited in each period. Accordingly, the second phase aims to determine the vehicle route for each period. In the case of single vehicle, the second phase is a multi-travelling salesman problem (multi-TSP) since the vehicle capacity is already taken into account in the first phase.

In our work, this method is redesigned for the reverse supply chain context. Therefore, the first phase consists of lot-sizing problem considering the number of components in each products as well as the availability of products in each collection centre at each period. This phase determines a number of decision variables:

- when and how many products to disassemble P_t,
- how many products to put into the warehouse I_t,
- how many penalty to be compensated for each component SO_{at},
- and when to collect used products by visiting collection centres.

It also uses SC_{it} as approximate visiting costs of collection centre i at period t. This phase is called *Reverse Lot-Sizing Problem with Approximate Visiting Costs* (RLSP-AVC) in the first step of the method and RLSP-AVC II in the second one.

The second phase computes the routing for the vehicle used in each period. Since only one vehicle is considered, this phase deals with classical TSP. Fig. 4 illustrates this method.

In each iteration, our method solves the problem by following three steps. First, the problem is solved as RLSP-AVC. Second, it tries to reduce the periods served by solving RLSP-AVC II. The details on these problems and their solution are given in Section 4.1. The proposed method is provided in Algorithm 1.

Algorithm 1. Efficient Two-Phase Iterative Heuristic.

$sol \leftarrow \varnothing$;

Initialise $Prob, SC_{it}, \forall i \in \mathcal{N}_c, t \in \mathcal{T}$

while *the maximum number of diversification mechanism is not met* **do**

  while *the standard deviation is less than a predetermined value* **do**

    FIRST STEP

    - $1^{st} Phase$: Solve RLSP-AVC and get $\gamma_{it}, \forall i \in \mathcal{N}_c, \forall t \in \mathcal{T}$
    - $2^{nd} Phase$: Solve *Routing Problem*
    - Update sol (if necessary) and SC_{it}

    Generate a random value $Rand$ from 0 to 1

    if $Rand <= Prob$ **then**

      SECOND STEP

      - $1^{st} Phase$: Solve RLSP-AVC II and get $\gamma_{it}, \forall i \in \mathcal{N}_c, \forall t \in \mathcal{T}$
      - $2^{nd} Phase$: Solve *Routing problem*
      - Update sol (if necessary) and SC_{it}

    else

      $Prob = Prob/2$

    end

  end

  Diversify SC_{it}

end

The method is started by setting the best solution found *sol* as an empty solution. The value of SC_{it} is initialised. The first phase minimises RLSP-AVC containing the decisions of P_t, I_t, SO_{at} and when to collect products by visiting collection centres. Regarding the decision of products collection, we introduce a binary variable, γ_{it} and a non negative variable $r_{it}, \forall i \in N_c, \forall t \in T$, denoting whether a collection centre i is visited at period t and number of products collected from collection centre i at period t.

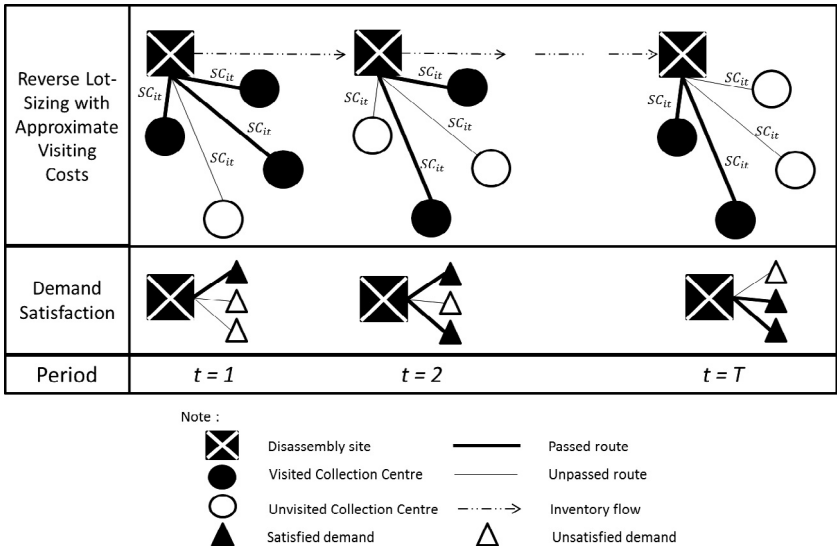

Fig. 4. Representation of reverse lot-sizing problem with approximate visiting costs.

Instead of using mileage cost c_{ij} and fixed vehicle dispatch cost CF, an approximate visiting cost, $SC_{it}, \forall i \in N_c, \forall t \in T$, is introduced. This cost has a prominent role to bridge the two phases since it contains an information regarding the cost occurred of visiting a node in a period. This cost is updated in each iteration of the method.

