Solving Inventory Routing with Transshipment and Substitution under Dynamic and Stochastic Demands using Genetic Algorithm and Deep Reinforcement Learning

Fatima Ezzahra Achamrah [ab], Fouad Riane [abc] and Sabine Limbourg [d]

[a]Complex Systems and Interations, Ecole Centrale of Casablanca; [b]Laboratoire Genie Industriel, CentraleSupelec, Paris Saclay University; [c] LIMII, Hassan First University, Settat, Morocco; [d] HEC-University of Liege (ULiege)

ARTICLE HISTORY
Compiled December 5, 2023

Abstract
In this paper, we investigate a two-level supply chain consisting of a company which manufactures a set of products and distributes them via its central warehouse to a set of customers. The problem is modelled as a dynamic and stochastic inventory routing problem (DSIRP) that considers two flexible instruments of transshipment and substitution to mitigate shortages at the customer level. A new resolution approach, based on the hybridisation of mathematical modelling, Genetic Algorithm and Deep Reinforcement Learning is proposed to handle the combinatorial complexity of the problem at hand. Tested on the 150 most commonly used benchmark instances for single-vehicle-product DSIRP, results show that the proposed algorithm outperforms the current best results in the literature for medium and large instances. Moreover, 450 additional instances for multi-products DSIRP are generated. Different demand distributions are examined in these experiments, namely Normal distribution, Poisson distribution for demand occurrence, combined with demands of constant size; Stuttering Poisson distribution and Negative Binomial distribution. In terms of managerial insights, results show the advantages of promoting inventory sharing and substitutions on the overall supply chain performance.

KEYWORDS
Dynamic and Stochastic Routing; Transshipment; Substitution; Genetic Algorithm; Deep Reinforcement Learning

1. Introduction

Highly competitive markets drive companies to efficiently and accurately satisfy their customers' demands across their supply chain. The lead times in most industries must be relatively short, and companies must be flexible enough to meet highly variable demands. Companies also should efficiently manage their capital assets to guarantee profitability. This highly depends on their capacity to maintain their manufacturing and logistical capacities to meet their customers' service requirements. In practice, to mitigate this issue, firms may promote inventory sharing among multiple locations within the same distribution network which leads to significant reductions in costs. This type of inventory sharing is commonly referred to as lateral transshipment (LT) (Paterson et al., 2011; Grahovac and Chakravarty, 2001).

CONTACT Sabine Limbourg, Email: sabine.limbourg@uliege.be

In such competitive markets, customers choose from a variety of products according to their needs. They may choose to buy their preferred products, or in case of unavailability, replace them with different ones. Substitutes can lead to a healthy market competition between products, which is in the customers' best interest and this prevents a market monopoly. This can be the case of food products; perishable products (e.g., artificial blood that can be used as a substitute to mitigate the risks of blood transfusions and shortage of supply or two different milk brands that can be substituted if their "milk" products have similar characteristics); and spare parts (e.g., original equipment manufacturer parts that can be substituted by aftermarket parts called replacement or pattern parts). Substitution could, therefore, serve as a new alternative to better meet customers' demands, particularly if decision makers are not fully aware of future events. This paper aims at highlighting the benefits of promoting both inventory sharing among customers and use of substitutes to remedy the shortage of products in a such stochastic environment. Products are therefore considered virtually pooled in the network and sent to a requesting location via LT from a location possessing a surplus of on-hand inventory; or they are substituted, if compatible, by each other.

This paper has four main contributions. First, we study a two-level supply chain in which a manufacturer supplies a central warehouse with a set of products. The central warehouse, distantly located from the manufacturer, distributes, under dynamic and stochastic demands, products to a given number of customers. Along with LT, substitutions of products, which is new to literature, are used to sidestep shortage at the customer level. We also assume that direct shipment, if necessary, can take place from the central warehouse to any customer. Secondly, we model the problem as a multi-product dynamic and stochastic inventory routing problem. The objective is to minimise the total cost, including the costs of holding inventory, transportation, transshipment, substitution and lost sales. Thirdly, a new resolution approach based on the hybridisation of mathematical modelling, Genetic Algorithm and Deep Reinforcement Learning is proposed to handle the combinatorial complexity of the problem at hand. And finally, tested on the 150 most commonly used benchmark instances for single-vehicle-product DSIRP, our algorithm outperforms the state-of-the-art algorithm. The experimental results show that the proposed algorithm outperforms the current best results in the literature for medium and large instances in terms of the quality of the solutions and run times. In addition, 450 additional instances for multi-product DSIRP are generated. Different demands distributions are examined in these experiments, namely Normal distribution, Poisson distribution for demand occurrence, combined with demands of constant size; Stuttering Poisson distribution and Negative Binomial distribution. Results confirm the efficiency of the proposed algorithm and highlight the benefits of both LT and substitutions on the supply chains overall.

The remainder of the paper is structured as follows. Section 2 presents related works. After describing the problem in Section 3, a mathematical formulation is provided in Section 4. In Section 5, a matheurstic based on hybridisation of mathematical modeling, a Genetic Algorithm and Deep Reinforcement learning is described. Section 6 provides computational experiments. We present conclusions and perspectives in Section 7.

2. Related work

First, we describe the Inventory Routing Problem (IRP) and its classifications, then papers on IRP with uncertainty are categorised, and finally, papers addressing IRP with transshipment are discussed.

2.1. *Inventory Routing Problem*

IRP includes inventory management, vehicle routing problem (VRP), and delivery scheduling decision making problems (Coelho et al., 2014b). Suppliers can reduce the overall costs of their activities in order to achieve a competitive advantage by integrating their routing, inventory and distribution decisions instead of independently optimising them. Such decisions can be streamlined by introducing a vendor managed inventory (VMI) approach, which incorporates replenishment and distribution processes, resulting in overall logistics cost reduction. In Coelho et al. (2012a), IRP is classified according to:

(1) the number of customers and suppliers:
   (a) one-to-one if only one supplier serves one customer (Dror and Levy, 1986).
   (b) one-to-many, in the most common cases of one supplier and several customers (Bell et al., 1983; Burns et al., 1985; Abdelmaguid, 2004).
   (c) many-to-many, which occurs less often, with multiple suppliers and multiple customers (Christiansen, 1999; Ronen, 2002).

(2) routing can be direct if there is only one client per route, multiple if there are multiple clients on the same route (Zhao et al., 2008), or continuous, as in several maritime applications, where there is no central depot (Savelsbergh and Song, 2008; Hewitt et al., 2013).

(3) pre-established inventory strategies to satisfy customers. The two most popular are the policy of the Maximum Level (ML) and the policy of Order-Up to level (OU). The replenishment level is flexible under an ML inventory strategy, but is restricted by the resources available to each customer (Savelsbergh and Song, 2008; Coelho and Laporte, 2013). Under an OU policy, the quantity delivered is required to fill its inventory capacity whenever a customer is visited (Archetti et al., 2007a). If the inventory is allowed to become negative, back-ordering will take place and the corresponding demand will be served at a later period (Abdelmaguid et al., 2009). If there is no back-order, the extra demand will be considered as a loss of sales (Mirzaei and Seifi, 2015). In both cases, a penalty for the shortage can be applied.

(4) composition and size of the fleet. The fleet can be homogeneous or heterogeneous, and the number of available vehicles can be set at one, set at many or unconstrained (Zhao et al., 2008; Coelho et al., 2012a).

In all of these papers, only one product is considered, whereas many VMI applications are concerned with multiple product distributions. Few papers address the multi-product inventory routing problem (MPIRP) (Coelho and Laporte, 2013). Most of the applications emerge in maritime logistics: Bertazzi and Speranza (2002); Grønhaug et al. (2010); Christiansen et al. (2011); Stålhane et al. (2012). Non-maritime cases include for example the delivery of perishable goods (Dehghani et al., 2021; Hssini et al., 2016), the transportation of gas by tanker trucks (Bell et al., 1983), the production and the distribution planning in gas filling industry (Strack et al., 2011), and the vehicle parts industry (Alegre et al., 2007).

2.2. *IRP with uncertainty*

IRP can be classified into four categories depending on the nature of the input data: (1) static and deterministic; (2) dynamic and deterministic; (3) static and stochastic and (4) dynamic and stochastic. Dynamic IRP (DIRP) differs from the static IRP (SIRP) in that the demands are known before planning in SIRP, while in DIRP demands are gradually revealed over time (Bertazzi et al., 2013). Stochastic and static IRP (SSIRP), is similar to the static

IRP except that the customer demand is known in a probabilistic sense (Bertazzi et al., 2013). In a dynamic and stochastic IRP (DSIRP), the objective is not to deliver a static result but a solution policy using the information revealed, outlining which measures need to be performed as time passes (Coelho et al., 2014a).

According to Coelho et al. (2014b), solving stochastic DSIRP relies on finding a solution policy which consists of one of the following:

(1) optimising a static instance whenever new information becomes available.
(2) applying a static algorithm only once and then re-optimising the problem through a heuristic whenever new information is made available.
(3) taking advantage of the probabilistic knowledge of future information and making use of forecasts.

Yu et al. (2013) solve a SSIRP with split delivery using a hybrid approach based on Lagrangian relaxation and local search improvement. Solyalı et al. (2012) solve a single product SSIRP with backorders. Authors apply a branch-and-cut algorithm for the robust proposed formulation. Bertazzi et al. (2013) address the same problem under an OU policy and consider shortage to be allowed. Authors present a dynamic programming formulation and a hybrid algorithm based on roll-out algorithm and a heuristic method. Huang and Lin (2010) solve a multi-item SSIRP using the conventional ant colony optimisation algorithm. Coelho et al. (2014a) propose an adaptive large neighborhood search with reactive and proactive policies to solve a single vehicle single product DSIRP with transshipment. Finally, Roldán et al. (2016) extend the work of Coelho et al. (2014a) by addressing the robustness of inventory replenishment and customer selection policies.

