Production Routing for Perishable Products

Aldair Alvarez[a,*], Pedro Miranda[a], Sonja U. K. Rohmer[a,b]

[a] *HEC Montréal, GERAD and CIRRELT, Canada*
[b] *Eindhoven University of Technology, The Netherlands*

Abstract

This paper introduces the production routing problem for perishable products with fixed shelf life and gradual decay, where the age of products impacts the price that can be obtained when satisfying customer demands. In this problem, a single supplier is responsible for the production and distribution of perishable products to a set of customers. Fixed setup and variable production costs occur at the supplier location when production takes place, and routing costs are charged for the distribution activities. Additionally, inventory holding costs are incurred at both the production facility and the customer locations. Transshipments between customer locations are allowed on the basis of a unit transshipment cost. The objective is to maximise the total profit given by the sales revenue minus the sum of production, routing, inventory and transshipment costs. The problem is formulated as a mixed integer linear program and solved using a branch-and-cut and a hybrid, iterated local search-based heuristic. Based on the results of the computational experiments, we analyse the impact of perishability on the structure of the solutions. We explore perishability in the form of both product shelf life and decay, observing that larger decay rates lead to a reduction in total profit, as a result of both a decrease in revenues and an increase in the total cost. We further analyse the impact of different decay rates on the cost structure of the problem and the significance of transshipments for the management of perishability across the system. Finally, we adapt our hybrid method to solve the standard production routing problem. Comparison of our results with state-of-the-art solution methods on benchmark instances from the literature shows that our heuristic provides good quality solutions that are competitive with the scientific literature.

Keywords: Production routing problem, Perishability, Product decay, Transshipments, Hybrid heuristic.

*Corresponding author.
Email addresses: aldair.alvarez@hec.ca (Aldair Alvarez), pedro.miranda@hec.ca (Pedro Miranda), sonja.rohmer@hec.ca (Sonja U. K. Rohmer)

1. Introduction

Integrated supply chain planning is a crucial, yet often challenging task for many companies that concerns decisions about a variety of supply chain activities related to the production, storage and distribution of products. The integration of these planning decisions holds substantial benefits, allowing companies to streamline their operations, cut costs and improve their overall performance in order to stay competitive within today's business environment (Díaz-Madroñero et al., 2015). Integrating supply chain decisions within a common planning framework is particularly relevant in the context of perishable products, where products have a limited shelf life and quality decays over time. As such, the time products spend in storage and transportation phases affects their quality, often resulting in product loss or a reduction in the value of the product. More coherent planning decisions, taking into account different stages in the chain, may therefore improve product quality and customer service levels and help prevent unnecessary losses.

Production routing presents an important tool in this context, aimed at supporting decision-making, by jointly optimising production and distribution planning. Production planning usually relates to lot-sizing and inventory decisions, determining what, when and how much to produce and keep in stock to minimise production, setup and inventory costs (Jans and Degraeve, 2008; Pochet and Wolsey, 2006). Distribution planning comprises the delivery schedule, quantities and routes to serve a given set of customer at minimum cost (Toth and Vigo, 2014). Combining production, inventory and routing decisions, the production routing problem (PRP) is a complex decision problem that has received growing attention in the scientific literature over recent years. An overview of this problem, different formulations and solution algorithms can be found in Adulyasak et al. (2015b).

Most of the research on the PRP has focused on solving a problem where the manufacturer has one production facility that produces a single product, and owns a limited fleet of homogeneous vehicles to perform deliveries to multiple customers. Diverse solution methods for this problem have been proposed, including branch-and-cut algorithms (Ruokokoski et al., 2010; Archetti et al., 2011; Adulyasak et al., 2014a), Benders-based branch-and-cut algorithms for stochastic settings (Adulyasak et al., 2015a), branch-and-price-based heuristics (Bard and Nananukul, 2009a, 2010), mathematical programming-based heuristics (Bertazzi et al., 2005; Absi et al., 2015; Chitsaz et al., 2019; Solyalı and Süral, 2017; Avci and Yildiz, 2019), and metaheuristics, such as greedy randomised adaptive search procedure (Boudia et al., 2007), genetic algorithm (Boudia and Prins, 2009), tabu search (Bard and Nananukul, 2009b), adaptive large neighbourhood search (ALNS) (Adulyasak et al., 2014b), and variable neighbourhood search (VNS) (Qiu et al., 2018b).

The problem with multiple products has been much less studied. Similar to the single product scenario, different solution methods have been proposed, such as branch-and-cut algorithms (Qiu et al., 2018a), decomposition heuristics (Chandra and Fisher, 1994), Lagrangian heuristics (Fumero and Vercellis, 1999), mathematical programming-based heuristics (Brahimi and Aouam, 2016), and tabu search algorithms (Armentano et al., 2011). Enriching vehicle routing features, such as heterogeneous fleet and multi-trips, were considered by Lei et al. (2006), in the context of the chemical industry, and Miranda et al. (2018a,b, 2019) in the furniture industry.

Despite this growing attention within the scientific literature, studies on PRPs for perishable products are still relatively scarce. Amorim et al. (2013) and Belo-Filho et al. (2015) study a scenario where products have a very short lifespan and, therefore, cannot be stocked (e.g., food-catering, industrial adhesive materials, ready-mixed concrete, and among others). Given the absence of inventory, production and distribution must be synchronised to guarantee that deliveries can only take place once customer orders are produced. Shaabani and Kamalabadi (2016) address a case with multiple products, where perishability is modelled by setting upper bounds on the inventory levels, in relation to the life span of the products, while at the same time forbidding products to be discarded. The research of Qiu et al. (2019), in contrast, models perishability by incorporating age-dependent inventory holding costs and deterioration rates, expressed in product loss, into the studied PRP. Within the context of their research, the authors also tested the impact of different delivery and selling priority policies. Neves-Moreira et al. (2019) model perishability within the framework of a PRP with multiple perishable products by setting a maximum difference between the consumption periods (at the customers) and the production periods (at the producer). Most of the literature on PRPs for perishable products, therefore, focuses on perishability in the form of a limited product shelf life or a fixed expiration date. As such, the gradual decay of products over time and the resulting loss in product quality in combination with its impact on the overall profit has been mostly neglected. Only recently, Li et al. (2018) and Li et al. (2020) gave more attention to the PRP with gradual decay. Li et al. (2018) study, in this context, a bi-objective PRP, considering one objective that minimises the total production, inventory and routing cost, and another one maximising the freshness of the products used to satisfy customer demands. The problem is defined mathematically for a single product with different quality levels and solved using an ϵ-constraint-based heuristic approach. The method is applied to a case study of a fresh-meat producer and tested further on a set of artificial instances. The research of Li et al. (2020) extends this work assuming a multi-plant environment with different packaging options. The shelf-life of products depends, in this case, on the choice of packaging and the objective function also includes packaging costs. The problem is formulated mathematically and

solved for a set of artificial instances using a hybrid matheuristic.

The problem presented in our paper is similar to the problems in Li et al. (2018) and Li et al. (2020), considering a PRP for a single product of perishable nature, which gradually decays over time until a predefined shelf life is reached. The value of the product is, thus, age-dependent, impacting the price that can be obtained. To model this impact of decay on the price and thus the resulting profit that can be achieved the objective of the proposed problem is to maximise the total profit rather than minimising the total cost as in the study of Li et al. (2018). Moreover, unlike Li et al. (2018) we do not restrict the nature of the age-dependent quality. In comparison to Li et al. (2020) our problem is more specific, allowing us to gain a deeper understanding of the impact of different forms of perishability. In this context, we also include the notion of transshipments to study the effect of redistributing inventories at the customer level. Real-life applications of this problem can be found in the context of the food system, related to the supply of fresh products to the grocery retail. Another potential application area is the production and distribution of vaccines as well as other pharmaceutical products. The problem is modelled as a mixed integer linear program (MILP) which is solved using a branch-and-cut algorithm as well as a hybrid heuristic which combines features from an iterated local search metaheuristic and mathematical programming techniques. Moreover, we perform extensive computational experiments on instances built upon well-known benchmark instance sets in the PRP literature. We conduct an analysis to gain managerial insights on the effect of different decay rates on the used resources and the optimal decisions made. In particular, we analyse the impact of perishability on the structure of the solutions, by exploring different types of perishability in the form of both product shelf life and gradual decay as well as the significance of transshipments for the management of perishability across the system. In this context, we also analyse the impact of the cost structure of the problem on the changes of the solutions for different rates of decay. From a computational perspective, the results show the effectiveness of our heuristic method to solve the problem, as well as its competitiveness when solving benchmark instances for the standard PRP.

As such, the contribution of this paper is threefold. First, we study the PRP with perishable products under consideration of gradual decay and transshipments to gain a deeper understanding of the impact of perishability and the role of transshipments in this context. Second, we develop a tailored optimisation model and suitable solution approaches and thirdly we show that our heuristic is capable of finding competitive solutions for the standard PRP.

The rest of this paper is organised as follows. Section 2 provides a formal problem description. Section 3 introduces the notation and the mathematical formulation of the problem, while Section 4 describes

the hybrid heuristic used to solve it. Computational results are presented in Section 5, and a general discussion and conclusions follow in Section 6.