Initially, the value of SC_{it} is fixed using $c_{0i} + c_{i0}$ to force the first phase serve retailers who are near from the disassembly centre since it imposes high transportation cost. Accordingly, the first phase is solved and we get the value of decision variables *sol* consisting $P_t, I_t, SO_{at}, \gamma_{it}$ and r_{it}. Using the value of γ_{it}, the set of nodes visited in each period is obtained. Consecutively, the routing problem is solved. After the route of each period constructed, the value of SC_{it} is updated.

The stopping criteria are as follows:

- the standard deviation of the last ten iterations' objective values
- the maximum number of diversification mechanisms of SC_{it}. (detail in Section 4.4).

4.1. Reverse Lot-Sizing Problems with Approximate Visiting Costs (RLSP-AVC)

In Algorithm 1, it is shown that our method poses two types of reverse lot-sizing problem. RLSP-AVC is to determine P_t, I_t, SO_{at}, γ_{it} and r_{it}. Based on the value γ_{it}, RLSP-AVC II is solved by introducing Z denoting the number of periods served. A binary variable z_t is used in RLSP-AVC II denoting whether period t is served or not. RLSP-AVC II attempts to find a solution in which the number of periods served in RLSP-AVC is reduced. The following formulations are dedicated to RLSP-AVC and RLSP-AVC II.

Formulation of RLSP-AVC

$$\textbf{Min} \sum_{t\in\mathcal{T}} \left(CH \cdot I_t + CD \cdot P_t + \sum_{a\in\mathcal{A}} CP_a \cdot SO_{at} + \sum_{i\in N_c} SC_{it} \cdot \gamma_{it} \right) \tag{13}$$

Subject to :

(7), (9), (10)

$$I_t = I_{t-1} + \sum_{i\in N_c} r_{it} - P_t \quad \forall t \in \mathcal{T} \tag{14}$$

$$r_{it} = S_{it} \cdot \gamma_{it} \quad \forall i \in N_c, \quad \forall t \in \mathcal{T} \tag{15}$$

$$\sum_{i\in N_c} r_{it} \leqslant min\left\{ Q : max_a \left\{ \frac{\sum_{t'=t}^{T} q_{at'}}{n_a} \right\} \right\} \quad \forall t \in \mathcal{T} \tag{16}$$

$$\gamma_{it} \in \{0,1\} \quad \forall i \in \mathcal{N}_c, \quad \forall t \in \mathcal{T} \tag{17}$$

$$SO_{at}, I_t, P_t, r_{it} \in \mathbb{Z}^+ \quad \forall a \in \mathcal{A}, \quad \forall i \in \mathcal{N}_c, \quad \forall t \in \mathcal{T}. \tag{18}$$

The objective function (13) minimises the total cost consisting total holding cost, total disassembly cost, total penalty cost and total visiting cost. As mentioned previously, constraints (7) impose the satisfaction of component demands. The capacity limitation regarding the inventory and disassembly is denoted by constraint (9) and (10). Constraints (14) balance the inventory level at the disassembly site. Constraints (15) state that r_{it} is limited by the number of products available at node i once it is visited. Constraints (16) imposes that the total of r_{it} at period t is limited by the minimum value between the capacity of vehicle and the biggest remaining demands among components. Constraints (17) and (18) are domain constraints for decision variables.

Table 1
Global average gaps (in %).

Data set	CPLEX	$\mathcal{H}$				$\mathcal{H}^*$			
	CPU time (s)	⩽25 s	⩽50 s	⩽75 s	⩽100 s	⩽25 s	⩽50 s	⩽75 s	⩽100 s
Random 1	1715.5[46]	5.5	4.77	4.45	4.22	1.68	1.08	0.83	0.74
Random 2	1168.5[33]	0.98	0.73	0.58	0.49	−0.17	−0.38	−0.48	−0.55
Cluster 1	1961.4[46]	10.29	8.82	8.04	7.56	1.02	0.74	0.57	0.46
Cluster 2	2005.9[47]	8.68	7.19	6.85	6.48	1.22	0.76	0.59	0.5

⟨—⟩ indicates number of instances that were not solved optimally. Note that each data set consists of 108 instances.

Table 2
Average gaps in data set - random 1 (in %).

