sustainability

MDPI

Article

Reducing Carbon Emissions in a Closed-Loop Production Routing Problem with Simultaneous Pickups and Deliveries under Carbon Cap-and-Trade

Xuanjing Fang [1], Yanan Du [1] and Yuzhuo Qiu [1,2,*]

1 Jiangsu Key Laboratory of Modern Logistics, School of Marketing and Logistics Management, Nanjing University of Finance and Economics, Nanjing 210023, China; ahjxfxj@126.com (X.F.); yanan_du2017@163.com (Y.D.)

2 Department of Industrial and Systems Engineering, Faculty of Engineering, University of Florida, Gainesville, FL 32611, USA

* Correspondence: jadeqyz@gmail.com

Received: 2 October 2017; Accepted: 26 November 2017; Published: 28 November 2017

Abstract: The incorporation of reverse logistics into production routing problems can promote and coordinate the implementation of sustainable development for supply chains. This study aims to incorporate reverse logistics into production routing problems and investigate the reduction of carbon emissions under carbon cap-and-trade. Mixed-integer programming models are proposed for the production routing problem with reverse logistics by considering simultaneous pickups and deliveries in vehicle routing subproblems. To solve this problem, we propose a solution method of a branch-and-cut guided search algorithm based on adaptation of known valid inequalities. Computational results highlight the trade-offs among various performance indicators, including emission levels and operational costs of production, inventory holding, fuel consumption, and drivers.

Keywords: routing; production planning and control; branch and bound; environmental studies; carbon emission; carbon cap-and-trade

1. Introduction

In the promotion and coordination of sustainable development for supply chains, environmental concerns about production and logistic activities have greatly increased in recent years. Since integrated operations can help achieve the goal of reduced environmental harm while remaining operational effectiveness [1], supply chain optimization problems such as the production routing problem (PRP) that aim at optimal joint decisions of production, inventory, distribution, and routing, have recently received a considerable attention [2]. Production-related and distribution-related CO_2 emissions along supply chains have also been a hot topic ([3,4], respectively).

Besides integrating operations forward, closed-loop supply chain optimization exhibited a reduction in environmental impact [5]. Return flow processes in a closed-loop supply chain usually consist of: (1) product collection from consumers; (2) reverse logistics to take collected products back; (3) screening, assorting, and disposal to specify the most economically attractive reuse alternatives; (4) remanufacturing; and (5) remarketing to produce and utilize new markets [6]. The PRP can be extended naturally to involve reverse logistics.

After the importance of considering production, inventory, and routing decisions simultaneously was stressed in [7], the PRP was extended in various ways to consider, for example, multiple plants and heterogeneous fleets of vehicles [8], incapacitated production [9], multiple homogeneous capacitated vehicles [10], demand uncertainty [11], multi-item back-order [12], perishable products [13], and multiscale production [14] in the past decade. The environmental impact of the PRP has seldom been addressed, with only a few notable exceptions such as the PRP with carbon emissions [1], and the multi-objective pollution production routing problem with a time window [15]. However, reverse logistics, to the best of our knowledge, have been largely ignored.

Pickup-and-delivery problems for goods transportation and various available algorithms were reviewed in [16]. The vehicle routing problem with simultaneous pickups and delivery (VRPSPD) has become increasingly popular. The VRPSPD has wide applications in the electric appliances industry, beverage industry, returnable/reusable transport items (RTI), and returnable/reusable logistical packaging. New solution methods of exact algorithms, e.g., the branch-and-cut method [17], and branch-price-and-cut method [18,19] have just appeared. Inventory routing problems with simultaneous pickups and deliveries (IRPSPD) have also been explored only recently [6,20,21]. A natural step forward is to extend these problems and methods to the PRP.

Among various carbon policies, the "cap-and-trade" policy works for energy-intensive industries. The carbon cap specifies the upper limit, i.e., tons of carbon dioxide equivalents (CO_2e), that a company may emit per year. Under "cap-and-trade" policy, if the cap is exceeded by a company, the company must buy additional allowances. If a company has not met with the carbon cap, the company can sell carbon credits on the carbon trading market.

Our aim is thus to design a model and algorithm for a closed-loop production routing problem with simultaneous pickups and deliveries (PRPSPD) under carbon cap-and-trade. The PRP involves combinatorial optimization of both delivery and routing decisions. Exact algorithms, such as branch-and-price [1,22,23] and branch-and-cut [9,10], can solve small and medium-sized problems. Other studies have often used heuristics, e.g., approximation algorithms [24], the decoupled heuristic [25], the greedy randomized adaptive search procedure [26], memetic algorithms [27], tabu searches [28,29], adaptive large neighborhood searches [30], iterative mixed-integer programming [31], particle-swarm optimization [15], the mathematical programming heuristic [32], and the multiphase heuristic [33]. Thus, we intend to develop a hybrid algorithm for the PRPSPD under carbon cap-and-trade.

The contributions of this paper can be summarized as follows. First, we introduce a real-world variant of the PRP with reverse logistics. Reverse logistics are modeled with simultaneous pickups and deliveries. Second, we formulate the PRPSPD under carbon cap-and-trade as a mixed-integer linear programming (MILP) problem. Third, we adapt known valid inequalities to tighten the MILP formulation and design a branch-and-cut guided search algorithm as the solution method. Finally, we conduct extensive computational experiments to assess the performance of the proposed algorithm and develop managerial implications through sensitivity analysis. The model, algorithm, and computational results can serve as a stepping stone for further research of the PRP with return flow [2].

The rest of the paper is organized as follows. Section 2 describes the PRPSPD under carbon cap-and-trade and introduces a mathematical formulation. Section 3 elaborates a solution method of a branch-and-cut guided search algorithm. Extensive computational results are provided in Section 4. We conclude in Section 5 with discussions on future research directions.

2. Problem Description and Mathematical Formulation

2.1. Problem Description

We first describe the PRP with simultaneous pickups and deliveries (PRPSPD) without considering carbon emissions. The PRP with simultaneous pickups and deliveries (PRPSPD) is defined on a complete directed graph $G = (N_0, A)$, where the node set $N_0 = N \cup \{0\}$ consists of a set $N = \{1, 2, \ldots, n\}$ of customers and a depot represented by node 0, and the arc set is $A = \{(i, j) : i, j \in N_0, i \neq j\}$. Triangular inequality holds for transportation cost over each arc, i.e., $c_{ij} + c_{jk} \geq c_{ik}$. Over a finite set $T = \{1, 2, \ldots, |T|\}$ of planning periods, a finite set $K = \{1, 2, \ldots, |K|\}$

of homogeneous vehicles with capacity Q is available to serve the customers. In every period, each customer has known pickup and delivery demands.

In each period, products can be shipped to customers, while pickups can be collected simultaneously. Given initial inventory levels of new products and pickups at the depot and customers, the problem is to determine the product amount to manufacture at the depot, the pickup and delivery amount for each customer, and the set of routes in each period while minimizing total costs of production, inventory, and routing.