2.3. *IRP with transshipment*

Coelho et al. (2012b) are, to the best of our knowledge, the first authors to propose the concept of transshipment within inventory-routing (IRPT). Authors propose a mixed-integer linear program to model a single-vehicle and single-product IRPT. Transshipment is allowed either from the manufacturer to customers or between customers. Lefever et al. (2018) model the same problem as in Coelho et al. (2012b) and strengthen their formulation by proposing a set of valid inequalities for IRPT based on the existing valid inequalities for the IRP, bounds, reformulation and variable eliminations on the linear relaxation of the problem of concern. Peres et al. (2017) model a multi-period, multi-product IRPT and use a Randomized Variable Neighbourhood Descent to solve the problem. Hssini et al. (2016) address MPIRP under a static and deterministic demand in a blood supply chain. The authors consider transshipment of blood products between hospitals and substitution between blood groups. On stochastic demand, Dehghani et al. (2021) develop a mathematical model that decides on transshipment under static and stochastic demand to reduce total costs, as well as shortages in a blood supply chain. Achamrah et al. (2021) model a two-level spare parts supply chain under static and stochastic demands. The authors consider transshipment of spare parts between depots and substitutions between original equipment manufacturer and pattern parts. Chrysochoou et al. (2015) propose a two–stage programming model in which transshipment is considered as a recourse action to address a single product and vehicle DSIRP. Coelho et al. (2014a) address the dynamic and stochastic version of the problem studied by Coelho et al. (2012b). Under OU and ML policies, the problem is solved using either a proactive or reactive policy all implemented in a rolling horizon fashion. In the proactive policy, once forecasts on demands are obtained, routes are constructed and LT takes place after the demand is realised to re-

duce shortages. The reactive policy (or wait and see policy) observes the state of the system and makes decisions accordingly. It is defined as an (r, S) replenishment system under which whenever the inventory reaches the reorder point r, it triggers a replenishment order to bring the inventory position up to S. Routing are constructed based on the threshold r, and as in proactive policy, LT takes place when demands are revealed. Authors also use an adaptive large neighborhood search to determine routing and exact method to determine the quantities to be transshipped. The setting of our problem description follows that of this paper.

2.4. *Paper main contributions*

Based on this literature review, apart from promoting transshipment between customers to avoid shortages of products, none of the existing papers incorporate products substitution within a dynamic and stochastic setting. The present paper extends the work of Coelho et al. (2014a) on a single product and single-vehicle DSIRP by addressing a more realistic configuration of the problem at hand. In the following, we study a one-to-many multi-product DSIRP under an ML policy and in which customers' demand follows a probability distribution which values of parameters are revealed over time. Moreover, we propose a model that integrates substitutions along with transshipment as alternatives to sidestep shortages at the customer level. As for resolution approach, the present study also contributes to existing literature by combining Genetic Algorithm and Deep Reinforcement Learning technique. The latter is used to analyse data related to the decision and the objective spaces that have been visited during the search process, moves and recombination. With the help of Deep Q-learning, useful knowledge is extracted and used to enhance the search performance and speed of the metaheuristic. Applied on a benchmark of 150 instances with up to a maximum 200 customers and 20 as number of periods for a single-product DSIRP and on a set of 450 generated instances for multi-products, the resolution approach obtains results that are advantageous compared to results stemming from the state-of-the-art algorithm, thus, proving its efficiency.

3. Problem description

The following problem description follows that of the paper of Coelho et al. (2014a) developed for a single-vehicle-product DSIRP. Our multi-vehicle-product DSIRP with Transshipment and Substitution (DSIRPTS) is defined on a graph $\mathcal{G} = (\mathcal{N}, \mathcal{A})$, where $\mathcal{N}$ is the vertex set indexed by $i \in \{0, ..., n\}$ and $\mathcal{A} = \{(i, j) : i, j \in \mathcal{N}, i \neq j\}$ is the edge set. Vertex 0 represents the manufacturer's central warehouse (CW), and the set $\mathcal{N}_0 = \mathcal{N} \backslash \{0\}$ denotes the customers. The planning horizon length is T with discrete time periods $t \in \mathcal{H} = \{1, ..., T\}$. The demand d_{pit} each customer $i \in \mathcal{N}_0$ has to satisfy for product $p \in \mathcal{P} = \{1, ..., m\}$ per period $t \in \mathcal{H}$ is a random variable D_{pit} per stock keeping unit (SKU). Moreover, each customer and the central warehouse, $i \in \mathcal{N}$, incur unit inventory holding costs, h_{pi} per product $p \in \mathcal{P}$, with inventory capacities K_i. Inventories are not allowed to exceed the holding capacity and must be positive. We further assume the CW has enough inventory to meet all demand during the planning horizon. At the beginning of each period, at each location $i \in \mathcal{N}$, the current inventory levels I_{pi0} of the product p are known.

A set of homogeneous vehicles $v \in V = \{1, ..., k\}$ is available, each with a capacity Q in terms of SKU with routing cost c_{ij} associated to all $(i, j) \in \mathcal{A}$. Direct deliveries along with multiple routing are permitted to guarantee that all planned deliveries are met (before demands are revealed). LT can occur when it is profitable to ship products between customers. LT between customers and direct shipment from CW to any customer are assumed to be outsourced and

the relative unit cost can be expressed as αc_{ij}, where $\alpha > 0$. α is used to express the fact that outsourced operations are volume-dependent rather than distance-dependent (as this is how often carriers define the terms of contracts).

The unit cost of substituting a product p by $s \in P$ is a_{ps}. All possible combinations according to the products' compatibility are represented by o_{ps}, which is equal to 1 if a product p is compatible, according to the customer, with a product s, and 0 otherwise. Compatible products can be used as substitutes to satisfy customer demand when preferred products are not available. We assume that the substitution of products is not bi-directional. That is, a product p is substituting product s but the inverse is not necessary implied. We also assume that the CW distributes multiple products including substitutes, and deliveries and transshipment can be performed during the same time period. The lost sales cost which is associated with the shortage of a product p at the customer i is f_{pi}. Finally, we assume that the manufacturer has enough inventory of products to service its CW and that the quantity of product p shipped from the manufacturer to the CW in period t is expressed by g_{pt}. As in Coelho et al. (2014a) and Archetti et al. (2012), we assume that g_{pt} is used only to account for inventory costs at the CW.

Regarding the sequence of the operations, we assume that first, the decisions related to routing, including direct shipments, are determined. Second, after demands are realised, LT and possible substitutions are performed to sidestep, as much as possible, shortages at the level of each customer. Decisions variables are as follows:

- I_{pit} the inventory level of product p at node $i \in \mathcal{N}$ at the end of a period t.
- q_{pitv} the quantity of product p delivered by vehicle v from the CW to the node $i \in \mathcal{N}$ in a period t.
- w_{pijt} the quantities of product p carried by the outsourced carrier from the the node $i \in \mathcal{N}$ to $j \in \mathcal{N}$, in a period t.
- l_{pit} the lost sales quantity of product p at customer $i \in \mathcal{N}_0$ in a period t.
- z_{spi} are defined for all $o_{sp} = 1$ as the quantity of a product s substitute for product p used at the customer i in a period t to satisfy a part of the unsatisfied demand.

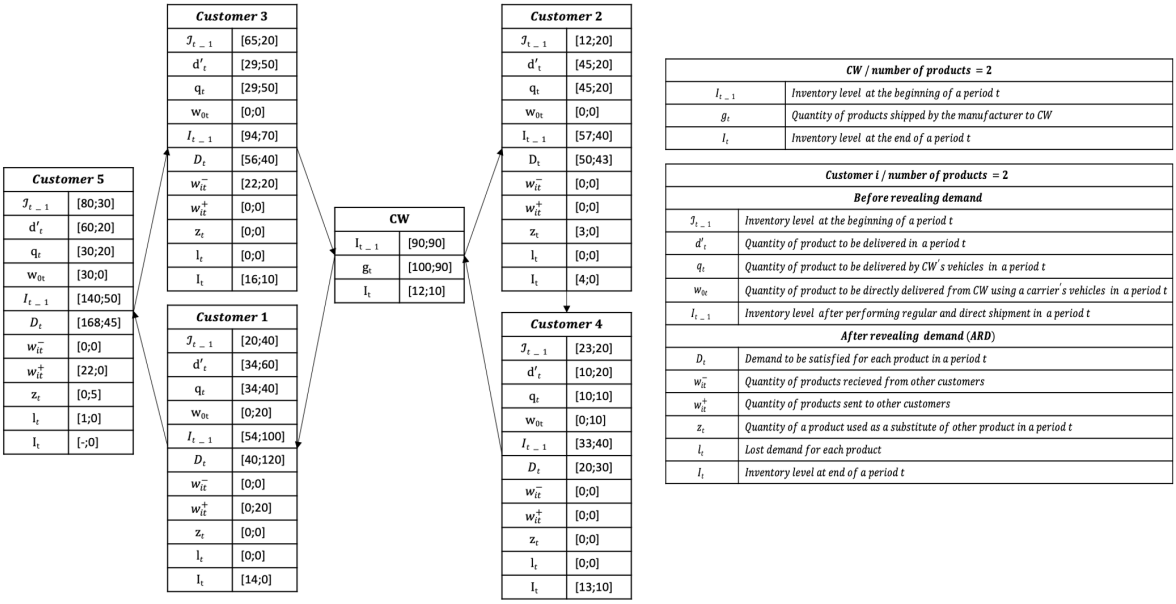
The inventory level at the end of each period at customer i is then:

$$I_{pit} = I_{pit-1} + l_{pit} + \sum_{j \neq i \in \mathcal{N}_0} (w_{pjit} - w_{pijt}) + \sum_{s \neq p \in \mathcal{P}} (z_{spit} - z_{psit}) \qquad \forall p \in \mathcal{P}, i \in \mathcal{N}_0, t \in \mathcal{H} \tag{1}$$

The objective function is to minimize the total cost which includes inventory holding, lost sales, substitutions, transshipment and routing costs:

$$\begin{aligned} \min & \sum_{t \in \mathcal{H}} \sum_{i \in \mathcal{N}} \sum_{p \in \mathcal{P}} h_{pi} I_{pit} + \sum_{t \in \mathcal{H}} \sum_{i \in \mathcal{N}_0} \sum_{p \in \mathcal{P}} f_{pi} l_{pit} + \\ & \sum_{t \in \mathcal{H}} \sum_{i \in \mathcal{N}_0} \sum_{p,s \in \mathcal{P}} a_{sp} z_{spit} + \alpha \sum_{t \in \mathcal{H}} \sum_{i,j \in \mathcal{N}_0, i \neq j} c_{ij} \sum_{p \in \mathcal{P}} w_{pijt} + \mathscr{C}_t \end{aligned} \tag{2}$$

where $\mathscr{C}_t$ is the cost of the routes performed in a period t. In the following, we present how the DSIRPT is solved using a reactive policy as in (Coelho et al., 2014a).



