Combining Production and Distribution in Supply Chains: the Hybrid Flow-Shop Vehicle Routing Problem

Leandro do C. Martins[a,*], Eliana M. Gonzalez-Neira[b,c], Sara Hatami[a], Angel A. Juan[a], Jairo R. Montoya-Torres[c]

[a] *IN3 – Computer Science Department, Universitat Oberta de Catalunya, Barcelona, Spain*
[b] *Departmento de Ingeniería Industrial, Pontificia Universidad Javeriana, Bogotá, Colombia*
[c] *Grupo de Investigación en Sistemas Logísticos, Universidad de La Sabana, Chía, Colombia*

Abstract

Many supply chains are composed of producers, suppliers, carriers, and customers. These agents must be coordinated to reduce waste and lead times. Production and distribution are two essential phases in most supply chains. Hence, improving the coordination of these phases is critical. This paper studies a combined hybrid flow-shop and vehicle routing problem. The production phase is modeled as a hybrid flow-shop configuration. In the second phase, the produced jobs have to be delivered to a set of customers. The delivery is carried out in batches of products, using vehicles with a limited capacity. With the objective of minimizing the service time of the last customer, we propose a biased-randomized variable neighborhood descent algorithm. Different test factors, such as the use of alternative initial solutions, solution representations, and loading strategies, are considered and analyzed.

Keywords: hybrid flow-shop problem, vehicle routing problem, biased randomization, metaheuristics

1. Introduction

In most supply chains, there is an increasing need to coordinate the efforts of suppliers, producers, and carriers to efficiently deliver products to customers, so that waste and lead times are reduced. The production and distributions phases are critical in any supply chain: finished

*Corresponding author

Email addresses: leandrocm@uoc.edu, eliana.gonzalez@javeriana.edu.co, shatami@uoc.edu, ajuanp@uoc.edu, jairo.montoya@unisabana.edu.co (Jairo R. Montoya-Torres), leandrocm@uoc.edu,shatami@uoc.edu,eliana.gonzalez@javeriana.edu.co,ajuanp@uoc.edu,jairo.montoya@unisabana.edu.co, marcone.freitas@gmail.com (Jairo R. Montoya-Torres)

Preprint submitted to Journal's Name *November 9, 2023*

products are transferred from production centers to a warehouse or distribution centers by cargo vehicles. In order to enhance the operational performance, both phases need to be considered while optimizing operations. Still, due to the complexity of these phases, traditional approaches usually consider them as two isolated problems (Chen, 2010).

In this paper, we offer a more holistic approach by considering the production and distribution phases altogether. This is the case, for example, of distributing medical tests or vaccines to local health centers –so they can be administrated to the population as soon as possible– while these items are being produced, in large quantities, at a central laboratory. Hence, the production phase is modeled as a hybrid flow-shop (HFS) environment, while the distribution phase is modeled as a vehicle routing problem (VRP). Accordingly, the combined problem can be referred to as a hybrid flow-shop vehicle routing problem (HFS-VRP). As shown in Figure 1, in the production phase a set J of jobs (items) are processed. Each job has to go through a set S of sequential stages. At each stage $s \in S$, a set M_s of parallel and identical machines are available to process the job. Given a job $j \in J$, its processing time in stage $s \in S$ is given by $p_{js} > 0$. Regarding the distribution phase, a set C of customers and a single vehicle that makes multiple trips are considered. In each trip the vehicle deliveries a batch of jobs. Each job $j \in J$ allows to a specific customer $c \in C$ and occupies a volume of $q_j > 0$, being $Q \gg \max_{i \in J}\{q_j\}$ the maximum loading capacity of the vehicle.

In order to speed up the delivery process, finished items are grouped into batches that can be delivered to customers while the production system is manufacturing new ones. In this context, the goal is to minimize the total time elapsed since the start of the manufacturing process and the delivery of the last customer's demand, i.e., the makespan of the hybrid problem. In order to solve the proposed HFS-VRP, three different and interrelated decisions have to be made: (*i*) determining the job sequence on each machine at the production phase; (*ii*) assigning the finished jobs to a proper batch for deliver; and (*iii*) determining adequate route for each trip of the vehicle in order to deliver jobs to customers.

To the best of our knowledge, and despite its many applications in supply chain management, this is the first time that such a combined hybrid flow-shop and vehicle routing problem has been discussed in the scientific literature. To cope with the complexity of the HFS-VRP, we propose a biased-randomized variable neighborhood descend (BR-VND) metaheuristic. Additionally, a new set of instances, which are based on some well-known benchmark instances of both the HFS and

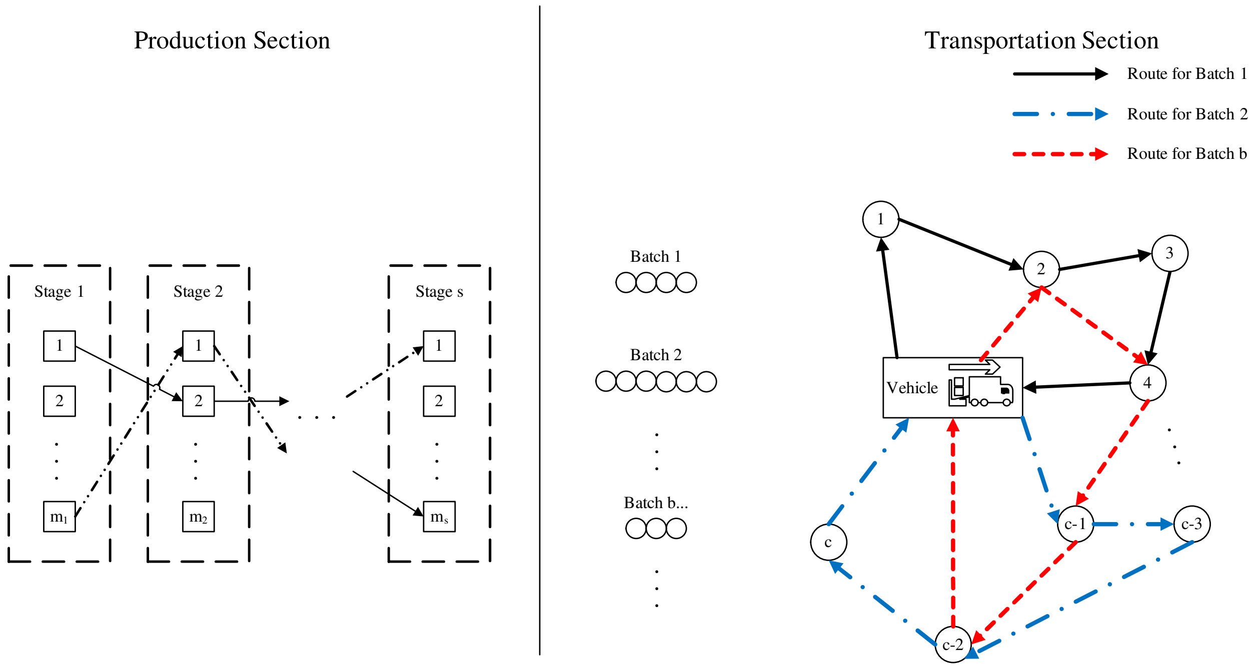

Figure 1: Combined production and distribution operations.

the VRP, are introduced.

The rest of the paper is arranged as follows: Section 2 provides a short literature review on related research. Section 5 describes the proposed BR-VND algorithm. In Section 6, a series of computational experiments are preformed. Finally, Section 7 provides the conclusions of the work and proposes some open research lines.

2. Literature Review

The analysis of combined production and distribution processes has been quite common from a tactical and strategical points of view. Hence, many review papers have been published on these areas, e.g.: Thomas and Griffin (1996), Cohen and Mallik (1997),Vidal and Goetschalckx (1997), Erengüç et al. (1999), Sarmiento and Nagi (1999), Goetschalckx et al. (2002), Chen (2004), Meixell and Gargeya (2005), Olhager et al. (2015) and Koç et al. (2017). However, research at the operational level is much more recent and scarce, with just a few articles discussing the combination of production scheduling and vehicle routing operations (Chen, 2010).

According to Karaoğlan and Kesen (2017), the integrated production scheduling and transportation problem can be classified into three categories, depending on the method employed for sorting the deliveries. The first category consists of simple methods like direct shipping, without a routing process: an order, a batch to a single client, or a batch to multiple customers delivered as soon as the production process is finished. Examples of the first category can be found in the works of Cakici et al. (2014) and Wang et al. (2016). The second category involves fixed transportation departure dates with a predetermined departure time for each vehicle, e.g.: Stecke and Zhao (2007) and Hajiaghaei-Keshteli et al. (2014). The third category consists of vehicle routing decisions to be made, involving the determination of departure times. A complete discussion on the integrated production scheduling and distribution operations can be found in Chen and Vairaktarakis (2005), Wang et al. (2015), and Moons et al. (2017). Our review will mainly focus on the third category, which is also the less studied one in the literature (Karaoğlan and Kesen, 2017).

Li et al. (2005) considered applications where one manufacturing factory and one delivery process are studied. Two objectives were analyzed: the customer service level and total distribution costs. Customer service was studied with two different measures: mean completion time and makespan. Dispatching costs included fixed and variable costs, the latter depending on the traveled distance. The authors proposed several mathematical models and, when the problems could not be solved exactly, they proposed heuristic approaches to obtain near-optimal solutions. Li and Vairaktarakis (2007) solved a bundling operations problem in which two dedicated machines perform two different tasks of the same job that can be executed in parallel. The job is finished when the two tasks are completed. Then, transportation is carried out by various vehicles. Decisions to be made are the sequencing of jobs into machines, the number of vehicles for transportation, and the routes they have to follow. The objective was to minimize the total cost of transportation and the waiting cost of customers. The authors proposed a polynomial-time algorithm and several heuristics to solve the problem. Armstrong et al. (2008) considered a single-machine problem in which jobs belonging to the same production order must be processed one after the other. Production orders had time windows for delivery, with no inventory allowed between the production and the transportation stages.

