Agri-Food Supply Chains with Stochastic Demands: A Multi-Period Inventory Routing Problem with Perishable Products

Bhakti Stephan Onggo[a], Javier Panadero[b,d], Canan G. Corlu[c], Angel A. Juan[b,d]

[a] *CORMSIS, Southampton Business School, University of Southampton, UK*
b.s.s.onggo@soton.ac.uk
[b] *IN3 – Computer Science, Multimedia and Telecommunication Dept.*
Universitat Oberta de Catalunya, Barcelona, Spain
{jpanaderom, ajuanp}@uoc.edu
[c] *Metropolitan College, Boston University, Boston, MA, USA*
canan@bu.edu
[d] *Euncet Business School, Terrassa, Spain*

Abstract

This paper considers an agri-food supply chain with a single fresh food supplier, who owns a central warehouse that serves several retail centers. Retail centers carry a certain amount of inventory of the fresh product, which is prone to deterioration. The supplier makes both inventory and routing decisions to minimize the inventory, transportation, food-waste, and stock-out costs in the face of stochastic customer demand and perishable products that need to be delivered to each retail center. This inventory routing problem is known as perishable inventory routing problem (PIRP) with stochastic demands in the literature. We model it using a mixed integer program and propose a simheuristic algorithm, which integrates Monte Carlo simulation within an iterated local search, to solve it. Our experiments show that the proposed algorithm can improve the initial solution with reasonable computational times. The resulting procedure is easy to implement and is applicable to other domains where a multi-period PIRP with stochastic demands may appear.

Keywords: agri-food supply chain, multi-period inventory routing problem, stochastic demands, perishable product, metaheuristics, simheuristics.

Preprint submitted to Journal Simulation Modelling Practice and Theory. August 27, 2019

1. Introduction

Agri-business plays a critical role in the world economy by serving as a key source of food supply to the world population. Over the last decades, the management of agri-food supply chains has received considerable attention both from practitioners and researchers. There are two types of agri-food supply chains: fresh produce agri-food supply chains and packaged products agri-food supply chains. Our focus in this paper is on the fresh produce agri-food supply chains.

Although fresh produce agri-food supply chains have similarities to conventional manufacturing supply chains, they have important characteristics that make their management more complicated. Specifically, fresh produce agri-food supply chains are characterized by the perishability of the food product, seasonality, long supply lead time, and uncertainty in harvest yield (**?**). Hence, their management mainly deals with transporting agri-food products from the production centres (farms) to the places of consumption at the right time, right quantity, and right quality with minimum cost. The cost includes components such as transportation cost, inventory cost (holding and stock-out), and food-waste cost.

One way to deal with minimizing the food waste is through a better integration between actors in agri-food supply chains. However, the study by **?** reveals that most research has focused on either harvesting or process planning. In general, research into the integration between processes from harvesting to processing and finally to distribution is lacking. This paper addresses the integration between inventory and transportation (or routing) management of a fresh produce agri-food supply chain.

The type of integration where inventory and routing decisions are made concurrently is known as vendor managed inventories (VMI). The benefit of VMI in a business setting has been well recognized in the literature. Back in 1980s, **?** discuss the importance of making the inventory and the routing decisions concurrently when making logistical decisions. **?** analyze the benefit of VMI and use real demand-data from different grocery supply chains to illustrate its value. Since then, there has been a significant amount of research on the use of VMI in several settings. We refer the reader to **?** and **?** for complete reviews of VMI practices.

Despite the popularity of VMI in several areas, the use of VMI practices in the management of agri-food supply chains is very limited. The lack of research into the integration between processes along an-food supply chain is

partly due to its complexity. **?** note that actors in an agri-food supply chain often have a high degree of autonomy with conflicting objectives. They often have limited perspective which makes it difficult for them to envisage how their individual decisions may affect the whole supply chain. Furthermore, their behaviour are often influenced by social, economical, and environmental factors (ibid). The characteristics of fresh agri-food products create further challenges to the implementation of VMI practices.

In this paper, we consider a single fresh food supplier (e.g., a big farm or a co-operative) who owns a central warehouse that serves n retail centers (RCs). Each RC carries a certain inventory of the fresh food product to maintain a target service level (e.g., low number of stock-outs). In retail-managed inventory (RMI) systems, there is no collaboration among RCs and thus each RC makes its own decision about when and how much to order. This practice directly affects the route planning process of the supplier (**??**). By contrast, the implementation of VMI practices takes the pressure away from the RCs to make inventory decisions and to hold inventories, thus allowing the supplier to make both the inventory and the routing decisions. This leads to the overall optimization of the supply chain, benefiting both the RCs and the supplier (**???**).

The implementation of VMI practices in the management of supply chains leads to a combinatorial optimization problem called inventory routing problem (IRP). IRP seeks to design routes by also managing inventories in the RCs. Being an extension of the vehicle routing problem (**?**), the IRP is also *NP-hard*. Therefore, most of the effort to solve this problem has either made simplifying assumptions (e.g., ignoring the natural uncertainty of agri-food supply chains) or has focused on metaheuristic methods (**??**). This paper presents a simheuristic algorithm (**?**) that extends a metaheuristic algorithm to deal with two important characteristics of fresh produce agri-food supply chains: stochastic customer demands at each RC and perishable products that need to be delivered to each RC (Figure 1).

The main contribution of this paper is to propose a new simheuristic algorithm for the multi-period IRP with stochastic demands for a *perishable product*. The stochasticity around customer demands is addressed by integrating the metaheuristic procedure with Monte Carlo simulation. The perishability of the product is addressed by reducing the value of the product over time until a certain duration, beyond which the product will perish. Product perishability adds a new complexity to the conventional IRP. As we know, in IRP, the less frequent deliveries reduce the routing cost at the

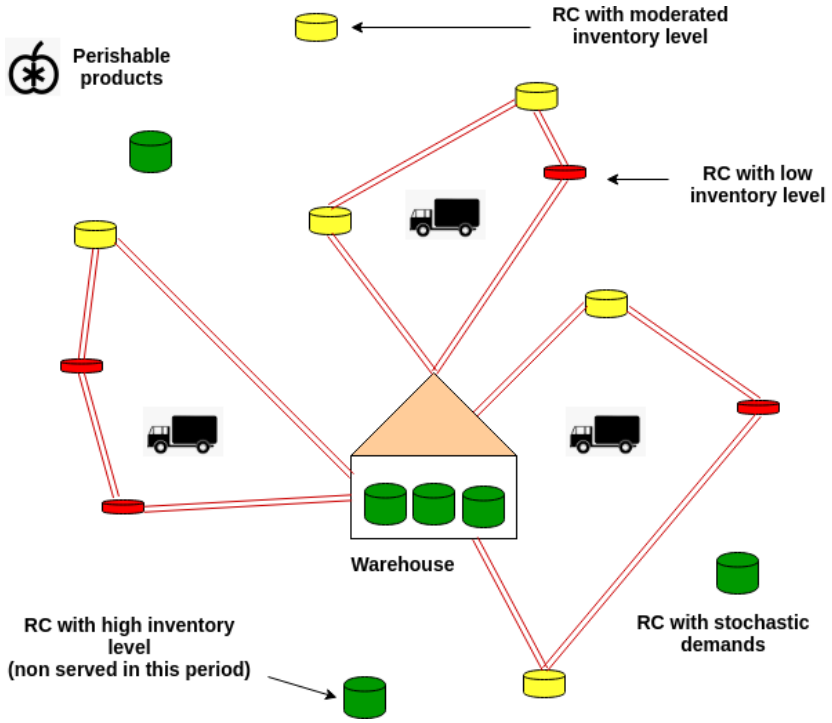

Figure 1: A stochastic inventory routing problem with perishable products.

expense of inventory holding cost. With product perishability (i.e., PIRP), in addition to higher inventory holding cost, the less frequent deliveries also increase the risk of higher product deterioration or product waste cost. It is therefore critical to balance the routing cost and the product deterioration cost. Finally, our algorithm also considers the stock-out cost. The proposed simheuristic algorithm finds the optimal refill policies for each RC in each period by minimizing the total expected cost over the periods. The total expected cost includes inventory holding cost, routing cost, product deterioration cost and stock-out cost. We believe this is the first work that uses a simulation-optimization method to solve a multi-period stochastic PIRP with stock-outs.

The remainder of this paper is structured as follows: relevant literature on the agri-food supply chain and IRP is reviewed in Section 2; a mathematical formulation of the multi-period IRP with stochastic demands for a perishable food product is provided in Section 3; Section 4 outlines the proposed algorithm; a range of computational experiments are described and

analyzed in Section 5; finally, Section 6 concludes this work and highlights future research lines.

2. Literature Review

This section provides an overview of the literature on inventory management and transportation in agri-food supply chains, focusing on how models are used to help making decisions. This is followed by a closer look at the perishable IRP problem and its solution from cases in agri-food supply chain and supply chain in general. We conclude the section with a discussion around simheuristic approaches for solving combinatorial optimization problems including stochastic IRPs.

2.1. Inventory and Transportation Management in Agri-Food Supply Chains

Fresh food products from farms to customers undergo various processes such as cultivation, harvest, and processing (e.g., washing and packing). The products are stored and transported at various points in the supply chain (**?**). For example, the crop yield from the harvest may be stored briefly at the farm before being transported to a processing facility. The final products from the processing facility are stored and then transported to retail centres (or customers). Hence, inventory and transportation management is an important part of fresh produce agri-food supply chain management, especially with the products being perishable. Pressures from regulatory bodies and consumers on issues such as food safety, traceability, and conformance (e.g., organic, environmental) have increased the complexity of inventory and transportation management of fresh food products.

From an inventory management perspective, the proliferation of fresh food products has added more complexity to the issue (**?**). For example, in the past, there were only commodity corn and high-value specialty corn. Nowadays, there are many different types of corn in the market. The different varieties of food products make the issue of product substitutions more prevalent, hence adding more complexity to the inventory management decisions. Global sourcing has an impact on agri-food supply chains, too. It is now possible to transport fresh food products from one continent to another at an acceptable transportation cost. Hence, the distance between supply chain actors has increased. At the same time, this globalization has also created more competition which affects profit margin. Global sourcing has also added more complexity to the routing decisions due to restrictions in

terms of local constraints –such as regulations and practices. In short, the challenges to transportation and inventory management in agri-food supply chain have become more complex. This is where Operational Research (OR) techniques may help.