[A numerical example]A numerical example with the CW and 6 customers.

Figure 1. A numerical example.

4. Solution policy

Reactive policy, also known as the "wait and see" policy, consists of observing the state of the system in order to make decisions regarding routing, transshipment and substitution. As in (Coelho et al., 2014a) we adopt an (r_{pit}, S_{pit}) replenishment system in which anytime the inventory level of a product p reaches a reorder point r_{pit}, a replenishment order to visit a customer i is triggered so that the inventory level is brought up to a value S_{pit}. Routing, including direct shipment, are constructed accordingly (Routing Model RM) in each period t. The objective is to deliver the quantities of products that have been determined using RM. Then, when visiting customers, the product p's demand is revealed in a period t and LT and substitution take place if the available inventory is insufficient to meet this demand (Transshipment and Substitution Model TSM). This is implemented in a rolling horizon framework.

Figure 1 provides a numerical example. Notations of the model are summarized in Table 1.

Table 1. Notation summary

	Sets
$\mathcal{H}$	Planning horizon indexed by t
V	Set of CW's vehicles indexed by v
$\mathcal{P}$	Set of products indexed by p
	Routing model
	Parameters
c_{ij}	Transportation unit cost associated to regular shipment using CW's vehicles for all $(i,j) \in \mathcal{A}$
α	Discount factor $0 < \alpha < 1$ used to represent the cost associated to the outsourced operations
Q	CW's vehicles capacity
I_{p00}	Inventory level at beginning of the planning horizon of a product p at CW (node 0)
g_{pt}	Quantity of a product p supplied to CW in period t
r_{pit}	Reorder point for product p, at customer i and period t
u_{pit}	Expected demand of product p in period t
σ_{pit}	Standard deviation of the demand of product p in period t
β	Shortage probability
$\Upsilon_{\beta pt}$	β-order quantile of the demand distribution for product p
$\mathscr{I}_{pit-1}$	Inventory level of product p at customer i at the beginning of period t (before the demand is revealed)
d'_{pit}	Quantity of product p that should be delivered to customer i in period t (using both regular and direct shipment/ before the demand is revealed)
	Decision variables
q_{pitv}	Quantity of product p delivered by vehicle v of CW to node $i \in \mathcal{N}$ in period t
w_{p0it}	Quantity of product p carried by the outsourced carrier from CW to node $i \in \mathcal{N}_0$, in period t
I_{p0t}	Inventory level of a product p at CW (node $\{0\}$) at the end of period t
x_{ijv}	Equal to 1 if the arc $(i,j) \in \mathcal{A}$ is visited by vehicle v in period t; 0 otherwise
y_{itv}	Equal to 1 if a customer i is visited by vehicle v in period t; 0 otherwise
	Transshipment and substitution model
	Parameters
h_{pi}	Unit inventory holding cost of product p at customer $i \in \mathcal{N}_0$ in period t
f_{pi}	Unit cost associated to the lost demand of product p at customer $i \in \mathcal{N}_0$ in period t
a_{sp}	Unit cost associated with the substitution of a primary preferred product p by a substitute s
D_{pit}	Random variable associated to revealed demand of product p in period t at node $i \in \mathcal{N}_0$
K_i	Maximum inventory capacity at customer $i \in \mathcal{N}_0$
I_{pi0}	Inventory level of product p at customer $i \in \mathcal{N}_0$ at beginning of the planning horizon
o_{sp}	Equal to 1 if a substitute product s is compatible with a primary preferred product p, and 0 otherwise
	Decision variables
I_{pit}	Inventory level of a product p at a customer i after the demand is revealed and the performance of transshipment and substitutions at the end of period t
w_{pijt}	Transshipment quantity of product p carried by the outsourced carrier from customer $i \in \mathcal{N}_0$ to the customer $j \in \mathcal{N}_0$, in period t after the demand is revealed
z_{spit}	Quantity of product s substitute for product p used at the customer i in period t to satisfy a part of the unsatisfied demand (defined for all $o_{sp} = 1$)
l_{pit}	Lost sales quantity of product p at customer $i \in \mathcal{N}_0$ in period t when the demand is revealed and performance of transshipment and substitutions

Note that the part of the satisfied demand of product p is represented by the quantity z_{ppi} of the product p used as a substitute of itself.

4.1. *Routing model*

Under ML policy, CW can freely decide on the quantity to supply the customer with, restricted only by the customer's inventory capacity and by the threshold r_{pit}. This quantity defines the parameter d'_{pit} that is proportional to $max[0, r_{pit} - \mathscr{I}_{pit-1}]$, where $\mathscr{I}_{pit-1}$ is the inventory level at the beginning of a period t. r_{pit} is defined as the expected demand during a lead time $\mathscr{L}$ (which is equal to 1 as deliveries taking place in a period t can be used to satisfy demand at period t), plus a safety stock which depends on demand variability, $\mathscr{L}$ and target service level. As in Coelho et al. (2014a), we assume a normally distributed demand. r_{pit} can be then

computed as follows:

$$r_{pit} = u_{pit} + \Upsilon_{\beta pt}\sigma_{pit} \tag{3}$$

where u_{pit} in a given period t is an estimate of the expected demand of product p at the customer i and σ_{pit} the related standard deviation. β is the shortage probability and $\Upsilon_{\beta pt}$ is the β-order quantile of the demand distribution for the product p. These values as well as r_{pit} are updated in each period t.

To construct vehicle routing in each period t, a mixed-integer linear program (MILP) is proposed. The objective is to decide which customer is allocated to which vehicle, the quantities delivered at each node, and direct shipments, if any.

For each period t, the RM is formulated as follows :

Objective function OF of routes X:

$$\min \left[\sum_{v\in V} \sum_{i,j\in\mathcal{N}, i\neq j} c_{ij}x_{ijvt} + \alpha \sum_{i\in\mathcal{N}_0} c_{0i} \sum_{p\in\mathcal{P}} w_{p0it} \right] \tag{4}$$

Subject to:

$$\sum_{v\in V} q_{pitv} + w_{p0it} = d'_{pit} \qquad \forall p \in \mathcal{P}, i \in \mathcal{N}_0 \tag{5}$$

$$\sum_{i\in\mathcal{N}_0} \sum_{p\in\mathcal{P}} q_{pitv} \leq Q \qquad \forall v \in V \tag{6}$$

$$\sum_{p\in\mathcal{P}} q_{pitv} \leq Q y_{itv} \qquad \forall i \in \mathcal{N}_0, v \in V \tag{7}$$

$$\sum_{j\in\mathcal{N}, i\neq j} x_{ijtv} + \sum_{j\in\mathcal{N}, i\neq j} x_{jitv} = 2y_{itv} \qquad \forall i \in \mathcal{N}, v \in V \tag{8}$$

$$\sum_{i\in\mathcal{S},} \sum_{j\in\mathcal{S}, i\neq j} x_{ijtv} \leq \sum_{i\in\mathcal{S}} y_{itv} - y_{\iota tv} \qquad \forall \mathscr{S} \subseteq \mathcal{N}_0, \iota \in \mathscr{S}, v \in V \tag{9}$$

$$I_{p0t} = I_{p0t-1} - \sum_{i\in\mathcal{N}} \sum_{v\in V} q_{pitv} - \sum_{i\in\mathcal{N}} w_{p0it} + g_{pt} \qquad \forall p \in \mathcal{P} \tag{10}$$

$$q_{pivt}, w_{p0it} \geq 0 \qquad \forall p \in \mathcal{P}, i \in \mathcal{N}_0, v \in V \tag{11}$$

$$x_{i0tv} \in \{0, 1, 2\} \qquad \forall i \in \mathcal{N}_0, v \in V \tag{12}$$

$$x_{ijtv} \in \{0, 1\} \qquad \forall i, j \in \mathcal{N}_0, v \in V \tag{13}$$

$$y_{itv} \in \{0, 1\} \qquad \forall i \in \mathcal{N}_0, v \in V \tag{14}$$

The objective function (4) is to minimise the cost of routing and direct shipments. Constraints (5) defines the total quantity supplied to a given customer d'_{pit} with respect to the delivery modes. Constraints (6) ensure that vehicle capacity is not exceeded and constraints (7) stipulate that CW's vehicle supplies quantities only to customers allocated to a visit. Constraints (8) and (9) are respectively degree and sub-tour elimination constraints. The conservation conditions of inventory at the central warehouse are expressed by constraints (10). Constraints (11)-(14) state the conditions of non-negativity and integrality on the variables.

4.2. *Transshipment and substitution model*

After the routing decisions have been constructed based on the (r_{pit}, S_{pit}) system, in TSM, the objective is therefore to use transshipment and substitution as emergency measures whenever demands that have been revealed exceeds the quantity of products made available to each customer. In TSM, an inventory level I_{pit-1} refers to the initial inventory per product at the beginning of each period of the rolling horizon. That is, TSM is solved for each period t after demands have been revealed.