$\mathcal{N}$	$\mathcal{T}$	$\mathcal{A}$	CPLEX	$\mathcal{H}$				$\mathcal{H}^*$			
			CPU time (s)	⩽25 s	⩽50 s	⩽75 s	⩽100 s	⩽25 s	⩽50 s	⩽75 s	⩽100 s
5	5	5	0.1	2.14	2.14	1.52	1.52	0	0	0	0
		10	0.1	1.65	1.6	1.6	1.6	0.97	0.97	0.97	0.94
	10	5	1.2	2.12	1.49	1.46	1.41	0.84	0.57	0.19	0.07
		10	3.5	6.25	4.89	4.89	4.69	0.71	0.4	0.38	0.38
	25	5	3806$^{(2)}$	8.23	7.85	6.97	6.97	3.99	3.74	2.9	2.85
		10	3856.1$^{(4)}$	7.24	6.85	6.39	6.25	5.28	3.03	2.75	2.51
10	5	5	3.3	2.75	1.54	1.07	1.07	0.06	0.06	0.02	0.02
		10	3.1	3.19	3.03	3.03	2.05	2.18	1.21	1.21	1.21
	10	5	1471.5	6.66	4.95	4.51	4.27	0.25	0.21	0.08	0.08
		10	2752.8$^{(3)}$	4.86	4.05	3.88	3.84	1.29	0.88	0.68	0.55
	25	5	3408.1$^{(6)}$	6.99	6.36	6.14	6.06	3.9	3.33	3.03	2.36
		10	4506.8$^{(6)}$	5.98	5.6	5.6	5.31	2.43	2.02	1.8	1.18
25	5	5	391.4	5.94	4.85	4.23	4.23	0.39	0.2	0.2	0.2
		10	851.4	5.97	5.08	5.05	4.59	1.39	1.39	1.11	1.11
	10	5	4513.2$^{(6)}$	8.11	7.08	6.23	5.98	1.57	0.93	0.93	0.91
		10	1825$^{(6)}$	5.77	5.09	4.71	4.31	0.73	−0.16	−0.41	−0.41
	25	5	1619.4$^{(6)}$	7.98	7.22	7.03	6.7	3.86	1.65	0.14	0.14
		10	1865.4$^{(6)}$	7.08	6.17	5.8	5.05	0.3	−1.02	−1.02	−0.88
			Max	8.23	7.85	7.03	6.97	5.28	3.74	3.03	2.85
			Min	1.65	1.49	1.07	1.07	0	−1.02	−1.02	−0.88
			Average	5.5	4.77	4.45	4.22	1.67	1.08	0.83	0.73

⟨−⟩ indicates number of instances that were not solved optimally. Note that each line consists of 6 instances.

Table 3
Average gaps in data set - random 2 (in %).

$\mathcal{N}$	$\mathcal{T}$	$\mathcal{A}$	CPLEX	$\mathcal{H}$				$\mathcal{H}^*$			
			CPU time (s)	⩽25 s	⩽50 s	⩽75 s	⩽100 s	⩽25 s	⩽50 s	⩽75 s	⩽100 s
5	5	5	0.1	0.33	0.12	0.12	0.12	0	0	0	0
		10	0.1	0.26	0.26	0.26	0.26	0.26	0.26	0.26	0.26
	10	5	0.7	3.1	3.03	2.3	2.07	0.63	0.31	0.24	0.24
		10	1.3	1.45	1.32	1.32	1.32	0.09	0.09	0.09	0.09
	25	5	2413.4$^{(1)}$	6.38	5.88	5.25	5.25	1.73	1.34	1.34	1.3
		10	1555.3$^{(2)}$	3.42	3.13	3.03	2.59	2.35	1.97	1.95	1.41
10	5	5	1.2	0.48	0.35	0.35	0.3	0	0	0	0
		10	2	2.02	1.99	1.99	1.99	1.12	1.12	1.12	1.12
	10	5	11	1.2	0.64	0.64	0.64	0.01	0.01	0.01	0.01
		10	1497.2$^{(1)}$	2.5	2.48	2.36	2.01	0.87	0.28	0.28	0.28
	25	5	3038.5$^{(3)}$	2.91	2.64	2.53	2.5	0.41	0.32	0.26	0.26
		10	5194$^{(5)}$	5.76	5.3	4.78	4.75	4.28	3.21	2.48	2.04
25	5	5	174.6	0.87	0.23	0.23	0.23	0.01	0.01	0.01	0.01
		10	302.3	0.86	0.64	0.59	0.46	0	0	0	0
	10	5	1350.1$^{(3)}$	−0.19	−0.41	−0.41	−0.67	−0.78	−0.94	−0.96	−0.96
		10	1817.7$^{(6)}$	−1.05	−1.42	−1.53	−1.63	−1.59	−1.66	−1.81	−1.83
	25	5	1833.1$^{(6)}$	−4.44	−4.68	−4.8	−4.9	−4.44	−4.95	−5.27	−5.34
		10	1840.4$^{(6)}$	−8.27	−8.41	−8.5	−8.51	−8.01	−8.27	−8.65	−8.76
			Max	6.38	5.88	5.25	5.25	4.28	3.21	2.48	2.04
			Min	−8.27	−8.41	−8.5	−8.51	−8.01	−8.27	−8.65	−8.76
			Average	0.98	0.73	0.58	0.49	−0.17	−0.38	−0.48	−0.55

⟨−⟩ indicates number of instances that were not solved optimally. Note that each line consists of 6 instances.