Armstrong et al. (2008), Geismar et al. (2008) and Geismar et al. (2011) considered the production and distribution of perishable products, which require avoiding waiting times before delivery. Armstrong et al. (2008) have included delivery time windows specified by the customer. Due to the limited resources of production and delivery processes, the whole demand cannot be met. Thus, the decision is to select the subset of customers that can be served, such in a way that the total satisfied demand is maximized. To solve this problem, the authors proposed a branch-and-bound algorithm. Geismar et al. (2008) have included the vehicle routing problem in the decision process. Since this is an *NP-hard* problem, these authors developed lower bounds that were used by a two-phase metaheuristic algorithm. The first phase is a genetic algorithm (GA) that provides a local optimum sequence for completing the products of the selected customers. The second phase divides the sequence into various subsets and use the Gilmore-Gomory algorithm (Gilmore and Gomory, 1961) to order the sub-sequences. Geismar et al. (2011) studied the same problem but considering intermediate hubs, which cluster some customers. The objective here was to minimize the total cost of production and transportation operations, while respecting the product lifetime and delivery capacity of vehicles.

Farahani et al. (2012) solved a cost minimization problem in the production and distribution scheduling of catering foods by employing an iterative hierarchical approach. In the first stage, the authors applied an aggregation procedure to create batches of orders with similar characteristics. Then, a block planning scheme is proposed to schedule the batches. Next, a heuristic is used to solve the delivery problem. Finally, the iterative approach is implemented to coordinate both schedules. Condotta et al. (2013) considers a single machine in the production stage, and a given fleet of vehicles with limited capacity to deliver final products. Jobs have a due date for delivery, and the goal is to minimize the lateness. A tabu search (TS) algorithm was proposed for obtaining partial solutions at the production stage. Later, the TS was hybridized with an optimal transportation schedule. Hajiaghaei-Keshteli and Aminnayeri (2014) proposed one heuristic procedure and two metaheuristics, GA and simulated annealing (SA), to maximize customer service at minimum total cost. Their GA obtained the best results, especially as the instance size increases.

Low et al. (2014) considered the production of a variety of products associated with one customer as a batch. The batches might be delivered immediately after completion, or might be grouped with other batches for delivering to the corresponding retailers. An heterogeneous fleet of vehicles was considered to minimize total costs. They proposed a mixed-integer linear programming (MILP) model and two GAs. Kang et al. (2016) solved a real case from a semiconductor industry. Constraints, such as job clusters, production costs depending on the job clusters, setup costs, and transportation costs of multiple vehicles were considered. The authors proposed a MILP model and a GA to minimize the total cost for large instances.

Karaoğlan and Kesen (2017) proposed a branch-and-cut algorithm to minimize the makespan in the production of a single product with limited shelf life. For delivery purposes, there is only one single vehicle with limited capacity. Fu et al. (2017) analyzed the problem with unrelated parallel machines and job splitting during the production stage. The transportation stage included delivery time windows and the delivery of jobs in batches using heterogeneous vehicles. Two objectives were evaluated with the use of an iterative heuristic: the setup costs minimization at the production stage and transportation costs for delivery.

3. Mixed Integer Linear Programming Model of HFS-VRP

In this section, we propose a MILP model of the HFS-VRP. Firstly, the sets and parameters are defined, thence, the decision variables, objective function and constraints.

Sets:

J: jobs $\{1...n\}$

S: stages $\{1...s\}$

M_s: machines at stage $s \in S$ $\{1...m_s\}$

R: trips (deliveries of batches) $\{1...r\}$

C: customers $\{1...c\}$

JC_c: jobs of customer $c \in C$ $\{1...jc_c\}$

Parameters:

B: very big constant

$P_{j,s}$: processing time of job $j \in J$ at stage $s \in S$

Q: capacity of *vehicle*

q_j: loading volume occupied by job $j \in J$

$TT_{c,a}$: travel time between customer $c \in C \cup \{0\}$ and $a \in C \cup \{0\}$ (where node 0 is he factory)

Variables:

$X_{j,h,s}$: binary variable that takes the value of 1 if job $j \in J$ is processed before job $h \in J$ at stage $s \in S$, and 0, otherwise

$Y_{j,s,m}$: binary variable that takes the value of 1 if job $j \in J$ is processed on machine $m \in E_s$ of stage $s \in S$, and 0, otherwise

$ST_{j,s}$: continuous variable for the starting time of job $j \in J$ processed on machine $m \in E_s$ of stage $s \in S$

$CT_{j,s}$: continuous variable for the completion time of job $j \in J$ processed on machine $m \in E_s$ of stage $s \in S$

SR_r: departure time of trip $r \in R$ of the vehicle

CR_r: completion time of trip $r \in R$ of the vehicle

$TV_{c,r}$: time of arrival at customer $c \in C \cup \{0\}$ on trip $r \in R$

$F_{c,a,r}$: binary variable that takes the value of 1 if customer $c \in C \cup \{0\}$ is visited before customer $a \in C \cup \{0\}$ in trip $r \in R$

$W_{j,r}$: binary variable that takes the value of 1 if job $j \in J$ is dispatched on trip $r \in R$

G_r: binary variable that takes the value of 1 if the vehicle performs the trip $r \in R$

$N_{c,r}$: binary variable that takes the value of 1 if the customer $c \in C$ is visited on trip $r \in R$

$Cmax$: makespan or maximum dispatching time of the jobs

$$min\ Z = C_{max} \tag{1}$$

s.t.:

$$\sum_{m \in M_s} Y_{j,s,m} = 1 \qquad \forall j \in J, \forall s \in S \tag{2}$$

$$CT_{j,s} = ST_{j,s} + P_{j,s} \qquad \forall j \in J, \forall s \in S, \forall m \in M_s \tag{3}$$

$$ST_{j,s} \geq CT_{j,s-1} \qquad \forall j \in J, \forall s \in S, s > 1 \tag{4}$$

$$ST_{h,s} \geq CT_{j,s} - B \cdot (3 - X_{j,h,s} - Y_{j,s,m} - Y_{h,s,m}) \qquad \forall j,h \in J, \forall s \in S, \forall m \in M_s, j \neq h \tag{5}$$

$$ST_{j,s} \geq CT_{h,s} - B \cdot X_{j,h,s} - B \cdot (2 - Y_{j,s,m} - Y_{h,s,m}) \qquad \forall j,h \in J, \forall s \in S, \forall m \in M_s, j \neq h \tag{6}$$

$$SR_r \geq CT_{j|S|} - B \cdot (1 - W_{j,r}) \qquad \forall j \in J, \forall r \in R \tag{7}$$

$$TV_{c,r} \geq TV_{a,r} + TT_{a,c} - B \cdot (1 - F_{j,r}) \qquad \forall c \in C, \forall a \in C \cup \{0\}, \forall r \in R, c \neq a \tag{8}$$

$$TV_{0,r} \geq SR_r \qquad \forall r \in R \tag{9}$$

$$\sum_{j \in J_c} Wj, r \leq N_{c,r} \cdot B \qquad \forall c \in C, \forall r \in R \tag{10}$$

$$N_{c,r} \leq \sum_{j \in J_c} Wj, r \qquad \forall c \in C, \forall r \in R \tag{11}$$

$$\sum_{a \in C \cup \{0\} a \neq c} F_{a,c,r} = N_{c,r} \qquad \forall c \in C, \forall r \in R \tag{12}$$

$$\sum_{a \in C \cup \{0\}, a \neq c} F_{c,a,r} = N_{c,r} \qquad \forall c \in C, \forall r \in R \tag{13}$$

$$\sum_{c \in C} F_{0,c,r} = G_{c,r} \qquad \forall r \in R \tag{14}$$

$$\sum_{c \in C} F_{c,0,r} = G_{c,r} \qquad \forall r \in R \tag{15}$$

$$\sum_{r \in R} W_{jr} = 1 \qquad \forall j \in J \tag{16}$$

$$\sum_{j \in J} q_j \cdot W_{j,r} \leq Q \cdot G_r \qquad \forall r \in R \tag{17}$$

$$CR_r \geq TV_r \qquad \forall c \in C, \forall r \in R \tag{18}$$

$$SR_{r+1} \geq CR_r + TT_{c,0} - B \cdot (1 - F_{c,0,r}) \qquad \forall c \in C, \forall r \in R, r < |R| \tag{19}$$

$$C_{max} \geq CR_r \qquad \forall r \in R \tag{20}$$

$$G_r \leq G_{r-1} \qquad \forall r \in R, r > 1 \tag{21}$$

$$X_{j,h,s} \in \{0,1\} \qquad \forall j, h \in J, \forall s \in S \tag{22}$$

$$Y_{j,s,m} \in \{0,1\} \qquad \forall j \in J, \forall s \in S, \forall m \in M_s \tag{23}$$

$$F_{c,a,r} \in \{0,1\} \qquad \forall c \in C \cup \{0\}, \forall a \in C \cup \{0\}, \forall r \in R \tag{24}$$