2. Problem description

We consider a production routing problem where a single supplier is responsible for the production and distribution of a perishable product. We assume a system consisting of a production plant, a set of customers, and storage facilities at both the plant and the customer locations. The supplier has to satisfy the customer demands during a finite multi-period planning horizon through new deliveries from the production plant and/or inventory from previous periods at the customer locations. In addition, the supplier may choose to make use of transshipments to fulfil demand by redistributing products between customer locations. Transshipments incur a cost per unit of product transshipped and can be used directly for demand fulfilment at the customer locations. At the production facility, the supplier faces a limited production capacity, a fixed setup cost for each period where production takes place, as well as a variable production cost for each unit produced. After production, products can either be stored at the plant or distributed to customer locations. Storage capacity is limited at both the plant and the customers, and inventory holding costs are site-dependent. Distribution to customers is carried out by a fleet of homogeneous vehicles with limited capacity. Each vehicle has to start and end its delivery route at the production plant, and incurs a cost for traveling between locations. We further assume that in the first period of the planning horizon each storage facility has a certain amount of initial inventory available to satisfy demand at the customer locations.

The product is considered to be perishable in nature, assuming both a fixed shelf life, specifying the time after which the product has to be discarded, as well as a gradual decay in product quality over time. Product quality defines the value of the product and thus affects the unit sales revenue, due to age-dependent pricing. The demand for the product at the customer locations is known for each time period, and specifies the minimum amount that the supplier must guarantee to make available at the customer location during that time period. These demands can be satisfied using products of different ages, but age-dependent pricing may apply. Overall, the problem contains the following decisions:

- When and how much to produce during the planning horizon?
- When and how much to deliver to the customer locations?
- What is the optimal routing of vehicles for the different time periods?
- When and how much to transship between the customer locations?

- How to satisfy the customer demands taking into account different product ages?

The problem therefore consists of jointly determining the optimal production and distribution plan over a finite multi-period planning horizon. The objective is to maximise the total profit within the system, while satisfying all customer demands and taking into account the loss in freshness of products over time.

3. Mathematical formulation

This section presents the main notation and the mathematical formulation of the problem. We first define the sets and parameters of the problem and then introduce a MILP for the problem. Consider the following notation:

Sets:

$\mathcal{C}$ Set of customers;

$\mathcal{N}$ Set of facilities: plant (node 0) and customers;

$\mathcal{A}$ Set of arcs;

$\mathcal{T}$ Set of time periods;

$\mathcal{S}$ Set of ages of the product;

$\mathcal{K}$ Set of vehicles.

Parameters:

f Fixed production setup cost;

v Unit production cost;

u_{is} Unit sales revenue for product of age $s \in \mathcal{S}$ at customer $i \in \mathcal{C}$;

h_i Unit inventory holding cost at facility $i \in \mathcal{N}$;

c_{ij} Unit travel cost between facilities $i \in \mathcal{N}$ and $j \in \mathcal{N}$;

b_{ij} Unit transshipment cost between customers $i \in \mathcal{C}$ and $j \in \mathcal{C}$;

d_i^t Demand of customer $i \in \mathcal{C}$ in time period $t \in \mathcal{T}$;

C_i Storage capacity of facility $i \in \mathcal{N}$;

I_{is}^0 Initial inventory of age $s \in \mathcal{S}$ at facility $i \in \mathcal{N}$;

S Maximum age of the product;

P Production capacity;

Q Capacity of each vehicle.

Decision variables:

z^t : 1 if there is production at the plant in period $t \in \mathcal{T}$, 0 otherwise;

x_{ij}^{kt} : 1 if arc $(i,j) \in \mathcal{A}$ is traversed by vehicle $k \in \mathcal{K}$ in period $t \in \mathcal{T}$, 0 otherwise;

y_i^{kt} : 1 if vehicle $k \in \mathcal{K}$ visits site $i \in \mathcal{N}$ in period $t \in \mathcal{T}$, 0 otherwise;

p^t : production quantity in period $t \in \mathcal{T}$;

I_{is}^t : inventory of age $s \in \mathcal{S}$ at facility $i \in \mathcal{N}$ at the end of period $t \in \mathcal{T}$;

q_{is}^{kt} : delivery quantity of age $s \in \mathcal{S}$ to customer $i \in \mathcal{C}$ by vehicle $k \in \mathcal{K}$ in period $t \in \mathcal{T}$;

w_{is}^t : consumption of product of age $s \in \mathcal{S}$ by customer $i \in \mathcal{C}$ in period $t \in \mathcal{T}$,

r_{ijs}^t : transshipment quantity of product of age $s \in \mathcal{S}$ from customer $i \in \mathcal{C}$ to customer $j \in \mathcal{C}$ in period $t \in \mathcal{T}$;

The mathematical formulation for the problem can then be stated as follows:

$$\max \quad \sum_{t\in\mathcal{T}} \Bigg(\sum_{i\in\mathcal{C}} \sum_{s\in\mathcal{S}} u_{is} w_{is}^t - f z^t - v p^t - \sum_{i\in\mathcal{N}} \sum_{s\in\mathcal{S}} h_i I_{is}^t - \sum_{(i,j)\in\mathcal{A}} \sum_{k\in\mathcal{K}} c_{ij} x_{ij}^{kt} - \sum_{i\in\mathcal{C}} \sum_{j\in\mathcal{C}:\, j\neq i} \sum_{s\in\mathcal{S}} b_{ij} r_{ijs}^t \Bigg) \tag{1}$$

$$\text{s.t.} \quad I_{0s}^t = p^t - \sum_{i\in\mathcal{C}} \sum_{k\in\mathcal{K}} q_{is}^{kt} \qquad t \in \mathcal{T}, s = 0, \tag{2}$$

$$I_{0s}^t = I_{0,s-1}^{t-1} - \sum_{i\in\mathcal{C}} \sum_{k\in\mathcal{K}} q_{is}^{kt} \qquad t \in \mathcal{T}, s \in \mathcal{S}\backslash\{0\}, \tag{3}$$

$$I_{is}^t = \sum_{k\in\mathcal{K}} q_{is}^{kt} + \sum_{j\in\mathcal{C}:\, j\neq i} r_{jis}^t - \sum_{j\in\mathcal{C}:\, j\neq i} r_{ijs}^t - w_{is}^t \qquad i \in \mathcal{C}, t \in \mathcal{T}, s = 0, \tag{4}$$

$$I_{is}^t = I_{i,s-1}^{t-1} + \sum_{k\in\mathcal{K}} q_{is}^{kt} + \sum_{j\in\mathcal{C}:\, j\neq i} r_{jis}^t - \sum_{j\in\mathcal{C}:\, j\neq i} r_{ijs}^t - w_{is}^t \qquad i \in \mathcal{C}, t \in \mathcal{T}, s \in \mathcal{S}\backslash\{0\}, \tag{5}$$

$$d_i^t = \sum_{s\in\mathcal{S}} w_{is}^t \qquad i \in \mathcal{C}, t \in \mathcal{T}, \tag{6}$$

$$p^t \leq \min\left\{ P, \sum_{i\in\mathcal{C}} \sum_{\tau=t}^{|\mathcal{T}|} d_i^\tau \right\} z^t \qquad t \in \mathcal{T}, \tag{7}$$

$$\sum_{s\in\mathcal{S}} I_{0s}^t \leq C_0 \qquad t\in\mathcal{T}, \qquad (8)$$

$$\sum_{s\in\mathcal{S}} I_{is}^t \leq C_i - d_i^t \qquad i\in\mathcal{C}, t\in\mathcal{T}, \qquad (9)$$

$$\sum_{s\in\mathcal{S}} q_{is}^{kt} \leq \min\{Q, C_i\} y_i^{kt} \qquad i\in\mathcal{C}, k\in\mathcal{K}, t\in\mathcal{T}, \qquad (10)$$

$$\sum_{i\in\mathcal{C}}\sum_{s\in\mathcal{S}} q_{is}^{kt} \leq Q y_0^{kt} \qquad k\in\mathcal{K}, t\in\mathcal{T}, \qquad (11)$$

$$\sum_{j\in\mathcal{N}:j\neq i} x_{ji}^{kt} = y_i^{kt} \qquad i\in\mathcal{N}, k\in\mathcal{K}, t\in\mathcal{T}, \qquad (12)$$

$$\sum_{j\in\mathcal{N}:j\neq i} x_{ij}^{kt} = y_i^{kt} \qquad i\in\mathcal{N}, k\in\mathcal{K}, t\in\mathcal{T}, \qquad (13)$$

$$\sum_{i\in\mathcal{B}}\sum_{j\in\mathcal{B}:j\neq i} x_{ij}^{kt} \leq \sum_{i\in\mathcal{B}} y_i^{kt} - y_\ell^{kt} \qquad \forall\mathcal{B}\subseteq\mathcal{C}, |\mathcal{B}|\geq 2, k\in\mathcal{K}, t\in\mathcal{T}, \ell\in\mathcal{B}, \qquad (14)$$

$$\sum_{k\in\mathcal{K}} y_i^{kt} \leq 1 \qquad i\in\mathcal{C}, t\in\mathcal{T}, \qquad (15)$$

$$p^t \geq 0 \qquad t\in\mathcal{T}, \qquad (16)$$

$$I_{is}^t \geq 0 \qquad i\in\mathcal{N}, t\in\mathcal{T}, s\in\mathcal{S}, \qquad (17)$$

$$q_{is}^{kt} \geq 0 \qquad i\in\mathcal{C}, k\in\mathcal{K}, t\in\mathcal{T}, s\in\mathcal{S}, \qquad (18)$$

$$w_{is}^t \geq 0 \qquad i\in\mathcal{C}, t\in\mathcal{T}, s\in\mathcal{S}, \qquad (19)$$

$$r_{ijs}^t \geq 0 \qquad i\in\mathcal{C}, j\in\mathcal{C}, t\in\mathcal{T}, s\in\mathcal{S}, \qquad (20)$$

$$z^t \in \{0,1\} \qquad t\in\mathcal{T}, \qquad (21)$$

$$y_i^{kt} \in \{0,1\} \qquad i\in\mathcal{N}, k\in\mathcal{K}, t\in\mathcal{T}, \qquad (22)$$

$$x_{ij}^{kt} \in \{0,1\} \qquad (i,j)\in\mathcal{A}, k\in\mathcal{K}, t\in\mathcal{T}. \qquad (23)$$