Formulation of RLSP-AVC II

Min (13)

Subject to :

(7), (9), (10), (14)−(18)

$$\sum_{i\in\mathcal{N}_c}\gamma_{it} \leqslant N_c \cdot z_t \quad \forall t \in \mathcal{T} \tag{19}$$

$$\sum_{t\in\mathcal{T}} z_t \leqslant Z - 1 \tag{20}$$

$$z_t \in \{0,1\} \quad \forall t \in \mathcal{T}. \tag{21}$$

Herein, the values of SC_{it} are identical to those used in RLSP-AVC. Constraints (19) impose that z_t is equal to one if at least one collection centre is visited at period t. The value of Z is fixed according to the value of γ_{it} obtained from RLSP-AVC. Correspondingly, constraints (20) force the number of periods visited to be slightly lower than the one obtained in RLSP-AVC. Constraint (21) is a domain constraint for z_t.

Our experience shows that RLSP-AVC II improves the solution in the very first iterations and often requires more CPU time than RLSP-AVC. Henceforth, a probability *Prob* is attached to the second step in order to stabilise the CPU time as shown in Algorithm 1.

Table 4
Average gaps in data set - cluster 1 (in %).

$\mathcal{N}$	$\mathcal{T}$	$\mathcal{A}$	CPLEX	$\mathcal{H}$				$\mathcal{H}^*$			
			CPU time (s)	⩽25 s	⩽50 s	⩽75 s	⩽100 s	⩽25 s	⩽50 s	⩽75 s	⩽100 s
5	5	5	0.1	2.95	2.08	1.54	1.47	0.83	0.83	0.71	0.71
		10	0.2	4.08	2.96	1.83	1.65	1.54	1.54	1.43	1.43
	10	5	8	8.45	6.92	6.68	6.44	2.01	1.96	1.95	1.69
		10	11.9	7.96	5.44	4.4	4.06	1.29	0.52	0.27	0.27
	25	5	158.6$^{(1)}$	8.69	8.69	8.53	8.53	3.9	3.5	3.33	2.66
		10	1906$^{(1)}$	15.19	14.13	13.45	13.4	4.76	4.47	3.76	3.75
10	5	5	4.3	5.32	4.84	3.67	3.67	0.12	0.12	0.12	0.12
		10	4.5	7.39	7.32	7.32	7.2	3.38	3.38	3.38	3.38
	10	5	1853.7$^{(1)}$	5.75	4.2	4.2	4.2	0.59	0.52	0.39	0.39
		10	2818.9$^{(2)}$	10.61	9.71	8.53	8.17	1.55	0.99	0.64	0.64
	25	5	6306.9$^{(6)}$	19.1	18.46	16.56	15.31	2.56	1.62	1.06	1.06
		10	7203.8$^{(6)}$	20.3	19.86	19.16	19.16	3.66	2.55	2.26	2.19
25	5	5	3615.9$^{(2)}$	4.96	1.91	1.68	1.68	0.06	0.04	0.03	0.03
		10	3640.7$^{(3)}$	10.57	8.42	7.51	5.74	2.43	2.43	2.43	2.43
	10	5	2313.5$^{(6)}$	1.59	−0.07	−1.85	−2.72	−9.6	−9.72	−9.72	−9.73
		10	1814.5$^{(6)}$	15.78	12.16	10.64	9.56	−0.3	−0.43	−0.59	−0.69
	25	5	1822$^{(6)}$	16.85	15.94	15.18	13.44	−0.84	−0.84	−0.9	−1.51
		10	1821$^{(6)}$	19.64	15.78	15.7	15.12	0.46	−0.1	−0.23	−0.53
			Max	20.3	19.86	19.16	19.16	4.76	4.47	3.76	3.75
			Min	1.59	−0.07	−1.85	−2.72	−9.6	−9.72	−9.72	−9.73
			Average	10.29	8.82	8.04	7.56	1.02	0.74	0.57	0.46

⟨−⟩ indicates number of instances that were not solved optimally. Note that each line consists of 6 instances.

Table 5
Average gaps in data set - cluster 2 (in %).