$$W_{j,r} \in \{0,1\} \qquad \forall j \in J, \forall r \in R \tag{25}$$

$$G_r \in \{0,1\} \qquad \forall r \in R \tag{26}$$

$$N_{c,r} \in \{0,1\} \qquad \forall c \in C, \forall r \in R \tag{27}$$

$$CT_{j,s} \geq 0 \qquad \forall j \in J, \forall s \in S \tag{28}$$

$$ST_{j,s} \geq 0 \qquad \forall j \in J, \forall s \in S \tag{29}$$

$$SR_r \geq 0 \qquad \forall r \in R \tag{30}$$

$$CR_r \geq 0 \qquad \forall r \in R \tag{31}$$

$$TV_{c,r} \geq 0 \qquad \forall c \in C \cup \{0\}, \forall r \in R \tag{32}$$

Equation (1) represents the objective function, that is the minimization of the makespan, that is, the time in which the last job is delivered. Constraints set (2) specifies that each job can be assigned at only one machine at each stage. Constraints set (3) calculates the completion time of each job at each stage. Constraints set (4) determines the minimum starting time of each job at each stage regarding the completion time of the job in the previous stage. Constraints sets (5) and (6) specify the minimum starting time of each job at each stage regarding the completion time of jobs processed before at the same machine. Constraints set (7) defines the minimum starting time of each (delivery of batch) regarding the maximum completion time of the jobs that are going to be dispatched on that trip. Constraints set (8) specifies the minimum time of the visit of a customer in a trip depending on the time of the visit of the previous customer in that trip, and the travel time between both customers. Constraints set (9) indicates that the time of the visit of the depot (node 0) in a trip is equal to the departure time of that trip. Constraints sets (10) and (11) guarantee that, if a job is dispatched on a trip, the customer who is the owner of that job is visited on that trip. Constraints sets (12) and (13) state that, if a customer is visited on a trip, that customer is a successor and a predecessor of another customer or depot. Constraints sets (14) and (15) ensure that each trip starts and ends at depot if the trip is performed (node 0). Constraints set (16) guarantees that each job is dispatched in exactly one trip. Constraints set (17) assures that the volume capacity of the vehicle on each trip is not surpassed. Constraints set (18) calculates the completion time of a trip regarding the time of the last customer visited in that trip. Constraints set (19) states that the starting time of a trip is greater or equal than the return time of the vehicle to the depot after the previous trip. Constraints set (20) specifies that the completion time of the last delivery is greater or equal than the completion time of the last trip. Constraints set (21) controls the binary variables of trips, ensuring that only consecutive trips can be performed. Finally, constraints sets (22)-(32) define the domain of decision variables.

3.1. Numerical Example of HFS-VRP Problem

As mentioned before, the following three decisions have to be made in order to solve HFS-VRP problem: (i) determining the job sequence at the production stage; (ii) assigning the finished jobs to a proper batch for delivery; and (iii) defining the routing plan for the single cargo vehicle. In order to give a better understanding of the problem, Figure 2 provides the following example with 6 jobs ($n = 6$) and 3 stages ($s = 3$), in which the first and third stages are composed of 3 machines each ($m_1 = 3$ and $m_3 = 3$), while the second stage is composed of a single machine ($m_2 = 1$).

1. At the HFS stage, each job is described by a tuple (j, c_j, q_j), in which j is the job identifier, c_j is the customer who requires the job j, and q_j is the loading volume of job j. For instance,

in tuple $(1, 1, 10)$, the job 1 is requested by customer 1, and consists of 10 demand size units. Once processed in the first stage –with a completion time of 100 time units– the job 1 can be processed in the following stage from time 100, and so on. The remaining jobs follow the same interpretation.

2. The second stage aims to join processed jobs into batches that meets the capacity constraint of the cargo vehicle. Each batch corresponds to one trip. In this example, the vehicle has a capacity of 50 demands units. A batch b are represented by the set of jobs and the tuple (CR_b, TD_b), where CR_b and TD_b represent the completion time and total volume of batch b, respectively. For example, the batch 1 is composed of jobs 3 and 2, has a completion time is 600 time units, and its total volume is 45 units.
3. The last stage regards the vehicle routing process. For the first batch 1, the vehicle starts its delivery at time SR_1, i.e., 600 time units. In this stage, each node is characterized by the tuple $[TV_{c,b}, RD_{c,b}]$, in which $TV_{c,b}$ represents the arrival time at node c of batch (trip) b, and $RD_{c,b}$ represents the remaining loaded demand at node c of batch (trip) b. For instance, the vehicle arrives at the depot after delivering the jobs at time 820 with no loaded demand. For the next route, the delivery starts at time 820, since the vehicle arrives at the depot after batch 2 being ready for delivery at time 800, i.e., the $max(800, 820)$. In case the vehicle is ready for delivery before the conclusion time of the batch, it must wait for the time needed for the batch to be ready and loaded. The same is done for the remaining batches.
4. Finally, the solution cost is given by the time in which the vehicle returns to the depot after delivering the jobs from the last batch. In this example, the integrated cost is 1210.

4. Lower Bound for HFS-VRP Problem

Considering the NP-hardness of the problem, we propose to calculate a lower bound for the problem, $LB_{HFS-VRP}$, in order to evaluate the performance of our proposed algorithms. Since the deliveries of some finished jobs can be performed simultaneously with the production of other jobs, it can be said that the HFS stage is overlapped partially with the VRP stage. For that reason, the proposed $LB_{HFS-VRP}$ consists in the maximum of the following two partial lower bounds (33):

(i) The $LB_{HFS-VRP_{part1}}$ (Equation 34), which consists in adding the HFS lower bound proposed by Haouari and Hidri (2008) (35) and the traveling time of the nearest customer to the factory.

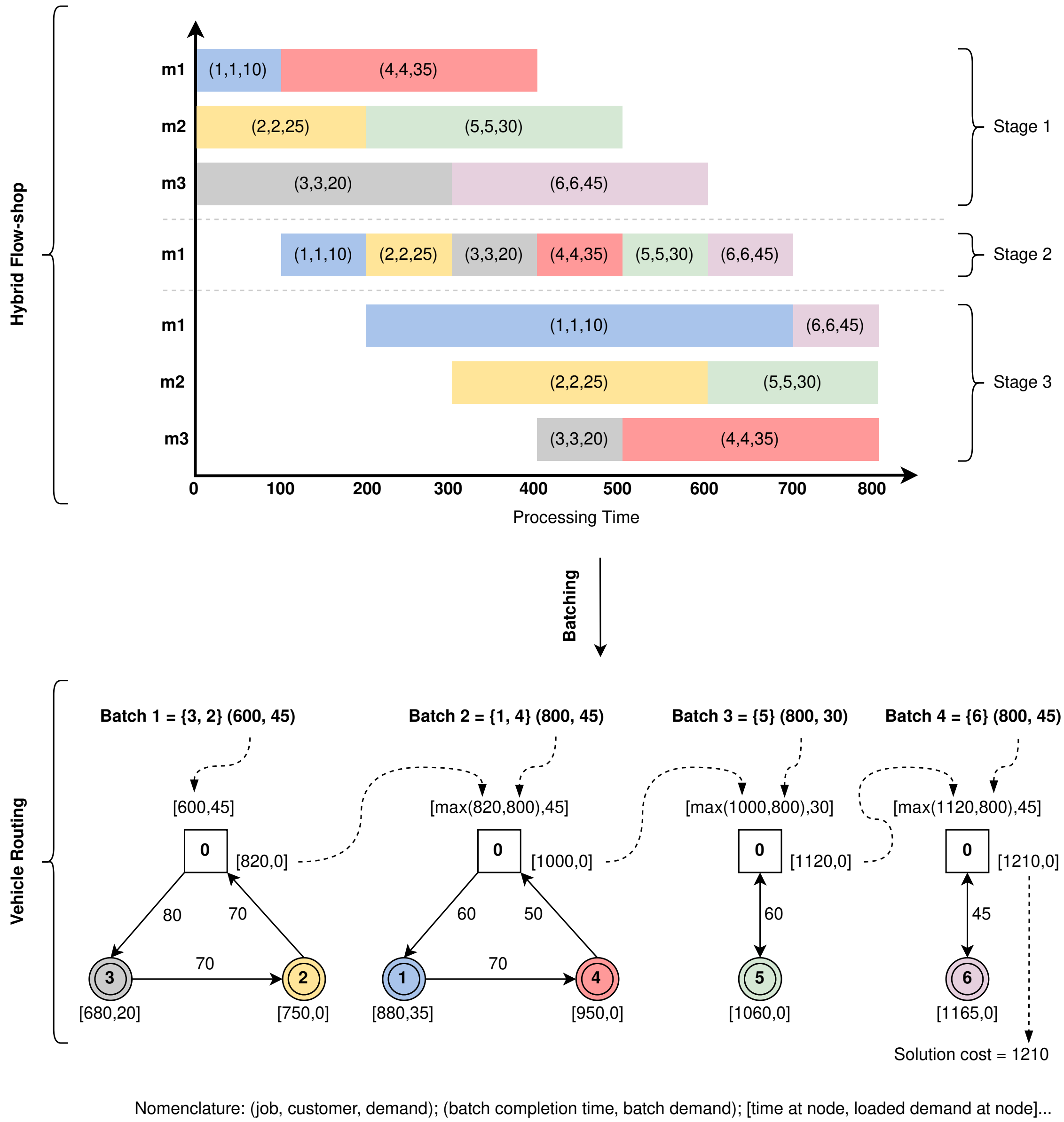

Figure 2: Numerical Example of of HFS-VRP Problem, the Combined Production and Distribution Operations.