When we consider carbon emissions, the problem is to minimize total costs of production, inventory, routing, and emission-related costs by making the same decisions under cap-and-trade regulations on carbon emissions. The carbon cap in this paper is interpreted as carbon quota, (see e.g., [1]). Companies can thus exchange more emission permits if their expected emissions surpass their carbon caps. Similarly, companies with less expected emissions than carbon caps could exchange their surplus for benefits. This also implies that carbon cap appears only in objective functions in the following formulations.

To formulate the problem, we use following notations for parameters and decision variables.

Parameters:

- l_{ij}: length of arc (i,j) with $l_{ij} = l_{ji}$;
- τ_{ij}: arc-specific traveling time constant;
- α_{ij}: $= a + g\sin\theta_{ij} + gC_r\cos\theta_{ij}$, arc-specific constant, where a is the vehicle acceleration (m/s^2), g is the gravitational constant (9.81 m/s^2), θ_{ij} is the angle of the arc (i,j), and C_r are the coefficients of rolling resistance;
- β: $= 0.5C_d A\rho$, vehicle-specific constant, where C_d are the coefficients of drag, A is the frontal surface area of the vehicle, and ρ is the air density;
- ω: empty vehicle weight;
- p_c: market price per unit of carbon;
- c_e: unit carbon emission of fuel;
- c_f: unit cost of fuel energy;
- $\tilde{s}$: fixed emissions of production setup;
- $\tilde{c}_0$: emissions per unit of production;
- $\tilde{h}_i$: emissions per unit of inventory held at node $i \in N_0$;
- w_t: driver wage per unit time in period t;
- C: manufacturing capacity;
- Q: vehicle capacity;
- c^{f}: fixed manufacturing setup costs;
- c_{ij}: transportation cost over arc (i,j);
- δ_{it}: delivery demand of customer $i \in N$ in period t;
- π_{it}: pickup demand of customer $i \in N$ in period t;
- h_i^{d}: unit inventory holding cost of products at node $i \in N_0$;
- h_i^{p}: unit inventory holding cost of pickups at node $i \in N_0$;
- L_i^{d}: storage capacity for delivered products at node $i \in N_0$;
- L_i^{p}: storage capacity for pickup requests at node $i \in N_0$;
- I_{i0}^{d}: initial product inventory at node $i \in N_0$;
- I_{i0}^{p}: initial pickup inventory at node $i \in N_0$;
- B_t: $= \min\{C, L_0^{\mathrm{d}}, \sum_{\tau=t}^{|T|}\sum_{i\in N}\delta_{i\tau}\}$, maximum amount of product that can be produced in period t;
- M_{1it}: $= \min\{Q, L_i^{\mathrm{d}}, \sum_{\tau=t}^{|T|}\delta_{i\tau}\}$, maximum amount of product that can be delivered to customer i in period t;
- M_{2it}: $= \min\{Q, L_i^{\mathrm{p}}, I_{i0}^{\mathrm{p}} + \sum_{\tau=1}^{t}\pi_{i\tau}\}$, maximum amount of product that can be picked up at customer i in period t;
- E: maximum carbon emissions allowed in the planning horizon.

Decision Variables:

- m_t: manufacturing quantity in period t;
- I_{it}^{d}: product inventory at node i at the end of period t;
- I_{it}^{p}: pickup inventory at node i at the end of period t;
- d_{it}: delivery amount to customer i in period t;
- p_{it}: pickup amount at customer i in period t
- u_{ijt}: pickup amount over arc (i,j) in period t if arc (i,j) is traversed in period t, 0 otherwise;
- v_{ijt}: delivery amount over arc (i,j) in period t if arc (i,j) is traversed in period t, 0 otherwise;
- x_{ijt}: binary variable, equal to 1 if arc (i,j) is traversed in period t, 0 otherwise;
- y_t: binary variable, equal to 1 if the product is set up for production in period t, 0 otherwise;

2.2. An MILP Formulation for the PRPSPD

Given the notations in Section 2.1, the arc-flow-based formulation of the PRPSPD is as follows:

$$\text{minimize} \quad \sum_{t\in T}\left(c^{\mathrm{u}}m_t + c^{\mathrm{f}}y_t\right) \tag{1a}$$

$$+\sum_{t\in T}\sum_{i\in N_0}\left(h_i^{\mathrm{d}}I_{it}^{\mathrm{d}} + h_i^{\mathrm{p}}I_{it}^{\mathrm{p}}\right) \tag{1b}$$

$$+\sum_{t\in T}\sum_{(i,j)\in A} c_{ij}x_{ijt} \tag{1c}$$

Subject to

$$I_{0,t-1}^{\mathrm{d}} + m_t - \sum_{i\in N} d_{it} = I_{0t}^{\mathrm{d}}, \qquad \forall t\in T, \tag{2}$$

$$I_{0,t-1}^{\mathrm{p}} + \sum_{i\in N} p_{it} = I_{0t}^{\mathrm{p}}, \qquad \forall t\in T, \tag{3}$$

$$I_{i,t-1}^{\mathrm{d}} + d_{it} - \delta_{it} = I_{it}^{\mathrm{d}}, \qquad \forall i\in N, t\in T, \tag{4}$$

$$I_{i,t-1}^{\mathrm{p}} - p_{it} + \pi_{it} = I_{it}^{\mathrm{p}}, \qquad \forall i\in N, t\in T, \tag{5}$$

$$m_t \le B_t y_t, \qquad \forall t\in T, \tag{6}$$

$$I_{it}^{\mathrm{d}} \le L_i^{\mathrm{d}}, \qquad \forall i\in N_0, t\in T, \tag{7}$$

$$I_{it}^{\mathrm{p}} \le L_i^{\mathrm{p}}, \qquad \forall i\in N_0, t\in T, \tag{8}$$

$$\sum_{j\in N_0} x_{ijt} \le 1, \qquad \forall i\in N, t\in T, \tag{9}$$

$$\sum_{j\in N_0} x_{ijt} - \sum_{j\in N_0} x_{jit} = 0, \qquad \forall i\in N_0, t\in T, \tag{10}$$

$$\sum_{j\in N_0} x_{0jt} \le |K|, \qquad \forall t\in T, \tag{11}$$

$$\sum_{j\in N_0} v_{jit} - \sum_{j\in N_0} v_{ijt} = d_{it}, \qquad \forall i\in N, t\in T, \tag{12}$$

$$\sum_{j\in N_0} u_{ijt} - \sum_{j\in N_0} u_{jit} = p_{it}, \qquad \forall i\in N, t\in T, \tag{13}$$

$$v_{ijt} + u_{ijt} \le Qx_{ijt}, \qquad \forall (i,j)\in A, t\in T, \tag{14}$$

$$d_{it} \le M_{1it}\sum_{j\in N_0} x_{ijt}, \qquad \forall i\in N, t\in T, \tag{15}$$

$$p_{it} \le M_{2it}\sum_{j\in N_0} x_{ijt}, \qquad \forall i\in N, t\in T, \tag{16}$$

$$m_t \geq 0, y_t \in \{0,1\}, \qquad \forall t \in T, \tag{17}$$
$$I^{\mathrm{d}}_{it}, I^{\mathrm{p}}_{it} \geq 0, \qquad \forall i \in N_0, t \in T, \tag{18}$$
$$d_{it}, p_{it} \geq 0, \qquad \forall i \in N, t \in T, \tag{19}$$
$$u_{ijt}, v_{ijt} \geq 0, \qquad \forall (i,j) \in A, t \in T, \tag{20}$$
$$x_{ijt} \in \{0,1\}, \qquad \forall (i,j) \in A, t \in T. \tag{21}$$