For each period t, TSM is formulated as follows:

$$\min \left[\sum_{i\in\mathcal{N}_0} \sum_{p\in\mathcal{P}} h_{pi} I_{pit} + \sum_{i\in\mathcal{N}_0} \sum_{p\in\mathcal{P}} f_{pi} l_{pit} + \sum_{i\in\mathcal{N}_0} \sum_{(p,s)\in\mathcal{P}} a_{sp} z_{spit} + \alpha \sum_{(i,j)\in\mathcal{N}_0, i\neq j} c_{ij} \sum_{p\in\mathcal{P}} w_{pijt} \right] \quad (15)$$

Subject to:

$$I_{pit-1} = \mathscr{I}_{pit-1} + d'_{pit} - D_{pit} \qquad \forall p \in \mathcal{P}, i \in \mathcal{N}_0 \quad (16)$$

$$I_{pit} = I_{pit-1} + l_{pit} + \sum_{j\neq i\in\mathcal{N}_0} (w_{pjit} - w_{pijt}) + \sum_{s\neq p\in\mathcal{P}} (z_{spit} - z_{psit}) \qquad \forall p \in \mathcal{P}, i \in \mathcal{N}_0 \quad (17)$$

$$0 \leq I_{pit} \leq K_i \qquad \forall p \in \mathcal{P}, i \in \mathcal{N}_0 \quad (18)$$

$$I'_{pit-1} = I_{pit-1} + \sum_{s\neq p\in\mathcal{P}} (z_{spit} - z_{psit}) \qquad \forall p, s \in \mathcal{P}, i \in \mathcal{N}_0 \quad (19)$$

$$0 \leq l_{pit} \leq -\min[0, I'_{pit-1}] \qquad \forall p \in \mathcal{P}, i \in \mathcal{N}_0 \quad (20)$$

$$0 \leq w_{pijt} \leq \min[\max[0, I'_{pit-1}], -\min[0, I'_{pjt-1}]] \qquad \forall p \in \mathcal{P}, i, j \in \mathcal{N}_0 \quad (21)$$

The objective function (15) is to minimise the cost of inventory holding, lost sales, substitution and transshipment costs. Constraints (16) define actual inventory level after demands is revealed. Constraints (17) state that the inventory level at the end of period of a product p at the level of each customer i is computed using the actual inventory level at i, quantities transshipped from and to customer i and the difference between the quantity of product s used as substitute of p and the quantity of p used as a substitute of the other products. Constraints (18) impose bound on inventory level. Constraints (19) define the inventory level after products' substitutions have taken place. Constraints (20) state that if the initial inventory of a product p is non-negative, both boundaries are equal to zero, and thus $l_{pi} = 0$, i.e. no demand is lost; otherwise the number of lost units is maximum -$I'pit - 1$. Constraints (20) state that if the initial inventory of a product p is non-negative, then no demand is lost, and both boundaries are equal to zero; otherwise, a minimum of zero and a maximum of I'_{pit-1} units are lost. Similarly, for each product p, constraints (21) place limits on the transshipment arc flows. For customers i and j, there are four possible combinations of inventory levels, all of which can be managed by these constraints:

- $I'_{pit-1} < 0$ and $I'_{pjt-1} < 0$: no transshipment is possible since there is not enough inventory to ship to j.
- $I'_{pit-1} \geq 0$ and $I'_{pjt-1} < 0$: I_{pjt-1} is the upper bound on the arc of the emergency transshipment from i to j.
- $I'_{pit-1} < 0$ and $I'_{pjt-1} \geq 0$: no transshipment is needed since customer j does not need

LT and i does not have enough inventory.

- $I'_{pit-1} \geq 0$ and $I'_{pjt-1} \geq 0$: no transshipment is needed since customer j has enough inventory.

5. Genetic Algorithm and Deep Reinforcement Learning

The classical Vehicle Routing problem (VRP) is NP-hard (Laporte, 2009). Consequently, exact methods can fail to find optimal solutions for large-size problems. In view of the complexity of the RM, our approach is, therefore, to use first a metaheuristic, namely Genetic Algorithm (GA) to determine routing decisions. Unlike Neighborhood Search Algorithms known for their propensity to deliver solution which are only local optima, GA is an efficient computational tool which known for its simplicity, great global search ability and adaptable topology (Baker and Ayechew, 2003). On the other hand, metaheuristics in general and GA in particular, through their iterative search processes generate a lot of data that can be turned to explicit knowledge if coupled with machine learning models (Talbi, 2020). This data relates to decision-making solutions and the objective spaces visited during the search process, solution or trajectory sequence, successive solution populations, movements, recombination, local optima, elite solutions, bad solutions, etc. In fact, machine learning techniques may assist in analyzing this data, learning useful knowledge and guiding to improve metaheuristics' search performance and speed. Thus, techniques for metasearch are *data-driven*, *well informed* and therefore *smarter*. In this paper, to speed up our GA we use Deep Q-learning (DQ) which combines reinforcement learning (RL) and deep learning techniques. We further explain these steps in the following sections.

Once RM is solved in the current period, the solution is used as a parameter for TSM which is solved exactly using CPLEX with default parameters. To do so, we use a matheuristic (noted DQ-GA) in a rolling horizon framework which hybridises the exact method and GA. The pseudo code of this procedure is provided in Algorithm 1.

Algorithm 1 DQ-GA executed for each period t

initialise
Input: RM's parameters, GA's parameters, DQ's parameters
$\mathscr{G} \leftarrow$ Generate an initial population using 2-opt heuristic
for all X in $\mathscr{G}$ **do**
    $F(X) \leftarrow \frac{1}{OF(X)}$
    Execute the constraints violation procedure
end for
repeat
    Select chromosomes from $\mathscr{G}$
    Execute the crossover operator based on DQ
    Execute mutation operator based on DQ
    Measure the fitness value $F(X)$ of the newly created chromosomes
    Execute the constraints violation procedure
    Execute cloning operator for $\mathscr{G}$
    $\mathscr{G} \leftarrow$ Construct new population comprising the 20% best chromosomes of the previous population and newly created chromosomes
until no improvement of the solution is noted or a time limit is reached
Retrieve the best solution found for RM and use the obtained values as parameters in TSM
Solve TSM using CPLEX
return transshipment, substitution and inventory-related solution values

We now describe these steps in detail.

5.1. *Genetic Algorithm*

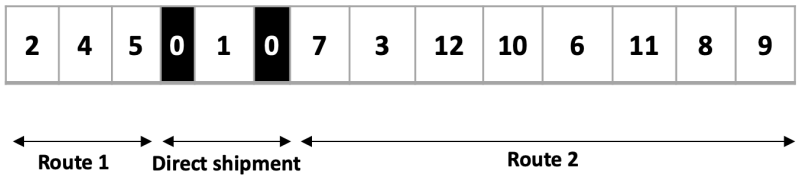
The algorithm begins with a set of initial solutions called population. For each slice time of the rolling horizon, each individual in the population is referred to as a chromosome, reflecting the sequence of assigned customers to each vehicle. During each generation, the fitness of each solution is measured, and solutions are evaluated and selected for cloning, crossover and mutation operations based on their fitness (computed using objective function values).

5.1.1. Chromosomes encoding

In this paper, each chromosome X is represented using a one-dimensional array of integer values, representing the nodes (customers) to be visited (see Figure 2). A repair heuristic is used to check the RM constraints. It ensures for instance that no customer with a non-zero d'_{pi} is missing on the routes or it belongs to several routes.

5.1.2. Generating the initial population

To generate an initial population for GA, we use a variant of 2-opt heuristic, an algorithm based on the conditional permutation of nodes (Sabba and Chikhi, 2012). The heuristic begins by randomly selecting two nodes in a tour and allowing permutation between segments as long as the total cost is reduced. Also, this permutation relies on inter-route moves. That is, we permit swapping nodes that belong to different tours. This process is repeated until tours are optimised.



[Representation of a chromosome]Representation of a chromosome for an RM with 12 customers, 2 vehicles, routes and direct shipment.

Figure 2. Representation of a chromosome for an RM with 12 customers and 2 vehicles.

5.1.3. Fitness function

The fitness function of a chromosome X is calculated from the objective function $OF(X)$ of RM as follows:

$$F(X) = \frac{1}{OF(X)} \tag{22}$$

5.1.4. Genetic Operators

In this algorithm, the following operators are used:

- Cloning operator which retains the best solutions found so far. The trade-off between the performance of the algorithm and its speed led to pick the best 20% of the present population of chromosomes to be copied into the next generation.
- Parent selection operator which uses a binary selection process that begins with two chromosome pairs. Two chromosomes are selected randomly from the existing population each. For crossover operations, the two best chromosomes are selected for each pair. This leads to two children, each counting in the new population.
- Crossover operator which is necessary to mate the chromosome pairs so that they produce their offspring. This paper implements double-point crossover to guarantee the preservation of the best chromosomes. A crossover is performed on the basis of a P_C probability.
- Mutation operator is a second operator used to explore new neighbours. It consists of producing random alterations in different chromosomes. A reversal mutation is used since it is shown to be effective (Zhang et al., 2010). A random set of two nodes are selected, and the nodes between are reversely ordered. Just like the crossover, the mutation process is performed with a P_M probability. Accordingly, each node in chromosome is checked for possible mutation by generating a random number between zero and one, and if this number is less than or equal to P_M, the node value is changed.

5.1.5. Constraints violation penalty

In order to respect the constraints of the model, a simple penalty strategy is adopted. In other words, the feasibility of each chromosome is tested in light of the violations of the model constraints during the generation of initial solutions along with the crossover and mutation operations. If there is an infeasibility in the solution, then the value of the fitness function of the corresponding chromosome is correlated with a penalty. In this way, infeasible chromosomes are less likely to integrate the next generation of chromosomes.

GA stops when a time limit is reached or no improvement in the quality of the solution is noted.

5.2. *Deep Reinforcement*

In this section, we present the deep RL algorithm used to speed up the convergence of GA.