The application of OR techniques in the context of agri-food supply chains has been reviewed in several papers. **?** review the application of OR on crop production (e.g., planning, harvesting, managing risk, etc.). Decisions surrounding crop production in which OR has been applied include optimum allocation of lands, optimum allocation of resources during harvest, and optimum cultivation to control weeds. They have found that linear programming is by far the most popular approach. In addition to production planning, **?** also include the distribution planning of crop-based food products in their review. They have also found that linear programming is the most popular method indicating that stochasticity has not been considered in most work. They have also found that only a few papers account for perishability of food products, and that there is a lack of studies integrating inventory and routing decisions for fresh product agri-food supply chains. **?** further reinforce the findings from the previous reviews that very few papers have addressed the inventory management issues of agri-fresh products, especially in incorporating the perishability of food products and stochastic demand. **?** focus on how OR techniques have been used to account for stochasticity in agri-food supply chains. They have observed that studies that incorporate stochasticity is growing with the most popular techniques being stochastic programming followed by robust optimization. The number of work that uses a hybrid simulation-optimization model to solve problems in agri-food supply chains is limited. *This paper fills in this gap in the literature by developing a simheuristic procedure, which combines a metaheuristic with Monte Carlo simulation, to solve the integrated inventory management and transportation problem that arises in fresh produce agri-food supply chains.*

2.2. The Perishable IRP

The term *inventory routing problem* first appeared in **?**. Since then, different variants of IRPs have been introduced. We refer the reader to **?**, **?**, **?**, and **?** for a comprehensive discussion of the IRP and its several variants. In this section, we restrict our focus to the variant of the IRP where perishable products are considered: the perishable inventory routing problem (PIRP).

Although the PIRP has first been motivated by a healthcare application (**???**) due to the perishability of the human blood, several papers then looked

at the PIRP in a more general context where the perishable items could be food or medical drugs. To the best of our knowledge, this paper is the first to study PIRP (with stochastic demands) in the context of agri-food supply chains.

Although the work on PIRP is scarce, several variants of it have already been proposed, including: multiple-product PIRP, PIRP with sustainability issues, PIRP with production decisions, PIRP with facility location decisions, and PIRP with transshipments. **?** provide a comprehensive review of these variants. In the following, we review the literature on single-product PIRP and classify the papers based on the nature of the demand that they assume. More specifically, Section 2.2.1 reviews the papers that study the single-product PIRP with a deterministic demand, while Section 2.2.2 reviews papers addressing stochastic demand in the context of a single-product PIRP. Throughout this section, we keep our focus on the papers that make a theoretical contribution to the solution of the PIRP.

2.2.1. The PIRP with deterministic demands

Much of the work on the single-product multi-period PIRP assumes that the demands at the delivery points (retail centers, hospitals, supermarkets, etc.) are deterministic. Motivated by the inventory routing problem faced by the blood bank of the Austrian Red Cross for Eastern Austria, **?** study a multi-period PIRP and propose two solution approaches based on integer programming and variable neighborhood search. The goal is to minimize the travel cost over a finite horizon, where additional constraints are considered due to the nature of blood. Examples of additional constraints include: *(i)* no out-of-stock situations; *(ii)* no vehicle capacity constraints; *(iii)* limited length of a route; and *(iv)* no inventory holding costs.

? consider a multi-period PIRP with deterministic demands in the context of the Academic Model for the Prevention and Treatment of HIV (AMPATH) program, which aims to provide food support to HIV-infected patients by transporting fresh foods from production farms to distribution sites. It is assumed that perishable goods have a fixed shelf life and are discarded after they expire. Authors model the PIRP as a mixed integer program whose goal is to minimize the sum of transportation costs and inventory holding costs. They propose a heuristic based on column-generation that can solve small- to medium-size problems.

? consider the problem in **?**, but without inventory holding costs. The authors present an arc-based formulation, which features a small number

of variables and a large number of constraints. This particular formulation enables the use of a tabu search algorithm to solve the problem, which is shown to outperform the column-generation based heuristic of **?** in terms of computational time.

? continue the study of single-product multi-period deterministic PIRP by formulating it as a mixed integer linear program and devising a branch-and-cut algorithm to solve the model. The model is general enough to accommodate different types of product perishability constraints (products that expire after a certain time versus products whose value degrades gradually over time) and the way retailer handles the aging inventory, i.e.: always sell the oldest available items first to avoid spoilage versus always sell the fresher items first to increase revenue because the selling price of the item depends on its age.

Recently, **?** find an inconsistency in the arc-based formulation presented in **?**. Authors propose four mathematical formulations and devise branch-and-cut algorithms to solve these formulations. Additionally, an iterated local search metaheuristic is proposed for larger instances.

? further consider the realistic situation that the end customers' demand depends on the age of the product and, therefore, a portion of the inventory that is not sold by the end of the period is considered lost sale. Authors model this problem as a mixed integer linear program whose goal is to minimize the sum of transportation costs, holding costs, and lost sales –where lost sales are assumed to be a linear or exponential function of the inventory age. Small instances are shown to be solved to optimality, while larger instances are solved via metaheuristic algorithms based on simulated annealing and tabu search.

2.2.2. The PIRP with stochastic demands

The literature on stochastic PIRP is extremely limited. **?** is the first to study the PIRP with random demands. These authors devise a heuristic algorithm to solve the problem but the method only works for a single-period. Very recently, **?** study the multi-period stochastic PIRP and propose four different solution methods to solve the problems that differ in their sophistication and the features considered, including: perishability of the product, stochasticity of demand, and the target service level. The distinguishing feature of our work is the design of a simheuristic method which allows the decision maker to solve a multi-period IRP with stochastic customer demands at each retailer center for perishable food product. *To the best of our knowl-*

edge, this is the first paper that proposes a simulation-optimization algorithm for the multi-period PIRP with stochastic demands.

2.3. Simheuristics for Combinatorial Optimization Problems

2.3.1. Simheuristic as a simulation-optimization technique

Simulation-optimization techniques have been extensively used both by practitioners and academics to solve complex optimization problems that involve uncertainty. **?** provide an excellent summary of simulation-optimization techniques used particularly for inventory replenishment decisions. **?** further review the applications of simulation-optimization techniques in fields including manufacturing and production, logistics and supply chain management, and healthcare.

Simheuristics is a special case of simulation- optimization techniques. As stated in **?** "Simheuristic algorithm is a particular simulation-optimization approach oriented to efficiently tackle a combinatorial optimization problem instance that typically contains stochastic components." By integrating Monte Carlo simulation into a metaheuristic optimization framework, simheuristic approaches are able to solve large-scale stochastic NP-hard problems in reasonable computational times. Another benefit of simheuristic approaches is their ability to provide performance statistics other than expected value, which are critical for risk-analysis purposes.

Due to their advantages over several other simulation- optimization techniques, simheuristic algorithms have been used extensively to solve large-scale, stochastic NP-hard combinatorial optimization problems. In the remainder of this section, we focus on the use of the simheuristic technique to solve stochastic IRPs and refer the reader to **?** and **?** for a thorough review of simheuristic approaches in different application areas.

2.3.2. Simheuristics to solve stochastic IRPs

? is the first to develop a simheuristic algorithm to solve a single-period stochastic IRP with stock-outs. The idea is to combine simulation with a vehicle-routing metaheuristic, where simulation is initially used to obtain the expected costs associated with each combination of retail center and refill policy of the retail center and a routing metaheuristic is then used for each refill policy to estimate the total cost including the inventory cost and the routing cost. **?** later extend this work to multi-period case, which bears additional complexities due to the interdependences between consecutive periods. This paper also deals with a multi-period IRP with stochastic demands but with

an additional consideration of perishable products. To the best of our knowledge, this is the first paper that solves a stochastic PIRP with a simheuristic algorithm, which provides good solutions in reasonable computational times.

3. Problem Description

The fresh produce agri-food supply chain problem considered in this paper is as follows. We develop the model from the perspective of an agri-food vendor that supplies a fresh produce product to several RCs, as shown in Figure 1. The RCs share their inventory level and expected demand information with the vendor. This is common in a VMI setting, where the vendor is responsible for the routing and inventory decisions. Fresh produce products include fruits, vegetables, flowers, and herbs that have not been processed, or altered by any preservation process, before being packaged. Hence, the quality of a fresh produce product degrades over time. After a certain time, final customers will not buy the product and it will be considered waste. The goal of the vendor is to minimize the total supply chain cost, which includes inventory holding cost, routing cost, product deterioration cost and stock-out cost. This problem is known as the multi-period stochastic PIRP with stock-outs.

This problem can be formulated as a mixed integer linear program (MILP). The symbols for data and variables used in the model are listed in Table 1 and will be explained next. In the following, we use the word "product" to refer to the fresh produce product.

We let $V = \{0, 1, \ldots, n\}$ denote a finite set of $n+1$ locations consisting of the depot (node 0) and n RCs. The set of RCs is denoted by $V^* = V \setminus \{0\}$. The planning horizon is denoted by P where $|P| \geq 1$ periods. The set of durations in which a product has been stored in the inventory (i.e., the age of the product) of an RC is denoted by B where $|B|$ is the longest duration that a product can be stored in the inventory. The product is discarded after $|B|$ periods.

The decision variables for the inventory management are the quantity of products, $q_{ip} \geq 0$, that must be delivered to RC i at the beginning of period p ($\forall i \in V^*$, $\forall p \in P$). $L_i^+ > 0$ denotes the maximum storage capacity of RC i. L_{ip} and L'_{ip} denote the inventory level at RC i at the start and the end of period p, respectively ($0 \leq L_{ip}, L'_{ip} \leq L_i^+$). Then, $q_{ip} \leq L_i^+ - L_{ip}$ follows. Since the quality of the products degrades over time, in period p, the

Table 1: Data and variables employed in the model.