The objective function (1) maximises the total profit, which is given by the total revenue minus the sum of production (setup and variable), inventory, transportation and transshipment costs. Constraints (2) and (3) balance the inventory at the plant for the product of age 0 and for the remaining ages of $\mathcal{S}$, respectively. Analogously, constraints (4) and (5) balance the inventory at the customers locations, taking into account possible transshipments. Constraints (6) ensure the customer demands satisfaction. Notice that this can be done with products of different ages simultaneously. Constraints (7) force the production setup variables to take value 1 if production takes place and also impose the production capacity. Constraints (8) limit the total amount stored in each time period at the plant, according to its maximum storage capacity. Similarly, constraints (9) impose that the inventory level after delivery at every cus-

tomer location cannot exceed its corresponding storage capacity. Constraints (10) link the delivery and visit variables, not allowing deliveries to customers that are not visited. Constraints (11) ensure that the capacity of each vehicle is respected and constraints (12) and (13) ensure the vehicle flow conservation at every site. Constraints (14) are subtour elimination constraints (SECs) while constraints (15) limit to one the number of visits to each customer in each period. The decision variables domain is stated in constraints (16)−(23).

Note that the way perishability is modelled, the problem can be easily adapted to also study other applications with age-dependent pricing, including ameliorating products for which the quality and thus the price increase over time. Other examples that take a similar approach to model the aging of products can be found in Coelho and Laporte (2014); Rohmer et al. (2019) and Alvarez et al. (2020). Note also that different types of formulations could be used to represent the problem, such as formulations without a vehicle index or using facility location-like delivery variables, as those studied by Alvarez et al. (2020). However, here we have opted for a standard arc-based network flow representation for the sake of clarity and explicitness.

Given that this formulation contains an exponential number of SECs, they are initially dropped out of the formulation and then checked (and generated) at each node of the branch-and-bound tree. For this purpose, we use a separation algorithm that relies on solving several minimum cut problems, as in Adulyasak et al. (2014a) and Alvarez et al. (2020, 2021). The algorithm constructs a weighted graph with the nodes that are visited in a given solution found in the tree. Then, solves a minimum cut problem for each customer node of the graph, setting it as the sink node and the plant node as the source. Every time a new subtour is found, the corresponding SEC is added for every vehicle and time period. The minimum cut problems are solved using a subroutine of the Concorde TSP solver (Applegate et al., 2018). Since in this problem the vehicle fleet is considered as homogeneous, we also included the following symmetry breaking constraints (SBCs) into the formulation to reduce the number of symmetric feasible solutions:

$$y_0^{kt} \leq y_0^{k-1,t} \qquad k \in \mathcal{K}\backslash\{1\}, t \in \mathcal{T}, \tag{24}$$

$$\sum_{i=1}^{j} 2^{(j-i)} y_i^{kt} \leq \sum_{i=1}^{j} 2^{(j-i)} y_i^{k-1,t} \qquad j \in \mathcal{C}, k \in \mathcal{K}\backslash\{1\}, t \in \mathcal{T}. \tag{25}$$

SBCs (24) allow the usage of vehicle $k+1$ only if vehicle k is also used, whereas SBCs (25) are lexicographic ordering constraints which order the vehicle routes according to a unique number (dependent on the subset of customers visited) assigned to them. Similar SBCs were used by Adulyasak et al. (2014a) and Alvarez

et al. (2021). These inequalities allowed the branch-and-cut algorithm to obtain a significantly larger number of optimal solutions (73% more than with the model without SBCs), while also reducing the average optimality gap and its running time.

4. A hybrid heuristic algorithm

Due to the complexity of the problem, solving the proposed model within reasonable computing times using state-of-the-art optimisation solvers is generally limited to small-sized instances. To circumvent this issue, we propose a hybrid method, based on the iterated local search (ILS) metaheuristic (Lourenço et al., 2019). The idea behind our method is to split the problem into its individual decisions and then improve these decisions in the different components of the heuristic. The heuristic starts by constructing an initial, feasible solution and improving this solution by applying two variants of a local search algorithm as well as a mixed-integer programming (MIP)-based component. Following from this, the heuristic enters into a loop, applying a perturbation algorithm in combination with a local search variant and a reduced LP formulation. The general structure of the method is shown in Algorithm 1.

Algorithm 1: Iterated local search-based hybrid heuristic algorithm

```
1  begin
2      O* ← construction_heuristic();
3      O* ← insertion_and_removal_MIP(O*);
4      O* ← local_search_variant_1(O*);
5      O* ← local_search_variant_2(O*);
6      while stopping criteria are not met do
7          O' ← perturbation(O*);
8          O' ← local_search_variant_1(O');
9          O' ← LP(O');
10         if f(O') > f(O*) then
11             O* ← insertion_and_removal_MIP(O');
12         end
13     end
14     O* ← local_search_variant_2(O*);
15 end
```

4.1. Construction heuristic

For the construction of an initial solution for our heuristic, we design a step-wise construction heuristic, which creates solutions by decomposing the planning horizon into the individual time periods t of the planning horizon $\mathcal{T}$. Before this decomposition, we apply a pre-processing step, where we use the available initial inventories I_i^0 at the customers to satisfy as many demands as possible. To cover the remaining

demands, we then address the different types of decisions for each of the individual time periods, starting from the first, in separate phases as outlined in the following and shown in Algorithm 2.

In the first phase, we decide on the delivery quantities $\tilde{q}_i$, based on the available storage capacity at the customer locations (such that $\tilde{q}_i = C_i - I_i^{t-1}$, where I_i^t is the total inventory level at customer i in period t). Note that the delivery decisions are preliminary at this point, as we do not consider feasibility with regards to vehicle capacity constraints. To deal with potential vehicle capacity restrictions, we assign a priority to each customer. This priority defines the likeliness of a customer to experience shortages in the short-term and is used to determine the likelihood of a customer being visited. In addition, we set the production quantities based on the available inventory at the plant and the just defined delivery quantities ($\tilde{q}_i$). This means, if the available stock at the plant is smaller than the sum of the deliveries to the high priority customers, we activate production at full capacity ($\tilde{p} = P$), otherwise production stays idle.

Following from this, we then apply, in the second phase, a greedy routing algorithm, based on the concept of a nearest neighbour insertion heuristic, to determine the routes to the customer locations. The planned delivery quantities for high priority customers, i.e., customers that would experience shortages of products if not visited in the current time period, are routed first in the algorithm. In case the amount available at the plant is not sufficient to cover all the quantities delivered, production decisions may be adjusted in this phase. This may happen due to the addition of lower-priority customers to the routes.

In a third phase, we define the actual delivery amounts q_{is}^{kt} based on the preliminary delivery quantities determined in phase II and the products available at the plant, taking into account the freshness of the products. For reasons of simplicity, we choose to deliver the freshest products to the customers with highest priorities and determine consumption decisions by applying a freshest first rule. Following from this, we compute the inventory levels I_{is}^t for all the facilities i. The heuristic stops when all the phases have been completed for the last time period of the planning horizon.

If the heuristic finds a feasible solution for the problem, we determine the optimal values for the continuous variables by applying the reduced LP formulation presented in Section 4.5. In order to find alternative solutions, we apply the heuristic within two outer loops. In these two outer loops, we vary the size of the preliminary delivery quantities $\tilde{q}_i$ and the priority of the customers in the first phase π_i, using a multiplier $\delta \in (0, 1]$ that reduces $\tilde{q}_i$ and a parameter τ that defines the number of future periods considered when assigning customer priorities. At the end of this process, we keep the solution with the largest profit and use it as the initial solution in our heuristic algorithm.