$\mathcal{N}$	$\mathcal{T}$	$\mathcal{A}$	CPLEX	$\mathcal{H}$				$\mathcal{H}^*$			
			CPU time (s)	⩽25 s	⩽50 s	⩽75 s	⩽100 s	⩽25 s	⩽50 s	⩽75 s	⩽100 s
5	5	5	0.1	2.55	2.02	2.02	1.68	0.25	0	0	0
		10	0.2	4.68	2.19	2.19	1.23	0.82	0.82	0.82	0.82
	10	5	1.3	4.59	4.44	3.33	3.33	1.05	0.82	0.82	0.82
		10	2.3	5.14	4.67	4.39	4.35	0.78	0.6	0.59	0.59
	25	5	3086.1	8.76	8.04	7.97	7.64	4.93	4.13	3.71	3.51
		10	3900.8$^{(2)}$	10.16	9.49	9.4	8.95	3.56	2.46	2.15	2.15
10	5	5	3.4	4.02	2.99	2.78	2.78	1.49	0.38	0.38	0.32
		10	4.7	9.16	7.44	7.27	6.48	3.59	3.14	3.14	3.14
	10	5	2015.2$^{(1)}$	9.93	7.68	7.36	6.8	1.5	1.02	0.95	0.59
		10	1802.4$^{(1)}$	8.07	6.95	6.24	6.11	0.85	0.33	0.33	0.33
	25	5	5411.3$^{(6)}$	13.11	12.18	12.18	12.12	1.95	1.3	0.64	0.6
		10	6305.5$^{(6)}$	19.13	16.98	16.18	15.2	3.51	2.83	2.67	2.29
25	5	5	1912.2$^{(3)}$	3.69	2.82	2.69	2.63	0	0	0	0
		10	3660.1$^{(3)}$	8.15	4.78	4.78	4.78	2.02	2.02	2.02	2.02
	10	5	2514.7$^{(6)}$	8.21	7.36	6.82	6.82	−1.03	−1.21	−1.51	−1.51
		10	1822.2$^{(6)}$	10.24	8.02	7.13	5.9	0.33	−0.67	−1.32	−1.32
	25	5	1829.3$^{(6)}$	14.74	11.74	11.07	10.83	−1.79	−2.17	−2.52	−2.72
		10	1833.8$^{(6)}$	11.92	9.68	9.55	8.99	−1.92	−2.07	−2.2	−2.55
			Max	19.13	16.98	16.18	15.2	4.93	4.13	3.71	3.51
			Min	2.55	2.02	2.02	1.23	−1.92	−2.17	−2.52	−2.72
			Average	8.68	7.19	6.85	6.48	1.22	0.76	0.59	0.5

⟨−⟩ indicates number of instances that were not solved optimally. Note that each line consists of 6 instances.

4.2. Routing problem

The first phase gives information regarding a set of nodes served in each period. Therefore, the routing problem becomes Multi-TSP. To construct the route, we use the Lin-Kernighan Heuristic (Lin & Kernighan, 1973) as the state-of-the-art heuristic of solving TSP.

4.3. Update of Approximate Visiting Cost SC_{it}

Suppose that $route_t$ is the route of period t obtained by solving Routing Problem. For $t \in T$ and $i \in route_t$, let denote i^- and i^+ as the predecessor and the successor of node i in $route_t$. For $t \in T$ and $i \notin route_t$, let Δ_{it} as the cheapest insertion of node i into $route_t$. For each step, SC_{it} is updated using Algorithm 2.

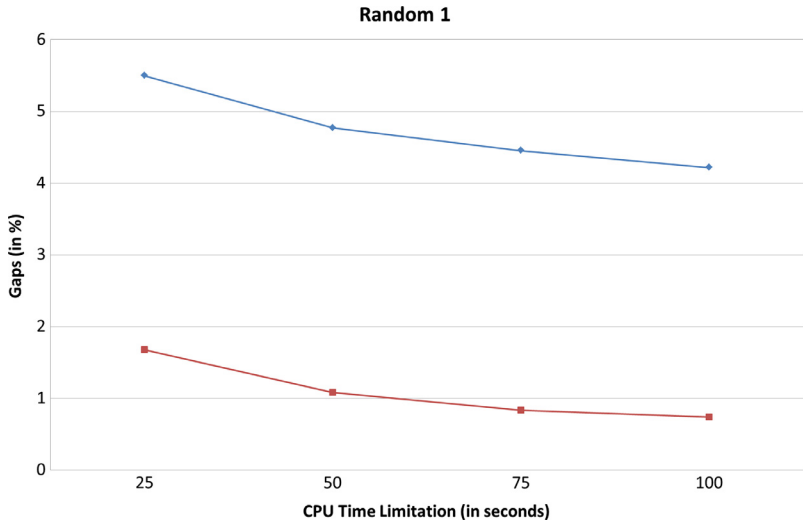

Fig. 5. Results of data set - random 1 using $\mathcal{H}$ (in blue) and $\mathcal{H}^*$ (in red). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