5.2.1. Q-learning

Q-Learning is an RL off-policy algorithm characterised by its strong self-adaptability and its environmental feedback signals (Alom et al., 2019). The main idea is to use the feedback signal to adjust an agent's action policy to optimise its choice when interacting with an environment. By performing actions (i.e., genetic operators), the agent (a chromosome here) arrives in different conditions known as states. Actions contribute to rewards that can be positive and negative. The idea behind Q-learning consists of putting the agent in sequences of state-action pairs, observing the resulting rewards, and adjusting the predictions of a table (called a Q-table) to those rewards until correctly predicted by the best policy. The "Q" stands for quality, which measures how beneficial a given action is in achieving a potential reward.

An agent communicates with the environment in one of two ways: exploration and exploitation (Silver, 2015). Exploration consists of allowing the agent to choose randomly the action it will take regardless of the reward, while in exploitation, the agent uses the Q-table and chooses an action depending on the maximum reward. Initially, the exploration rate noted ϵ (also called ϵ-greedy policy) is set to 1 as all the actions have a Q-value of 0. As the agent starts to learn more about its environment, ϵ is decayed by a certain rate so that the probability of exploration decreases.

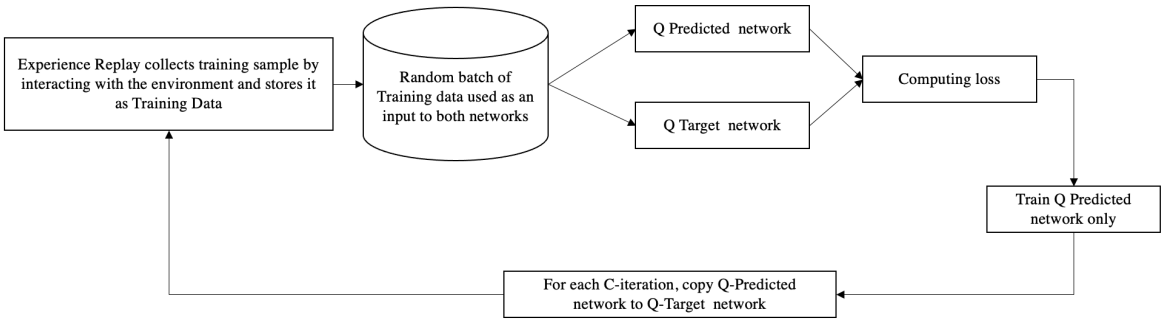
5.2.2. Deep Q-learning

Q-learning is a very simple and efficient algorithm for our GA to construct a Q-table with. This allows the latter to figure out exactly the best actions to perform for crossover, cloning and mutation operators in terms of the best moves. However, it could be time consuming since the amount of memory needed to save and update the table will increase as the number of states increases and the amount of time needed to explore each state to build the appropriate Q-table would be impractical. Since computational time is our primary concern, we estimate these Q-values with deep learning models, namely neural networks, which is known as DQ (Adams et al., 2021; Zhang et al., 2010). Indeed, to approximate the Q-value function, we use a neural network. This function maps a state to the Q values of all the actions that can be taken from that state. It learns the network's parameters (weights) such that it can output the optimal Q-values. Choosing the correct action means comparing the possible rewards of each action and choosing the best one.

5.2.3. DQ exploitation

As it is depicted in Figure 3, DQ starts with random Q-value estimations and uses the ϵ-greedy policy to explore the environment. DQ improves its Q-value estimations by using the same concept of dual actions, a current action with a current Q-Predicted value and a target action with a Target Q-value. As the network and its weights are identical, the direction of the predicted Q-Target values also changes; they remain unchanged but may fluctuate following each update. The stabilisation of the Q-Target values is ensured by using a second network

which is not trained. The learned weights from the Q-Predicted Network are copied to the Q-Target network after a pre-set number of steps noted C-iteration. From Figure 3, we can see that there are two neural networks in the DQ architecture (Q-Predicted and Q-Target) and an agent Experience Replay. Replay Experience interacts with the environment for data generation for Q-network training. This information contains all moves carried out by GA's operators and saved as <st,a,$\mathscr{R}$,st'> tuples (see notation below Equation 23). Then a sample is picked randomly from this data such that it consists a mix of older and more recent samples. This batch of training data is used in the Q-Predicted and Q-Target networks. The Q-Predicted network takes the current state and move out of each sample, and for that particular move predicts the Q value. Q-Predicted value, Q-Target value and the observed data sample reward are used to compute the loss for the Q network training (see Equation 23). To reduce variance and guarantee the stability of the algorithm, in C-iteration, a batch of data is selected from all prior experiences.



[DQ architecture]Two neural networks in the DQ architecture (Q-Predicted and Q-Target) and an agent Experience Replay that interacts with the environment for data generation for Q-network training.

Figure 3. DQ architecture

$$Loss = [\mathscr{R}_{t+1} + \gamma \max_{a}(\theta^T Q(st', a') - \theta^T Q(st, a))]^2 \tag{23}$$

Where:

- γ: discount-rate parameter. It measures how much weight the future awards are given.
- a, a': current and future action respectively.
- st, st' : current and future state respectively.
- $\mathscr{R}_{t+1}$: future reward.
- $Q(st, a)$: learned action-value function.
- θ^T: Transpose matrix of network weights.

Finally, to further speed up the GA, all genetic moves we have gotten as of yet are stored. Instead of "starting from scratch" every time the algorithm is run to solve the RM either for the current instance, or for a different period, or for a new given instance, which happens to be similar to the chromosomes that have already been treated, we use the "memory" to rapidly exploit the best optimal policies. The selection of the best moves depends on how the instance to be solved is similar to the previously addressed instances. K-Nearest Neighbours algorithm is used to determine clusters of instances that are closer to a given "new and unseen" instance

(Mohammed et al., 2017).

6. Computational Results

6.1. *Experimental design*

First, to test the effectiveness and validate the proposed resolution approach, we perform experiments on a set of well-known benchmark instances developed for the single-vehicle and product DSIRP with and without transshipment. It is composed of 150 instances proposed by Coelho et al. (2014a): 5 to 200 customers and planning horizon of 5 to 20 periods for a total of 10 instances for each set of customers. The instances follow some standards defined for the deterministic IRP instances of Archetti et al. (2012, 2007b), namely the mean customer demand, initial inventories, vehicle capacity and distances matrix. The demand follows a normal distribution, the mean of which is generated as an integer random number after an interval of discrete uniform distribution [10, 100], and the standard deviation is generated as an integer random number after an interval of discrete uniform distribution [2, 10]. A negative demand value, if generated, is replaced by zero. The maximum inventory capacity is a multiple of the average demand, and initial inventories are equal to the maximum capacity minus the average demand. In the interval [0.02, 0.10], holding costs are generated randomly from a continuous uniform distribution, and the shortage penalty cost equals 200 times the cost of holding. Finally, the unit cost α is set to 0.01 and the vehicle capacity to $1.5 \sum_{i \in \mathcal{N}_0} \sum_{p \in \mathcal{P}} u_{pi}$. Where u is the expected demand. All instances available from: https://www.leandro-coelho.com/instances/inventory-routing/ and solutions are retrieved from Coelho et al. (2014a). Finally, a fair comparison between algorithms, hardware benchmarking is used in order to compare the speed of the algorithms. The reported CPU of the matheuristic is thus recalculated to align the computational time with regards to the performance of the computer used in Coelho et al. (2014a). Further information on the CPU speed of both computers can be found on: www.cpubenchmark.net.

Second, to evaluate the DSIRPTS for multi-products and multi-vehicle and highlight the benefit of transshipment and substitution on the supply chain's overall performance, we consider a set of randomly generated instances. The generation precisely follows the standards defined for the single-vehicle-product DSIRP. As the supplier has a fleet of homogeneous vehicles, the vehicle capacity no longer needs to be expressed as a function of the expected demands. In this paper, we consider a set of 10 homogeneous vehicles, each having a capacity of Q=2000 units. For products' substitution, we consider a constant unit cost of $c = 0.1\$/\text{product}$ (for identical products: $s = p$, $c = 0\$/\text{product}$). All steps of optimisation were carried out with a personal computer (MacBook Pro, macOS Big Sur, 3.3 GHz Quad-Core Intel Core i7 CPU with 8 GB of RAM) and with CPLEX 12.9 for the resolution of TSM and Python for RM, Python and Pytorch for DQ.

6.2. *Parameters tuning*

Sophisticated optimisation algorithms typically require a large number of parameters to be set in order to enhance their performance. The immediate purpose of the automated configuration of the algorithm is to automatically find the optimiser's best parameter settings. Automatic configuration of algorithms essentially has the ability to contribute to new design paradigms for optimisation applications. The Irace package is a software package that implements a variety of automated configuration procedures (López-Ibáñez et al., 2016). It provides particularly

iterated racing procedures which have been used effectively to automatically configure various state-of-the-art algorithms. Irace's iterated racing processes include the iterated F-race algorithm and several improvement and extensions. In this paper, a set of training instances representing the problem (40 instances with 5, 10, 15... 50 customers each) is used to choose the best algorithm configuration (see Table 2). The selected algorithm configuration can then be used to solve new instances of the same problem.

Table 2. Parameters tuning using Irace package

Parameters	Range	Chosen values
Crossover probability P_C	[0.60,0.81]	0.7
Mutation probability P_M	[0.33,0.37]	0.34
Population size	[100;140]	110
Maximum number of iteration	[100;140]	110

6.3. *Computational results*

6.3.1. Computational results for the single-vehicle-product DSIRP with and without transshipment

In this section, we present the results of the experiments performed on the set of small to large scale dataset generated by Coelho et al. (2014a). To assess the performance of the matheuristic, we compare it with the result obtained using the best known ALNS of Coelho et al. (2014a) developed to solve a single vehicle-product DSIRP. A statistical analysis using ANOVA is also conducted in order to assess the randomness or not of the differences between the obtained results (p-value > 0.05). For each size of instances (small, medium and large) we present the average total cost and the CPU time in second. Results are summarised in the Table 3.