Algorithm 2: Construction heuristic

begin
  Use the initial inventories at the customers to satisfy as many demands as possible;
  for $t \in \mathcal{T}$ **do**
    Phase I;
    Set preliminary delivery quantity for each customer: $\tilde{q}_i = C_i - I_i^{t-1}$;
    Assign priorities to customers: π_i = No. of demand periods not fully covered;
    Activate production: $\tilde{p} = P$ (if required);
    Phase II;
    Apply nearest neighbour insertion heuristic, considering customer priorities π_i to determine routes;
    Activate production: $\tilde{p} = P$ (if required);
    Phase III;
    Set actual delivery amounts q_{is}^{kt}, using the delivery quantities defined in phase II;
    Determine consumption quantities w_{is}^t according to the freshest first rule;
    Compute inventory levels I_{is}^t;
  end
  If a feasible solution O is found, do `LP(`O`)`;
end

4.2. Local search variant 1

In this local search we try to improve the incumbent solution by optimising its delivery routes. For this purpose, we apply a randomised variable neighbourhood descent heuristic (Subramanian et al., 2013; Alvarez et al., 2018). This algorithm makes use of a set of local search operators to gradually reach better routing solutions. In each iteration, an operator is randomly selected from the set and applied to the incumbent solution. If the operator cannot improve the solution, it is removed from the set. On the other hand, if the operator improves the solution, the set is restored to its initial form, containing all the local search operators. The heuristic stops when none of the operators can improve the incumbent solution, resulting in an empty set. The structure of this heuristic is shown in Algorithm 3.

Algorithm 3: Local search: randomised variable neighbourhood descent heuristic

```
begin
    O* ← O^0 (save initial solution);
    L ← L^0 (initialise local search operators set);
    while |L| > 0 do
        l ← choose a local search operator at random from L;
        O' ← l(O*);
        if f(O') > f(O*) then
            O* ← O';
            L ← L^0;
        else
            L ← L\{l};
        end
    end
end
```

The set of local search operators ($\mathcal{L}$) applied, in this context, is composed of the following classical vehicle routing operators:

- **Or-opt-**k, $k \in \{1, 2, 3\}$: transfers k consecutive customers from its current position in the route to another one in the same route;
- **2-opt**: deletes two nonconsecutive arcs from a route, reverses the arcs in-between them and adds two more arcs such that a new route is generated;
- **Shift**(k), $k \in \{1, 2, 3\}$: transfers k consecutive customers from its current position in a route to another route in the same time period;
- **Swap**(k_1, k_2), $k_1, k_2 \in \{1, 2\}, k_1 \geq k_2$: exchanges k_1 consecutive customers from a route with k_2 consecutive customers in another route in the same period.

Note that the first two operators represent intra-route operators that only change one route at the time, while the last two correspond to inter-route operators that may alter two routes every time the operator is called. Also, it is worth mentioning that we apply a first improvement strategy, and only consider feasible solutions during the search.

4.3. Local search variant 2

This second variant of the local search follows the same structure as the first variant (presented in Algorithm 3), but uses different kinds of operators, aimed at improving the visit decisions of the solutions. The operators applied are listed below:

- **Insert visits**: for each route of the solution, try to insert the customers (one at the time) that are not currently visited by the route (or other routes in the same time period);

- **Remove visits**: for each route of the solution, try to remove the customers (one at the time) that are currently visited by the route;
- **Transfer visit**: for each route of the solution, try to transfer its customers (one at the time) to another route in a different time period;

Every time one of these operators performs a move, we use the reduced LP formulation (presented in Section 4.5) to compute the optimal values for the continuous variables and check the feasibility of the resulting solution. As in local search variant 1, we apply a first improvement strategy, and only accept feasible solutions during the search.

4.4. Perturbation

While the local search variant 1 (on line 8 in Algorithm 1) focuses on improving the routing decisions within the solutions, the perturbation mechanism deals with the decisions regarding production setup and customer visits. For this purpose, several operators are used with the aim of altering different attributes of the solution simultaneously.

The perturbation mechanism applied, in this context, works in a similar way as the local search algorithms described in Sections 4.2 and 4.3. Using a set of perturbation operators, the algorithm selects in each iteration one of the operators at random and applies it to the incumbent solution. If the operator does not change the solution, it is removed from the set and the algorithm continues with the remaining set of operators. Otherwise, if the operator changes the solution, the set is reestablished to its initial form. The size of the perturbation is determined by $maxP$, denoting the maximum number of operators that can be applied successfully. The algorithm stops once $maxP$ is reached or the remaining set of operators is empty. An overview of the general structure of the perturbation algorithm is shown in Algorithm 4.

The set of perturbation operators ($\mathcal{P}$) contains the following operators:

- **Change setup**: choose randomly a time period t and change its production setup status, i.e., if $z^t = 1$ then it is changed to 0, otherwise, if $z^t = 0$, it is changed to 1;
- **Transfer setup**: choose randomly two different time periods t_1 and t_2, with $z^{t_1} = 1$ and $z^{t_2} = 0$, then change their production setup status to $z^{t_1} = 0$ and $z^{t_2} = 1$;
- **Insert visits**: choose randomly a route and a customer not visited in the time period of the route. Then the customer is inserted into the cheapest position of the route;
- **Remove visits**: choose a random route and remove randomly one of its customers;

Algorithm 4: Perturbation mechanism

```
begin
    O* ← O^0 (save initial solution);
    P ← P^0 (initialise perturbation operators set);
    n ← 0;
    while n < maxP and |P| > 0 do
        p ← choose a perturbation operator at random from P;
        O' ← p(O*);
        if O' if different from O* then
            O* ← O';
            P ← P^0;
            n ← n + 1;
        else
            P ← P\{p};
        end
    end
end
```

- **Transfer visit**: choose a random route and then transfer one of its its customer to another route in a different time period, choosing both, the route and period, at random;
- **Transfer route**: choose a random route and transfer it to another time period;
- **Copy route**: choose a random route and create an identical route in a different time period, removing from it the customers that are already visited in the given time period.

Only feasible changes are accepted, taking into account the necessary capacity and demand constraints. To further improve the performance of our heuristic algorithm we also included a shaking mechanism, which temporarily increases the size of the perturbation when stagnation occurs, i.e., the incumbent solution has not been improved over α iterations. For this shaking mechanism, the algorithm increases $maxP$ for a predefined number of iterations β in order to attempt escaping from potential local optima.

4.5. Reduced LP formulation

Given the values of the setup and visit variables ($\bar{z}$ and $\bar{y}$, respectively) for a given solution, we can find values of the continuous variables (production and delivery quantities, consumption amounts, and inventory levels) that maximise the total profit of the solution. For this, we make use of the following reduced LP formulation of the model presented in Section 3.

$$\max \quad \sum_{t\in\mathcal{T}}\left(\sum_{i\in\mathcal{C}}\sum_{s\in\mathcal{S}} u_{is}w_{is}^{t} - vp^{t} - \sum_{i\in\mathcal{N}}\sum_{s\in\mathcal{S}} h_i I_{is}^{t} - \sum_{i\in\mathcal{C}}\sum_{j\in\mathcal{C}:\ j\neq i}\sum_{s\in\mathcal{S}} b_{ij}r_{ijs}^{t}\right) \tag{26}$$

s.t. $(2)-(6),(8)-(9),$ and $(16)-(20)$,

$$p^t \leq P\bar{z}^t \qquad t \in \mathcal{T}, \tag{27}$$

$$\sum_{s\in\mathcal{S}} q_{is}^{kt} \leq min\{Q, C_i\}\bar{y}_i^{kt} \qquad i \in \mathcal{C}, k \in \mathcal{K}, t \in \mathcal{T}, \tag{28}$$

$$\sum_{i\in\mathcal{C}}\sum_{s\in\mathcal{S}} q_{is}^{kt} \leq Q\bar{y}_0^{kt} \qquad k \in \mathcal{K}, t \in \mathcal{T}. \tag{29}$$

Note that in this LP formulation the former y_i^{kt} and z^t variables are now known and thus replaced by the parameters $\bar{y}_i^{kt}$ and $\bar{z}^t$, while the constraints (12)-(15) have been removed. Following from this, we remove all empty setups (where $p^t = 0$ and $\bar{z}^t = 1$ for a given $t \in \mathcal{T}$) and visits (where $\sum_{s\in\mathcal{S}} q_{is}^{kt} = 0$ and $\bar{y}_i^{kt} = 1$ for a given $i \in \mathcal{C}$, $k \in \mathcal{K}$, $t \in \mathcal{T}$) that may remain after applying this LP to the solution.

4.6. Insertion and Removal MIP

In order to improve the performance of the heuristic, we use a mathematical programming-based component which consists of an insertion and removal MIP. This MIP is based on the original formulation of the problem and models the insertion and removal of customers from the routes of a given input solution ($\bar{y}$). This type of component is used to perform a MIP-based local search over the input solution and was originally proposed for the inventory routing problem by Archetti et al. (2012).