The objective function (1a)–(1c) minimizes total operational costs, where (1a), (1b), and (1c) measure production, inventory, and routing costs, respectively. Constraints (2) and (3) guarantee product and pickup inventory flow balance at the depot, respectively. Constraints (4) and (5) ensure product and pickup inventory flow balance for the customers, respectively. Constraints (6) enforce that the setup binary variable is one, if the manufacturing amount is positive in each period. Constraints (6) also set the limit of the manufacturing amount to the lower value between total delivery demand in the remaining periods and manufacturing capacity. Constraints (7) and (8) stipulate that the product and pickup inventory should not exceed their corresponding capacity, respectively. Constraints (9) serve as degree constraints. These constraints ensure that each customer is visited at most once in every period, and are referred as degree constraints because of their origins in the traveling salesman and vehicle routing problems. Constraints (10) represent vehicle flow balance. Constraints (11) impose the number of vehicles available in each period. Constraints (12) and (13) are the flow conservation constraints for pickups and deliveries, respectively. Constraints (14) bind the product flow transportation over each arc with a maximum value for vehicle capacity. Constraints (15) specify that each customer is visited if the delivery quantity at the customer is nonzero. Constraints (16) enforce a restriction that each customer is visited if the pickup quantity at the customer is nonzero. Finally, constraints (17)–(21) introduce the model's decision variables.

2.3. Emission Models in PRPSPD and Formulation of PRPSPD under Carbon Cap-and-Trade

In this section, we firstly describe production and inventory-related emissions, and vehicle routing-related emissions. We then model PRPSPD under the carbon cap-and-trade regulatory mechanism by integrating the pollution routing model and the pollution lot-sizing model.

2.3.1. Production and Inventory-Related Emissions

Carbon emissions mainly originate from production, inventory, and routing decisions in a typical two-echelon supply chain. As for production and inventory-related emissions, we assume

$$e_{1t} = \tilde{c}_0 m_t + \tilde{s} y_t + \sum_{i \in N_0} \tilde{h}_i \left(I^{\mathrm{p}}_{it} + I^{\mathrm{d}}_{it} \right), \forall t \in T, \tag{22}$$

as in [1]. Given carbon price p_{c} and production and inventory-related carbon cap E_1, we can formulate a lot-sizing model under carbon cap-and-trade as follows:

$$\text{minimize} \quad \sum_{t \in T} \left(c^{\mathrm{u}} m_t + c^{\mathrm{f}} y_t \right) + \sum_{t \in T} \sum_{i \in N_0} \left(h^{\mathrm{d}}_i I^{\mathrm{d}}_{it} + h^{\mathrm{p}}_i I^{\mathrm{p}}_{it} \right) + \sum_{t \in T} \sum_{(i,j) \in A} c_{ij} x_{ijt} \tag{23a}$$
$$+ p_{\mathrm{c}} \left(\sum_{t \in T} e_{1t} - E_1 \right) \tag{23b}$$

subject to (2)–(8), (17) and (18), where the term (23b) measures carbon trade costs/benefits from carbon cap-and-trade regulation on emissions.

2.3.2. Vehicle Routing-Related Emissions

To account for vehicle routing-related emissions, we adopt the linear approximation as in [1,3]. An arc-specific traveling time τ_{ij} is assumed by linking with each arc a speed of l_{ij}/τ_{ij}. The energy amount exhausted on arc (i, j) in period t is then as follows:

$$P_{ijt} = \alpha_{ij}(\omega + u_{ijt} + v_{ijt})l_{ij} + \beta \left(l_{ij}/\tau_{ij}\right)^2 l_{ij}. \tag{24}$$

where $\alpha_{ij} = a + g\sin\theta_{ij} + gC_r\cos\theta_{ij}$ is an arc-specific constant, and $\beta = 0.5C_d A\rho$ is a vehicle-specific constant. In the definitions, a is the vehicle acceleration (m/s^2), g is the gravitational constant (9.81 m/s^2), θ_{ij} is the angle of the arc (i, j), A is the frontal surface area of the vehicle (m^2), ρ is the air density (kg/m^3), and C_r and C_d are the coefficients of rolling resistance and drag, respectively [3]. The derivation of $\alpha_{ij} = a + g\sin\theta_{ij} + gC_r\cos\theta_{ij}$ and $\beta = 0.5C_d A\rho$ can be found in [3], where the energy amount P_{ijt} exhausted on arc (i, j) was approximated as $P_t l_{ij}/v_{ijt}$, where P_t is the total tractive power demand requirement. The calculation of the total tractive power demand requirement involves the mechanics of vehicles. Interested readers are referred to [3] and the references therein. Consequently, vehicle routing-related emissions in period t can be formulated as follows

$$e_{2t} = c_e \sum_{i\in N_0}\sum_{j\in N_0} P_{ijt}, \forall t \in T, \tag{25}$$

which become

$$\begin{aligned} e_{2t} = c_e &\sum_{i\in N_0}\sum_{j\in N_0} \left(\omega\alpha_{ij}l_{ij}x_{ijt} + \alpha_{ij}l_{ij}(u_{ijt} + v_{ijt})\right) \\ &+ c_e \sum_{i\in N_0}\sum_{j\in N_0} \beta\left(l_{ij}/\tau_{ij}\right)^2 l_{ij}x_{ijt}, \qquad \forall t \in T. \end{aligned} \tag{26}$$

by substituting (24) into (25).

Given vehicle routing-related carbon cap E_2, and production and inventory decisions from previous stages, we develop a pollution routing model under carbon cap-and-trade as follows.

$$\min \quad c_f \sum_{t\in T}\sum_{i\in N_0}\sum_{j\in N_0} \left(\omega\alpha_{ij}l_{ij}x_{ijt} + \alpha_{ij}l_{ij}(u_{ijt} + v_{ijt})\right) \tag{27a}$$

$$+ c_f \sum_{t\in T}\sum_{i\in N_0}\sum_{j\in N_0} \beta\left(l_{ij}/\tau_{ij}\right)^2 l_{ij}x_{ijt} \tag{27b}$$

$$+ \sum_{t\in T} w_t \sum_{i\in N_0}\sum_{j\in N_0} \tau_{ij}x_{ijt} \tag{27c}$$

$$+ p_c\left(\sum_{t\in T} e_{2t} - E_2\right) \tag{27d}$$

subject to (9)–(16), and (19)–(21). The terms (27c), (27a) and (27b) measure total driver wages and fuel costs, respectively. These terms also measure vehicle routing costs. The term (27d) represents carbon trade costs/benefits from carbon cap-and-trade regulation on emissions.