(ii) The $LB_{HFS-VRP_{part2}}$ (Equation 42), which is obtained by the aggregation of a proposed lower bound for the multi-trip single VRP and the minimum summation of processing times across all jobs.

$$LB_{HFS-VRP} = \max\{LB_{HFS-VRP_{part1}}, LB_{HFS-VRP_{part2}}\} \tag{33}$$

As stated, Haouari and Hidri (2008) proposed a HFS lower bound. This bound summed with the smallest traveling time from the factory to a customer gives a possible lower bound for the HFS-VRP (34). Equations 35-41, which were taken from Haouari and Hidri (2008), supports the calculation of $LB_{HFS-VRP_{part1}}$.

$$LB_{HFS-VRP_{part1}} = LB_{HFS} + \min_{c \in C}\{TT_{0,c}\} \tag{34}$$

$$LB_{HFS} = \max_{2 \leq s \leq |S|}\{LB_s^{'}\} \tag{35}$$

$$LB_s^{'} = JL_{1,s-1} + \frac{SPT_{s-1}(|M_s|) + \sum_{j \in J} P_{j,s} + \sum_{k \in M_s} JR_{k,s}}{|M_s|} \tag{36}$$

$$LS_{j,s} = \begin{cases} \sum_{k=1}^{s-1} P_{j,k} & \text{if } j \in J, s > 1 \\ 0 & \text{if } j \in J, s = 1 \end{cases} \tag{37}$$

$$RS_{j,s} = \begin{cases} \sum_{k=s+1}^{|S|} P_{j,k} & \text{if } j \in J, j < s \\ 0 & \text{if } j \in J, s = |S| \end{cases} \tag{38}$$

$JL_{l,s}$: the lth smallest value of $LS_{j,s}$ (39)

$JR_{l,s}$: the lth smallest value of $RS_{j,s}$ (40)

$SPT_{l,s}(k)$: the minimum-sum of completion times of the k smallest $(s-1)$-stage jobs $LS_{j,s}$ (41)

The lower bound that we propose for multi-trip single VRP $LB_{HFS-VRP_{part2}}$ (42) is constructed considering that:

(i) the total departing times from the depot to the first customer of the trips should be at least the minimum distance from the factory to a customer multiplied by the number of trips.

(ii) the total traveling time of the vehicle should be at least the minimum travel time from the factory to a customer multiplied by two times the number of trips (departure and return of each trip). Nevertheless, this multiplication considers the last return to the factory, thence this distance should be subtracted once. Therefore, the total traveling time of the vehicle should be at least two times the minimum travel time from the factory to a customer times the minimum number of trips minus the minimum travel time from the factory to a customer.

(iii) the number of arcs visited between customers (that does not include the arcs that connect with the depot) is at least the number of customers minus the minimum number of trips $|C|-MNT$. Thence, the total traveling time across arcs is at least the sum of $|C|-MNT$ smallest distances between customers.

(iv) if the vehicle only has to do only one trip, then the traveling time is at least the sum of the minimum travel time from the factory to a customer with the $|C|-1$ smallest distances between customers and with the LB_{HFS}.

Equations (43)-(46) support the calculation of $LB_{HFS-VRP_{part2}}$.

$$LB_{HFS-VRP_{part2}} = \begin{cases} \min_{c\in C} TT_{0,c} + \sum_{i=1}^{|C|-1} STTO_i + LB_{HFS} & \text{if } MNT = 1 \\ 2 \cdot MNT \cdot \min_{c\in C} TT_{0,c} - \min_{c\in C} TT_{0,c} + \min_{j\in J} SUMPT_j & \text{if } MNT \geq |C|, MNT > 1 \\ 2 \cdot MNT \cdot \min_{c\in C} TT_{0,c} - \min_{c\in C} TT_{0,c} + \sum_{i=1}^{|C|-MNT} STTO_i + \min_{j\in J} SUMPT_j & \text{if } |C| > MNT > 1 \end{cases} \tag{42}$$

$$MNT = \frac{\sum_{j\ inJ} q_j}{Q}\text{: the minimum trips of the vehicle} \tag{43}$$

$$STT_c = \min_{h\in C, h\neq c} \{TT_{c,h}\} \tag{44}$$

$$STTO_l\text{: the } l\text{-th smallest traveling time of } STT \text{ values} \tag{45}$$

$$SUMPT_j = \sum_{s\in S} P_{j,s} \tag{46}$$

5. BR-VND Algorithm

Since the HFS and the VRP are both *NP-hard* problems (Lenstra and Kan, 1981; Ruiz and Vázquez-Rodríguez, 2010), so it is the composed HFS-VRP. Therefore, the use of metaheuristic approaches becomes necessary to solve large-sized instances in reasonable computing times. Hence, we propose an algorithm that combines biased-randomization (BR) techniques (Gonzalez-Martin et al., 2012) with the well-known variable neighborhood descent (VND) framework. The latter is a variant of the variable neighborhood search (VNS) metaheuristic framework (Mladenović and Hansen, 1997). The VNS is an enhanced local search strategy that systematically explores the solution space by changing the neighborhood structure. The local optimum provided by one neighborhood structure is not necessarily the same as the one provided by another neighborhood structure. In this way, the search becomes more flexible by exploring different neighborhood structures (Burke et al., 2008). The VND starts by employing an initial structure N_1. The searching process continues until no further improvement is reached. Then, a new neighborhood structure, N_2, is explored. If a new local optimum is obtained, the VND returns back and starts again with N_1. Otherwise, it continues with the next neighborhood structure, N_3. This process goes on until the last neighborhood structure is reached. In our biased-randomized variable neighborhood descent (BR-VND) algorithm, we first create an initial solution, which is then iteratively improved by employing a set of neighborhood structures (Figure 1). More details on our algorithm are provided in the following subsections.

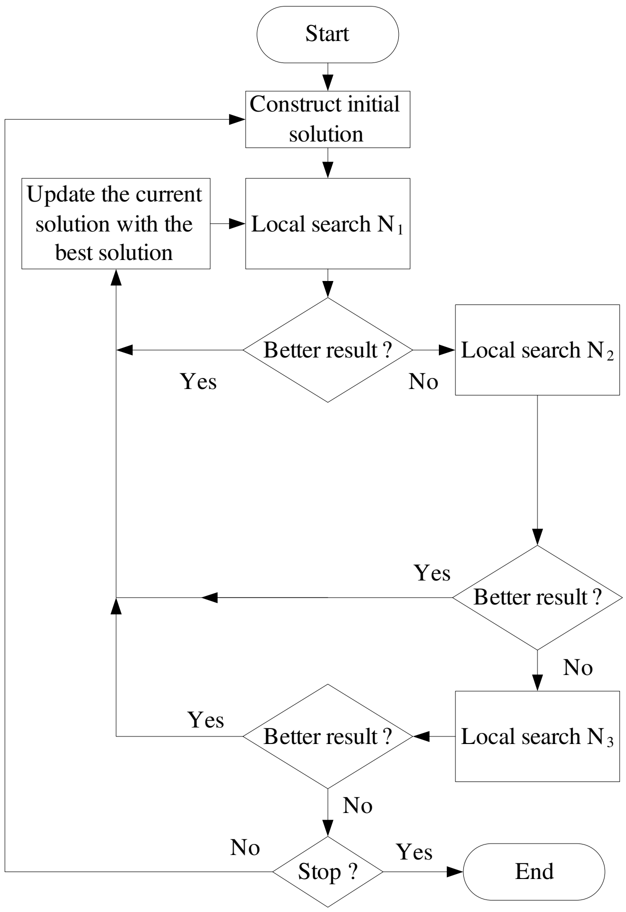

Figure 3: Flow-chart of the BR-VND algorithm.

5.1. Solution Representation and Loading Strategy

We consider two different solution representations. The first one, SR_1, is a complete sequence (permutation) of all jobs, and does not make any assumption about the assignment of jobs to customers. Hence, using this representation it is possible to consider $n!$ different permutations.

The second solution representation, SR_2, also employs a sequence of jobs. This time, however, jobs belonging to the same customer appear together in the sequence.

The BR-VND starts by generating an initial solution. We consider biased-randomized versions of the following constructive heuristics to generate this initial solution: the NEH heuristic (BR-NEH); the short processing time bottleneck heuristic (BR-SPTB); and the backward largest processing time bottleneck heuristic (BR-bLPTB). Each constructive heuristic is applied to solution representations SR_1 and SR_2. In order to load batches of jobs on a vehicle with limited capacity, we consider two different vehicle loading strategies. In the first one, VLS_1, jobs are loaded by increasing order of completion times. Let us consider, for example, a single vehicle with a maximum capacity of 50 unit per trip, and 5 jobs $(j_1, j_2, \ldots, j_5)$ with the following completion times (second element in the list) and volume capacities (third element in the list): $\{j_1, 38, 15\}$, $\{j_2, 24, 35\}$, $\{j_3, 31, 5\}$, $\{j_4, 20, 10\}$, and $\{j_5, 15, 30\}$. Thus, VLS_1 will determine the following loading plan of jobs: $\{j_5, j_4\}$ in trip 1, $\{j_2, j_3\}$ in trip 2, and $\{j_1\}$ in trip 3. In the second loading strategy, VLS_2, the goal is to load the maximum possible volume in each trip. Hence, when applying this second loading strategy to the previous numerical example, the loading plan will be as follows: $\{j_5, j_4, j_3\}$ in trip 1, $\{j_2\}$ in trip 2, and $\{j_1\}$ in trip 3. In order to investigate the effects of different initial solutions ($IniSol$) and loading strategies (VLS), we design twelve variants of the algorithm. These variants employ the same solution representation and have the same neighborhood structures, but they use different initial solutions and loading strategies.