First, for all 150 instances under consideration, we notice that on average our algorithm provides better solutions both in terms of CPU and costs than those of Coelho et al. (2014a) apart from small instances, with a very small gap between the costs. Thus, our algorithm is found to be competitive and efficient compared to the most known state of the art algorithm applied to solve a such specific DSIRPT under a reactive policy. Finally, as it is expected, we notice that sharing inventories between customers helps to significantly reduce the lost sales and thus total costs.

Table 3. Summary of comparison between the results obtained by Coelho et al. (2014a) and this paper

Instances	Average cost				Average CPU (s)			
	Coelho et al. (2014a)		DQ-GA		Coelho et al. (2014a)		DQ-GA	
	DSIRP	DSIRPT	DSIRP	DSIRPT	DSIRP	DSIRPT	DSIRP	DSIRPT
Small	10,225.9	7,926.7	9,473.6	8,788.9	46.3	44.6	42.7	11.6
Medium	30,360.7	26,527.1	27,797.7	26,244.4	452.7	444.1	125.2	129.0
Large	61,250.2	54,292.4	50,550.0	47,352.2	3,860.1	4,100.1	136.6	127.0
Average	33,945.6	29,582.0	29,273.8	27,461.8	1,453.0	1,529.6	101.5	89.2

6.3.2. Computational results for the multi-vehicle-product DSIRP with substitution and transshipment

We now evaluate the impact of transshipment and substitution on solution cost for a more realistic DSIRP. We first consider DSIRP for 20 products and compare the results obtained for DSIRP without transshipment (DSIRP), DSIRP with transshipment (DSIRPT), DSIRP with substitution (DSIRPS) and finally DSIRP with transshipment and substitution (DSIRPTS). Later, we apply the same logic for 40 products. The aim is to confirm the representativeness of the results, highlight the benefits of both transshipment and substitution and to re-evaluate the

accuracy of the proposed algorithm. The generation of the 300 instances (150 for the of 20 and 40 products respectively), follows exactly the standards defined for the single-vehicle-product DSIRP. Results are summarised in the Tables 4 and 7. Also, for an illustrative purpose, Tables 5 and 6 report the breakdown of total costs namely: transportation, transshipment, inventory, substitutions and lost sales cost. The nomenclature of instances follows that of Coelho et al. (2014a): each instance is generated 5 times. 3-4-2 thus refers to the second instance consisting of 3 customers and 4 periods.

Table 4. Summary of computational results for 20 products

Instances	Average cost				Average CPU (s)			
	DSIRP	DSIRPT	DSIRPS	DSIRPTS	DSIRP	DSIRPT	DSIRPS	DSIRPTS
Small	112,479.0	90,647.5	88,195.7	73,306.5	169.2	182.3	176.9	141.5
Medium	332,970.5	265,625.6	260,424.7	214,757.3	279.2	280.2	291.0	218.6
Large	601,088.3	490,336.0	472,007.6	389,921.1	307.4	247.4	293.0	263.9

Table 5. Breakdown of costs for a number of customers varying between 5 and 15 and a number of periods between 1 and 20 (20 products)

Instance	Transportation	Transshipment	Inventory	Substitution	Lost sales	Total cost
5-5-1.txt	12,024	2,812	1,893	2,817	1,549	21,095
5-5-2.txt	11,097	2,672	1,475	1,346	1,025	17,615
5-5-3.txt	7,838	1,537	1,281	980	1,006	12,642
5-5-4.txt	5,720	1,280	757	1,093	526	9,377
5-5-5.txt	10,097	2,253	1,235	1,471	971	16,026
5-10-1.txt	12,646	2,971	1,788	1,426	1,242	20,073
5-10-2.txt	11,991	2,394	1,779	1,580	1,288	19,033
5-10-3.txt	13,341	3,878	2,649	2,113	1,841	23,824
5-10-4.txt	15,118	3,093	2,531	1,972	2,070	24,784
5-10-5.txt	18,028	4,420	2,153	2,784	1,692	29,077
5-20-1.txt	45,864	10,380	6,902	12,110	5,207	80,462
5-20-2.txt	39,415	8,214	5,998	5,421	4,525	63,572
5-20-3.txt	30,386	7,090	4,278	5,662	3,227	50,643
5-20-4.txt	35,879	9,175	4,239	6,056	3,468	58,817
5-20-5.txt	40,886	11,081	5,137	8,319	3,875	69,298
10-5-1.txt	20,962	4,801	2,918	2,478	2,113	33,273
10-5-2.txt	20,618	4,398	2,879	4,206	2,262	34,363
10-5-3.txt	12,048	2,773	1,518	2,169	1,242	19,750
10-5-4.txt	18,588	3,203	2,017	3,267	1,522	28,597
10-5-5.txt	11,948	2,708	2,109	1,682	1,466	19,913
10-10-1.txt	35,697	6,782	5,077	6,491	3,528	57,576
10-10-2.txt	35,077	11,024	5,093	7,276	4,167	62,638
10-10-3.txt	34,773	6,366	3,782	6,055	2,521	53,497
10-10-4.txt	24,531	4,554	2,589	4,622	2,034	38,329
10-10-5.txt	42,985	9,913	5,544	4,708	4,014	67,164
10-20-1.txt	65,252	16,538	10,873	12,285	7,556	112,504
10-20-2.txt	72,421	18,829	10,555	10,602	8,293	120,701
10-20-3.txt	53,299	10,045	6,874	6,156	5,624	81,998
10-20-4.txt	95,696	20,190	12,396	18,857	9,740	156,879
10-20-5.txt	77,009	18,730	12,314	13,914	8,557	130,524
15-5-1.txt	14,952	3,898	2,211	3,118	1,601	25,779
15-5-2.txt	20,843	6,551	3,125	4,618	2,083	37,220
15-5-3.txt	23,444	5,470	3,357	4,470	2,333	39,074
15-5-4.txt	23,209	3,999	2,522	3,994	1,982	35,706
15-5-5.txt	17,191	4,062	3,146	2,365	2,373	29,138
15-10-1.txt	45,246	12,778	7,255	6,795	5,936	78,010
15-10-2.txt	42,919	11,332	6,482	9,470	5,093	75,296
15-10-3.txt	52,269	11,172	6,890	6,745	4,594	81,670
15-10-4.txt	39,809	9,682	5,729	8,271	3,981	67,473
15-10-5.txt	42,037	8,760	6,591	5,442	4,972	67,802
15-20-1.txt	99,142	22,819	13,858	15,821	10,888	162,528
15-20-2.txt	81,608	16,322	15,081	12,948	10,054	136,013
15-20-3.txt	60,237	15,662	9,355	8,084	7,057	100,395
15-20-4.txt	84,307	18,748	12,669	18,724	8,446	142,893
15-20-5.txt	81,834	25,311	12,464	14,934	9,026	143,568

Table 6. Breakdown of costs for a number of customers varying between 125 and 200 and a number of periods between 1 and 20 (20 products)

Instance	Transportation	Transshipment	Inventory	Substitution	Lost sales	Total cost
125-5-1.txt	72,582	19,355	10,792	10,742	7,499	120,971
125-5-2.txt	99,770	19,568	12,782	19,543	9,256	160,919
125-5-3.txt	92,134	19,199	12,754	15,653	8,863	148,603
125-5-4.txt	62,202	15,016	9,413	15,954	6,541	109,126
125-5-5.txt	84,576	19,936	15,109	20,467	10,941	151,029
125-10-1.txt	131,385	22,639	17,780	16,356	13,970	202,131
125-10-2.txt	191,388	45,219	25,517	43,012	19,250	324,386
125-10-3.txt	185,475	43,572	21,209	29,411	14,738	294,405
125-10-4.txt	109,188	30,885	16,241	26,904	11,761	194,978
125-10-5.txt	142,719	39,671	21,030	22,613	15,864	241,897
125-20-1.txt	352,853	84,343	43,613	55,407	32,901	569,117
125-20-2.txt	321,885	72,258	55,037	67,549	38,246	554,975
125-20-3.txt	283,157	63,710	33,830	39,182	22,553	442,433
125-20-4.txt	287,268	80,183	45,914	55,976	34,637	503,978
125-20-5.txt	326,592	82,114	69,623	52,348	52,522	583,199
150-5-1.txt	120,356	28,471	18,557	16,482	13,438	197,305
150-5-2.txt	82,792	16,409	11,048	16,436	9,040	135,725
150-5-3.txt	99,676	23,551	15,909	19,201	10,606	168,943
150-5-4.txt	90,557	17,948	14,282	13,982	11,685	148,454
150-5-5.txt	96,354	18,108	12,941	15,777	9,762	152,943
150-10-1.txt	136,464	39,078	25,147	26,852	20,575	248,117
150-10-2.txt	232,048	47,699	28,701	40,748	19,134	368,331
150-10-3.txt	145,821	32,631	16,968	30,300	13,332	239,051
150-10-4.txt	259,283	51,857	50,093	37,510	33,396	432,138
150-10-5.txt	241,995	53,425	25,855	31,521	19,504	372,301
150-20-1.txt	309,588	72,400	53,244	66,071	41,834	543,137
150-20-2.txt	359,755	93,279	70,830	60,603	57,952	642,420
150-20-3.txt	363,111	100,932	56,093	52,990	42,316	615,442
150-20-4.txt	397,142	82,076	72,892	54,806	54,988	661,904
150-20-5.txt	411,929	96,771	53,882	52,256	39,018	653,856
200-5-1.txt	114,984	29,564	18,324	17,619	14,397	194,888
200-5-2.txt	100,311	32,008	14,809	24,532	10,724	182,384
200-5-3.txt	122,907	25,612	15,870	21,379	12,469	198,237
200-5-4.txt	114,141	25,111	17,727	19,883	13,373	190,235
200-5-5.txt	108,411	25,614	16,707	21,878	11,138	183,748
200-10-1.txt	227,646	40,976	24,424	42,667	19,984	355,697
200-10-2.txt	208,150	61,240	32,213	39,272	24,301	365,175
200-10-3.txt	189,240	44,606	34,971	41,633	27,478	337,928
200-10-4.txt	335,300	97,476	57,775	68,049	40,149	598,749
200-10-5.txt	397,885	78,037	58,670	64,673	42,485	641,749
200-20-1.txt	519,553	131,680	99,776	78,255	66,517	895,782
200-20-2.txt	459,192	86,299	62,754	75,188	45,443	728,876
200-20-3.txt	533,211	122,726	83,781	78,545	55,854	874,117
200-20-4.txt	492,376	105,623	57,121	94,155	44,881	794,155
200-20-5.txt	509,961	115,646	70,396	68,919	57,596	822,518