2.3.3. Formulation of PRPSPD under Carbon Cap-and-Trade

Given carbon price p_c and total carbon cap $E = E_1 + E_2$, we can formulate the PRPSPD under carbon cap-and-trade as follows:

$$
\begin{aligned}
\text{minimize} \quad & \sum_{t\in T}\left(c^{\mathrm{u}}m_t + c^{\mathrm{f}}y_t\right) + \sum_{t\in T}\sum_{i\in N_0}\left(h_i^{\mathrm{d}} I_{it}^{\mathrm{d}} + h_i^{\mathrm{p}} I_{it}^{\mathrm{p}}\right) + \sum_{t\in T}\sum_{(i,j)\in A} c_{ij}x_{ijt} \\
& + c_f \sum_{t\in T}\sum_{i\in N_0}\sum_{j\in N_0} \left(\omega\alpha_{ij}l_{ij}x_{ijt} + \alpha_{ij}l_{ij}(u_{ijt}+v_{ijt})\right) \\
& + c_f \sum_{t\in T}\sum_{i\in N_0}\sum_{j\in N_0} \beta\left(l_{ij}/\tau_{ij}\right)^2 l_{ij}x_{ijt} \\
& + \sum_{t\in T} w_t \sum_{i\in N_0}\sum_{j\in N_0} \tau_{ij}x_{ijt} \\
& + p_{\mathrm{c}}\left(\sum_{t\in T} e_{1t} + \sum_{t\in T} e_{2t} - E\right)
\end{aligned} \tag{28}
$$

subject to (2)–(21).

Theorem 1. *The PRPSPD under carbon cap-and-trade is $\mathcal{NP}$-hard.*

Proof. By combining terms in (28), we obtain

$$
\begin{aligned}
\min \quad & \sum_{t\in T}\left(\hat{c}_0 m_t + \hat{s}y_t + \sum_{i\in N_0}(\hat{h}_i^{\mathrm{d}} I_{it}^{\mathrm{d}} + \hat{h}_i^{\mathrm{p}} I_{it}^{\mathrm{p}}) + \sum_{(i,j)\in A} \hat{c}_{ij}x_{ijt}\right) \\
& + \sum_{t\in T}\sum_{i\in N_0}\sum_{j\in N_0} \tilde{c}_{ij}(u_{ijt}+v_{ijt}) \\
& - p_{\mathrm{c}}E
\end{aligned} \tag{29}
$$

subject to (2)–(21), where $\hat{c}_0 = c^u + p_c\tilde{c}_0, \hat{s} = c^f + p_c\tilde{s}, \hat{h}_i^{\mathrm{d}} = h_i^{\mathrm{d}} + p_c\tilde{h}_i, \hat{h}_i^{\mathrm{p}} = h_i^{\mathrm{p}} + p_c\tilde{h}_i, \hat{c}_{ij} = (c_f + p_c c_e)(\omega\alpha_{ij}l_{ij} + \beta l_{ij}^3/\tau_{ij}^2) + w_t\tau_{ij}, \tilde{c}_{ij} = (c_f + p_c c_e)\alpha_{ij}l_{ij}$.

With this compact formulation, we can see that the PRPSPD under carbon cap-and-trade is similar with PPRP [1] with additional inventory and flow terms as well as similar flow constraints. Because PPRP is $\mathcal{NP}$-hard [1], it follows that the PRPSPD under carbon cap-and-trade is also $\mathcal{NP}$-hard. □

3. Solution Method

In this section, after strengthening the linear relaxations of the formulations (2)–(21) with valid inequalities, we propose a branch-and-cut guided search algorithm.

3.1. Valid Inequalities

Denote by $I_{i0,s}^{\mathrm{d}} := \max\{0, I_{i0}^{\mathrm{d}} - \sum_{\tau=1}^{s}\delta_{i\tau}\}$ the product quantity left from the initial product inventory of customer $i \in N$ at the end of period s, and let $I_{i0,0}^{\mathrm{d}} := I_{i0}^{\mathrm{d}}$. The residual delivery demands at customer i in period s can also be defined as $\hat{\delta}_{is} := \max\{0, \delta_{is} - I_{i0,s-1}^{\mathrm{d}}\}$. With these notations, we can strengthen the linear relaxations of formulations (2)–(21) with the following valid inequalities:

$$
\sum_{\tau=1}^{t}\sum_{j\in N_0}\sum_{k\in K} x_{ijk\tau} \geq \left\lceil \frac{\sum_{\tau=1}^{t}\hat{\delta}_{i\tau}}{\min\{Q, L_i^{\mathrm{d}}\}} \right\rceil, \quad \forall i \in N, t \in T, \tag{30}
$$

$$
\sum_{\tau=1}^{t}\sum_{j\in N_0}\sum_{k\in K} x_{0jk\tau} \geq \left\lceil \frac{\sum_{i\in N}\sum_{\tau=1}^{t}\hat{\delta}_{i\tau}}{Q} \right\rceil, \quad \forall t \in T, \tag{31}
$$

$$
\sum_{\tau=1}^{t}\sum_{j\in N_0}\sum_{k\in K} x_{ijk\tau} \geq \left\lceil \frac{I_{i0}^{\mathrm{p}} + \sum_{\tau=1}^{t}\pi_{i\tau} - L_i^{\mathrm{p}}}{\min\{Q, L_i^{\mathrm{p}}\}} \right\rceil, \quad \forall i \in N, t \in T, \tag{32}
$$

where constraints (30) and (32) present lower bounds on the number of visits to each customer in each period, respectively. The lower bound in constraints (30) is calculated from residual delivery requests.

The lower bound in constraints (32) guarantee that accumulated pickup requests do not exceed the pickup inventory capacity. Constraints (31) specify a lower bound on the number of vehicles to use in each period. This lower bound is calculated from total residual delivery requests.

We can also strengthen the routing constraints (9)–(16) with the following valid inequalities

$$\sum_{i \in N} d_{it} \leq Q \sum_{j \in N_0} x_{0jt}, \qquad \forall t \in T, \tag{33}$$

$$\sum_{i \in N} \pi_{it} \leq Q \sum_{j \in N_0} x_{0jt}, \qquad \forall t \in T, \tag{34}$$

$$\sum_{j \in N_0} x_{ijt} \leq \sum_{j \in N_0} x_{0jt}, \qquad \forall i \in N, t \in T, \tag{35}$$

where constraints (33) and (34) enforce that the total delivery and pickup quantity do not exceed the total vehicle capacity, respectively, and constraints (35) ensure that each customer is visited only if the depot is also traversed.

Finally, we can add the following valid inequalities

$$\left(\sum_{j=0}^{s} \delta_{i,t-j}\right)\left(1 - \sum_{k=0}^{s} \sum_{j \in N_0} x_{ij,t-k}\right) \leq I^{\mathrm{d}}_{i,t-s-1}, \qquad \forall i \in N, t \in T, 0 \leq s \leq t-1, \tag{36}$$

$$L^{\mathrm{p}}_i - \left(\sum_{j=0}^{s} \pi_{i,t-j}\right)\left(1 - \sum_{k=0}^{s} \sum_{j \in N_0} x_{ij,t-k}\right) \geq I^{\mathrm{p}}_{i,t-s-1}, \qquad \forall i \in N, t \in T, 0 \leq s \leq t-1, \tag{37}$$

where constraints (36) ensure that in each time instant, product inventory is enough to meet accumulated delivery demand, and constraints (37) guarantee that in each time instant, accumulated pickup requests do not exceed the corresponding inventory capacity.