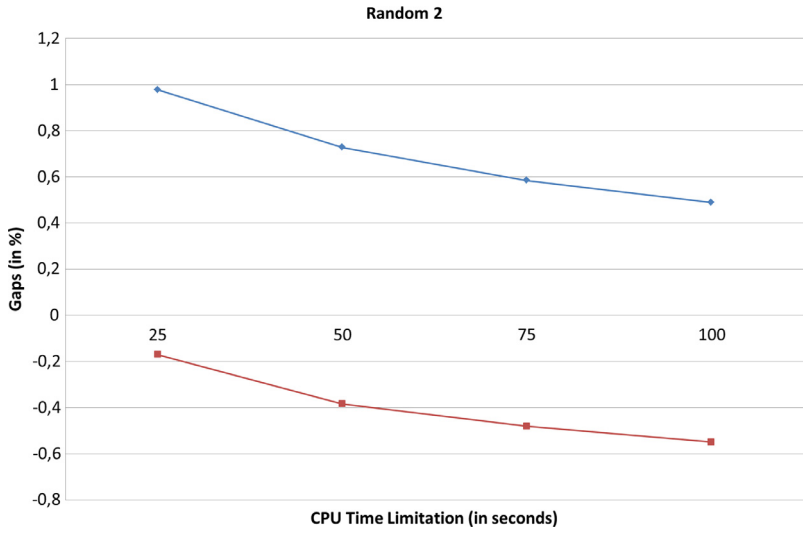

Fig. 6. Results of data set - random 2 using $\mathcal{H}$ (in blue) and $\mathcal{H}^*$ (in red). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Algorithm 2. Update of visiting cost.

```
forall the t ∈ 𝒯 do
    forall the i ∈ 𝒩_c do
        if i ∈ route_t then
            SC_it ← c_{i⁻i} + c_{ii⁺} − c_{i⁻i⁺}
        else
            SC_it ← Δ_it
        end
    end
end
```

4.4. Diversification mechanisms

The method uses two types of diversification mechanism: the multi-start procedure and the update diversification.

The multi-start procedure initialises the value of SC_{it} through multiplication with a random value, ρ_{it}. Its value is drawn between [0.0,1.5]. Thus, SC_{it} is set to $\rho_{it} \cdot (c_{0i} + c_{i0})$. Each start is stopped when its standard deviation of last ten iterations' objective values is less than a predetermined value.

The update diversification aims to reject a period having high number of retailers visited. It helps the method to move to the solution space that is not explored recently. For each t, the value of SC_{it} is multiplied by the number of retailers served plus one. Plus one is kept to avoid zero multiplication.

5. Numerical experiments

In this section, we compare the obtained results by CPLEX, Two-Phase Iterative Heuristic of (Absi et al., 2014) (denoted as $\mathcal{H}$) and our proposed heuristic (denoted as $\mathcal{H}^*$). All formulations and algorithms were implemented in Java using Concert Technology and were solved by CPLEX 12.6 on a PC with processor Intel® Core™i7 CPU 2.9 GHz and 4 G RAM under Windows 7 Professional.

Four data sets were taken from Habibi et al. (2017). All instances were solved with CPLEX. In order to avoid memory issues with CPLEX and obtain its lower bounds, the following solution procedure was adopted. Initially, all instances were solved under 2 h of CPLEX execution. The instances with memory issues were resolved under 30 min of CPLEX execution. Those with persistent memory issues were resolved under 10 min of CPLEX execution.

The gaps of CPLEX were obtained based on its best found solutions and their corresponding lower bounds *LB*s (linear relaxation). Similarly, the heuristics' ones were also obtained between its best solutions and *LB*s. The computational time takes into account the whole multi-start procedure.

The global average gaps are provided in Table 1. The results of each data set are provided in Tables 2–5. Both heuristics were executed in 25, 50, 75 and 100 s of CPU time. In these tables, we only

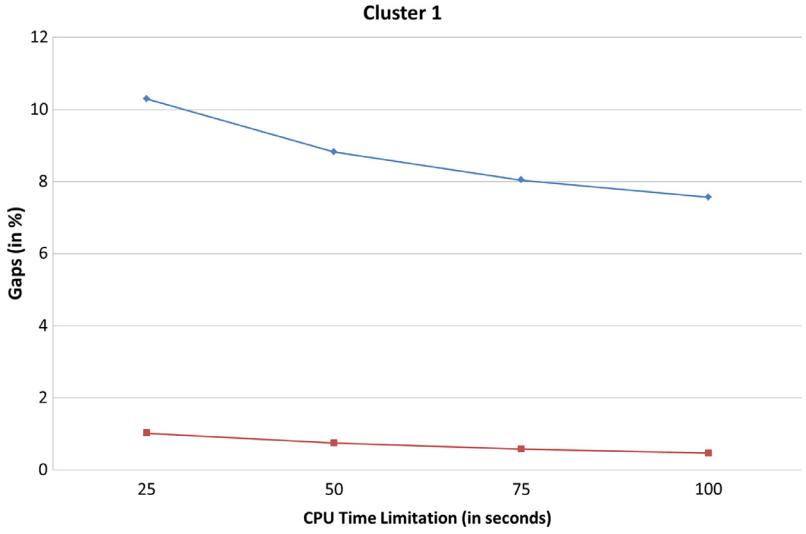

Fig. 7. Results of data set - cluster 1 using $\mathcal{H}$ (in blue) and $\mathcal{H}^*$ (in red). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