5.2. *Generating an Initial Solution*

For the generation of the initial solution ($IniSol$), we propose the implementation of simple dispatching rules, such as the shortest processing time (SPT) and longest processing time (LPT) ones, which are adapted to the HFS problem. Both the SPT and the LPT generate job permutations that are based on sorting the total processing time of jobs according to an ascending and a descending order, respectively. Once all operations of one job are completed on the production phase, the job can be delivered to its demanding customer. Therefore, a vehicle is loaded and routed. For the routing process, we employ a biased-randomized version of the popular savings heuristic (Quintero-Araujo et al., 2017). The biased-randomization processes employed in this paper, both during the scheduling and the routing phases, make use of Geometric probability distributions, as proposed in (Ferrer et al., 2016) for the scheduling and in Gonzalez-Martin et al. (2018) for the routing.

5.2.1. The BR-NEH Heuristic

The first and second initial solutions are generated through the biased-randomized version of the NEH heuristic (Nawaz et al., 1983). The extension of the NEH to the HFS problem provides good results (Naderi et al., 2010). In the first initial solution, $BR - NEH1$, the biased-randomization process is applied to a job sequence β in order to generate a job sequence π. Then, a 'shift-to-left' operator (Juan et al., 2014) is used to improve the current solution. In the second initial solution, $BR - NEH2$, the NEH heuristic and a biased-randomization processes are jointly applied to all jobs belonging to each customer $c \in C$. Hence, a partial job sequence π_c is constructed for each customer $c \in C$ via the biased-randomized NEH heuristic. Next, the list of customers is sorted by ascending order of their jobs' makespan. A complete job sequence, π_T, is obtained by putting together all the partial job sequences.

5.2.2. The BR-SPTB Heuristic

The idea of the third and fourth initial solutions make use of a biased-randomized version of the short processing time bottleneck heuristic proposed by Pan et al. (2014). In many cases, bottlenecks in a system are generated by a single component (Liao et al., 2012). As Paternina-Arboleda et al. (2008) mentioned, a stage is a bottleneck when it has the largest flow ratio between the workload and the total available capacity. The SPTB heuristic sorts jobs by their total processing time, from the first stage to the bottleneck one. To generate the third initial solution, *BR-SPTB1*, the biased-randomized process is applied to the job sequence. The fourth initial solution, *BR-SPTB2*, is similar to the second one –it also works separately with the jobs that belong to each customer. The only difference is that in *BR-SPTB2* the partial job sequence for each customer is obtained via a biased-randomized version of the SPTB heuristic.

5.2.3. The BR-bLPTB Heuristic

The last two initial solutions are the *BR-bLPTB1* and *BR-bLPTB2*, which make use of biased-randomized version of the bLPTB heuristic. In the *BR-bLPTB1*, the jobs are sorted by their total processing time, in descending order, from the bottleneck stage to the last stage. In the *BR-bLPTB2*, the biased-randomized heuristic is applied to jobs belonging to each customer.

5.3. Neighborhood Structures

Our proposed BR-VND algorithm employs three neighborhood structures for the first solution representation, SR_1, and two for the second solution representation, SR_2.

5.3.1. The SR_1 Neighborhood Structures

Pseudo-codes 1 to 3 show the three proposed neighborhood structures. The first neighborhood for SR_1 is referred to as LS_{C1} and attempts to improve the objective function by examining different complete job sequences. LS_{C1} provides a list of complete job sequences by removing a single job from π_T and inserting it into all possible $n-1$ positions of π_T. The newly created job sequences are evaluated by assigning the jobs to the machines on the stages. If a new sequence provides a better objective function, then π_T is updated and all jobs are reinserted again. Otherwise, the search continues with the next job. The second neighborhood for this solution representation, LS_{P1}, works with partial job sequences: given a machine g in a stage k, it takes all jobs in π_{kg} and inserts them, considering all possible positions, both in the same as well as in any other machines at the stage k. When all jobs in machine g are considered, the search is continued with the next machine in stage k and, once these have been covered, with the machines in the next stage. The last neighborhood for SR_1, LS_{CS1}, is similar to LS_{C1}. It works over the jobs on a complete job sequence, π_T. The complete job sequence for each stage k, $\pi_{T(k)}$ is constructed. It extracts and reinserts each job into all possible $n-1$ positions of $\pi_{T(k)}$.

5.3.2. The SR_2 Neighborhood Structures

The first neighborhood proposed for the SR_2 solution representation, LS_{CS2}, works over the jobs that belong to the same customer: given an entire sequence π_T, it extracts all jobs associated with each customer as a block, and insert this block in all possible positions of π_T. The second neighborhood designed for SR_2, LS_{C2}, is similar to LS_{C1} and works over the jobs on the complete job sequence π_T. However, in the case of LS_{C2}, all jobs belonging to a customer $c \in C$ are extracted and inserted into all possible positions of the partial sequence associated with c.

Algorithm 1 Neighborhood structures, LS_{C1}.

```
l = 1
while l ≤ n do
  - Remove job a located at position l of π_T
  - Insert a into all n − 1 possible positions of π_T
  - Evaluate all obtained π_T by assigning jobs to machines of the stages
  if a better objective function is obtained then
    - update π_T
  else
    l = l + 1
  end if
end while
```

6. Computational Experiments

6.1. Generation of Instances

As mentioned before, this is the first time that a combined hybrid flow-shop and vehicle routing problem is discussed in the scientific literature. So, no benchmark instances are available in the literature to experimentally evaluate the proposed solution approaches. Hence, a new set of instances, which are based on some well-known benchmark instances of both the HFS and the VRP, is introduced, by considering the four instance factors listed in Table 1.

		Instance type			
		Small		Large	
Instance factor	Symbol	Number of levels	Values	Number of levels	Values
Number of jobs	n	3	6, 8, 10	3	60, 80, 100
Number of stage	s	3	2, 3, 4	3	5, 8, 10
Number of customer	c	3	2, 3, 4	8	16, 19, 20, 22, 23, 31, 32, 33
Vehicle loading capacity	v	2	20, 30	2	100, 200

Table 1: Instance factor for the small and large instances.

The number of identical parallel machines at each stage $k \in S$, m_k is generated using a uniform distribution $U[1, 5]$. Processing times of jobs on the HFS section are fixed to be integer

Algorithm 2 Neighborhood structures, LS_{P1}.

```
k = 1
while r ≤ s do
  g = 1, w ∈ M_k, w ≠ g
  while g ≤ m_k do
    j = 1
    while j ≤ |N(π_kg)| do
      - Remove job a located at position j of π_kg
      - Insert a into all possible positions of current π_kg and other ρ_kw
      if a better objective function is obtained then
        - update π_kg and other π_kw
      else
        j = j + 1
      end if
    end while
    g = g + 1
  end while
  k = k + 1
end while
```

Algorithm 3 Neighborhood structures, LS_{CS1}.

```
k = 1, t ∈ s, t ≥ k
while k ≤ s do
  l = 1
  while l ≤ n do
    Obtain complete job sequence for stage k, π_T(k)
    - Remove job a located at position l of π_T(k)
    - Insert a into all n − 1 possible positions of π_T(k)
    - Evaluate all obtained π_T(k) by assigning jobs to machines of the stages t
    if a better objective function is obtained then
      - update partial job sequences on machines at stage k and stages t
    else
      l = l + 1
    end if
  end while
  k = k + 1
end while
```

values from a uniform distribution $U[1, 99]$, as commonly defined in the scheduling literature. The volume capacity of each job $j \in N$, l_j is uniformly generated in the range of $U[5, 10]$ for small instances and $U[10, 30]$ for large instances. Since the customers should place in a certain geographic location, we have used some well-known set of benchmarks in the VRP literature. Eight VRP instances have been selected from a set of instances A, B, E and P, available at `http://vrp.atd-lab.inf.puc-rio.br/index.php/en/`. Regarding the number of the jobs, we have selected some acceptable instances where $n > c$. These instances are different among their scattered or clustered topology. In the case of small instances, the customer data was taken from the first customers of VRP mentioned instances. In particular, for small instances type, one test instance was generated for each combination of n, s, c, and v, obtaining a total of 54 small-sized instances. On the other hand, for large-sized instances, five test instances were created for each combination of n, s, c, and v, leading to a total of 720 large instances.

6.2. Results of Small Instances

The MILP model presented in section 3 was implemented in GLPK language with stooping criteria of 3600 seconds. Table 2 shows the results of the makespan of the best integer solution found after 3600s of running. As it can be seen in 75.92% of the small problem instances, i.e., 41 of the 54, no integer solution was found after one hour of execution. From the 13 instances in which an integer solution could be found, 9 of them were optimal. The table also presents the results of our proposed lower bound ($LB_{HFS-VRP}$), the percentage that the proposed LB is below the optimal value (LB_{Dev}) (47), the minimum value found by our BR-VND algorithm (Min_{BR-VND}), the minimum GAP of the BR-VND in comparison with the MILP model ($GAP_{BRVND_{MILP}}$) (48), and the minimum GAP of the BR-VND in comparison with the proposed lower bound ($GAP_{BRVND_{LB}}$) (49).

$$LB_{Dev} = \frac{LowerBound_{sol} - MILP_{sol}}{MILP_{sol}} \cdot 100\% \tag{47}$$

$$GAP_{BR-VND_{MILP}} = \frac{BRVND_{sol} - MILP_{sol}}{MILP_{sol}} \cdot 100\% \tag{48}$$

$$GAP_{BR-VND_{LB}} = \frac{BRVND_{sol} - LB_{HFS-VRP_{sol}}}{LB_{HFS-VRP_{sol}}} \cdot 100\% \tag{49}$$

In the 13 instances with integer solution found by the MILP model, the $LB_{HFS-VRP}$ is below the MILP result in an average of 32.83% and the $GAP_{BR-VND_{MILP}}$ is in average 8.22%. Specifically, for the problem instance $n = 6, s = 2, v = 30, c = 2$, the BR-VND found the optimal solution, and for the problem instance $n = 6, s = 3, v = 20, c = 4$, the BR-VND found a better solution than the best integer solution reached by the MILP model after an hour of execution.