3.2. Branch-and-Cut Guided Search Algorithm

To solve the model, we propose a branch-and-cut guided search algorithm. This algorithm is outlined in Algorithm 1.

Initial Solution

An upper bound U^* and an incumbent solution at the root node of the branching tree is obtained through the following heuristic.

First, we set the delivery amount of each customer to its residual delivery requests in each period. We also set the pickup quantity of each customer to its pickup demand in each period, except that the pickup quantity in the first period also includes the initial pickup inventory. These pickup and delivery quantity are inputs to VRPSPD subproblems in each period. We obtain an initial feasible solution by the well-known Clark-and-Wright heuristic. The VRPSPD solution is then improved by a guided variable neighborhood descent (GVND). The procedure of the GVND is provided in Algorithm 2.

Algorithm 1: Branch-and-cut guided search algorithm

1 Obtain initial solutions and update the upper bound U^* and the incumbent solution.
2 Construct the node pool $\mathcal{N}$, which is initialized with the root node.
3 Generate and insert the proposed valid inequalities into the program at the root node of the search tree.
4 **repeat**
5   *Selection*: Choose the next node in $\mathcal{N}$, evaluate, and remove it from $\mathcal{N}$.
6   *Lower bound*: Obtain lower bound of the current node U^l by solving the LP relaxation at the current node.
7   **if** *the current solution is feasible* **then**
8   **if** $U^l > U^*$ **then**
9   go to the termination check.
        else
10             $U^* \leftarrow U^l$.
11   Update the incumbent solution.
12   Prune nodes with lower bound $U > U^*$.
        end
    end
13   *Cut generation*:
14   **if** *any cut is violated by the current solution of the LP relaxation* **then**
15   determine the violated subtour elimination cuts by connected component heuristics adapted from [34].
16   Add violated cuts.
    end
17   *Branching*: If $U^l > U^*$, go to the termination check.
until $\mathcal{N} = \emptyset$ *or time limit is met (termination check)*
18 Stop with the optimal solution and the corresponding cost U^*.

Algorithm 2: Guided variable neighborhood descent (GVND)

<u>**Function**</u> `GVND(s)`
1 **foreach** $t \in T$ **do**
2   Store routes in period t from the feasible solution s to R_t
3   **repeat**
4   **repeat**
5   Select next route pair $\{R_1, R_2\}, R_1 \in R_t, R_2 \in R_t$.
6   Set threshold $\tau = \lambda \times \max\{c_{ij} | (i,j) \in A(\{R_1, R_2\})\}$, where $A(\{R_1, R_2\})$ is the arc set for the route pair. Set $P_{ij} = 0, \forall (i,j) \in A(\{R_1, R_2\})$.
7   Utilize *exchange*, *relocate*, *2-opt**, and *cross-exchange* to route pair $\{R_1, R_2\}$ until no more improvements can be found. If an improvement has been found, utilize the LKH implementation of the Lin-Kernighan heuristic to both R_1 and R_2 separately.
8   Choose an arc according to Equation (38) and increase its P_{ij} by 1. If $\sum P_{ij} \times \lambda > \tau$, go to line 7; otherwise, go to line 6 evaluating moves according to Equation (39).
        until *all route pairs have been selected*
    until *no more improvements for any pair of routes can be detected*
end
return s

In the GVND, we apply a set of improvement heuristics [35]. The "inter-route" operators which we use and which modify several routes simultaneously are *exchange*, *relocate*, *2-opt**, and *cross-exchange*. In the *exchange* heuristic, the position of two customers in two routes are swapped at a time, while in the *relocate* heuristic a single customer is reinserted into an alternate route after being deleted from its original route. The *2-opt** heuristic removes two arcs $(i, i+1)$ and $(j, j+1)$ from two distinct routes, and reconnects the routes by inserting the arcs $(i, j+1)$ and $(j, i+1)$. The *cross-exchange* operator removes arcs $(i-1, i)$ and $(k, k+1)$ in one route, and arcs $(j-1, j)$ and $(l, l+1)$ in another route, then swaps the segments *i*-*k* and *j*-*l* by forming new arcs $(i-1, j)$,$(l, k+1)$,$(j-1, i)$ and $(k, l+1)$. These heuristics are implemented with the VRPH library [36].

First, we apply the inter-route operators for each route pair, and if an improvement is found, the LKH implementation [37] of the Lin-Kernighan heuristic [38] is applied to each single route. Then we adopt the guided local search strategy [35]. A modified objective function is used when the inter-route VND procedure is repeated, such that long arcs in the current route pairs are penalized. These long arcs will then be forced out of the candidate routes. We penalize the arc with the highest "utility function" value. The function is as follows:

$$\mathcal{U} = \frac{\lambda \times P_{ij} + c_{ij}}{1 + P_{ij}} \tag{38}$$

where P_{ij} is the number of penalized times of arc (i, j) and λ is a user-defined value. We set $\lambda = 8$ based on the sensitivity analysis in [35]. If current total penalty factors do not exceed a threshold, we repeat the inter-route procedures and LKH implementation of the Lin-Kernighan heuristic for single route with the modified objective function:

$$g'(s) = g(s) + \lambda \sum P_{ij} \tag{39}$$

where s is the current VRPSPD feasible solution, $g(s)$ is the original objective function for the VRPSPD subproblem.

4. Computational Results

This section summarizes computational experiments conducted to assess the performance of our algorithm and investigate how variations in key parameters affect carbon emissions.

The algorithm was coded in C++ with IBM ILOG CPLEX version 12 release 7 as the LP solver. The experiments were run on a 64-bit Windows 7 PC with Intel Core i7-6700 3.40GHz CPU and 16GB RAM.

4.1. Data and Experiment Settings

We generate instances by adapting the data set of [10] and adding parameters for customers' pickup requests. The data set of [10] was created from a subset of the dataset of [9] for the PRP, which consists four classes of instances. Five instances with different node coordinates were generated for each instance type. The first class is the base class. Unit production costs are higher in the second class. Transportation costs are higher in the third class. Instances in the fourth class have no customer inventory cost.

Instances for multi-vehicle PRP are generated by reducing initial inventory level for instances with three periods [10]. Production capacity and depot inventory capacity were set accordingly in [10] while uncapacitated production and unlimited depot inventory capacity were assumed in [9]. We keep the settings of initial inventory level, production capacity, and depot inventory capacity as those of [10], while taking the vehicle capacity as given in [9]. Moreover, pickup inventory holding costs are set equal to delivery inventory holding costs. Pickup requests of customers are set to half of delivery requests in the previous period except for pickup requests in the first period, which are set according to the initial delivery inventory level. There are no initial pickup inventories. Pickup inventory capacity is set to cover the pickup requests throughout planning periods.

We calculate carbon emission-related parameters according to [3,39]. Values for carbon emission-related parameters are shown in Table 1. As for τ_{ij}, we choose l_{ij}/τ_{ij} = 11 m/s (40 km/h) as in [1]. Empty vehicle weight w is set equal to vehicle capacity Q.

The average CPU times in seconds and the average number of nodes are reported in the "CPU" and "Nodes" columns, respectively, in Tables 2 and 3. The column "Gap" represents the difference between the final lower bound and the best upper bound as a percentage of the best upper bound. We also report the number of cuts added by CPLEX and the number of subtour elimination constraints separated in the columns "CPLEX Cuts" and "SEC Cuts", respectively, in Table 3 to assess the performance of our algorithm on different classes of instances.