Symbol	Indices	Type	Meaning
P	$p = 1, ..., \|P\|$	Data	Set of periods included in the planning horizon
B	$b = 0, 1, ..., \|B\|$	Data	Set of the durations in which a product has been in the inventory of RC i
V	$i, j = 0, 1, ..., n$	Data	Set of locations, including the depot 0 and n RCs
V^*	$i, j = 1, ..., n$	Data	Set of RCs ($V^* = V \setminus \{0\}$)
K	$k = 1, ..., K$	Data	Set of vehicles
w_b	$b = 0, 1, ..., \|B\|$	Data	Deterioration cost associated with product age b
c_{ij}	$i, j \in V$	Data	Cost of traveling from $i \in V$ to $j \in V$
$h > 0$		Data	Total capacity of vehicles
$L_i^+ > 0$	$i \in V^*$	Data	Maximum storage capacity of RC i
L_{ip}	$i \in V^*, p \in P$	Var	Inventory level at RC i at the start of period p
L'_{ip}	$i \in V^*, p \in P$	Var	Inventory level at RC i at the end of period p
l_{ipb}	$i \in V^*, p \in P, b \in B$	Var	Inventory level of products age b at RC i at the start of period p
l'_{ipb}	$i \in V^*, p \in P, b \in B$	Var	Inventory level of products age b at RC i at the end of period p
D_{ip}	$i \in V^*, p \in P$	Var	Customers' demand at RC i in period p
d_{ipb}	$i \in V^*, p \in P, b \in B$	Var	Customers' demand at RC i in period p to be fulfilled by product age b
$q_{ip} \geq 0$	$i \in V^*, p \in P$	Var	Quantity of fresh produce delivered to RC i at the start of period p
$y_{ip} \geq 0$	$i \in V^*, p \in P$	Var	Binary variable that defines if RC i is visited in period p
x_{ij}^{pk}	$i, j \in V, k \in K, p \in P$	Var	Binary variable that defines if vehicle k goes from i to j in period p

aggregated inventory L_{ip} at RC i can be split into the inventory of products with different ages ($b \in |B|$), as stated in Equation 1:

$$L_{ip} = \sum_{b \in B} l_{ipb} \tag{1}$$

The initial inventory level for the first period ($p = 1$) is given as an input, $L_{i1} = l_{i10}, \forall i \in V^*$ (i.e., all products in the initial inventory are fresh). As for the remaining periods ($p > 1$), the inventory level of product age b at RC i at the start of period p is shown in Equation 2. We assume that all products delivered in period p to all RCs are fresh (i.e., $b = 0$). This assumption is common in agri-food supply chain literature (**?**). The degradation speed is controlled by $|B|$ in which a smaller value means that the product degrades faster.

$$l_{ipb} = \begin{cases} q_{ip} & \text{if} \quad b = 0 \\ l'_{i(p-1)(b-1)} & \text{if} \quad 0 < b \leq |B| \end{cases} \tag{2}$$

The customers' demand at each RC $i \in V^*$ during a period $p \in P$ is a random variable, D_{ip}, which follows a known probability distribution. In this work, it is assumed that these random demands are independent across RCs and throughout the periods –although they do not need to be identically distributed. We assume that each RC always satisfies its demand using the older products first. Hence, the demand for product age b is:

$$d_{ipb} = \begin{cases} \max(D_{ip} - l_{ipb}, 0) & \text{if} \quad b = |B| \quad \text{i.e., oldest} \\ \max(d_{ip(b+1)} - l_{ip(b+1)}, 0) & \text{if} \quad 0 \leq b < |B| \end{cases} \tag{3}$$

Once the customers' demand at RC i in period p is known, the stock level of product age b at the end of period p can be computed using Equation 4. The aggregated inventory level at RC i at the end of period p is given in Equation 5. As the product deteriorates, its value decreases. We model this by applying a penalty cost w_b to product age b (where $w_0 = 0$ and $w_b < w_{b+1}$). The deterioration cost (W) is calculated using Equation 6.

$$l'_{ipb} = l_{ipb} - \min(d_{ipb}, l_{ipb}) \tag{4}$$

$$L'_{ip} = \sum_{b \in B} l'_{ipb} \tag{5}$$

$$W = \sum_{p \in P} \sum_{i \in V^*} \sum_{b \in B} w_b \cdot l'_{ipb} \tag{6}$$

The inventory cost at RC i in period p can be obtained by using Equation 7, where λ represents the unit cost of holding surplus inventory at the end of period p. The total inventory cost can be calculated using Equation 8.

$$s_{ip} = \lambda L'_{ip} \tag{7}$$

$$S = \sum_{p \in P} \sum_{i \in V^*} s_{ip} \tag{8}$$

As for the routing decision, in each period $p \in P$, a VRP needs to be solved for those RCs with $q_{ip} > 0$ (i.e. we do not need to deliver to an RC that does not need any replenishment). As discussed in **?**, the VRP can be defined on a complete and undirected graph $G = (V, E)$, where V includes the depot from which n demand points (RCs) are served with a set K of homogeneous vehicles, and E is the set of edges connecting each pair of locations in V. Each of the vehicles in the fleet has a maximum loading capacity given by $h > 0$. There is a traveling cost, $c_{ij} = c_{ji} > 0$ associated with a vehicle moving from location i to location j ($\forall i, j \in V, i \neq j$). The routing cost in period p depends on the binary decision variables x_{ij}^{pk}, which define whether or not the edge connecting locations i and j is traversed in period p by a vehicle $k \in K$. Notice that this depends on the realization of the random variables representing the customers' demands D_{ip}. Accordingly, the total routing cost across all periods can be expressed as shown in Equation 9:

$$R(x_{ij}^{pk}, D_{ip} \mid i, j \in V, p \in P, k \in K) = \sum_{p \in P} \sum_{i \in V} \sum_{j \in V} \sum_{k \in K} c_{ij} x_{ij}^{pk} \tag{9}$$

Finally, we assume that the customers' demand at each RC at any period (i.e., D_{ip}) will always be satisfied. Thus, should a stock-out occur during a period p at RC i, i.e., when $d_{ip0} > l_{ip0}$, additional $d_{ip0} - l_{ip0}$ fresh produce will be delivered immediately from the depot. Equation 10 shows the cost associated with the stock-out at RC i in period p. It shows that when a stock-out occurs at RC i, we need to transport the product from the depot to RC i immediately. The total stock-out cost is shown in Equation 11.

$$g_{ip} = \begin{cases} 0 & \text{if} \quad d_{ip0} - l_{ip0} \leq 0 \\ 2 \cdot c_{i0} & \text{if} \quad d_{ip0} - l_{ip0} > 0 \end{cases} \tag{10}$$

$$G = \sum_{p \in P} \sum_{i \in V^*} g_{ip} \tag{11}$$

The objective of our multi-period stochastic PIRP model with stock-out is to find the optimum combination of q_{ip} and x_{ij}^{pk} (i.e. the decision variables) in order to minimize the expected overall cost as formulated in Equation 12, where S is the total inventory cost, W is the total deterioration cost, R is the total routing cost and G is the total stock-out cost.

$$E[S + W + R + G] \tag{12}$$

Constraint 13 ensures that we do not deliver more than the capacity of each RC.

$$0 \leq q_{ip} \leq L_i^+ - L_{ip}^0 \qquad \forall i \in V^*, \forall p \in P \tag{13}$$

Constraints 14 and 15 make sure that we only deliver to RCs that need replenishment. If RC i needs a replenishment in period p (i.e. $q_{ip} > 0$) then $y_{ip} = 1$. Otherwise, $y_{ip} = 0$. M is a very large number.

$$q_{ip} \leq L_i^+ y_{ip} \qquad \forall i \in V^*, \forall p \in P \tag{14}$$

$$y_{ip} \leq M q_{ip} \qquad \forall i \in V^*, \forall p \in P \tag{15}$$

Constraint 16 makes sure that each vehicle k makes one round trip per period p. A round trip always starts from the depot and ends at the depot and the trip is done within one period.

$$\sum_{i \in V^*} x_{0i}^{pk} = \sum_{i \in V^*} x_{i0}^{pk} = 1 \qquad \forall k \in K, \forall p \in P \tag{16}$$

Constraint 17 guarantees that each RC $i \in V^*$ that needs replenishment (i.e. $y_{ip} = 1$) will be visited and the vehicle k that visits the RC in period p will leave the RC on the same period p. Constraint 18 is needed for the sub-tour elimination.

$$\sum_{j \in V \setminus \{i\}} x_{ij}^{pk} + \sum_{j \in V \setminus \{i\}} x_{ji}^{pk} = 2 \cdot y_{ip} \qquad \forall i \in V^*, \forall k \in K, \forall p \in P \tag{17}$$

$$\sum_{i \in S} \sum_{j \notin S} x_{ij}^{pk} \leq 2 \sum_{i \in S} y_{ip} \qquad \forall S \subset V^*, \forall k \in K, \forall p \in P \tag{18}$$

Constraint 19 makes sure that in each round trip, we cannot deliver more than the vehicle capacity.

$$\sum_{i \in V^*} \sum_{j \in V} x_{ij}^{pk} q_{ip} \leq h \qquad \forall k \in K, \forall p \in P \tag{19}$$

Constraints 20 and 21 enforce binary conditions on the auxiliary and routing decision variables, respectively.

$$y_{ip} \in \{0, 1\} \qquad \forall i \in V^*, \forall p \in P \tag{20}$$

$$x_{ij}^{pk} \in \{0, 1\} \qquad \forall i, j \in V, \forall k \in K, \forall p \in P \tag{21}$$

4. Our Simulation-Optimization Approach

We propose a simheuristic method to minimize the expected total cost of the multi-period stochastic PIRP with stock-out as described in the previous section. Our proposed method is comprised of three stages:

4.1. Construction of the initial solution

The main objective of this stage is to generate a decent initial solution in a fast manner. To achieve this, we use a constructive heuristic algorithm that finds the most optimum refill policy when the same policy is applied to all RCs in every period. The algorithm compares several refill policies. For each refill policy, total costs including holding cost, routing cost, deterioration cost, and stock-out cost are estimated using a Monte Carlo simulation. The constructive heuristic is presented in detail in Algorithm 1.