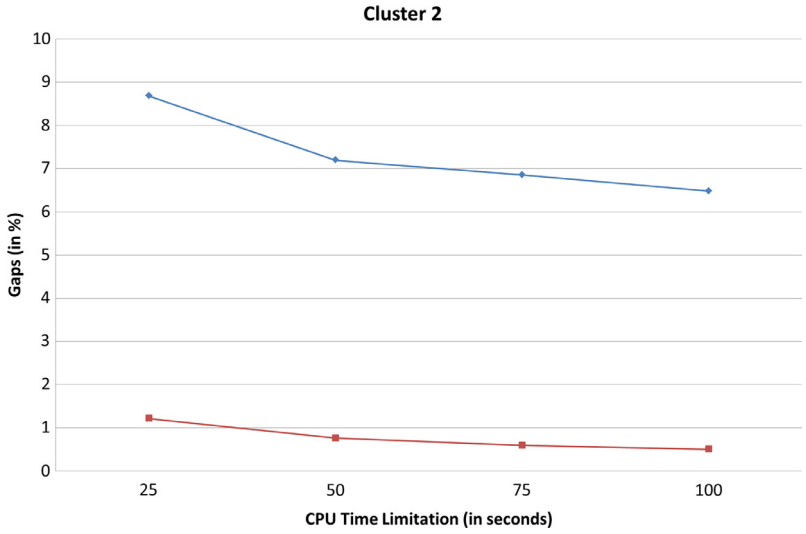

Fig. 8. Results of data set - cluster 2 using $\mathcal{H}$ (in blue) and $\mathcal{H}^*$ (in red). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

display the gaps between their obtained results and CPLEX's ones in percentage.

Based on these tables, we found that both heuristics propose solutions with good quality under faster CPU time rather than CPLEX. However, $\mathcal{H}^*$ always provides better solutions than $\mathcal{H}$ within the same CPU time as shown in Figs. 5–8.

6. Conclusion and future works

In order to improve the sustainability of closed-loop supply chains, the collection of used products and their disassembly have to be coordinated. However, the optimisation problems to be considered at each step are $\mathcal{NP}$-hard. Consequently, the integrated problem called Collection-Disassembly Problem is difficult to be dealt notably for large size instances. To tackle such instances in a more efficient way, we propose an Efficient Two-Phase Iterative Heuristic. In each iteration, this method solves the Collection-Disassembly Problem by following two steps. First, Reverse Lot-Sizing Problems with Approximate Visiting Costs (RLSP-AVC) is solved. Second, the method tries to reduce the served periods by solving RLSP-AVC II. The proposed method has been compared to available commercial solver and the original form of Two-Phase Iterative Heuristic. The obtained results show that our method provides the best solutions for all tested instances. Future research should notably focus on the extension of the current formulation by introducing multiple products and multiples vehicles. Then, uncertainties related to the collection amount and quality of the product as well as to the demand in components and materials need to be considered.

Acknowledgement

The government of Auvergne-Rhône-Alpes region of France is gratefully acknowledged for funding this work.

References

Absi, N., Archetti, C., Dauzère-Pérès, S., & 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., & Jans, R. (2014). Formulations and branch-and-cut algorithms for multivehicle production and inventory routing problems. *INFORMS Journal on Computing, 26*(1), 103–120. http://dx.doi.org/10.1287/ijoc.2013.0550.

Adulyasak, Y., Cordeau, J.-F., & Jans, R. (2015). The production routing problem: A review of formulations and solution algorithms. *Computers & Operations Research, 55*, 141–152. http://dx.doi.org/10.1016/j.cor.2014.01.011.

Adulyasak, Y., Cordeau, J., & Jans, R. (2015). Benders decomposition for production routing under demand uncertainty. *Operations Research, 63*(4), 851–867.

Adulyasak, Y., Cordeau, J., & Jans, R. Optimization-based adaptive large neighborhood search for the production routing problem, *Transportation Science* (November 2015).

Amorim, P., Belo-Filho, M., Toledo, F., Almeder, C., & Almada-Lobo, B. (2013). Lot sizing versus batching in the production and distribution planning of perishable goods. *International Journal of Production Economics, 146*(1), 208–218. http://dx.doi.org/10.1016/j.ijpe.2013.07.001.

Archetti, C., Bertazzi, L., Paletta, G., & Speranza, M. G. (2011). Analysis of the maximum level policy in a production-distribution system. *Computers & Operations Research, 38*(12), 1731–1746. http://dx.doi.org/10.1016/j.cor.2011.03.002.

Barba-Gutiérrez, Y., Adenso-Díaz, B., & Gupta, S. (2008). Lot sizing in reverse MRP for scheduling disassembly. *International Journal of Production Economics, 111*(2), 741–751. http://dx.doi.org/10.1016/j.ijpe.2007.03.017.

Bard, J. F., & Nananukul, N. (2010). A branch-and-price algorithm for an integrated production and inventory routing problem. *Computers & Operations Research, 37*(12), 2202–2217. http://dx.doi.org/10.1016/j.cor.2010.03.010.