Instance size varies and results in 370 instances. Specifically, 90 instances are generated for the results in Table 2, 120 for Table 3, 60 for Table 4, and 100 for Table 5.

Table 1. Values for carbon-emission-related parameters.

α_{ij}	β	c_e	c_f	w_t	$\tilde{s}$	$\tilde{c}_0$	$\tilde{h}_i$	E	E_1	E_2
0.981	2.1	0.00094	0.0006	2.2	22.6	22.6	0.8	15,000	10,000	5000

4.2. Results and Discussions

4.2.1. Performance of the Algorithm

First, we report the performance of the algorithm on the first class of instances in Table 2. For these base settings, the CPU time limit was set to 10 min. Given the notation ac/bp/cv, where a, b, and c are the number of customers, periods, and vehicles, respectively, these tests showed that the proposed algorithm can obtain the optimal solutions for instances up to 20c/3p/2v and 10c/6p/4v. For instances up to 50c/3p/4v, the algorithm can obtain near-optimal solutions with a gap of at most 4.7% in 10 min. For instances up to 50c/6p/6v, the near-optimal solutions obtained by the algorithm are also within gaps less than 8% in 10 min. Current exact algorithms for multi-vehicle PRP can only solve instances of up to 35c/3p/3v in 2 h [10]. Since vehicle routing subproblems in the PRPRSPD involve pickups, the routing subproblems are notably more difficult than that in the original PRP. The overall performance of our algorithm for the PRPRSPD is thus quite acceptable.

Table 2. Performance of the algorithm on the first class of instances: base settings.

n	$\lvert T \rvert$	$\lvert K \rvert$	CPU (s)	Gap (%)	Nodes (#)
10	3	2	5.6	0.0	628.8
15	3	2	65.6	0.0	3205.6
20	3	2	438.4	0.0	4491.2
25	3	3	600.0	1.5	4966.4
30	3	3	600.0	1.8	2996.8
35	3	3	600.0	2.8	1951.2
40	3	4	600.0	3.4	1909.6
45	3	4	600.0	3.4	791.4
50	3	4	600.0	4.7	665.2
10	6	4	498.0	0.0	3652.2
15	6	4	600.0	0.9	5877.2
20	6	4	600.0	2.7	2874.8
25	6	5	600.0	3.7	1852.8
30	6	5	600.0	4.5	890.2
35	6	5	600.0	5.3	595.8
40	6	6	600.0	5.4	190.6
45	6	6	600.0	6.2	91.4
50	6	6	600.0	7.6	65.2

Note: $p_c = 0.5$.

In Table 3, we report performance of the algorithm on four classes of instances with up to 20 customers. The CPU time limit was extended to 2 h to better compare the performance of the algorithm for different classes.

It can be shown from Table 3 that instances in class I are easiest to solve, and instances in class IV are the most difficult to solve. When there are only 10 and 15 customers, instances in class III are more difficult to solve than those in class II. However, when there are 20 customers, instances in class II become more difficult to solve than those in class III.

Table 3. Performance of the algorithm on instances with 10, 15, and 20 customers: four classes of instances.

Class	n	$\lvert T\rvert$	$\lvert K\rvert$	CPU (s)	Gap (%)	CPLEX Cuts	SEC Cuts	Nodes
I	10	3	2	5.6	0	260.8	35.0	628.8
II	10	3	2	17.5	0	144.3	47.1	2256.7
III	10	3	2	37.8	0	216.5	66.0	3225.1
IV	10	3	2	40.7	0	197.0	62.2	3880.7
I	10	3	3	7.8	0	211.3	31.9	936.7
II	10	3	3	19.4	0	124.8	68.5	2267.7
III	10	3	3	49.6	0	242.4	39.5	5885.6
IV	10	3	3	42.6	0	145.6	64.8	3370.6
I	15	3	2	65.6	0	323.1	56.6	3205.6
II	15	3	2	207.7	0	188.6	89.3	9875.4
III	15	3	2	578.2	0	434.2	143.7	27,364.8
IV	15	3	2	818.0	0	207.3	114.9	29,695.0
I	15	3	3	123.1	0	367.4	93.2	4486.5
II	15	3	3	327.4	0	306.6	108.8	18,028.9
III	15	3	3	1575.4	0.2	263.7	101.8	30,023.1
IV	15	3	3	372.6	0	174.0	140.2	24,796.2
I	20	3	2	438.4	0	378.1	223.8	4491.2
II	20	3	2	4103.5	0.3	271.4	185.9	46,195.4
III	20	3	2	4444.2	1.8	363.1	108.5	37,556.1
IV	20	3	2	5635.5	0.8	477.7	238.6	67,714.4
I	20	3	3	3799.3	0.6	460.8	206.5	29,192.2
II	20	3	3	3299.1	0.4	325.7	219.0	50,894.6
III	20	3	3	4697.3	1.6	317.8	86.2	28,345.1
IV	20	3	3	5906.0	0.8	453.6	315.2	103,188.3

Note: $p_c = 0.5$; Class I: base settings; Class II: high production unit cost; Class III: large transportation costs; Class IV: no retailer inventory costs.

4.2.2. Comparison of Costs and Sensitivity Analysis

In Table 4, we provide a comparison of solution costs on four classes of instances with up to 20 customers. Although unit production costs are 10 times higher, the manufacturing quantity and carbon emissions of class II are the same as those of class I. The reason is that delivery demand must be met, and backorder is not allowed in the model.

In contrast, when transportation costs increase for instances in class III, lower driver costs and routing emissions are achieved at the cost of larger production costs and emissions, and larger inventory costs and emissions. However, the total costs are reduced. This implies that the benefits of carbon cap-and-trade increase as transportation costs increase. Likewise, when there are no retailer inventory costs for instances in class IV, total costs increase. This also implies that the benefits of carbon cap-and-trade decrease as inventory costs get lower. The managerial insights are consistent with [1].

In Table 5, we provide a sensitivity analysis on the effect of carbon cap-and-trade. It can be seen that when the carbon price increases, total costs increase. However, total emission levels do not decrease monotonically. Total emission levels in the base settings achieved a minimum when carbon price was 0.3. The phenomenon reappears when transportation costs increase for instances in class III. However, when production costs increase for instances in class II, and when there are no retailer inventory costs for instances in class IV, total emissions first increase and then decrease. The reason may be that the change of production costs and inventory costs is too drastic, and the benefits of carbon cap-and-trade are thus twisted as compared to the base settings. Nevertheless, the managerial insights are still consistent with [1].

Table 4. Comparison of costs with instances with 10, 15, and 20 customers: four classes of instances.