For each of the 54 problem instances presented in Table 2, it is shown the minimum value found by our BR-VND algorithm (Min_{BR-VND}), resulted from the twelve different algorithm combinations. Since each instance is executed 5 times for each proposed algorithm, 3240 executions have been performed. The CPU times are not reported as they are so small. As a matter of fact, among the 3240 observed CPU times in the results, the maximum reported is 1.5 seconds. The average observed CPU time in all results is only 0.06 seconds.

n	s	v	c	$MILP$	**Time(s)**	$LB_{HFS-VRP}$	LB_{Dev}	Min_{BRVND}	$GAP_{BRVND_{MILP}}$	$GAP_{BRVND_{LB}}$
	2	20	2	282.44*	12.1	176.22	-37.61	296.29	4.90	68.14
	2	20	3	396.02*	81.3	239.66	-39.48	416.63	5.21	73.84
	2	20	4	370.40*	155.3	202.66	-45.29	442.3	19.41	118.25
	2	30	2	451.07*	65.6	437.22	-3.07	451.07	0.00	3.17
	2	30	3	272.81*	112.3	179.66	-34.15	291.9	7.00	62.48
	2	30	4	311.41	3600.0	206.00	-33.85	347.18	11.49	68.54
	3	20	2	-	3600.0	348.10	-	433.65	-	24.58
	3	20	3	-	3600.0	381.22	-	468.5	-	22.9
6	3	20	4	492.88	3600.0	276.22	-43.96	486.41	-1.31	76.1
	3	30	2	-	3600.0	303.22	-	542.07	-	78.77
	3	30	3	294.24*	12.0	182.00	-38.15	357.77	21.59	96.58
	3	30	4	421.28	3600.0	372.22	-11.65	424.03	0.65	13.92
	4	20	2	388.58*	24.7	254.22	-34.58	427.65	10.05	68.22
	4	20	3	-	3600.0	359.22		481.43	-	34.02
	4	20	4	-	3600.0	306.66		500.24	-	63.13
	4	30	2	346.07*	3.6	229.00	-33.83	404.21	16.80	76.51
	4	30	3	388.90*	89.2	222.00	-42.92	423.23	8.83	90.64
	4	30	4	-	3600.0	467.00	-	537.5	-	15.1
	2	20	2	-	3600.0	282.10	-	534.22	-	89.37
	2	20	3	-	3600.0	399.54	-	534.03	-	33.66
	2	20	4	-	3600.0	301.10	-	470.02	-	56.1
	2	30	2	-	3600.0	194.66	-	385.54	-	98.06
	2	30	3	-	3600.0	337.22	-	417.13	-	23.7
	2	30	4	-	3600.0	210.66	-	482.07	-	128.84

Continued on next page

*Optimal solution

Table 2 – continued from previous page

n	s	v	c	$MILP$	**Time(s)**	$LB_{HFS-VRP}$	LB_{Dev}	Min_{BRVND}	$GAP_{BRVND_{MILP}}$	$GAP_{BRVND_{LB}}$
	3	20	2	-	3600.0	312.10	-	448.73	-	43.78
	3	20	3	-	3600.0	238.66	-	450.2	-	88.64
8	3	20	4	-	3600.0	371.10	-	530.36	-	42.92
	3	30	2	-	3600.0	231.66	-	434.69	-	87.64
	3	30	3	-	3600.0	320.22	-	385.24	-	20.31
	3	30	4	-	3600.0	251.66	-	381.06	-	51.42
	4	20	2	520.51	3600.0	373.22	-28.30	532.24	2.25	42.61
	4	20	3	-	3600.0	470.22	-	584.38	-	24.28
	4	20	4	-	3600.0	353.10	-	620.77	-	75.81
	4	30	2	-	3600.0	384.22	-	473.43	-	23.22
	4	30	3	-	3600.0	317.66	-	527.48	-	66.05
	4	30	4	-	3600.0	281.22	-	426.87	-	51.79
	2	20	2	-	3600.0	382.54	-	565.22	-	47.76
	2	20	3	-	3600.0	390.54	-	632.55	-	61.97
	2	20	4	-	3600.0	407.54	-	587.16	-	44.08
	2	30	2	-	3600.0	285.22	-	632.22	-	121.66
	2	30	3	-	3600.0	324.22	-	427.24	-	31.78
	2	30	4	-	3600.0	234.22	-	433.61	-	85.13
	3	20	2	-	3600.0	567.22	-	596.07	-	5.09
	3	20	3	-	3600.0	417.54	-	553.4	-	32.54
	3	20	4	-	3600.0	533.22	-	640.28	-	20.08
	3	30	2	-	3600.0	657.22	-	663.22	-	0.91
10	3	30	3	-	3600.0	347.22	-	468.43	-	34.91
	3	30	4	-	3600.0	328.22	-	482.73	-	47.08
	4	20	2	-	3600.0	528.22	-	593.22	-	12.31
	4	20	3	-	3600.0	458.54	-	660.51	-	44.05
	4	20	4	-	3600.0	559.22	-	714.47	-	27.76
	4	30	2	-	3600.0	294.66	-	468.69	-	59.06
	4	30	3	-	3600.0	612.22	-	620.22	-	1.31
	4	30	4	-	3600.0	624.22	-	779.88	-	24.94
Average					3010.3		-32.83		8.22	51.95
										*Optimal solution

Table 2: Results of MILP and proposed LB for small instances.

6.3. Results of Large Instances

An experimental design was carried out to test the performance of the proposed algorithms. The experiment has considered the factors n, s, v, c, $IniSol$, SR, and VLS. The levels considered for the factors n, s, v, and c were presented in Table1 for the large-sized instances. The levels of $IniSol$, SR, and VLS were those presented in Table 3. Therefore, the treatments of the experiment were 1728 and the observations per treatment were 25. Considering that the BR-VND is a stochastic algorithm, each one of the 720 large instances was run for five different replications. Thence, there was a total of 25 observations per treatment, since 5 instances were generated for each combination of n, s, v, and c, and each one of them was run 5 times. This represented a total of $43,200$ replications for the experiment. Each replication was limited to $40 \times n \times s$ milliseconds of running as stopping criteria. For each replication, we have calculated the GAP as $GAP = ((Algorithm_{sol} - LowerBound_{sol})/LowerBound_{sol})$ where $Algorithm_{sol}$ is the solution obtained by a given algorithm and $LowerBound_{sol}$ is the lower bound obtained by applying the calculations presented on Section 4 for the corresponding instance.

Table 3: Test factors for instances.

Test factor	Symbol	Number of levels	Values
Initial solution	$IniSol$	3	BR-NEH, BR-SPTB, BR-bLPTB
Solution representation	SR	2	SR_1, SR_2
Loading strategy	VLS	2	VLS_1 , VLS_2

Table 4 presents a summary of results on the average GAP of proposed algorithms in comparison with the proposed lower bound. The results are categorized by all the instance factors, n, s, v and c. As shown in Table 4, from the descriptive point of view, the algorithm that combines the second solution representation SR_2 with the second loading strategy LR_2 and the initial solution BR-bLPTB is able to provide better solutions than the other ones, with a GAP of 17.91%. Besides, the algorithm with the worst performance is the one that combines SR_1, LR_2, and BR-SPTB, with a GAP of 21.98%. The behavior of the instance size factors, presented in Table 4, show that the problem becomes easier to solve when increasing the number of jobs (n), number of stages (s), and vehicle capacity (v). In the case of the number of customers (c), the best performance is presented in instances with 31 customers.

Despite the overall average GAP being 19.75%, it should be highlighted that in 48.70% of the instances, the obtained average GAP is smaller than %5, and in 8.83% of the instances, the average GAP is between 5% and 10%.

The algorithms' CPU time consumption for large instances is summarised in Table 5, grouped

Table 4: Average GAP over the proposed lower bound, grouped by instance characteristics.

		SR_1						SR_2					
		VLS_1			VLS_2			VLS_1			VLS_2		
		BR-NEH	BR-SPTB	BR-bLPTB	BR-NEH	BR-SPTB	BR-bLPTB	BR-NEH	BR-SPTB	BR-bLPTB	BR-NEH	BR-SPTB	BR-bLPTB
	60	23.31	22.46	23.54	23.83	22.94	24.17	21.05	21.15	20.84	21.08	21.2	20.98
n	80	20.61	19.50	20.67	21.15	19.83	21.21	17.19	17.62	17.32	16.85	17.24	17.00
	100	20.22	19.21	20.26	20.52	19.23	20.55	16.40	16.99	16.66	15.89	16.42	16.14
	5	26.38	25.05	26.54	27.07	25.45	27.40	21.61	21.96	21.57	21.39	21.68	21.34
s	8	21.82	20.94	21.9	22.18	21.21	22.24	18.95	19.31	18.97	18.63	18.99	18.73
	10	15.93	15.18	16.03	16.24	15.34	16.29	14.07	14.49	14.28	13.81	14.19	14.05
v	100	24.08	23.06	24.12	24.26	22.84	24.30	20.15	20.71	20.25	19.43	19.93	19.57
	200	18.68	17.72	18.85	19.4	18.50	19.66	16.27	16.46	16.29	16.45	16.65	16.51
	16	13.24	12.90	13.36	13.77	13.15	13.66	11.46	11.85	11.68	11.53	11.94	11.80
	19	12.32	12.00	12.37	12.80	12.08	12.91	10.58	10.68	10.56	10.66	10.68	10.61
	20	13.86	13.46	14.15	14.33	13.63	14.37	12.18	12.38	12.21	12.23	12.34	12.12
c	22	12.23	11.49	11.93	12.66	11.64	12.16	10.25	10.64	10.40	10.26	10.53	10.37
	23	48.91	46.25	49.27	49.16	46.59	49.40	41.42	42.23	41.62	40.15	40.82	40.45
	31	10.16	9.84	10.42	10.70	10.50	11.08	9.19	9.36	9.25	9.43	9.65	9.53
	32	31.68	30.04	31.93	33.41	31.32	34.24	24.76	25.50	24.7	24.77	25.75	25.07
	33	28.63	27.12	28.48	27.81	26.43	28.01	25.83	26.07	25.79	24.49	24.60	24.38
	Average	21.38	20.39	21.49	21.83	20.67	21.98	18.21	18.59	18.27	17.94	18.29	18.04

by the instance characteristics. The algorithms that use the first solution representation (SR_1) use almost three times more CPU time than the algorithms that use the alternative solution representation (SR_2). The algorithm with BR-SPTB as the initial solution, SR_1 as solution representation, and VLS_2 as loading strategy, consumes an average of 42.18 seconds, the longest CPU time consumption compared to other algorithms. Moreover, notice how the CPU times clearly depend on the size of the instance (number of jobs n, number of stages s, and number of customers c).