The inputs to the algorithm are the set of RCs and a depot (V^*), the set of time periods (P), the set of duration in which a product has been stored in the inventory (B), the initial inventory levels (L_{i1}^0), the maximum storage capacity of each RC (L_i^+), the random variable D_{ip} that represents the random demand for each RC in each time period, the set of refill policies (T), and the maximum number of simulation runs that must be executed.

The first phase (lines 1 to 22) estimates the inventory cost (holding cost and stock-out cost) and the deterioration cost of each refill policy. For the

initial solution, we consider 11 possible refill policies: 10%-refill policy, 20%-refill policy, ..., 100% (full) refill policy, and, in addition, a no-stock (0%) refill policy in which an RC can only use its current stock to fulfill the demand in the subsequent period. The loop in line 1 iterates each refill policy. In each iteration, the same refill policy will be applied to all RCs (line 4) in every period (line 9).

In line 7, we execute a small number of Monte Carlo simulation runs to estimate the costs associated with refill policy t applied to RC i for all periods P. In each period (the loop in line 9), we first update the age of each product (line 10). When a product reaches its maximum age, it is discarded. Then, we determine the quantity to be delivered to RC i in period p (line 11) taking into account the initial inventory level of the RC and its maximum storage capacity. Next, we generate a realization of the random variable D_{ip}, which is the demand at RC i in period p (line 12). With the generated demand, we can now compute the inventory level at the end of the current period (line 13), which will become the inventory level at the beginning of the next period. In line 14, we compute the inventory cost. Should a stock-out occurs, the inventory cost also includes the stock-out cost which is the cost of a round trip to the depot. In line 15, we compute the deterioration (aging) cost. By the end of the first phase, we obtain the expected inventory and deterioration costs for each refill policy. Figure 2 shows the flowchart of phase 1 to complement the pseudocode in Algorithm 1.

The second phase (lines 23 to 36) estimates the routing cost for each refill policy using the plan generated in the first phase (i.e. the number of products to be delivered to each RC in every period). In phase 1, for each simulation run, we store the number of products that must be delivered to each RC in every period. We use the same data structure as in **?** whereby a matrix of $|V^*|$ rows and $|P|$ columns is used to store the inventory requirement plan. Each cell (i, p) in the matrix represents the quantity to deliver to RC i in period p. We maintain one matrix for each simulation run. We use the quantities from these matrices as the parameters to a biased-randomized version of the savings heuristic reported in **?** to estimate the associated routing cost in period p (line 30). Next, we calculate the total routing cost for the refill policy t (line 34). Finally, the refill policy that produces the lowest expected total cost is chosen as our initial solution (lines 35 and 36). We complement the pseudocode of phase 2 of Algorithm 1 with a flowchart shown in Figure 3.

Algorithm 1: Generate Initial Solution

Inputs:
$V = \{0, 1, \ldots, |V|\}$: Set of depot (0) and retail centers (RCs) (V^*)
$P = \{1, 2, \ldots, |P|\}$: Set of time periods
$B = \{1, 2, \ldots, |B|\}$: Age index of a product
L^0_{i1}: Initial inventory level of RC i in period 1
L^+_i: Maximum storage capacity of RC i
D_{ip}: Random variable for demand at RC i in period p
T: Refill policies as a % of L^+_i (e.g., 0%, 10%, 20%,... 90%, 100%)
$maxRuns$: Maximum number of runs of the Monte Carlo simulation
% Phase 1: Compute expected multi-period inventory costs for each refill policy
foreach *refill policy* $t \in T$ **do**
  $expInvCost[t] \leftarrow 0$ % set expected inventory cost associated with policy t to 0
  $expAgingCost[t] \leftarrow 0$ % set expected deterioration cost associated with policy t to 0
  foreach *RC* $i \in V^*$ **do**
    $accumInvCost \leftarrow 0$ % set accumulated inventory cost to 0
    $accumAgingCost \leftarrow 0$ % set accumulated deterioration cost to 0
    $iter \leftarrow 0$
    while $iter < maxRuns$ **do**
      foreach *period* $p \in P$ **do**
        $updateAging(L^0_{ip}, B)$ % update the age of each product
        $q_{ip}[t][iter] \leftarrow \max\{t \cdot L^+_i - L^0_{ip}, 0\}$ % determine the quantity to deliver to RC i
        $d_{ip} \leftarrow$ generate a realization of demand D_{ip} % generate a demand variable
        $L^0_{i(p+1)} \leftarrow \max\{L^0_{ip} + q_{ip}[t][iter] - d_{ip}, 0\}$ % compute inventory level at the end of the current period
        $invCost \leftarrow computeInventoryCost(t, L^0_{i(p+1)})$ % compute inventory cost for RC i and policy t
        $agingCost \leftarrow computeAgingCost(L^0_{ip})$ % compute deterioration cost for each RC and policy
        $accumAgingCost \leftarrow accumAgingCost + agingCost$ % compute accumulated deterioration cost
        $accumInvCost \leftarrow accumInvCost + invCost$ % compute accumulated inventory cost
      end
      $iter \leftarrow iter + 1$
    end
    $avgInvCostRC \leftarrow accumInvCost/maxRuns$ % average inventory cost of RC i at policy t
    $avgAgingCostRC \leftarrow accumAgingCost/maxRuns$ % average deterioration cost of RC i at policy t
    $expInvCost[t] \leftarrow expInvCost[t] + avgInvCostRC$
    $expAgingCost[t] \leftarrow expAgingCost[t] + avgAgingCostRC$
  end
end
% Phase 2: Compute average multi-period routing cost and total cost for each policy
$initSol \leftarrow emptySol$
$cost(initSol) \leftarrow \infty$
foreach *refill policy* t *in* T **do**
  $accumRoutingCost \leftarrow 0$
  foreach *period* $p \in P$ **do**
    $iter \leftarrow 0$
    while $iter < maxRuns$ **do**
      $routingCost \leftarrow estimateRoutingCost(q_{1p}[t][iter], \ldots, q_{|V|p}[t][iter])$ % use biased-randomized heuristic
      $accumRoutingCost \leftarrow accumRoutingCost + routingCost$
      $iter \leftarrow iter + 1$
    end
    $expRoutingCost \leftarrow accumRoutingCost/maxRuns$
  end
  $totalCost \leftarrow expInvCost[t] + expAgingCost[t] + expRoutingCost$
  if $totalCost < cost(initSol)$ **then**
    $initSol \leftarrow setAllRefillDecisionsToValue(t)$
    $cost(initSol) \leftarrow totalCost$
  end
end
return $initSol$

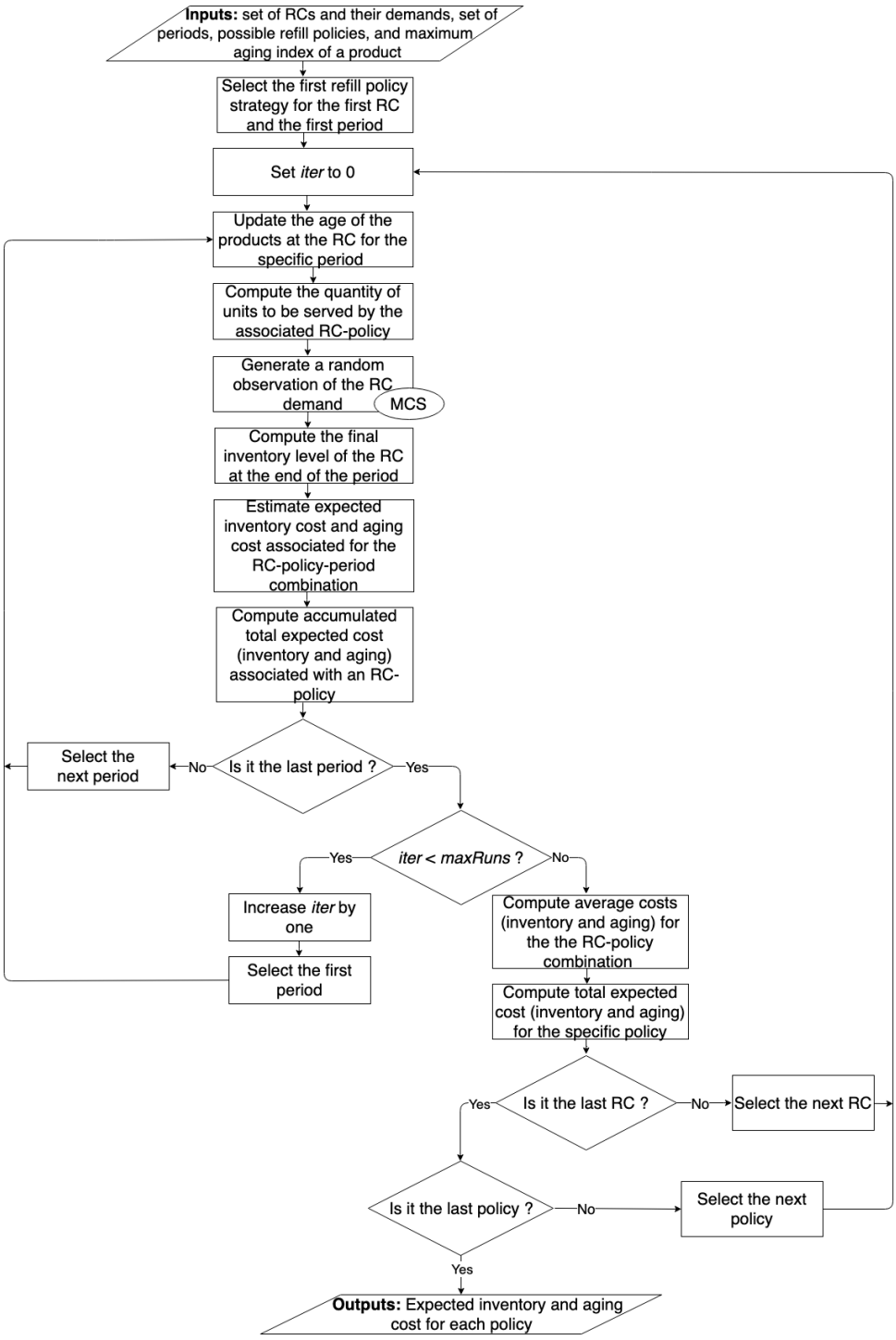

Figure 2: Flowchart for phase 1 of algorithm 1.