Class	n	TC	PC	IC	FC	EL	PE	IE	RE	DC	NVI	NVE
I	10	4684	4047.9	1069.7	407.8	12,521	9322.9	1307.2	1891	398.1	9.6	2
II	10	41,111	40,479	996.3	434.4	12,555	9322.9	1217.5	2015	424.1	9.6	2
III	10	4513	4361.6	1375.8	279.8	11,444	8719.8	1426.9	1298	273.2	7.2	2
IV	10	5664	4558.3	1123.2	411.3	13,339	10,238.6	1192.7	1908	401.6	8.5	2
I	15	9410	5641.2	1535.1	583.8	17,160	12,612.0	1840.8	2708	570.0	13.8	3
II	15	60,250	56,412	1475.7	622	17,266	12,612.0	1769.6	2884	607.2	13.8	3
III	15	10,750	6852.4	1898	419.6	17,341	13,404.0	1991.3	1946	409.7	10.8	3
IV	15	10,172	6217.6	1610.3	592.8	17,345	12,816.0	1780.3	2749	578.8	12.2	3
I	20	15,743	7535.8	2149.3	798.6	23,960	17,609.7	2646.6	3704	779.7	18.9	4
II	20	83,850	75,358	1977	911.7	24,428	17,609.7	2590.5	4228	890.0	20.2	4
III	20	15,673	7616.7	2568	588.6	23,651	18,030.6	2891.2	2729	574.7	15.1	4
IV	20	16,070	7823	2321.1	828	23,579	17,210.9	2527.9	3840	808.4	17.0	4

Note: $|T| = 6$, $|K| = 4$, $p_c = 0.5$; TC: total cost; PC: total production cost; IC: inventory cost; FC: fuel cost; DC: driver cost; EL: emission levels; PE: production emissions; IE: inventory emissions; RE: routing emissions; NVI: number of visits; NVE: number of vehicles used.

Table 5. Comparison of costs under different carbon prices: four classes of instances.

Class	p_c	TC	PC	IC	FC	EL	PE	IE	RE	DC	NVI	NVE
I	0	8284	5641.2	1307.6	675.8	17,314	12,612	1568	3134	659.8	38.4	3
I	0.1	8529	5641.2	1410.2	634.3	17,245	12,612	1691.0	2941.5	619.3	36.2	3
I	0.3	8914	5641.2	1465.8	592.7	17,119	12,612	1757.8	2748.8	578.7	25.6	3
I	0.5	9410	5641.2	1535.1	583.8	17,160	12,612	1840.8	2707.5	570.0	13.8	3
I	0.7	9904	5641.2	1580.3	579.7	17,195	12,612	1895.0	2688.1	565.9	10.6	3
II	0	59,055	56,412	1307.6	675.8	17,314	12,612	1568.0	3134	659.8	38.4	3
II	0.1	59,345	56,412	1395.8	659.1	17,342	12,612	1673.8	3056.7	643.5	36.2	3
II	0.3	59,733	56,412	1407.1	630.8	17,225	12,612	1687.4	2925.3	615.9	25.6	3
II	0.5	60,250	56,412	1475.7	622	17,266	12,612	1769.6	2884.4	607.2	13.8	3
II	0.7	60,553	56,412	1519.7	577.7	17,114	12,612	1822.4	2679.2	564.1	10.6	3
III	0	9401	6852.4	1588.3	485.7	17,323	13,404	1666.5	2252.6	474.2	27.6	3
III	0.1	9748	6852.4	1725.1	470.8	17,397	13,404	1809.9	2183.1	459.6	26.2	3
III	0.3	10,155	6852.4	1734.8	445.5	17,290	13,404	1820.1	2066.1	435.0	15.4	3
III	0.5	10,750	6852.4	1898	419.6	17,341	13,404	1991.3	1945.9	409.7	10.8	3
III	0.7	11,252	6852.4	1995.9	393.4	17,323	13,404	2094.0	1824.6	384.1	10.8	3
IV	0	8883	6217.6	1441.2	619.5	17,282	12,816	1593.4	2872.8	604.8	35.2	3
IV	0.1	9134	6217.6	1461.7	619.3	17,304	12,816	1616.0	2872	604.6	33.4	3
IV	0.3	9680	6217.6	1564.7	604.1	17,347	12,816	1729.9	2801.6	589.8	24.8	3
IV	0.5	10,172	6217.6	1610.3	592.8	17,345	12,816	1780.3	2749.1	578.8	12.2	3
IV	0.7	10,574	6217.6	1739.9	535.9	17,225	12,816	1923.6	2485.2	523.2	10.6	3

Note: $n = 15$, $|T| = 6$, $|K| = 4$; TC: total cost; PC: total production cost; IC: inventory cost; FC: fuel cost; DC: driver cost; EL: emission levels; PE: production emissions; IE: inventory emissions; RE: routing emissions; NVI: number of visits; NVE: number of vehicles used.

5. Conclusions

We have introduced, modeled, and analyzed the PRPSPD, a generalization of the VRPSPD and IRPSPD. The contributions of this paper are: (1) to describe a modeling approach enriching the production-routing problems with simultaneous pickups and deliveries; (2) to offer a mixed-integer linear programming formulation for the PRPSPD under carbon cap-and-trade; (3) to provide a branch-and-cut guided search algorithm; and (4) to discuss reductions in carbon emissions under different carbon price, from which managerial insight can be drawn.

There are certain limitations associated with our work. For example, we assume an arc-specific traveling time τ_{ij} by associating with each arc a traveling speed of l_{ij}/τ_{ij}. However, in the computational experiments, the traveling speed on arc is assumed to be a discrete constant. Thus, estimation of routing-related carbon-emission costs might be not so accurate, although this assumption is usually adopted by researchers in related disciplines. Better treatment is expected to enhance the overall performance of supply chains.

Several extensions are possible for the PRPSPD. One worth mentioning here is the possibility of incorporating remanufacturing in the production part of the problem. Another extension would be to consider the simultaneous pickups and deliveries in multi-level production and routing problems with time windows. Finally, a branch-price-and-cut algorithm could be developed when dealing with a limited number of customers per vehicle.

Acknowledgments: Yuzhuo Qiu's work is supported by the NSFC under Grant No. 71571092. Yuzhuo Qiu's work is also supported in part by General Research Project for Humanities and Social Sciences from Chinese Ministry of Education under Grant No. 11YJCZH137 and the Priority Academic Program Development of Jiangsu Higher Education Institutions (PAPD).

Author Contributions: Yuzhuo Qiu conceived and designed the experiments; Xuanjing Fang performed the experiments; Xuanjing Fang and Yanan Du analyzed the data; Yuzhuo Qiu wrote the paper.

Conflicts of Interest: The authors declare no conflict of interest. The founding sponsors had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, and in the decision to publish the results.