We can see from Tables 4, 5 and 6 that the results of comparison confirm that any reduction in total cost is made possible by either considering transshipment between customers or substitution of products. Moreover, considering transshipment combined with substitution enhances considerably the performance of the overall supply chain. By substituting products, less inventory is being held and by sharing further their inventory, customers are allowed to meet better their demands and decrease the lost sales and inventory costs. This is reconfirmed in the case of 40 products as it is shown in Table 7. On average, both DISIRPT and DSIRPS may lead to the same results as they both can be used to mitigate shortage, lower inventory and transportation costs. Combining these two emergency measures allows for a considerable reduction of costs for all the instances under consideration. Finally, the algorithm again proves to be efficient and competitive as it allows to find a solution in a reasonably short amount of

time.

Table 7. Summary of computational results for 40 products

Instances	Average cost				Average CPU			
	DSIRP	DSIRPT	DSIRPS	DSIRPTS	DSIRP	DSIRPT	DSIRPS	DSIRPTS
Small	958,123.8	814,014.9	808,906.6	643,880.2	396.6	378.1	385.2	386.5
Medium	2,828,370.3	2,427,250.4	2,401,466.8	1,922,767.0	565.4	564.9	517.4	594.4
Large	5,157,985.9	4,389,631.7	4,382,510.6	3,493,632.6	1,127.3	1,176.2	1,123.3	1,192.0

6.3.3. Computational results for other demand patterns

Intermittency of demands of products such as spare part can be characterised by the infrequent demands that occur at irregular intervals, often of variable size. Modelling demand from constituent components, i.e. the demand size and inter-demand interval, is thus preferable. Compound theoretical distributions (which explicitly take into account the combination of size and interval) are therefore commonly used in these application contexts (Conceição et al., 2015; Syntetos et al., 2012; Turrini and Meissner, 2019). In this paper, d_{pit} represents demand that each customer i has to satisfy per product p and per period t. Different distributions have been studied. We have chosen discrete distributions as they provide a better fit for intermittent demands compared to the continuous ones. According to Syntetos et al. (2012) these distributions are: (1) Poisson Distribution (PD) for demand occurrence, combined with demands of constant size over the planning horizon. (2) Stuttering Poisson distribution (SPD) which is a combination of a Poisson distribution for demand occurrence and a Geometric distribution for demand size over the planning horizon. (3) Negative Binomial Distribution (NBD) which a combination of a Poisson distribution for demand occurrence and a Logarithmic distribution for demand size over the planning horizon.

For $\zeta = 0, 1, 2, ...$ the distribution functions can be written as:
Poisson distribution occurrence PD_λ:

$$PD_\lambda(\zeta) = \frac{\lambda^\zeta \ e^\zeta}{\zeta!} \tag{24}$$

Stuttering Poisson distribution $SPD_{(\lambda,\omega)}(\zeta)$:

$$SPD_{(\lambda,\omega)}(\zeta) = \sum_{1 \leq i \leq \zeta} e^\zeta \frac{\lambda^\zeta}{i!} \binom{\zeta - 1}{i - 1} \omega^i (1 - \omega)^{\zeta - i} \tag{25}$$

where λ and ω are the Poisson and geometric distribution parameters.
Negative Binomial distribution $NBD_{(r,l)}(\zeta)$:

$$NBD_{(r,l)}(\zeta) = \binom{\zeta + r - 1}{\zeta} l^r (1 - l)^\zeta \tag{26}$$

where r is the number of successes, and l is the probability of success.

We used the Inverse Transform Sampling (algorithm 2) to generate independent and identical distributed (i.i.d.) random sample for d_{pit} realizations for each distribution under consideration.

Algorithm 2 Inverse Transform Sampling

1: **procedure** ITS(F) ▷ F is a distribution function
2: $\chi \leftarrow$ Generate random number from the standard uniform distribution in $[0, 1]$;
3: $\zeta \leftarrow F^{-1}(\chi)$
4: **end procedure**

To get more insight about the advantage of both transshipment and substitution, we conduct extra experiments on 150 instances generated based on the experimental design. We consider a set of customers varying between 5 and 200, periods between 5 and 20 and a number of products of 40. For each customer, period and product, Poisson and geometric distribution parameters λ and ω as well as the NBD parameters r and l are random numbers generated between 0 and 1. Table 8 reports the summary of computational results for the different demand's distributions under consideration.

Table 8. Computational results for the different distribution patterns

Distribution	Instances	Average cost				Average CPU			
		DSIRP	DSIRPT	DSIRPS	DSIRPTS	DSIRP	DSIRPT	DSIRPS	DSIRPTS
PD	Small	579,620.0	510,199.0	512,898.8	395,199.3	172.5	206.2	185.5	191.6
	Medium	1,717,250.6	1,514,218.5	1,515,691.4	1,157,961.0	913.9	895.7	879.7	975.9
	Large	3,094,277.0	2,717,394.1	2,692,526.4	2,104,930.6	1,785.5	1,920.2	1,671.7	1,723.1
	Average	1,797,049.2	1,580,603.9	1,573,705.6	1,219,363.6	957.3	1,007.4	912.3	963.5
SPD	Small	659,197.4	582,248.4	580,904.5	450,600.6	203.2	183.5	187.6	206.8
	Medium	1,870,371.9	1,661,277.5	1,649,561.1	1,281,203.8	201.5	747.4	947.4	939.4
	Large	3,590,243.8	3,175,741.8	3,141,875.6	2,407,310.5	1,818.8	1,996.7	1,714.0	1,744.5
	Average	2,039,937.7	1,806,422.6	1,790,780.4	1,379,705.0	741.2	975.9	949.7	963.5
NBD	Small	706,205.3	626,912.6	617,771.5	472,025.9	180.1	180.0	219.1	181.3
	Medium	2,081,957.6	1,845,195.7	1,820,708.2	1,391,390.4	739.7	827.9	982.4	690.8
	Large	3,772,317.6	3,338,938.7	3,287,841.0	2,539,575.3	1,860.2	1,800.6	1,952.2	1,690.9
	Average	2,186,826.8	1,937,015.6	1,908,773.6	1,467,663.9	926.7	936.2	1,051.2	854.3

We can notice that allowing transshipment and substitution permit to reduce considerably the total cost. When these two options are not taken into account, the supply chain experiences a high cost of inventory and loss of sales. Both transshipment and substitution reduce lost sales and inventory holding costs at the level of each customer. When transshipment is permitted, results show that it offers a number of advantages: customers receiving the quantity latterly transshipped, are able to satisfy even more demand and consequently reduce lost sales. Customers from which the transshipment is carried out, are in counterpart able to lower their inventory holding costs. When the substitution is also allowed along with transshipment, compared to the other configurations, we observe a considerable reduction of the costs (to about 32%). In addition to what it can be received through transshipment, each customer is able to use the quantities of products, if compatible, that could constitute idle stock (which leads to high holding cost) to meet the demand of other products. Furthermore, by means of substitution a product of what could be transshipped could be used as a substitute for other products. As for demands patterns, very low size variability and value of demands (as in the case of PD), regardless of the average inter-demand interval may be less stressful than the case when demands experiences high variability and size. Indeed, when demands to satisfy are higher than the available quantity to promise, there would not be enough quantity for substitution and transshipment to lessen any possible lost sales. For this reason, the impact of transshipment and substitution depends whether the distribution of demands to meet may or not be considered as a stressful scenario.

7. Conclusions & Perspectives

In this paper, we consider a two-level supply chain. At the upper level, a manufacturer-owned central warehouse distributes products to a given number of customers (the lower level). We model the problem as a dynamic and stochastic inventory-routing problem that considers the two flexible instruments of transshipment and substitution to mitigate shortages. We assume that lost sales are allowed when shortage occurs. We solve the problem using a new matheuristic which combines the mathematical modeling, the strong global search ability of Genetic Algorithm and the self-adaptability of the Deep Q-learning. The matheuristic is first applied to a set of 150 known instances and is found to be competitive and efficient as it enhances the best known solutions of the single-vehicle-product DSIRP. We later solve the problem for multi-product-vehicle DSIRP. Four demand distributions have been studied, namely Normal distribution, Poisson distribution for demand occurrence, combined with demands of constant size; Stuttering Poisson distribution and Negative Binomial distribution. Regarding the managerial insights, for all the demand patterns under consideration, we demonstrate the benefits of promoting either inventory sharing or substitutions as emergency measures to sidestep shortages. In addition, we show that combining these two flexible instruments can be seen as a viable solution for supply chain managers aiming to improve the system's wide service level under dynamic and stochastic demands. Moreover, results show that the impact of transshipment and substitution on the overall performance depends on the size variability of demands, regardless of the average inter-demand interval.

This paper can be expanded to investigate a multi-echelon either of centralised or decentralised supply chains. One can consider non-parametric methods for demand, whereby the empirical demand distribution is instead directly constructed from the data. The effect of the forecasting method and the resulting error can also be integrated in the model. Also, a stochastic optimisation on demands such as sample average approximation could be applied. In this paper, we have assumed that the lead times are long between the facility and the central warehouse. Additionally stochastic lead time and and production rate can also be investigated. Furthermore, it would be interesting to consider a substitution rate instead of a compatibility matrix. This would allow customers to more explicitly express their preference with regards to the available products.