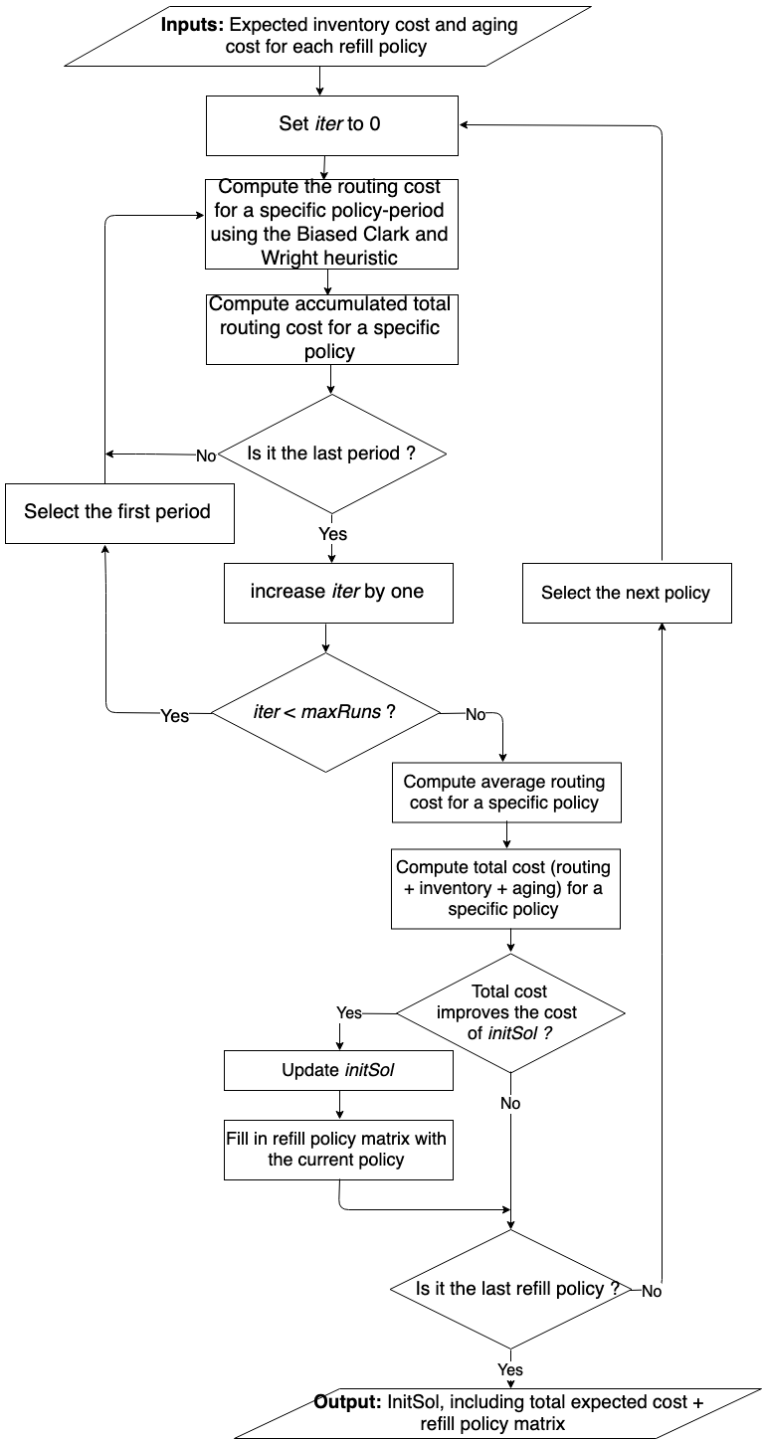

Figure 3: Flowchart for phase 2 of of algorithm 1.

4.2. Local Search stage

The objective of the local search (or local optimization) stage is to improve the homogeneous (global) refill policy obtained from the previous stage by finding the optimum "local" refill policy. The local refill policy refers to the refill policy applied to a specific RC in a specific period (i.e., the cell (i, p) in the matrix described in Section 4.1). This local search method may result in a different refill policy to be applied to each RC in each period. The detailed pseudocode and its flowchart are given in Algorithm 2 and Figure 4, respectively.

The local search algorithm starts by assigning the initial solution *initSol* generated by Algorithm 1 both to the base solution *baseSol*, and to the current best solution *bestSol* (lines 1-2). Then, it randomly picks a retail center and a period (lines 6-7, lines 21-22) without a replacement from $\{V^* \times P\}$ (i.e., a set of all possible combinations of RCs and periods). For each combination (line 8), we compare the base refill policy t with $t - 10\%$ and $t + 10\%$ refill policies (lines 13-19). In line 15, we estimate the total supply chain cost using the same constructive heuristic method described in Section 4.1 (the only difference is that now it only affects a subset of elements in the policy matrix). If the cost of *newSol* improves over the cost of *bestSol*, the current best solution is updated (line 17). If any of the two refill policies outperforms the current best solution *bestSol*, the process (lines 13-19) is repeated by comparing the base refill policy t with $t-20\%$ and $t+20\%$ refill policies, and so on (line 20) until there is no improvement that can be made. The loop (lines 8-22) is then repeated until all the elements in the matrix have been visited.

4.3. Refinement stage

The aim of this stage is to refine the solution achieved by the local search algorithm by applying a large number of Monte Carlo simulation runs to the best solution obtained in the previous stage. Since the simulation process is time-consuming, we limit the number of iterations to 10,000.

Notice that an important advantage of our approach is its relative simplicity and its reduced number of parameters. This significantly reduces the need for fine tuning processes and its sensitivity to particular problem characteristics, providing a robust method that performs efficiently across different instances.

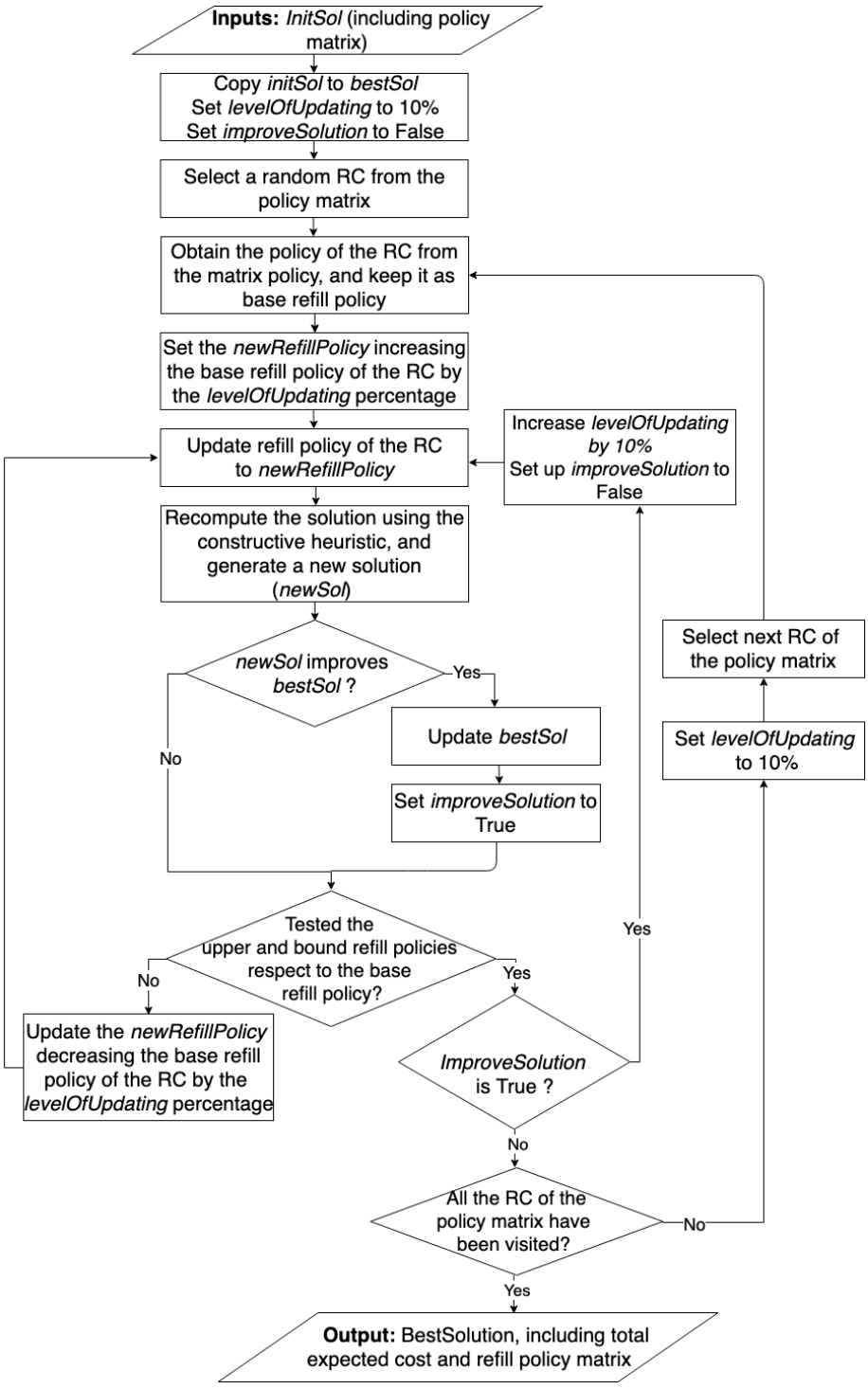

Figure 4: Flowchart for algorithm 2.