References

1. Qiu, Y.; Qiao, J.; Pardalos, P. A branch-and-price algorithm for production routing problems with carbon cap-and-trade. *Omega* **2017**, *68*, 49–61.
2. Adulyasak, Y.; Cordeau, J.-F.; Jans, R. The production routing problem: A review of formulations and solution algorithms. *Comput. Oper. Res.* **2015**, *55*, 141–152.
3. Bektas, T.; Laporte, G. The pollution-routing problem. *Transp. Res. Part B Meth.* **2011**, *45*, 1232–1250.
4. Palak, G.; Ekşioğlub, S.D.; Geunes, J. Analyzing the impacts of carbon regulatory mechanisms on supplier and mode selection decisions: An application to a biofuel supply chain. *Int. J. Prod. Econ.* **2014**, *154*, 198–216.
5. Savaskan, R.C.; Bhattacharya, S.; Wassenhove, L.N.V. Closed-loop supply chain models with product remanufacturing. *Manag. Sci.* **2004**, *50*, 239–252.
6. Iassinovskaia, G.; Limbourg, S.; Riane, F. The inventory-routing problem of returnable transport items with time windows and simultaneous pickup and delivery in closed-loop supply chains. *Int. J. Prod. Econ.* **2017**, *183*, 570–582.
7. Chandra, P. A dynamic distribution model with warehouse and customer replenishment requirements. *J. Oper. Res. Soc.* **1993**, *44*, 681–692.
8. Lei, L.; Liu, S.; Ruszczynski, A.; Park, S. On the integrated production, inventory, and distribution routing problem. *IIE Trans.* **2006**, *38*, 955–970.
9. Archetti, C.; Bertazzi, L.; Paletta, G.; Speranza, M.G. Analysis of the maximum level policy in a production-distribution system. *Comput. Oper. Res.* **2011**, *38*, 1731–1746.
10. Adulyasak, Y.; Cordeau, J.-F.; Jans, R. Formulations and branch-and-cut algorithms for multivehicle production and inventory routing problems. *INFORMS J. Comput.* **2014**, *26*, 103–120.
11. Adulyasak, Y.; Cordeau, J.-F.; Jans, R. Benders decomposition for production routing under demand uncertainty. *Oper. Res.* **2015**, *63*, 851–867.
12. Brahimi, N.; Aouam, T. Multi-item production routing problem with backordering: A MILP approach. *Int. J. Prod. Res.* **2015**, *54*, 1076–1093.
13. Vahdani, B.; Niaki, S.; Aslanzade, S. Production-inventory-routing coordination with capacity and time window constraints for perishable products: Heuristic and meta-heuristic algorithms. *J. Clean. Prod.* **2017**, *161*, 598–618.
14. Zhang, Q.; Sundaramoorthy, A.; Grossmann, I.E.; Pinto, J.M. Multiscale production routing in multicommodity supply chains with complex production facilities. *Comput. Oper. Res.* **2017**, *79*, 207–222.
15. Kumar, R.S.; Kondapaneni, K.; Dixit, V.; Goswami, A.; Thakur, L.; Tiwari, M. Multi-objective modeling of production and pollution routing problem with time window: A self-learning particle swarm optimization approach. *Comput. Ind. Eng.* **2015**, *99*, 29–40.

16. Battarra, M.; Cordeau, J.-F.; Iori, M. Pickup-and-delivery problems for goods transportation. In *Vehicle Routing: Problems, Methods, and Applications*; Toth, P., Vigo, D., Eds.; MOS/SIAM Series on Optimization; SIAM: Philadelphia, PA, USA, 2014; Volume 18, pp. 161–192.
17. Subramanian, A.; Uchoa, E.; Pessoa, A.A.; Ochi, L.S. Branch-cut-and-price for the vehicle routing problem with simultaneous pickup and delivery. *Optim. Lett.* **2013**, *7*, 1569–1581.
18. Cherkesly, M.; Desaulniers, G.; Laporte, G. Branch-price-and-cut algorithms for the pickup and delivery problem with time windows and last-in-first-out loading. *Transp. Sci.* **2015**, *49*, 752–766.
19. Qu, Y.; Bard, J.F. A branch-and-price-and-cut algorithm for heterogeneous pickup and delivery problems with configurable vehicle capacity. *Transp. Sci.* **2015**, *49*, 254–270.
20. Soysal, M. Closed-loop inventory routing problem for returnable transport items. *Transp. Res. Part D Transp. Environ.* **2016**, *48*, 31–45.
21. Van Anholt, R.G.; Coelho, L.C.; Laporte, G.; Vis, I.F.A. An inventory-routing problem with pickups and deliveries arising in the replenishment of automated teller machines. *Transp. Sci.* **2016**, *50*, 1077–1091.
22. Bard, J.F.; Nananukul, N. Heuristics for a multiperiod inventory routing problem with production decisions. *Comput. Ind. Eng.* **2009**, *57*, 713–723.
23. Bard, J.F.; Nananukul, N. A branch-and-price algorithm for an integrated production and inventory routing problem. *Comput. Oper. Res.* **2010**, *37*, 2202–2217.
24. Chandra, P.; Fisher, M.L. Coordination of production and distribution planning. *Eur. J. Oper. Res.* **1994**, *72*, 503–517.
25. Fumero, F.; Vercellis, C. Synchronized development of production, inventory, and distribution schedules. *Transp. Sci.* **1999**, *33*, 330–340.
26. Boudia, M.; Louly, M.; Prins, C. A reactive GRASP and path relinking for a combined production-distribution problem. *Comput. Oper. Res.* **2007**, *34*, 3402–3419.
27. Boudia, M.; Prins, C. A memetic algorithm with dynamic population management for an integrated production-distribution problem. *Eur. J. Oper. Res.* **2009**, *195*, 703–715.
28. Bard, J.F.; Nananukul, N. The integrated production-inventory-distribution-routing problem. *J. Sched.* **2009**, *12*, 257–280.
29. Armentano, V.; Shiguemoto, A.; Lokketangen, A. Tabu search with path relinking for an integrated production-distribution problem. *Comput. Oper. Res.* **2011**, *38*, 1199–1209.
30. Adulyasak, Y.; Cordeau, J.-F.; Jans, R. Optimization-based adaptive large neighborhood search for the production routing problem. *Transp. Sci.* **2014**, *48*, 20–45.
31. Absi, N.; Archetti, C.; Dauzere-Peres, S.; Feillet, D. A two-phase iterative heuristic approach for the production routing problem. *Transp. Sci.* **2015**, *49*, 784–795.
32. Russell, R.A. Mathematical programming heuristics for the production routing problem. *Int. J. Prod. Econ.* **2017**, *193*, 40–49.
33. Solyalı, O.; Süral, H. A multi-phase heuristic for the production routing problem. *Comput. Oper. Res.* **2017**, *87*, 114–124.
34. Lysgaard, J.; Letchford, A.N.; Eglese, R.W. A new branch-and-cut algorithm for the capacitated vehicle routing problem. *Math. Program.* **2004**, *100*, 423–445.
35. Kytöjoki, J.; Nuortio, T.; Bräysy O.; Gendreau M. An efficient variable neighborhood search heuristic for very large scale vehicle routing problems. *Comput. Oper. Res.* **2007**, *34*, 2743–2757.
36. Groër, C.; Golden, B.; Wasil, E. A library of local search heuristics for the vehicle routing problem. *Math. Program. Comput.* **2010**, *2*, 79–101.
37. Helsgaun, K. LKH. 2012. Available online: http://www.akira.ruc.dk/~keld/research/LKH (accessed on 28 November 2017).
38. Lin, S.; Kernighan, B.W. An effective heuristic algorithm for the traveling-salesman problem. *Oper. Res.* **1973**, *21*, 498–516.
39. Franceschetti, A.; Honhon, D.; Van Woensel, T.; Bektaş, T.; Laporte, G. The time-dependent pollution-routing problem. *Transp. Res. Part B Meth.* **2013**, *56*, 265–293.