Acknowledgements

We would like to show our gratitude to the (Prof. Leandro C. Coelho, Ph.D., Canada Research Chair in Integrated Logistics, Laval University) for sharing valuable information about the single-vehicle-product DSIRP data-set. His help is sincerely appreciated. The authors thank the editor-in-chief, the associate editor and three anonymous referees for their constructive comments and encouragements that have helped improve our paper greatly.

Data Availability Statement

The data that support the findings of this study are openly available in the Open Repository and Bibliography (ORBi) at `http://hdl.handle.net/2268/263351`.

References

Abdelmaguid, T. F. (2004). *Heuristic approaches for the integrated inventory distribution problem*. PhD thesis, University of Southern California.

Abdelmaguid, T. F., Dessouky, M. M., and Ordóñez, F. (2009). Heuristic approaches for the inventory-routing problem with backlogging. *Computers & Industrial Engineering*, 56(4):1519–1534.

Achamrah, F. E., Riane, F., and Limbourg, S. (2021). Spare parts inventory routing problem with transshipment and substitutions under stochastic demands. *Applied Mathematical Modelling*.

Adams, D., Oh, D.-H., Kim, D.-W., Lee, C.-H., and Oh, M. (2021). Deep reinforcement learning optimization framework for a power generation plant considering performance and environmental issues. *Journal of Cleaner Production*, 291:125915.

Alegre, J., Laguna, M., and Pacheco, J. (2007). Optimizing the periodic pick-up of raw materials for a manufacturer of auto parts. *European Journal of Operational Research*, 179(3):736–746.

Alom, M. Z., Taha, T. M., Yakopcic, C., Westberg, S., Sidike, P., Nasrin, M. S., Hasan, M., Van Essen, B. C., Awwal, A. A. S., and Asari, V. K. (2019). A state-of-the-art survey on deep learning theory and architectures. *Electronics*, 8(3).

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

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

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

Baker, B. M. and Ayechew, M. (2003). A genetic algorithm for the vehicle routing problem. *Computers & Operations Research*, 30(5):787 – 800.

Bell, W. J., Dalberto, L. M., Fisher, M. L., Greenfield, A. J., Jaikumar, R., Kedia, P., Mack, R. G., and Prutzman, P. J. (1983). Improving the distribution of industrial gases with an on-line computerized routing and scheduling optimizer. *Interfaces*, 13(6):4–23.

Bertazzi, L., Bosco, A., Guerriero, F., and Laganà, D. (2013). A stochastic inventory routing problem with stock-out. *Transportation Research Part C: Emerging Technologies*, pages –.

Bertazzi, L. and Speranza, M. G. (2002). Continuous and discrete shipping strategies for the single link problem. *Transportation Science*, 36(3):314–325.

Burns, L. D., Hall, R. W., Blumenfeld, D. E., and Daganzo, C. F. (1985). Distribution strategies that minimize transportation and inventory costs. *Operations research*, 33(3):469–490.

Christiansen, M. (1999). Decomposition of a combined inventory and time constrained ship routing problem. *Transportation Science*, 33(1):3–16.

Christiansen, M., Fagerholt, K., Flatberg, T., Haugen, Ø., Kloster, O., and Lund, E. H. (2011). Maritime inventory routing with multiple products: A case study from the cement industry. *European Journal of Operational Research*, 208(1):86–94.

Chrysochoou, E., Ziliaskopoulos, A., and Lois, A. (2015). An exact algorithm for the stochastic inventory routing problem with transshipment.

Coelho, L., Cordeau, J.-F., and Laporte, G. (2014a). Heuristics for dynamic and stochastic inventory-routing. *Computers & Operations Research*, 52:55–67.

Coelho, L. C., Cordeau, J.-F., and Laporte, G. (2012a). Consistency in multi-vehicle inventory-

routing. *Transportation Research Part C: Emerging Technologies*, 24:270–287.

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

Coelho, L. C., Cordeau, J.-F., and Laporte, G. (2014b). Thirty years of inventory routing. *Transportation Science*, 48(1):1–19.

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

Conceição, S. V., da Silva, G. L. C., Lu, D., Nunes, N. T. R., and Pedrosa, G. C. (2015). A demand classification scheme for spare part inventory model subject to stochastic demand and lead time. *Production Planning & Control*, 26(16):1318–1331.

Dehghani, M., Abbasi, B., and Oliveira, F. (2021). Proactive transshipment in the blood supply chain: A stochastic programming approach. *Omega*, 98:102112.

Dror, M. and Levy, L. (1986). A vehicle routing improvement algorithm comparison of a "greedy" and a matching implementation for inventory routing. *Computers & Operations Research*, 13(1):33–45.

Grahovac, J. and Chakravarty, A. (2001). Sharing and lateral transshipment of inventory in a supply chain with expensive low-demand items. *Management Science*, 47(4):579–594.

Grønhaug, R., Christiansen, M., Desaulniers, G., and Desrosiers, J. (2010). A branch-and-price method for a liquefied natural gas inventory routing problem. *Transportation Science*, 44(3):400–415.

Hewitt, M., Nemhauser, G., Savelsbergh, M., and Song, J.-H. (2013). A branch-and-price guided search approach to maritime inventory routing. *Computers & Operations Research*, 40(5):1410–1419.

Hssini, I., Meskens, N., and Riane, F. (2016). Blood products inventory pickup and delivery problem under time windows constraints. In *International Conference on Operations Research and Enterprise Systems*, volume 2, pages 349–356.

Huang, S.-H. and Lin, P.-C. (2010). A modified ant colony optimization algorithm for multi-item inventory routing problems with demand uncertainty. *Transportation Research Part E: Logistics and Transportation Review*, 46(5):598 – 611.

Laporte, G. (2009). Fifty years of vehicle routing. *Transportation science*, 43(4):408–416.

Lefever, W., Aghezzaf, E.-H., Hadj-Hamou, K., and Penz, B. (2018). Analysis of an improved branch-and-cut formulation for the inventory-routing problem with transshipment. *Computers & Operations Research*, 98:137–148.

López-Ibáñez, M., Dubois-Lacoste, J., Pérez Cáceres, L., Birattari, M., and Stützle, T. (2016). The irace package: Iterated racing for automatic algorithm configuration. *Operations Research Perspectives*, 3:43 – 58.

Mirzaei, S. and Seifi, A. (2015). Considering lost sale in inventory routing problems for perishable goods. *Computers & Industrial Engineering*, 87:213 – 227.

Mohammed, M. A., Ghani, M. K. A., Hamed, R. I., Mostafa, S. A., Ibrahim, D. A., Jameel, H. K., and Alallah, A. H. (2017). Solving vehicle routing problem by using improved k-nearest neighbor algorithm for best solution. *Journal of Computational Science*, 21:232 – 240.

Paterson, C., Kiesmüller, G., Teunter, R., and Glazebrook, K. (2011). Inventory models with lateral transshipments: A review. *European Journal of Operational Research*, 210(2):125–136.

Peres, I. T., Repolho, H. M., Martinelli, R., and Monteiro, N. J. (2017). Optimization in inventory-routing problem with planned transshipment: A case study in the retail industry. *International Journal of Production Economics*, 193:748–756.

Roldán, R. F., Basagoiti, R., and Coelho, L. C. (2016). Robustness of inventory replenishment and customer selection policies for the dynamic and stochastic inventory-routing problem. *Computers & Operations Research*, 74:14 – 20.

Ronen, D. (2002). Marine inventory routing: Shipments planning. *Journal of the Operational Research Society*, 53(1):108–114.

Sabba, S. and Chikhi, S. (2012). Integrating the best 2-opt method to enhance the genetic algorithm execution time in solving the traveler salesman problem. *Advances in Intelligent and Soft Computing*, 170:195–208.

Savelsbergh, M. and Song, J.-H. (2008). An optimization algorithm for the inventory routing problem with continuous moves. *Computers & Operations research*, 35(7):2266–2282.

Silver, D. (2015). RL exploration and exploitation.

Solyalı, O., Cordeau, J.-F., and Laporte, G. (2012). Robust inventory routing under demand uncertainty. *Transportation Science*, 46:327–340.

Strack, G., Fortz, B., Riane, F., and Van Vyve, M. (2011). Comparison of heuristic procedures for an integrated model for production and distribution planning in an environment of shared resources. LIDAM Discussion Papers CORE 2011016, Université catholique de Louvain, Center for Operations Research and Econometrics (CORE).

Stålhane, M., Andersson, H., Christiansen, M., Cordeau, J.-F., and Desaulniers, G. (2012). A branch-price-and-cut method for a ship routing and scheduling problem with split loads. *Computers and Operations Research*, 39:3361–3375.

Syntetos, A. A., Babai, M. Z., and Altay, N. (2012). On the demand distributions of spare parts. *International Journal of Production Research*, 50(8):2101–2117.

Talbi, E.-G. (2020). Machine learning into metaheuristics: A survey and taxonomy of data-driven metaheuristics. working paper or preprint.

Turrini, L. and Meissner, J. (2019). Spare parts inventory management: New evidence from distribution fitting. *European Journal of Operational Research*, 273(1):118–130.

Yu, Y., Chu, C., Chen, H., and Chu, F. (2013). Large scale stochastic inventory routing problems with split delivery and service level constraints. *Annals of Operations Research*, 197:135–.

Zhang, D., Liu, Y., M'Hallah, R., and Leung, S. C. (2010). A simulated annealing with a new neighborhood structure based algorithm for high school timetabling problems. *European Journal of Operational Research*, 203(3):550 – 558.

Zhao, Q.-H., Chen, S., and Zang, C.-X. (2008). Model and algorithm for inventory/routing decision in a three-echelon logistics system. *European Journal of Operational Research*, 191(3):623–635.