To present this component, we introduce the following notation. Let Λ_i^{kt} be the travel cost decrease resulting from the removal of customer i from the route of vehicle k in time period t. The value of this parameter can be computed as $c_{hi} + c_{ij} - c_{hj}$, where h and j are the predecessor and successor of the customer in the route, respectively. Let Ω_i^{kt} be the cheapest insertion cost of customer i into the route of vehicle k in time period t. Let λ_i^{kt} be a binary variable equal to 1, if customer i is removed from the route of vehicle k in period t, and 0, otherwise; and let ω_i^{kt} be a binary variable equal to 1, if customer i is inserted into the route of vehicle k in time period t, and 0, otherwise. Then, the Insertion and Removal MIP can be written as follows:

$$\max \quad \sum_{t\in\mathcal{T}}\Bigg(\sum_{i\in\mathcal{C}}\sum_{s\in\mathcal{S}} u_{is}w_{is}^t - fz^t - vp^t - \sum_{i\in\mathcal{N}}\sum_{s\in\mathcal{S}} h_i I_{is}^t - \sum_{i\in\mathcal{C}}\sum_{j\in\mathcal{C}:\, j\neq i}\sum_{s\in\mathcal{S}} b_{ij}r_{ijs}^t + \sum_{i\in\mathcal{C}}\sum_{k\in\mathcal{K}}\Lambda_i^{kt}\lambda_i^{kt} - \sum_{i\in\mathcal{C}}\sum_{k\in\mathcal{K}}\Omega_i^{kt}\omega_i^{kt}\Bigg) \tag{30}$$

s.t. $(2)-(9),(16)-(21)$,

$$\sum_{s\in\mathcal{S}} q_{is}^{kt} \leq \min\{Q, C_i\}(\bar{y}_i^{kt} - \lambda_i^{kt} + \omega_i^{kt}) \qquad i \in \mathcal{C}, k \in \mathcal{K}, t \in \mathcal{T}, \tag{31}$$

$$\sum_{i\in\mathcal{C}}\sum_{s\in\mathcal{S}} q_{is}^{kt} \leq Q(\bar{y}_0^{kt} - \lambda_i^{kt} + \omega_i^{kt}) \qquad k \in \mathcal{K}, t \in \mathcal{T}, \tag{32}$$

$$\sum_{k\in\mathcal{K}} (\bar{y}_i^{kt} - \lambda_i^{kt} + \omega_i^{kt}) \leq 1 \qquad i \in \mathcal{C}, t \in \mathcal{T}, \tag{33}$$

$$\omega_i^{kt} \leq 1 - \bar{y}_i^{kt} \qquad i \in \mathcal{C}, k \in \mathcal{K}, t \in \mathcal{T}, \tag{34}$$

$$\lambda_i^{kt} \leq \bar{y}_i^{kt} \qquad i \in \mathcal{C}, k \in \mathcal{K}, t \in \mathcal{T}, \tag{35}$$

$$\sum_{i\in\mathcal{C}} (\lambda_i^{kt} + \omega_i^{kt}) \leq 1 \qquad k \in \mathcal{K}, t \in \mathcal{T}, \tag{36}$$

$$\lambda_i^{kt} \in \{0, 1\} \qquad i \in \mathcal{C}, k \in \mathcal{K}, t \in \mathcal{T}, \tag{37}$$

$$\omega_i^{kt} \in \{0, 1\} \qquad i \in \mathcal{C}, k \in \mathcal{K}, t \in \mathcal{T}. \tag{38}$$

The objective function (30) maximises the total profit, which now includes the insertion and removal costs. Constraints (31) and (32) correspond to (10) and (11), respectively, while considering the insertion and removal variables. Constraints (33) ensure that each customer is visited by at most one vehicle per time period. Consistency of the insertion and removal operations with respect to the input solution $\bar{y}$, is ensured by constraints (34) and (35), while constraints (36) allow at most one insertion or removal per route. Finally, the domain of the insertion and removal variables is defined in constraints (37) and (38).

5. Computational experiments

In order to validate the MILP formulation and test the proposed heuristic, we carried out a number of computational experiments. The MILP formulation as well as the heuristic were coded in C++ and CPLEX 12.8 running on a single thread. All computations were executed on a machine equipped with a 2.1 GHz AMD Opteron 6172 processor and a limit of 8GB of RAM. The time limit for CPLEX was set to two hours when solving the full problem formulation and 20 seconds for the insertion and removal MIP.

5.1. Instance description

For our experiments, we used the benchmark test set proposed by Adulyasak et al. (2014a) for the standard PRP. This set consists of 168 instances, ranging from 10 to 50 customers, including 3, 6 or 9 time periods, and a fleet size of 2, 3 or 4 vehicles. To be applicable to the case of perishable products with transshipments, these instances were adapted using the following assumptions:

- We defined the maximum age of the products (S) to be equal to 2 for the instances with a planning horizon of 3 time periods. For each of the instances with a longer planning horizon, we create two instances, one with a maximum age of 2 and another one with a maximum age of 4. This results in a total of 264 instances for the perishable case.

- To ensure that the objective function values remain positive, the initial prices of products for a customer (u_{i0}) were defined as three times the unit production cost. Given the aging of the product, we assume a decrease in the price according to the rate of decay γ, which defines the percent of price reduction per time period, i.e., $u_{is} = (1 - \gamma)u_{i,s-1}$, for $s > 0$. Unless stated otherwise, the rate of decay γ is set to 5%.

- For the transshipments, the costs were determined as in Avci and Yildiz (2020) for the standard PRP with transshipments. As such, the transshipment cost b_{ij} was set equal to $\sigma \times c_{ij}/\bar{d}$, where $\bar{d}$ is the average demand value of the given instance and σ is a weight term for the unit transshipment cost. Unless stated otherwise, σ is set equal to 3.

As in Adulyasak et al. (2014a), the resulting set of instances can be categorised into four classes. The first class represents the base case, the second class depicts a case with higher production costs, the third class presents a case with higher transportation costs and the fourth class describes the case with no inventory costs at the customer locations.

5.2. Managerial Insights

Using the instances described in Section 5.1, this section presents valuable managerial insights into different aspects of the problem and the impact of perishability in the context of the PRP. Considering a number of different transshipment cost scenarios, Section 5.2.1 starts by analysing the use and effect of transshipments within the problem setting. Following from this, Section 5.2.2 gives insights into the impact of different forms of perishability while Section 5.2.3 addresses the value of integration by comparing the proposed problem formulation with the decomposed approach.

5.2.1. Insights into the impact of transshipments

To evaluate the impact of transshipments on perishability, and the benefits the former brings to the management of PRPs with perishable products, a series of experiments was carried out, altering the values for the transshipment cost. Changing the weight term σ that determines the relative importance of the unit transshipments cost, the experiments consider five different scenarios with gradually increasing costs for the transshipments, where the scenario with weight 5 has the highest associated cost.

A comparison between these scenarios, with the different values of σ and a decay rate γ of 5%, is provided in Figure 1, presenting the average percentage change in the different objective function components with respect to the case where no transshipments are allowed. The values shown are averages derived from the optimal solutions obtained with the branch-and-cut algorithm for all instances.

The results show that allowing transshipments in the system increases the profit of the solutions. This is in line with expectations and can be explained with the savings generated in the total cost, as lower transshipment costs lead to a higher use of transshipments which, in turn, significantly reduces the transportation and inventory holding costs. The flexibility that transshipments offer to the system permits more efficient management of the inventories at the customer level, allowing significant economies at the routing level since less frequent visits are required to maintain the same revenue. This effect also translates into savings at the plant level, requiring less frequent setup and production operations.

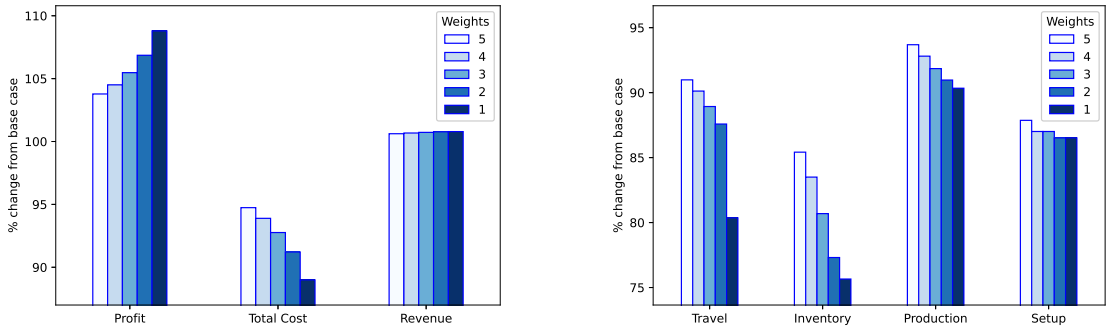

Figure 1: Changes in the objective function components for different weights of the transshipment cost

To further illustrate the effect of transshipments, Figure 2 shows the change of the average inventory level as well as the change in the proportion of fresh (age 0) and old (age S) inventory at the customers. This analysis reveals that transshipments allow for a significant reduction of the total inventory quantities kept at the customer locations. This is caused mostly by a reduction in the storage of old products, whereas the proportion of fresh products in the inventory increases. A partial explanation for these changes is the better management of the initial inventories at the customer locations, which can be redistributed using transshipments. These observations highlight the benefits of transshipments for the management of perishability in the context of integrated production and distribution planning.

5.2.2. Insights into the effect of different forms of perishability

In this section, we investigate the impact of different types of perishability on profits, revenues and the costs involved. For this reason, we consider the following three scenarios:

- **No perishability:** The quality of products stays the same and there is no expiry date. As such,

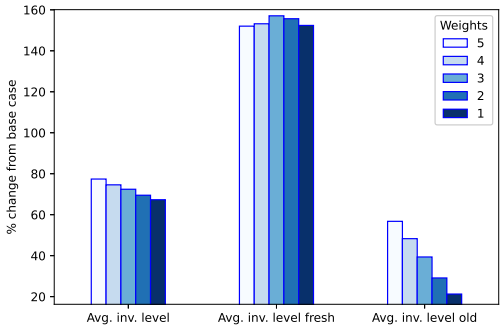

Figure 2: Changes in the inventories for different weights of the transshipment cost

the produced products can be sold at stable prices throughout the time horizon.