Algorithm 2: Local search procedure

Inputs:
$V = \{0, 1, \ldots, |V|\}$: Set of depot (0) and retail centers (RCs) (V^*)
$P = \{1, 2, \ldots, |P|\}$: Set of time periods
initSol: initial solution

```
1  baseSol ← initSol % assign the initial solution generated by Algorithm 1
2  bestSol ← initSol % assign the initial solution generated by Algorithm 1
3  improves ← true
4  levelOfUpdating ← 10%
5  RCs = V*
6  rc ← selectRandRC() % randomly select an RC
7  p ← selectRandPeriod() % randomly select a time period
8  foreach (i, p) in {V* × P} do
9      baseRefillPolicy ← getRefillPolicy(baseSol, rc, p) % iterate the refill
         policy matrix
10     while improves is true do
11         improves ← false
12         newRefillPolicy ← baseRefillPolicy + levelOfUpdating % set the
             new refill policy by increasing the base policy by levelOfUpdating
13         foreach incr = 1 To 2 do
14             newSol ← modifyRefillPolicy(bestSol, rc, p, newRefillPolicy)
                 % generate a new solution by increasing the current refill
                 policy by 10%
15             newSol ← constructiveHeuristic(newSol) % recalculate the
                 solution by using the constructive heuristic in Algorithm 1.
16             if cost(newSol) < cost(bestSol) then
17                 bestSol ← newSol
18                 improves ← true
               end
19             newRefillPolicy ← baseRefillPolicy − levelOfUpdating
           end
20         levelOfUpdating ← levelOfUpdating + 10%
       end
21     rc ← selectNewRC()
22     p ← selectNewPeriod()
   end
23 return bestSol
```

5. Computational Experiments and Analysis of Results

We perform a set of experiments to illustrate the effectiveness of our approach for solving the multi-period stochastic PIRP with stock-out. We use the 27 VRP instances proposed by **?** and adapted for the IRP by **?** and Gruler et al. (2018b). These instances contain a central depot, between 27 and 80 RC nodes, and a fleet of 5–10 homogeneous vehicles. We implement our approach as a Java application and execute it on a workstation with 64 GB of RAM and an Intel Xeon at 3.7 GHz.

The algorithms are set-up with the following parameters:

- λ is set to 0.25 to ensure that the inventory holding cost has the same magnitude as the routing cost (see **?** and Gruler et al. (2018b)) and deterioration cost; hence, none of the terms in the objective function dominates over the others.
- Product age: $|B|$ is set to 3, allowing that each product passes through three different states in its cycle of life (fresh, semi-fresh and expired).
- Deterioration unit cost: We set $w = \{0, 0.4, 1.0\}$. This means that at the end of the second period the product loses 40% of its value; and at the end of the third period the product loses its value completely and is discarded.
- Number of simulation runs in the initial phase: This value is set to a number that is big enough to provide a good and quick estimate for the costs of promising solutions. A number up to 100 provides a good compromise for all instances used in the experiment.
- Number of simulation runs in the refinement phase: After carrying out several tests, this value is set to $10,000$ experimentally.

We assume that the customer demand at RC i in period p follows a Log-Normal probability distribution, i.e. $D_{ip} \sim LogNormal(\mu_{ip}, c \cdot \sigma_{ip})$. Log-Normal distribution is commonly used to model non-negative random variables (**?**). The parameter $c > 0$ is a design parameter that allows us to experiment with different levels of uncertainty. It is expected that as c converges to zero, the results from the stochastic version converge to those obtained in the deterministic scenario. We consider three different levels of uncertainty: low ($c = 0.10$), medium ($c = 0.25$), and large ($c = 0.75$).

Tables 2 to 4 present the results for the case where there are three planning horizons ($P = 3$). The number of retail centers n and vehicles K in each problem setting are reflected in the instance name. Column (1) depicts the results of the initial solution (i.e., generated by the constructive heuristic shown in Algorithm 1). Column (2) shows the total expected cost of our best solution (OBS), generated by our simheuristic method. This expected total cost is disaggregated in the following columns: routing (3), inventory (4-6), and deterioration (7) costs. Furthermore, we break down the expected total inventory cost (4) into stock-out cost (5) and holding (6) cost. Column (8) shows the number of units of product with age 1 or 2 (i.e. semi-fresh units), and (9) shows the total number of units of product which have exceeded their maximum age and have perished. Finally, the last two columns show the computational time to obtain the OBS (in seconds), and the average percentage gaps between the initial solution and the OBS, respectively. Gaps are calculated as follows:

$$Gap\ (Cost_{OBS}, Cost_{InitSol}) = 100 \left(\frac{Cost_{OBS} - Cost_{InitSol}}{Cost_{InitSol}} \right) \qquad (22)$$

A closer look at Tables 2–4 reveals that a higher level of demand-uncertainty leads to a higher total cost due to a higher inventory cost (holding and stock-out costs) and a higher deterioration cost. It can further be inferred from the tables that a higher inventory level increases the risk of higher deterioration cost and product waste.

We have also carried out experiments for five and seven planning horizons ($P = 5$ and $P = 7$). Figure 5 illustrates the average total costs for various planning horizons $P \in \{3, 5, 7\}$ and different variance levels, $c \in \{0.1, 0.25, 0.75\}$ for each horizon. In the same figure, we also compare the expected total cost of the initial solution –in which the same replenishment policy is applied at all retail centers in each period–, and the OBS found by the proposed simheuristic method. It shows that the proposed simheuristic method improves the initial solution provided by the constructive heuristic (i.e. Algorithm 1). It implies that the local search (i.e. Algorithm 2) has done well in reducing the expected total cost.

We present the computational times needed to obtain the OBS in Figure 6. The figure presents the average computing times of all 27 VRP instances in various planning horizons and variance levels. As expected, the average computational time increases with the number of planning periods (i.e., a bigger matrix to solve). We also notice that the levels of variance in demand

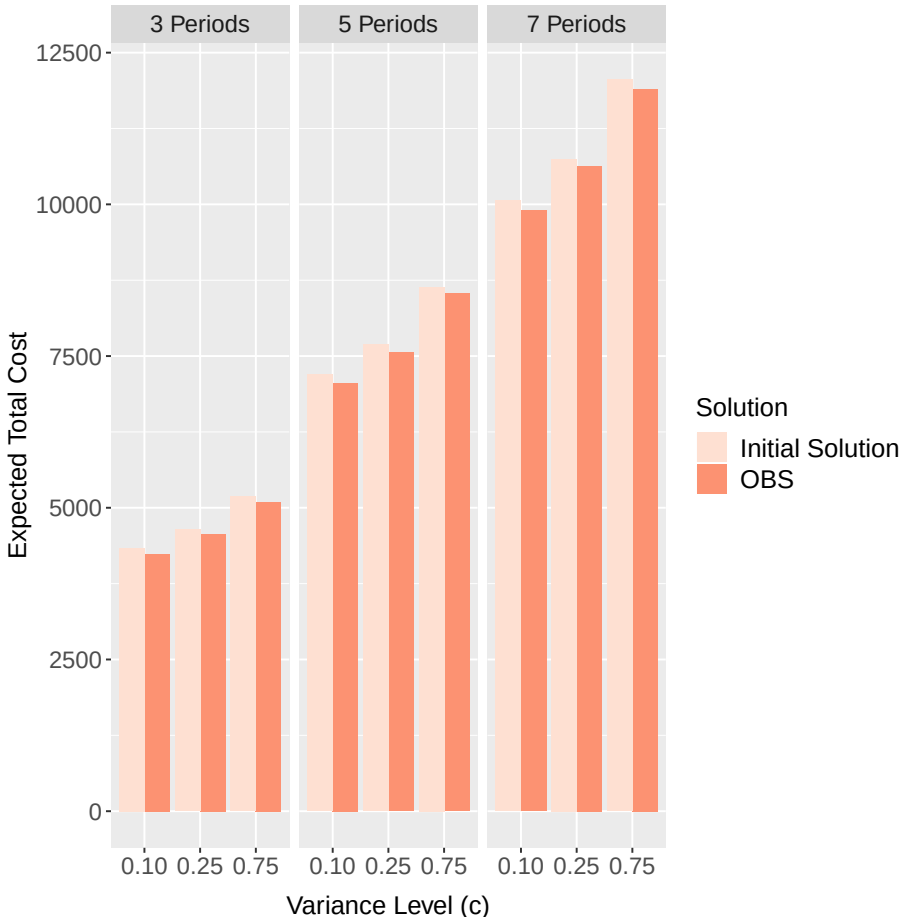

Figure 5: Expected total costs over all instances for different variance levels and planning horizons

does not affect the computational time. This is expected because the variance in the demand does not make the computation more expensive. It simply affects the quality of the solutions.