In order to determine if there is a significant statistical difference among the results of Table 4, a multifactor Analysis of Variance (ANOVA) was also carried out. The response variable is the GAP, and the control variables are n, s, v, c, $IniSol$, SR and VLS. We tested the three assumptions of ANOVA that are normality, homoscedasticity, and independence of residuals.

Since in this experiment the hypotheses of normality and homoscedasticity of samples were not fulfilled, we have performed an ANOVA-Type statistic (Brunner et al., 1997), which is a rank-based test that does not consider the assumptions of normality and homoscedasticity. According to the ANOVA-Type, all main effects are statistically significant with p-values very close to zero (lower than 0.001). Moreover, 18 of the 21 double interactions were significant. Specifically, regarding solution methods, the interaction between VLS and SR, and the interaction between $IniSol$ and SR, were statistically significant. According to the confidence intervals of rankings, with a confidence level of 95%, the best initial solution is $BR-NEH$, the best solution representation is $SR2$, and the best loading strategy is $VLS1$.

As it is known, not necessarily the combination of the best levels of factors leads to the best results. Then, the performance of the algorithm using a different combination of these factors is also studied. This combination generates twelve different algorithm configurations. In order to determine which configuration performs better, it is necessary to carry out the analysis of the triple interaction of factors $IniSol$, SR, and VLS. According to the confidence intervals of ranks for the twelve solution methods, obtained from the ANOVA-Type statistic, all of the combinations that consider the $SR2$ as solution representation present, statistically, the best performance. Figure 4 present the 95% Tukey confidence intervals for these configurations.

Table 5: CPU times (seconds) of proposed algorithms for the large instances.

		SR_1						SR_2					
		VLS_1			VLS_2			VLS_1			VLS_2		
		BR-NEH	BR-SPTB	BR-bLPTB	BR-NEH	BR-SPTB	BR-bLPTB	BR-NEH	BR-SPTB	BR-bLPTB	BR-NEH	BR-SPTB	BR-bLPTB
	60	15.91	17.06	16.22	17.28	18.13	15.82	7.61	8.22	7.01	8.58	8.38	7.23
n	80	30.15	32.99	31.04	33.82	35.10	30.73	11.03	11.43	9.26	11.33	11.68	9.73
	100	60.99	67.90	61.01	68.41	72.89	59.73	14.03	14.35	12.23	15.15	15.00	12.34
	5	21.03	23.71	21.52	23.84	24.99	21.00	6.80	6.80	5.75	07.09	7.30	5.46
s	8	36.39	39.44	36.12	40.05	42.37	34.74	11.18	11.60	9.69	12.24	11.72	10.19
	10	49.01	54.09	50.00	54.91	57.99	49.91	14.62	15.53	13.00	15.64	15.96	13.59
v	100	37.07	38.05	31.35	39.36	42.15	29.40	11.01	11.33	8.47	11.79	12.01	8.63
	200	33.84	40.12	40.48	39.83	41.40	41.14	10.72	11.29	10.50	11.52	11.30	10.88
	16	29.63	33.37	30.17	34.54	37.64	29.93	8.70	9.35	8.15	9.33	9.83	8.60
	19	29.89	34.11	29.48	35.22	38.01	29.99	10.06	9.70	8.30	10.56	10.46	09.07
	20	29.87	33.93	32.46	35.68	38.41	32.31	8.76	9.54	8.52	11.17	11.07	8.89
c	22	31.95	34.02	32.81	34.81	36.35	32.01	10.15	10.46	8.99	11.29	11.43	8.88
	23	43.46	46.37	38.14	46.15	45.51	36.86	12.57	13.12	10.46	13.24	12.57	10.14
	31	31.98	35.21	35.81	37.11	40.35	36.05	10.86	11.59	9.94	11.57	11.27	10.04
	32	34.07	39.25	37.27	40.61	44.92	38.30	10.99	11.95	10.23	12.05	12.82	11.05
	33	53.29	56.71	50.97	52.96	53.20	46.39	14.91	14.85	11.29	14.12	13.86	11.29
	Average	35.84	39.55	36.32	39.97	42.18	35.63	10.92	11.35	9.52	11.70	11.71	9.82

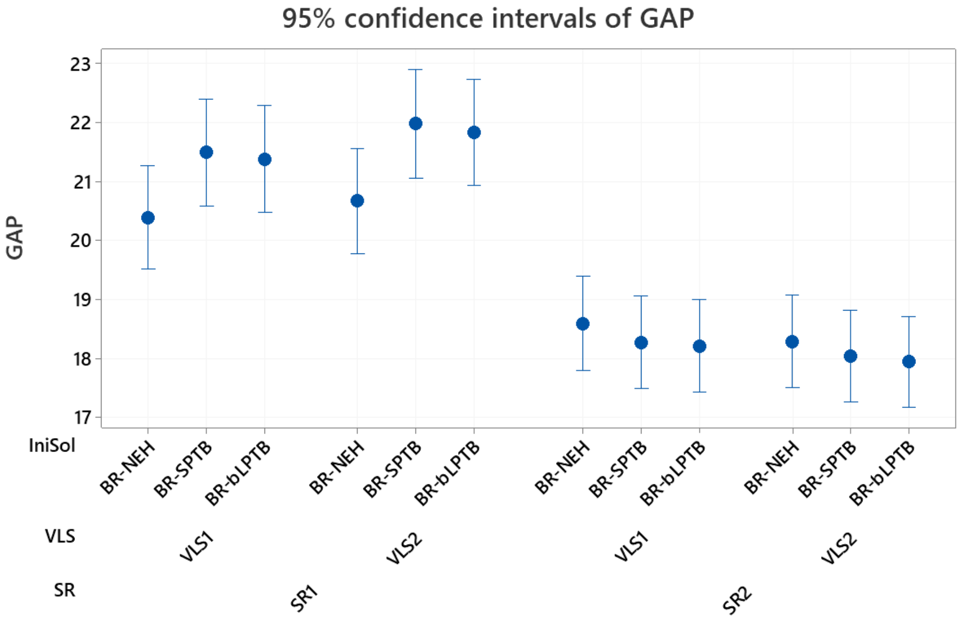

Figure 4: Means plot and 95% Tukey confidence intervals for different combinations of test factors

7. Conclusions and Future Work

This paper considered a combination of the Hybrid Flow Shop (HFS) scheduling problem with the Vehicle Routing Problem (VRP). To the best of our knowledge, this is the first time in the academic literature that this problem is approached. The problem, denoted as HFS-VRP, consists of a production section with HFS configuration to process jobs and a set of customers with a defined batch of jobs demand, followed by a distribution process where one capacitated vehicle is available to deliver the finished batches of jobs to the final customers. The optimization objective is the minimization of the service time to the last customer, i.e. the makespan of the joint problem. As pointed out, this problem is of practical relevance, for example, to distribute medical tests or vaccines to local health centers, so they can be administrated to the population as soon as possible, while these items are being produced, in large quantities, at a central laboratory.

To solve the problem, three stages were proposed. Firstly, the MILP model. Secondly, a first lower bound of the HFS-VRP problem. Thirdly, this paper proposed a Biased-Randomized Variable Neighborhood Descent (BR-VND) metaheuristic. Twelve different configurations of the algorithm, which consists of three methods of initial solutions, two solution representations, and two vehicle loading strategies, were developed. Since no benchmark data sets are not available, a complete set of instances was generated to test these configurations, inspired by existing benchmarks of HFS and VRP from the literature.

Computational evaluations were carried out in two phases. In the first one, the MILP model was executed for very small instances, in which 75% of them did not obtain an integer solution after 3600s of executions. The small instances that obtained a result after an hour of execution were compared with the proposed lower bound, obtaining that the lower bound is 32% lower, on average, than the objective function obtained for the best solution found by the MILP model. In the second phase, an experimental design was performed with 720 generated large instances, and the results were analyzed through the ANOVA statistical test. Seven factors, including four instance factors (number of jobs, stages, customers, and the capacity of the vehicle) and three test factors (initial solution, solution representation, and vehicle loading strategy) were considered in ANOVA as control factors. The response variable was the GAP versus the proposed lower bound. Results showed that all main effects are statistically significant. Due to the assumptions of the ANOVA were not fulfilled, the ANOVA-Type statistic was performed confirming the initial results given by the ANOVA. Particularly, instances with the highest level of stages ($s = 10$) presented the best GAP (14.99%). Also when the number of customers was set to $c = 31$, the average GAP was 9.92%. When the capacity of vehicles is 100, the GAP presents better performance than when it is set to 200. The computational analysis shows that BR-NEH initial solution, solution representation SR_2, and loading strategy VLS_1 perform statistically better than the others. It is important to note that, for 48.07% of the instances, the GAP was less than 5%.