- **Fixed shelf life:** The quality of products remains the same until a designated expiry date is reached and the product has to be discarded. Products can, thus, be sold at stable prices until their expiry date is reached.

- **Gradual decay:** Products deteriorate gradually over time until they are no longer suitable for sale. This is reflected in a gradual decrease in the price of products until the product is discarded.

Figure 3 provides a comparison of these three scenarios, highlighting the relative differences in profit, revenue and costs. The results are presented in terms of a percentage change from the scenario with no perishability, which functions as the base case. All results are based on average values of the solutions found by CPLEX, considering only the instances that could be solved to optimality for all three scenarios.

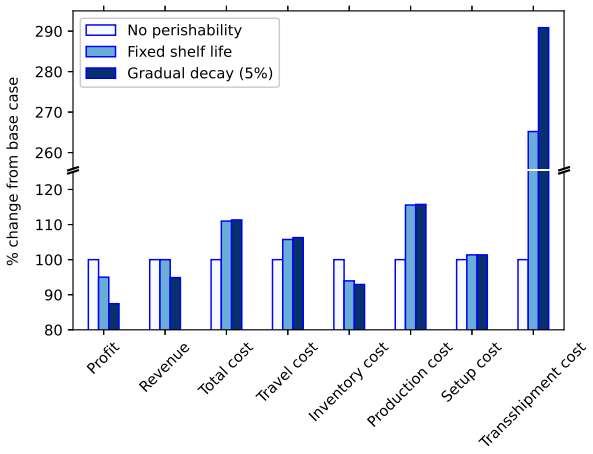

Figure 3: Change in profit, revenue and costs for different perishability scenarios

The comparison in Figure 3 shows that all types of perishability lead to an increase in most of the costs involved as well as a reduction in the overall profit. The revenue, however, remains unaffected in the

case of a fixed shelf life as the prices of products remain the same over time. An exception from the general increase in costs can be observed for the part of the inventory costs as perishable products are usually sold faster and thus spend less time in storage. The sharp increase in the use of transshipments reveals their significance in the management of perishable products, as transshipments play a key role in the inventory control and distribution across the system. In the case of gradual decay these trends are amplified in comparison to the case of a fixed shelf life, given the continuous and more apparent value loss over time.

Results for alternative rates of decay

Taking a more detailed look at the effect of gradual decay, this section investigates the impact of changes in the rate of decay γ, which describes the reduction in the price as the age of the product increases. Four different values for the rate of decay have been analysed and compared with regards to the characteristics of the resulting solutions. Figure 4 provides a summary of this comparison, highlighting the relative difference in profit, revenue and total cost between scenarios for each instance class. The results are presented as a percentage change with respect to a base case ($\gamma = 5\%$), for three scenarios assuming a decay rate of 10, 15 and 20 percent. All the results are based on average values of the solutions found by CPLEX, considering only the instances that could be solved to optimality for all rates of decay. It is worth mentioning, however, that the solutions of the heuristic follow a similar pattern with regards to the different cost components involved.

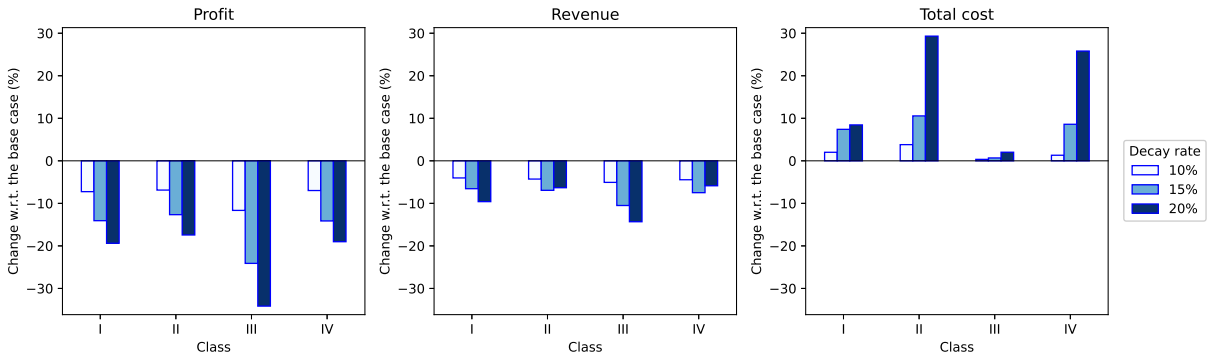

Figure 4: Change of the profit, revenue and total costs with respect to the base case for the different instance classes.

From Figure 4, we observe that, for all instance classes, higher rates of decay generally result in a decrease in both revenue and profit, as well as an increase in the total cost incurred. This behavior is expected as the profit margins decrease more quickly with a higher rate of decay. Extending this analysis, Figure 5 shows the relative differences in the individual costs components for each of the classes. This analysis reveals that higher rates of decay result in larger travel costs as there is a need to replenish customers

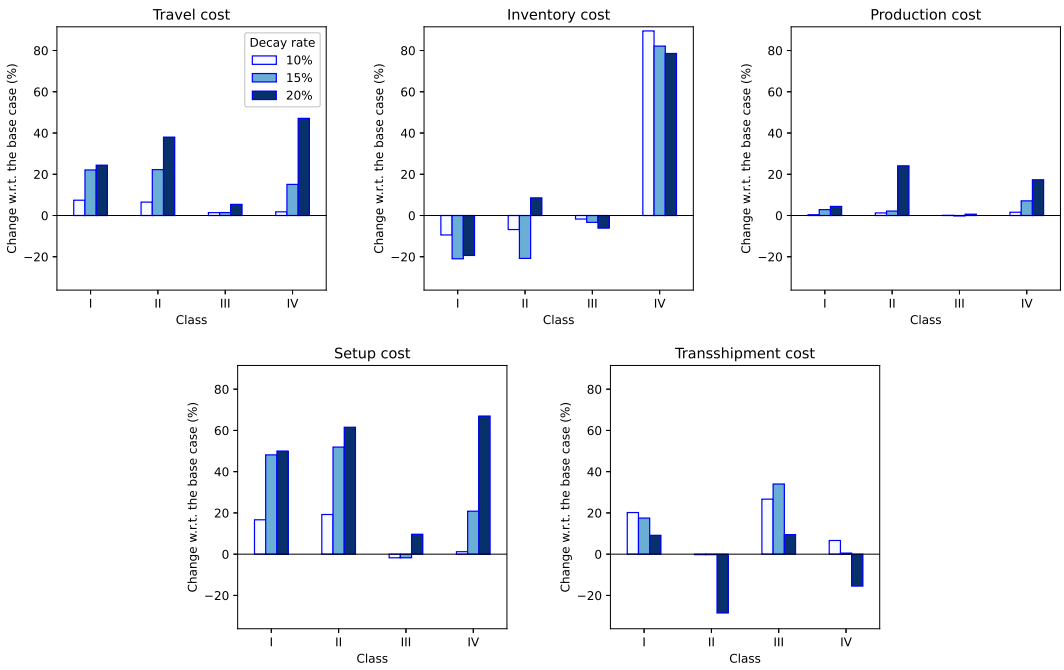

Figure 5: Change of the cost components with respect to the base case for the different instance classes

more frequently in order to maximise the revenue obtained. Moreover, in general, the inventory cost decreases for higher values of γ since products tend to spend less time in storage due to the rapid decay in quality. At the same time, given the faster product deterioration, production takes place in smaller lot sizes, which results in lower inventory levels throughout the system. This also implies that production setups and deliveries occur more frequently, in order to stock and deliver smaller batches of fresh products on a more regular basis, leading to higher setup and routing costs.

We also notice that the impact of different decay rates is not the same for all instance classes due to differences in the cost structures. For example, for Class III (with high routing costs) the impact of higher decay rates is much less pronounced, with routing, production and setup costs increasing by 5%, 1% and 10%, respectively, for $\gamma = 20\%$. These increases, however, are large enough to cut the profits by 34%. On the other hand, for Class II (with high production costs), we observe that higher decay rates have a stronger impact on production costs, increasing the relative differences from 1% for $\gamma = 10\%$ to 24% for $\gamma = 20\%$. Similarly for the transshipment costs, the relative differences decrease from 0% for $\gamma = 10\%$ to -28% for $\gamma = 20\%$. Finally, for Class IV (with no inventory costs at the customers), we observe, besides large increases in setup and routing cost, a sharp increase in inventory costs. This increase can be misleading as it is mostly a result of relative increases in the very small values for the inventory costs of these instances. For the transshipments costs, we observe that the optimal strategy depends on the considered rate of decay, impacting the profitability of using transshipments to redistribute inventories.