Bentaha, M., Battaïa, O., & Dolgui, a. (2014a). A sample average approximation method for disassembly line balancing problem under uncertainty. *Computers & Operations Research, 51*, 111–122. http://dx.doi.org/10.1016/j.cor.2014.05.006.

Bentaha, M. L., Battaïa, O., & Dolgui, A. (2014b). Disassembly line balancing and sequencing under uncertainty. *Procedia CIRP, 15*, 239–244. http://dx.doi.org/10.1016/j.procir.2014.06.016.

Bertazzi, L., Paletta, G., & Speranza, M. (2005). Minimizing the total cost in an integrated vendor managed inventory system. *Journal of Heuristics, 11*(5), 393–419.

Boudia, M., Louly, M., & Prins, C. (2007). A reactive GRASP and path relinking for a combined production distribution problem. *Computers & Operations Research, 34*(11), 3402–3419. http://dx.doi.org/10.1016/j.cor.2006.02.005.

Boudia, M., & 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. http://dx.doi.org/10.1016/j.ejor.2007.07.034.

Brown, G., Keegan, J., Vigus, B., & Wood, K. (2001). The Kellogg company optimizes production, inventory, and distribution. *Interfaces, 31*(6), 1–15.

Buer, M. V., Woodruff, D., & Olson, R. (1999). Solving the medium newspaper production/distribution problem. *European Journal of Operational Research, 115*(2), 237–253.

Calvete, H. I., Galé, C., & Oliveros, M.-J. (2011). Bilevel model for production distribution planning solved by using ant colony optimization. *Computers & Operations Research, 38*(1), 320–327. http://dx.doi.org/10.1016/j.cor.2010.05.007.

Çetinkaya, S., Üster, H., Easwaran, G., & Keskin, B. B. (2009). An integrated outbound logistics model for Frito-lay: Coordinating aggregate-level production and distribution decisions. *Interfaces, 39*(5), 460–475. http://dx.doi.org/10.1287/inte.1090.0450.

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

Chen, H.-K., Hsueh, C.-F., & Chang, M.-S. (2009). Production scheduling and vehicle routing with time windows for perishable food products. *Computers & Operations Research, 36*(7), 2311–2319. http://dx.doi.org/10.1016/j.cor.2008.09.010.

Desrochers, M., & Laporte, G. (1991). Improvements and extensions to the Miller-Tucker-Zemlin subtour elimination constraints. *Operations Research Letters, 10*(February), 27–36.

Díaz-Madroñero, M., Peidro, D., & Mula, J. (2015). A review of tactical optimization models for integrated production and transport routing planning decisions. *Computers & Industrial Engineering, 88*, 518–535.

Ferrer, G., Heath, S. K., & Dew, N. (2011). An RFID application in large job shop remanufacturing operations. *International Journal of Production Economics, 133*(2), 612–621. http://dx.doi.org/10.1016/j.ijpe.2011.05.006.

Fumero, F., & Vercellis, C. (1999). Synchronized development of production, inventory, and distribution schedules. *Transportation Science, 33*(3), 330–340. http://dx.doi.org/10.1287/trsc.33.3.330.

Godichaud, M., & Amodeo, L. Efficient multi-objective optimization of supply chain with returned products. *Journal of Manufacturing Systems*, doi:http://dx.doi.org/10.1016/j.jmsy.2014.12.004.

Habibi, M. K., Battaïa, O., Cung, V.-D., & Dolgui, A. (2017). Collection-disassembly problem in reverse supply chain. *International Journal of Production Economics, 183*, 334–344. http://dx.doi.org/10.1016/j.ijpe.2016.06.025.

Kumar, R. S., Kondapaneni, K., Dixit, V., Goswami, A., Thakur, L., & Tiwari, M. (2015). Multi-objective modeling of production and pollution routing problem with time window: A self-learning particle swarm optimization approach. *Computers & Industrial Engineering, 99*, 29–40. http://dx.doi.org/10.1016/j.cie.2015.07.003.

Lin, S., & Kernighan, B. W. (1973). An effective heuristic algorithm for the traveling-salesman problem. *Operations Research, 21*(2), 498–516.

McGovern, S., & Gupta, S. (2011). *The disassembly line: Balancing and modeling* (1st ed.). New York: McGraw-Hill.

Shiguemoto, A. L., & Armentano, V. A. (2010). A tabu search procedure for coordinating production, inventory and distribution routing problems. *International Transactions in Operational Research, 17*(2), 179–195. http://dx.doi.org/10.1111/j.1475-3995.2009.00741.x.

Yeh, W.-C. (2012). Simplified swarm optimization in disassembly sequencing problems with learning effects. *Computers & Operations Research, 39*(9), 2168–2177. http://dx.doi.org/10.1016/j.cor.2011.10.027.