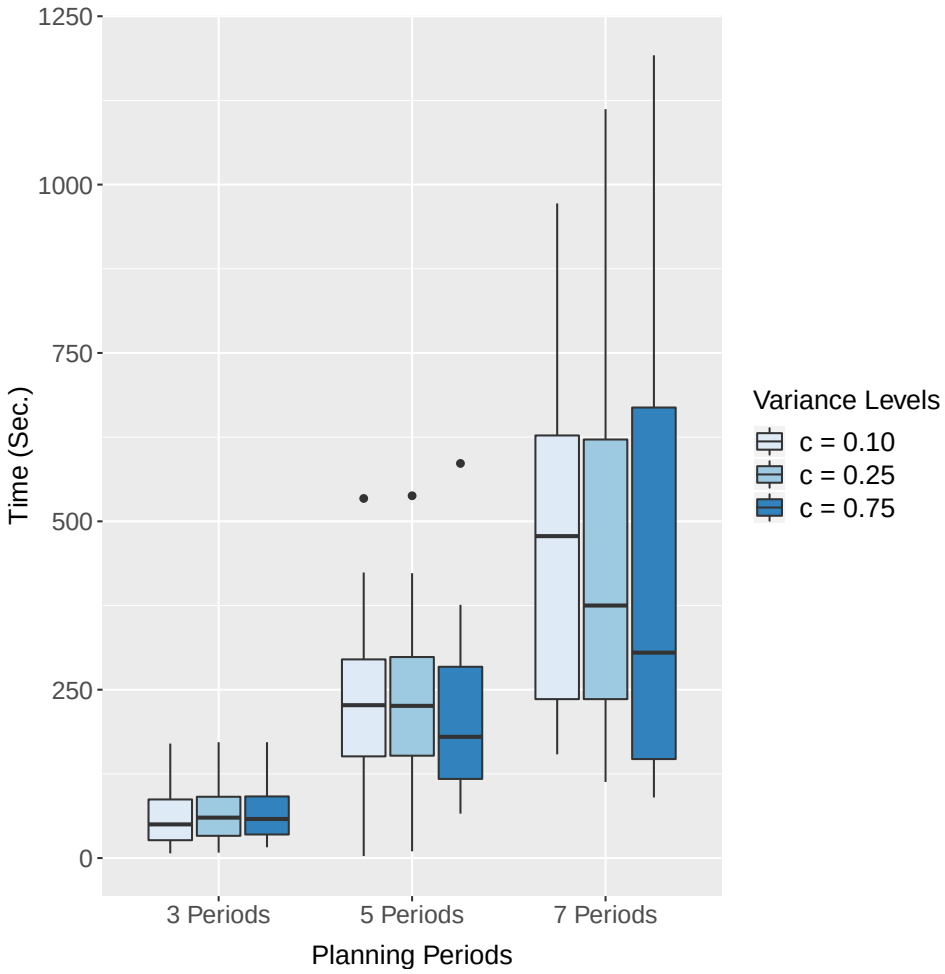

Figure 6: Average computational times for different variance levels and planning horizons

Finally, Figure 7 shows the disaggregated total expected cost when the same refill policy is applied to all RCs in all periods with planning horizon $P = 5$ and a medium variance level ($c = 0.25$). This is shown in the first eleven bars. We compare this with the optimum solution (OBS) from our simheuristic method (the rightmost bar). A number of interesting conclusions can be made from this figure. First, a policy with lower refill rate leads

to a higher inventory cost, mainly due to a higher stock-out cost. The no-refill policy (inventory level at 0%) is a special case, in which all the costs are associated with stock-out costs. Second, as the replenishment level increases, the holding cost becomes more dominant than the stock-out cost. This is expected as stock-outs rarely happen when the inventory level is high. We also observe that deterioration cost increases. This is also expected since a high inventory level means that more products are subject to the deterioration. Hence, it increases the risk of product waste. Third, although the difference in the expected total cost between 50% refill policy applied globally (i.e., to all RCs) and different refill policies applied locally at each RC in each period (i.e., OBS) is not significant, applying refill policies locally leads to a slightly better result due to a lower deterioration cost. Hence, this solution is preferred when the concern about food waste is high.

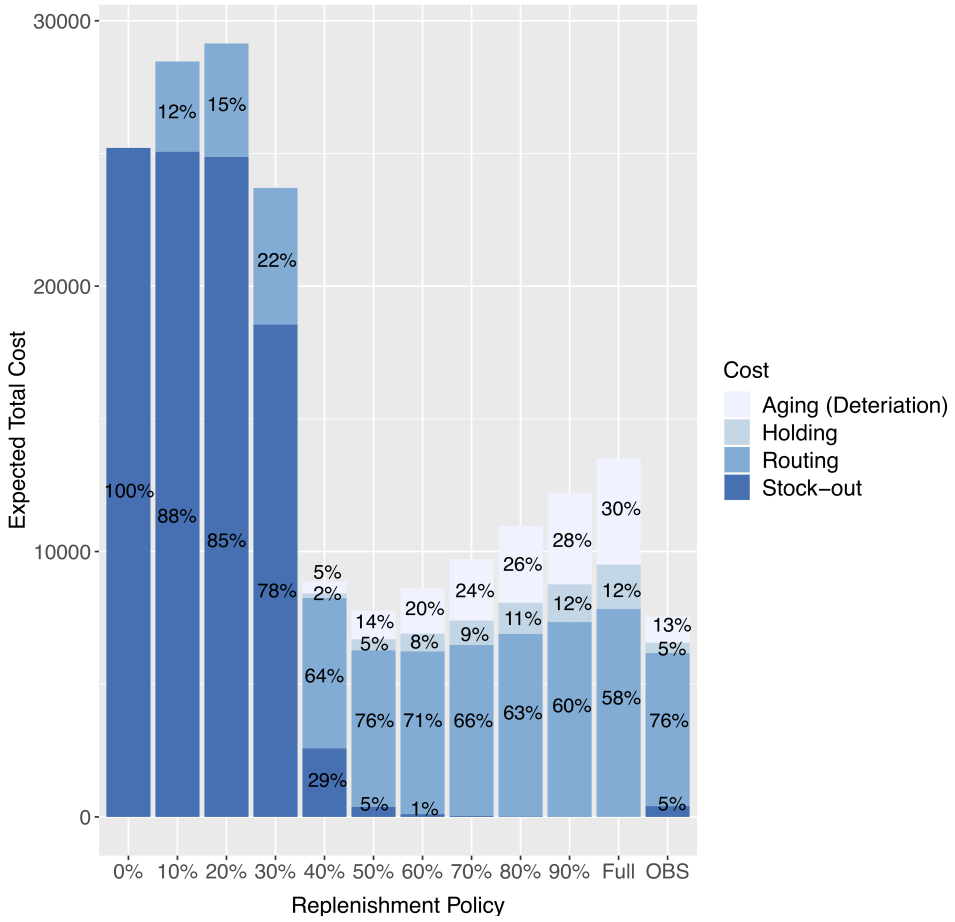

Figure 7: Expected routing, inventory, and deterioration cost for the different replenishment policies analyzed (P = 5 planning horizons and c = 0.25 variance level)

Table 2: Results for P = 3 time periods and a low variance level (c = 0.10)

		OBS									
	Initial Solution [1]	OBS Total Cost [2]	Routing Cost [3]	Total Inventory Cost [4]	Stock-out Cost [5]	Holding Cost [6]	Deterioration Cost [7]	Semi-Fresh Units [8]	Unit Loss [9]	Time (s)	GAP(%) [1-2]
A-n32-K5	3416.54	3177.12	2726.82	282.36	201.33	81.03	167.94	206.40	2	37	-7.01
A-n33-K5	2851.21	2750.60	2384.34	196.39	114.74	81.66	169.87	208.97	2	40	-3.53
A-n33-K6	2959.90	2951.47	2533.80	234.54	146.04	88.50	183.13	228.89	0	31	-0.28
A-n37-K5	2888.12	2793.52	2354.56	207.53	86.40	121.13	231.43	288.85	0	51	-3.28
A-n38-K5	3311.49	3273.72	2730.94	361.65	275.20	86.44	181.13	226.25	0	29	-1.14
A-n39-K6	3526.04	3418.88	2871.61	356.81	258.83	97.98	190.46	237.92	0	24	-3.04
A-n45-K6	3936.75	3780.55	3163.50	382.47	269.45	113.02	234.58	290.34	2	32	-3.97
A-n45-K7	4937.86	4915.49	4252.97	449.37	344.46	104.91	213.14	266.34	0	77	-0.45
A-n55-K9	4574.26	4446.53	3851.74	292.21	137.33	154.88	302.58	377.95	0	76	-2.79
A-n60-K9	6003.72	5791.00	4772.87	707.01	562.73	144.28	311.13	387.03	1	122	-3.54
A-n61-K9	4350.11	4345.50	3570.30	484.89	330.49	154.40	290.30	362.88	0	107	-0.11
A-n63-K9	7443.74	7385.97	6296.30	399.25	77.76	321.49	690.43	860.85	1	126	-0.78
A-n65-K9	5307.30	5166.78	4194.78	670.06	524.62	145.44	301.94	377.29	0	77	-2.65
A-n80-K10	7641.80	7486.76	6293.38	502.74	180.87	321.87	690.64	859.11	3	179	-2.03
B-n31-K5	3078.20	2942.15	2484.85	316.80	243.94	72.86	140.50	175.46	0	10	-4.42
B-n35-K5	4283.36	4133.24	3526.32	237.42	72.46	164.96	369.50	455.54	4	24	-3.50
B-n39-K5	2806.68	2679.66	2167.37	216.92	77.59	139.34	295.36	369.01	0	54	-4.53
B-n41-K6	3544.84	3517.36	2978.70	330.06	235.25	94.80	208.61	248.08	8	21	-0.78
B-n45-K5	3238.35	3112.60	2498.73	252.43	87.25	165.19	361.44	451.20	0	56	-3.88
B-n50-K7	3478.18	3389.06	2626.85	538.95	430.12	108.84	223.25	278.85	0	95	-2.56
B-n52-K7	3446.28	3430.86	2467.86	752.52	656.42	96.10	210.48	254.66	6	39	-0.45
B-n56-K7	3234.78	3156.04	2386.53	276.55	49.04	227.51	492.96	616.13	0	96	-2.43
B-n57-K9	6984.75	6899.21	5861.46	373.24	77.15	296.09	664.52	807.49	15	50	-1.22
B-n64-K9	3916.44	3867.04	2939.35	634.37	492.52	141.84	293.32	366.63	0	58	-1.26
B-n67-K10	4712.71	4575.76	3690.78	557.29	403.09	154.20	327.69	405.18	3	92	-2.91
B-n68-K9	5686.91	5497.42	4494.21	367.65	69.54	298.12	635.56	794.37	0	137	-3.33
B-n78-K10	5608.46	5487.81	4311.17	449.14	108.79	340.35	727.49	909.21	0	82	-2.15
Average:	4339.58	4236.00	3497.48	401.13	241.24	159.90	337.38	418.92	2	84	-2.39

Table 3: Results for P = 3 time periods and a medium variance level (c = 0.25)