5.2.3. Insights into the value of integration

In this section we compare the results of CPLEX against a sequential approach that solves the problem in two steps. The first step consists of solving an IRP that maximises the total profit, without taking into account production and inventory decisions at the plant. The second step, then uses the solution to this IRP solving a lot-sizing problem that aims to minimise production, setup and inventory costs at the plant. Our main goal is to highlight the benefits that can be achieved by integrating production, setup, inventory and delivery decisions for perishable products in a single decision-making framework. Appendix A describes the two MIP components of this sequential procedure, which are solved sequentially in order to find a feasible solution to the problem.

Table 1 summarises the relative differences of our integrated model with respect to the sequential approach, in terms of total profit, revenue and cost. For a fair comparison, we only consider instances with 10–25 customers for which CPLEX could optimally solve model (1)–(25) within two hours of computing time. The analysis clearly shows that our integrated model is able to provide solutions that are, on average, 27.38% more profitable than those found by the sequential approach. Such gains come mainly from reductions in the total cost (32.58%, on average) that are achieved due to a better coordination between production, setup, inventory and delivery decisions. Interestingly, it is also possible to observe a small reduction (3.34%, on average) in the revenue obtained by our integrated model. This can be explained by the fact that the integrated model utilises inventories of different ages better, satisfying customer demands in a more cost-effective way than the sequential approach. However, making use of less fresh products also leads to a lower revenue due to the age-dependent pricing. Nevertheless, this reduction in revenue is clearly offset by a much more substantial reduction in the total cost of the problem.

Table 1: Relative difference of profit, revenue and cost of the integrated model with respect to the sequential approach.

T	C	Profit	Revenue	Cost
3	10	41.34	-3.24	-37.70
	15	23.70	-3.18	-32.48
	20	23.87	-3.09	-32.90
	25	21.92	-3.54	-34.81
6	10	32.02	-2.74	-29.80
	15	23.11	-3.81	-33.17
	20	22.41	-3.44	-30.97
	25	22.12	-4.36	-36.58
9	10	34.03	-2.83	-29.36
	15	32.52	-5.24	-44.25
	20	27.39	-4.08	-36.20
	25	7.38	-3.56	-18.03
Average		27.38	-3.34	-32.58

Table 2: Changes on solution structure for the integrated model with respect to the sequential approach.

C	#Routes	#Visits	#Setups	Avg. Inventory.	Avg. Delivery.	Avg. Lot-size	Avg. Transshipment
10	-22.71	-36.77	-55.56	-15.38	29.51	78.32	-10.56
15	-35.89	-42.50	-59.07	-15.90	29.62	77.18	29.29
20	-28.03	-44.28	-57.22	-27.56	36.95	75.60	157.89
25	-36.68	-39.82	-57.87	-30.52	15.16	67.38	53.19
Average	-29.04	-40.34	-57.06	-21.33	28.93	75.46	69.38

These results also point out the disadvantages of using a myopic sequential procedure that initially aims to optimise inventory and delivery decisions at the customers, disregarding their effect and interaction with production, setup and inventory decisions at the plant. In order to maximise the profit, the initial inventory routing plan delivers small batches of fresh products, which leads to more routes and visits over the planning horizon. This inventory and delivery pattern then forces small lot-sizes and frequent setups at the plant, which results in high production costs. The final result is then, after considering production costs, a much lower total profit.

These findings are also confirmed by Table 2, which shows the relative changes on the structure of the solutions found by our integrated model with respect to the sequential approach. In general, optimal solutions for the integrated model tend to deliver larger batches of products, which results in fewer routes and customer visits. Also, it is possible to observe that solutions tend to perform fewer setups and produce larger lot-sizes. As a final remark, we note that transshipment is an effective way of repositioning inventory of different ages between customers, leading to a significant reduction in the overall inventory level across the system (21.33%, on average).

5.3. Results with the heuristic algorithm

This section presents the results of different experiments carried out to evaluate the performance of the proposed heuristic algorithm. In this context, we first assess the performance of the heuristic for the perishable case with different configurations of the transshipment costs and the rates of decay. We then evaluate its performance for the standard PRP with respect to the state-of-the-art heuristics from the literature.

5.3.1. Parameter settings

For the tuning of the heuristic parameters, a test set consisting of 40 instances was generated by selecting instances at random from the different instance classes. The tuning on this test set was carried out using a sequential approach, where three test runs were executed for each parameter setting. The stopping criteria for all the experiments in the tuning process were set to a maximum of 10,000 iterations and

a maximum run time of 600 seconds. The following formula was used to determine the size of the perturbation $maxP = \pi(N + \lambda T)$, where π is a multiplier, N is the number of customers of the instance, T represents the number of time periods, and λ is a weight for the number of periods. The rational behind this approach is that the perturbation size should depend on the size of the instance, with the number of customers and time periods being the main determinants of the instance size. Based on these experiments, π was set to 0.07 and λ to 2. The best parameter settings for the shaking mechanism, resulted in α to be set to 600 iterations and β to 100 iterations. Whenever this mechanism is called, $maxP$ is increased by one. The execution of the heuristic is stopped once one of the following three stopping criteria is reached: a maximum run time of 600 seconds, a maximum of 10,000 iterations or a maximum of 5,000 iterations without improvement.

5.3.2. Results for the perishable case

This section presents the results for the set of instances described in Section 5.1, focusing on the performance of the model in CPLEX and the proposed hybrid heuristic algorithm. The analysis with regards to the heuristic is based on results from one heuristic run carried out for each of the instances. A comparison between the two solution approaches with regards to different instance characteristics is shown in Tables 3-4. The column headings of the tables present the instance class (Class), the number of instances (#), the number of feasible (#F) and optimal (#O) solutions found by CPLEX as well as the resulting average optimality gap (Opt gap) per class and the average run time (Time), in seconds. For the heuristic, 'Opt gap' refers to the optimality gap between the heuristic solutions and the solutions found by CPLEX for all the instances solved to optimality. Extending the analysis to all the instances where CPLEX finds feasible solutions, the relative differences (Rel diff) compare the solutions found by the heuristic with the best solutions found by CPLEX within the given time limit. The average run time (Time), in seconds, and the average number of iterations (Iter) used are presented in the final columns of the table.

Table 3 shows a comparison of the average performance of the model and the heuristic for each of the four instance classes. Overall, CPLEX is able to find 250 feasible solutions and 108 optimal solutions for the 264 instances, with an average optimality gap of 4.60%. The heuristic in contrast finds feasible solutions for all instances with an average optimality gap of 0.51% to the optimal solutions of CPLEX. Looking at the relative differences, the heuristic outperforms CPLEX by 0.83%. Moreover, the average run time of the heuristic corresponds to only about 11% of the run time of CPLEX. Looking at the different instance classes, it can be seen that the performance of both the model and the heuristic differ between classes. Finding feasible and/or optimal solutions with CPLEX seems to be particularly

challenging for instance class III, with CPLEX only proving the optimality of 30.3% of the instances in the class and an average optimality gap of 17.14%. The heuristic in comparison finds solutions that are on average -3.62% better than the results of the model. These observations can be partially explained by the fact that, for this instance class, CPLEX takes significantly longer to find the first feasible solution (on average more than twice as long as for the other instance classes). In contrast, the construction heuristic of our proposed solution method is able to find feasible solutions typically in less than a second for all of the considered instance classes and thus has more time to improve the quality of the found solutions.

Table 3: Comparison between CPLEX and the performance of the heuristic for the four instance classes

	CPLEX					Heuristic			
Class	#	#F	#O	Opt gap (%)	Total time	Opt gap* (%)	Rel diff (%)	Total time	# of iterations
I	66	65	30	0.95	4,269.73	0.23	-0.11	478.30	4,602
II	66	64	30	0.07	4,039.50	0.02	0.00	489.18	4,112
III	66	60	20	17.14	4,980.22	1.80	-3.62	465.43	5,413
IV	66	61	28	0.93	4,329.87	0.41	0.28	524.33	3,897
Total	264	250	108	4.60	4,395.98	0.51	-0.83	489.31	4,506

In addition to the differences between instance classes, we also investigate other structural features that may impact the performance. Table 4 presents in this context an overview of the results found by CPLEX and the heuristic detailed for the number of time periods (T) and the number of customers (N) considered. It can be seen that the number of time periods plays an important role for the performance of CPLEX, showcasing an increase in the optimality gap as the number of time periods increases. The heuristic again outperforms the model, while staying relatively consistent in terms of the differences to the model when comparing between different values of T. Looking at the instances with 3 and 6 time periods, an increase in the number of customers generally negatively impacts the performance of CPLEX, while for the instances with 9 time periods no clear trend could be identified. This finding is, however, in line with the findings presented in Adulyasak et al. (2014a) and most likely caused by structural traits of the instances. The main observation for the heuristic, in this context, is the increase in run time as the instance size increases.