Future work could be directed to incorporate various vehicles in the routing to test the best configuration, not only in terms of makespan but also including due date related measures. Of course, some other solution procedures can be proposed and evaluated.

Acknowledgments

This work has been partially supported by the Spanish Ministry of Science (PID2019-111100RB-C21, RED2018-102642-T) and the Erasmus+ program (2019-I-ES01-KA103-062602).

References

Armstrong, R., Gao, S., and Lei, L. (2008). A zero-inventory production and distribution problem with a fixed customer sequence. *Annals of Operations Research*, 159(1):395–414.

Brunner, E., Dette, H., and Munk, A. (1997). Box-Type Approximations in Nonparametric Factorial Designs. *Journal of the American Statistical Association*, 92(440):1494–1502.

Burke, E. K., Curtois, T., Post, G., Qu, R., and Veltman, B. (2008). A hybrid heuristic ordering and variable neighbourhood search for the nurse rostering problem. *European Journal of Operational Research*, 188(2):330 – 341.

Cakici, E., Mason, S. J., Geismar, H. N., and Fowler, J. W. (2014). Scheduling parallel machines with single vehicle delivery. *Journal of Heuristics*, 20(5):511–537.

Chen, Z.-L. (2004). Integrated Production and Distribution Operations. In Simchi-Levi, D., Wu, S. D., and Shen, Z.-J., editors, *Handbook of Quantitative Supply Chain Analysis*, chapter 17, pages 711–745. Springer US, New York.

Chen, Z.-L. (2010). Integrated Production and Outbound Distribution Scheduling: Review and Extensions. *Operations Research*, 58(1):130–148.

Chen, Z.-L. and Vairaktarakis, G. L. (2005). Integrated Scheduling of Production and Distribution Operations. *Management Science*, 51(4):614–628.

Cohen, M. and Mallik, S. (1997). Global supply chains: research and applications. *Production and Operations Management*, 6(3):193–210.

Condotta, A., Knust, S., Meier, D., and Shakhlevich, N. V. (2013). Tabu search and lower bounds for a combined production–transportation problem. *Computers & Operations Research*, 40(3):886–900.

Erengüç, Ş., Simpson, N., and Vakharia, A. J. (1999). Integrated production/distribution planning in supply chains: An invited review. *European Journal of Operational Research*, 115(2):219–236.

Farahani, P., Grunow, M., and Günther, H.-O. (2012). Integrated production and distribution planning for perishable food products. *Flexible Services and Manufacturing Journal*, 24(1):28–51.

Ferrer, A., Guimarans, D., Ramalhinho, H., and Juan, A. A. (2016). A BRILS metaheuristic for non-smooth flow-shop problems with failure-risk costs. *Expert Systems with Applications*, 44:177–186.

Fu, L.-L., Aloulou, M. A., and Triki, C. (2017). Integrated production scheduling and vehicle routing problem with job splitting and delivery time windows. *International Journal of Production Research*, 7543(April):1–16.

Geismar, H. N., Dawande, M., and Sriskandarajah, C. (2011). Pool-Point Distribution of Zero-Inventory Products. *Production and Operations Management*, 20(5):737–753.

Geismar, H. N., Laporte, G., Lei, L., and Sriskandarajah, C. (2008). The Integrated Production and Transportation Scheduling Problem for a Product with a Short Lifespan. *INFORMS Journal on Computing*, 20(1):21–33.

Gilmore, P. C. and Gomory, R. E. (1961). A linear programming approach to the cutting-stock problem. *Operations research*, 9(6):849–859.

Goetschalckx, M., Vidal, C. J., and Dogan, K. (2002). Modeling and design of global logistics systems: A review of integrated strategic and tactical models and design algorithms. *European Journal of Operational Research*, 143(1):1–18.

Gonzalez-Martin, S., Juan, A. A., Riera, D., Castella, Q., Muñoz, R., and Perez, A. (2012). Development and assessment of the SHARP and RandSHARP algorithms for the arc routing problem. *AI Communications*, 25(2):173–189.

Gonzalez-Martin, S., Juan, A. A., Riera, D., Elizondo, M. G., and Ramos, J. J. (2018). A simheuristic algorithm for solving the arc routing problem with stochastic demands. *Journal of Simulation*, 12(1):53–66.

Hajiaghaei-Keshteli, M. and Aminnayeri, M. (2014). Solving the integrated scheduling of production and rail transportation problem by Keshtel algorithm. *Applied Soft Computing*, 25:184–203.

Hajiaghaei-Keshteli, M., Aminnayeri, M., and Ghomi, S. F. (2014). Integrated scheduling of production and rail transportation. *Computers & Industrial Engineering*, 74:240–256.

Haouari, M. and Hidri, L. (2008). On the hybrid flowshop scheduling problem. *International Journal of Production Economics*, 113(1):495–497.

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

Kang, H.-Y., Pearn, W. L., Chung, I.-P., and Lee, A. H. I. (2016). An enhanced model for the integrated production and transportation problem in a multiple vehicles environment. *Soft Computing*, 20(4):1415–1435.

Karaoğlan, I. and Kesen, S. E. (2017). The coordinated production and transportation scheduling problem with a time-sensitive product: a branch-and-cut algorithm. *International Journal of Production Research*, 55(2):536–557.

Koç, U., Toptal, A., and Sabuncuoglu, I. (2017). Coordination of inbound and outbound transportation schedules with the production schedule. *Computers & Industrial Engineering*, 103:178–192.

Lenstra, J. K. and Kan, A. H. G. R. (1981). Complexity of vehicle routing and scheduling problems. *Networks*, 11(2):221–227.

Li, C.-L. and Vairaktarakis, G. (2007). Coordinating production and distribution of jobs with bundling operations. *Iie Transactions*, 39(2):203–215.

Li, C.-L., Vairaktarakis, G., and Lee, C. Y. (2005). Machine scheduling with deliveries to multiple customer locations. *European Journal of Operational Research*, 164(1):39–51.

Liao, C.-J., Tjandradjaja, E., and Chung, T.-P. (2012). An approach using particle swarm optimization and bottleneck heuristic to solve hybrid flow shop scheduling problem. *Applied Soft Computing*, 12(6):1755 – 1764.

Low, C., Chang, C.-M., Li, R.-K., and Huang, C.-L. (2014). Coordination of production scheduling and delivery problems with heterogeneous fleet. *International Journal of Production Economics*, 153:139–148.

Meixell, M. J. and Gargeya, V. B. (2005). Global supply chain design: A literature review and critique. *Transportation Research Part E: Logistics and Transportation Review*, 41(6):531–550.

Mladenović, N. and Hansen, P. (1997). Variable neighborhood search. *Comput. Oper. Res.*, 24(11):1097–1100.

Moons, S., Ramaekers, K., Caris, A., and Arda, Y. (2017). Integrating production scheduling and vehicle routing decisions at the operational decision level: A review and discussion. *Computers & Industrial Engineering*, 104:224–245.

Naderi, B., Ruiz, R., and Zandieh, M. (2010). Algorithms for a realistic variant of flowshop scheduling. *Computers & Operations Research*, 37(2):236 – 246.

Nawaz, M., Enscore, E. E., and Ham, I. (1983). A heuristic algorithm for the m-machine, n-job flow-shop sequencing problem. *Omega*, 11(1):91 – 95.

Olhager, J., Pashaei, S., and Sternberg, H. (2015). Design of global production and distribution networks. *International Journal of Physical Distribution & Logistics Management*, 45(1/2):138–158.

Pan, Q.-K., Wang, L., Li, J.-Q., and Duan, J.-H. (2014). A novel discrete artificial bee colony algorithm for the hybrid flowshop scheduling problem with makespan minimisation. *Omega*, 45(Supplement C):42 – 56.

Paternina-Arboleda, C. D., Montoya-Torres, J. R., Acero-Dominguez, M. J., and Herrera-Hernandez, M. C. (2008). Scheduling jobs on a k-stage flexible flow-shop. *Annals of Operations Research*, 164(1):29–40.

Quintero-Araujo, C. L., Caballero-Villalobos, J. P., Juan, A. A., and Montoya-Torres, J. R. (2017). A biased-randomized metaheuristic for the capacitated location routing problem. *International Transactions in Operational Research*, 24(5):1079–1098.

Ruiz, R. and Vázquez-Rodríguez, J. A. (2010). The hybrid flow shop scheduling problem. *European Journal of Operational Research*, 205(1):1 – 18.

Sarmiento, A. M. and Nagi, R. (1999). A review of integrated analysis of production-distribution systems. *IIE Transactions*, 31(11):1061–1074.

Stecke, K. E. and Zhao, X. (2007). Production and Transportation Integration for a Make-to-Order Manufacturing Company with a Commit-to-Delivery Business Mode. *Manufacturing & Service Operations Management*, 9(2):206–224.

Thomas, D. J. and Griffin, P. M. (1996). Coordinated supply chain management. *European Journal of Operational Research*, 94(1):1–15.

Vidal, C. J. and Goetschalckx, M. (1997). Strategic production-distribution models: A critical review with emphasis on global supply chain models. *European Journal of Operational Research*, 98(1):1–18.

Wang, D. Y., Grunder, O., and Moudni, A. E. (2015). Integrated scheduling of production and distribution operations: a review. *International Journal of Industrial and Systems Engineering*, 19(1):94–122.

Wang, K., Ma, W., Luo, H., and Qin, H. (2016). Coordinated scheduling of production and transportation in a two-stage assembly flowshop. *International Journal of Production Research*, 54(22):6891–6911.