		OBS									
	Initial Solution [1]	OBS Total Cost [2]	Routing Cost [3]	Total Inventory Cost [4]	Stock-out Cost [5]	Holding Cost [6]	Deterioration Cost [7]	Semi-Fresh Units [8]	Unit Loss [9]	Time (s)	GAP(%) [1-2]
A-n32-K5	3572.87	3471.96	2800.86	335.56	181.44	154.12	335.54	418.98	0	28	-2.82
A-n33-K5	3085.55	3037.70	2477.07	235.22	82.03	153.20	325.40	405.99	1	43	-1.55
A-n33-K6	3357.63	3300.45	2591.36	283.53	88.46	195.07	425.56	531.68	0	15	-1.70
A-n37-K5	3034.53	2974.88	2322.49	372.79	236.39	136.39	279.60	344.64	3	31	-1.97
A-n38-K5	3564.91	3441.40	2772.53	298.40	125.44	172.96	370.47	461.98	1	21	-3.46
A-n39-K6	3738.52	3696.02	2920.46	368.04	177.59	190.45	407.52	508.95	0	24	-1.14
A-n45-K6	4120.31	4072.30	3248.31	359.07	143.47	215.60	464.92	574.22	5	39	-1.17
A-n45-K7	5371.67	5296.58	4445.30	343.72	107.44	236.28	507.57	633.16	1	74	-1.40
A-n55-K9	5100.08	5015.85	3973.40	402.05	102.03	300.01	640.40	798.49	1	82	-1.65
A-n60-K9	6331.65	6253.53	4988.67	616.19	318.58	297.61	648.67	809.94	1	112	-1.23
A-n61-K9	4886.88	4840.46	3681.82	456.95	137.16	319.79	701.69	876.61	0	121	-0.95
A-n63-K9	7735.20	7668.57	6261.80	710.53	391.33	319.20	696.23	868.23	1	130	-0.86
A-n65-K9	5632.93	5602.29	4316.57	577.00	249.28	327.71	708.73	884.97	1	60	-0.54
A-n80-K10	8161.45	8145.25	6209.57	959.24	708.17	351.08	776.44	954.92	10	172	-0.20
B-n31-K5	3182.80	3164.27	2563.49	262.45	107.43	155.02	338.34	422.44	0	8	-0.58
B-n35-K5	4470.57	4366.50	3550.36	449.53	288.07	161.46	366.60	452.86	4	34	-2.33
B-n39-K5	2965.87	2923.87	2168.23	431.14	278.47	152.67	324.50	403.83	1	52	-1.42
B-n41-K6	3871.62	3820.65	3090.30	266.65	53.59	213.06	463.70	575.81	3	32	-1.32
B-n45-K5	3435.74	3295.76	2487.91	450.29	284.34	165.94	357.57	440.13	5	63	-4.07
B-n50-K7	3650.89	3605.91	2677.62	424.76	194.97	229.79	503.54	628.60	1	85	-1.23
B-n52-K7	3656.11	3586.02	2653.46	443.14	217.56	225.58	489.42	598.55	9	54	-1.92
B-n56-K7	3394.45	3378.00	2407.88	482.27	256.26	226.01	487.85	608.94	1	105	-0.48
B-n57-K9	7205.45	7128.95	5841.23	625.30	319.30	305.99	662.42	814.82	9	44	-1.06
B-n64-K9	4508.42	4411.61	3167.43	529.08	203.08	326.00	715.10	889.89	3	68	-2.15
B-n67-K10	5354.59	5262.80	3968.47	595.45	267.77	327.68	698.88	870.18	2	97	-1.71
B-n68-K9	5945.65	5757.34	4489.91	620.01	318.29	301.72	647.42	805.98	2	149	-3.17
B-n78-K10	5935.37	5888.49	4300.23	861.35	518.62	342.74	726.90	906.45	1	78	-0.79
Average:	4639.69	4570.64	3569.51	472.58	235.43	240.86	521.15	647.82	3	67	-1.59

Table 4: Results for P = 3 time periods and a high variance level (c = 0.75)

		OBS									
	Initial Solution [1]	OBS Total Cost [2]	Routing Cost [3]	Total Inventory Cost [4]	Stock-out Cost [5]	Holding Cost [6]	Deterioration Cost [7]	Semi-Fresh Units [8]	Unit Loss [9]	Time (s)	GAP(%) [1-2]
A-n32-K5	3996.01	3922.24	2886.51	683.62	525.47	158.15	352.11	425.87	10	35	-1.85
A-n33-K5	3350.84	3253.20	2387.82	494.35	318.67	175.68	371.03	448.87	10	35	-2.91
A-n33-K6	3590.10	3563.72	2615.47	506.72	305.11	201.62	441.52	548.74	2	24	-0.73
A-n37-K5	3409.02	3376.44	2328.30	711.00	549.09	161.91	337.14	406.70	10	41	-0.96
A-n38-K5	3937.39	3746.95	2696.93	640.35	451.28	189.07	409.68	486.06	17	22	-4.84
A-n39-K6	4190.86	4180.73	2962.20	792.47	591.62	200.85	426.06	527.14	4	17	-0.24
A-n45-K6	4616.19	4584.67	3211.94	897.77	678.23	219.54	474.96	586.71	5	41	-0.68
A-n45-K7	5810.66	5574.95	4224.15	796.60	544.07	252.53	554.19	680.56	8	70	-4.06
A-n55-K9	5487.41	5412.57	3921.76	827.72	517.75	309.97	663.08	823.07	4	82	-1.36
A-n60-K9	6977.31	6910.97	5022.00	1218.77	903.17	315.59	670.21	828.25	6	117	-0.95
A-n61-K9	5344.02	5289.92	3673.94	938.08	619.16	318.92	677.91	840.81	4	109	-1.01
A-n63-K9	8565.00	8321.54	5965.02	1635.56	1303.50	332.06	720.96	881.49	13	121	-2.84
A-n65-K9	6215.88	6208.09	4308.25	1155.71	815.73	339.97	744.13	911.68	12	70	-0.13
A-n80-K10	9597.16	9432.94	6746.42	1487.48	933.69	553.79	1199.03	1424.92	49	172	-1.71
B-n31-K5	3588.52	3526.37	2564.57	620.81	461.59	159.22	340.99	421.14	3	16	-1.73
B-n35-K5	4985.25	4960.42	3565.09	1010.33	839.08	171.25	385.00	463.65	12	30	-0.50
B-n39-K5	3399.01	3284.92	2133.75	795.38	616.90	178.48	355.79	420.59	16	53	-3.36
B-n41-K6	4192.25	4093.28	2993.25	599.85	373.46	226.39	500.18	604.49	14	29	-2.36
B-n45-K5	3888.76	3709.09	2492.49	816.15	624.79	191.36	400.45	482.17	12	65	-4.62
B-n50-K7	4157.56	4058.93	2713.34	870.08	648.43	221.65	475.52	582.83	8	86	-2.37
B-n52-K7	4236.80	4199.11	2631.52	1044.08	805.72	238.36	523.51	629.20	17	49	-0.89
B-n56-K7	4006.22	3909.47	2350.93	1040.44	799.69	240.75	518.09	630.17	12	102	-2.42
B-n57-K9	8023.31	7968.14	5892.24	1376.01	1047.33	328.68	699.89	837.06	25	40	-0.69
B-n64-K9	4912.12	4813.19	3087.32	1009.04	681.05	327.99	716.84	887.14	6	58	-2.01
B-n67-K10	5902.16	5861.28	4001.76	1130.46	788.05	342.41	729.05	895.49	11	97	-0.69
B-n68-K9	6864.70	6725.79	4500.37	1563.83	1252.81	311.02	661.60	814.01	9	140	-2.02
B-n78-K10	6740.99	6731.71	4292.88	1672.36	1313.43	358.93	766.47	945.29	9	78	-0.14
Average:	5184.65	5097.06	3561.86	975.37	715.14	260.23	559.83	682.74	12	67	-1.78

6. Conclusions

We have studied a fresh produce agri-food supply chain where there is a single fresh food supplier who owns a central warehouse that serves several retail centers (RCs) with stochastic demands. The supplier makes decisions on the inventory level of each RC and the transportation of perishable products from the warehouse to the RCs. This can be viewed as a multi-period inventory routing problem with stochastic demands and perishable products in the context of agri-food supply chains. It is also known as Perishable Inventory Routing Problem (PIRP) in the combinatorial optimization field.

We have modelled this problem as a mixed integer program and developed a simheuristic algorithm, which is based on a local search metaheuristic, to solve it. The algorithm integrates Monte Carlo simulation into the metaheuristic, allowing us to model stochastic demands. The perishability of the products is included in the model as constraints and costs. We have shown that the proposed simheuristic algorithm can produce an optimal solution that minimizes the total expected costs (i.e., inventory costs, routing costs and costs related to food waste). Although the focus of this paper is on agri-food supply chains, the proposed method is applicable to other perishable products including blood supply chains and short-life technology products.

We have identified several research lines based on the limitations of our model. First, our model only considers one fresh produce product. Hence, we plan to extend our work to include multiple products and to consider product substitutions, both are common in agri-food supply chains. Secondly, our model only integrates the routing and inventory management of agri-food supply chains. Hence, another extension will include further integration between harvesting, processing and distribution processes in fresh produce agri-food supply chains. Thirdly, our model assumes that the demand at all RCs in all periods are independent. We plan to extend our model to include seasonal and correlated demands. Finally, we plan to extend the proposed simheuristic algorithm to include different types of heuristics with a more iterative process between the local search stage and the refinement stage.

Acknowledgements

This work has been partially supported by the Spanish Ministry of Economy and Competitiveness (TRA2015-71883-REDT), FEDER. and the Erasmus+ programme (2018-1-ES01-KA103-04976).

References

Appendix A. List of abbreviations

Table A.5 contains the list of abbreviations used in the text.

Abbreviation	Description
PIRP	Perishable Inventory Routing Problem
IRP	Inventory Routing Problem
VMI	Vendor managed inventories
RMI	Retail Managed Inventory
VRP	Vehicle Routing Problem
RC	Retail Center
OR	Operational Research
MILP	Mixed Integer Linear Program
LS	Local Search
MCS	Monte Carlo Simulation
OBS	Our Best Solution

Table A.5: List of abbreviations