To further evaluate the performance of the heuristic, Table 5 presents the average optimality gap and Table 6 the relative differences for the solutions of the heuristic, computed with respect to the solutions found by CPLEX. The results are displayed for different configurations of the transshipment cost weights and rates of decay, including the case in which no transshipments are allowed. The observations show that the performance of the heuristic remains competitive and consistent with respect to the branch-and-cut algorithm for all the considered configurations. Moreover, the heuristic finds feasible solutions for all the

Table 4: Comparison between CPLEX and the performance of the heuristic for different instance sizes

		CPLEX					Heuristic			
T	N	#	#F	#O	Gap (%)	Time	Opt gap* (%)	Rel diff (%)	Total time	# of iterations
3	10	8	8	8	0.00	1.56	0.01	0.01	38.62	5,554
	15	8	8	8	0.00	42.08	0.07	0.07	99.52	7,761
	20	8	8	8	0.00	24.53	0.35	0.35	198.92	6,444
	25	8	8	8	0.00	89.33	0.05	0.05	268.23	6,627
	30	8	8	7	0.61	2,779.77	0.45	-0.15	435.82	7,294
	35	8	8	1	1.62	6,667.99	1.43	-0.16	514.69	5,323
	40	8	8	1	1.98	6,348.26	1.72	-0.24	600.04	5,268
	45	8	8	0	4.09	7,200.03	2.12	-1.72	594.53	4,305
	50	8	8	2	2.60	5,654.32	1.52	-0.95	600.05	2,670
6	10	16	16	16	0.00	35.82	0.86	0.86	170.91	6,696
	15	16	16	9	1.80	3,715.12	2.57	0.76	433.49	7,751
	20	16	16	14	0.28	1,290.65	1.39	1.10	575.69	4,875
	25	16	16	8	1.55	4,180.07	1.94	0.41	600.08	3,657
	30	16	16	0	5.92	7,200.05	3.83	-1.66	600.12	2,956
	35	16	12	0	7.01	7,200.14	3.30	-2.71	600.12	1,549
	40	16	13	0	8.01	7,200.06	2.81	-3.78	600.28	1,063
9	10	16	16	13	1.42	2,299.47	3.22	1.67	429.34	7,406
	15	16	16	1	12.68	7,037.35	8.49	-2.30	587.80	5,852
	20	16	16	3	4.71	6,185.77	4.30	-0.03	600.11	3,753
	25	16	16	1	8.27	7,038.79	6.15	-1.06	600.18	1,977
	30	16	9	0	32.18	7,200.09	7.55	-11.00	600.29	1,192
3		72	72	43	1.21	3,200.88	0.86	-0.30	372.27	5,694
6		112	105	47	3.25	4,216.66	2.34	-0.55	511.53	4,078
9		80	73	18	9.90	5,832.64	5.79	-1.74	563.54	4,036
Total		264	250	108	4.60	4,395.98	2.92	-0.83	489.31	4,506

instances, while CPLEX fails to do so.

Table 5: Optimality gap of the solutions of the heuristic for different configurations

Decay	Weight σ					No
rate	1	2	3	4	5	transs.
5%	0.16	0.34	0.51	0.69	0.58	0.42
10%	0.24	0.5	0.57	0.77	0.85	0.55

Table 6: Relative difference of the solutions of the heuristic for different configurations

Decay	Weight σ					No
rate	1	2	3	4	5	transs.
5%	-0.83	-0.49	-0.83	-0.89	-0.81	-0.11
10%	-0.65	-0.29	-0.38	-0.17	-0.56	-0.11

5.3.3. Results for the standard PRP

This section presents the results of the heuristic algorithm when applied to solve the standard PRP. For these experiments, we used the benchmark instance set of Archetti et al. (2011). This instance set is composed of three subsets of different size, consisting of 14 (set A1), 50 (set A2) and 100 (set A3) customer locations. Each subset contains 480 instances divided in the same four classes as the instances

described in Section 5.1. For the tuning of the heuristic we apply the same procedure as outlined in Section 5.3.1 on a test set of 160 instances. This results in the same parameter setting as described for the perishable case with the exception of parameters λ, which is now set to 0, and $maxP$ during the shaking mechanism, which is now increased by five (instead of by one). The execution of the main loop in the heuristic in this case is stopped once one of the following three stopping criteria is reached: a maximum run time of 600 seconds, a maximum of 5,000 iterations or a maximum of 4,000 iterations without improvement.

We compare the performance of our heuristic (best-out-of 5 runs) with respect to the best known solutions found by our (ILS) and several other heuristics presented in the scientific literature. In this context, the following methods have been included in the comparison: the optimisation-based ALNS of Adulyasak et al. (2014b) (ALNS), the multi-start iterative method of Absi et al. (2015) (IM-MS), the multi-phase heuristic of Solyalı and Süral (2017) (5P), the VNS heuristic of Qiu et al. (2018b) (VNS), a three-level heuristic of Li et al. (2019) (3LH), the decomposition matheuristic of Chitsaz et al. (2019) (CCJ-DH), the matheuristic algorithm of Avci and Yildiz (2019) (MA) and the infeasible space exploration matheuristic of Manousakis et al. (2021) (ISMA).

An overview of this comparison is shown in Table 7 for set A1 and Table 8 for sets A2 and A3. The performance for the small sized instances of A1 is shown in terms of the optimality gaps for the solutions found by the heuristics computed with regards to the optimal solutions found by Ruokokoski et al. (2010) (Solyalı and Süral, 2017). It can be seen from the table that for most classes our heuristic finds competitive solutions close to optimality. The largest gap can be observed for class III with an average optimality gap of 1.66%, which is significantly better than the performance of the ALNS of Adulyasak et al. (2014a) and comparable to the VNS of Qiu et al. (2018a) and the CCJ-DH method of (Chitsaz et al., 2019) for this class. The slightly worse performance for Class III can be partially explained by the relatively large impact of the visit decisions due to the higher routing costs considered in this class. The same issue can also be observed for the ALNS method proposed in Adulyasak et al. (2014b), which was specifically designed for the standard PRP.

Table 7: Average optimality gaps of the solutions for set A1

Class	ALNS	IM-MS	5P	VNS	3LH	CCJ-DH	MA	ISMA	ILS
I	1.70	0.09	0.03	0.28	0.15	0.24	0.01	0.05	0.50
II	0.36	0.01	0.00	0.04	0.03	0.03	0.00	0.01	0.09
III	8.43	0.57	0.18	1.52	0.76	1.49	0.04	0.39	1.66
IV	0.93	0.02	0.03	0.56	0.08	0.11	0.01	0.00	0.19
Avg	2.85	0.17	0.06	0.60	0.25	0.47	0.02	0.11	0.61

For the larger instances of 50 and 100 customers the results of the heuristics are compared with regards

to the best solutions found by the considered methods. Table 8 shows, for each method, the average deviations from these solutions for each of the instance classes. It can be observed that our heuristic finds reasonably good results with average deviations from the best known solutions of about 1.5%. The average run times for set A1, A2 and A3 are 22, 330 and 600 seconds, respectively. These results indicate that our heuristic shows a reasonable performance that is competitive with other methods that have been designed specifically for the standard PRP.

Table 8: Relative deviations with respect to the best known solution for sets A2 and A3

Set	Class	ALNS	IM-MS	5P	VNS	3LH	CCJ-DH	MA	ISMA	ILS
A2	I	1.35	0.38	0.27	0.26	0.21	0.16	0.18	0.02	1.70
	II	0.22	0.10	0.06	0.08	0.05	0.06	0.06	0.00	0.13
	III	4.39	1.83	0.92	0.87	0.82	0.62	0.78	0.06	2.33
	IV	0.35	0.25	0.21	0.13	0.18	0.13	0.17	0.06	0.87
	Avg	1.57	0.64	0.36	0.34	0.32	0.24	0.30	0.04	1.26
A3	I	1.05	0.27	0.12	0.23	0.08	0.23	0.07	0.19	2.03
	II	0.16	0.06	0.04	0.08	0.03	0.04	0.05	0.00	0.27
	III	3.97	1.54	0.46	0.62	0.47	1.38	0.55	0.37	3.54
	IV	0.35	0.26	0.07	0.10	0.06	0.08	0.11	0.14	1.11
	Avg	1.38	0.53	0.17	0.26	0.16	0.43	0.19	0.17	1.74

6. Conclusions

In this paper, we introduced the PRP for a perishable product with fixed shelf life and gradual decay under consideration of transshipments. We formulated the problem as an MILP and solved it by using a branch-and-cut as well as a hybrid heuristic algorithm that combines features from the iterated local search metaheuristic and mathematical programming techniques. We tested the proposed methods on a set of instances generated on the basis of well-known benchmark instances from the literature. Computational experiments on these instances showed that the branch-and-cut algorithm is capable of optimally solving small-sized instances, while the heuristic method provides high-quality feasible solutions within reasonable running times for all instance sizes.

Furthermore, we investigated the impact of perishability, in the form of different decay rates, on the solutions structure to gain managerial insights with respect to the optimal decision making and resource utilisation. In particular, we observed that larger rates of decay result in reductions of the total profit which, in turn, come from both a decrease in revenue (due to quality/price decay) and an increase in the total costs (due to higher setup and routing costs). In this context, we also studied the significance of transshipments for the management of perishability across the system. Finally, we applied our heuristic method to solve the standard PRP, and compared its performance to a number of state-of-the-art solution methods for the problem. The results reveal that our method is competitive when compared to the

methods from the literature, finding good quality solutions within relatively short running times. Future research could build on these findings and further explore and extend the proposed problem and solution methods, by considering a multi-product setting and a stochastic environment.

References

Absi, N., Archetti, C., Dauzère-Pérès, S., and Feillet, D. (2015). A two-phase iterative heuristic approach for the production routing problem. *Transportation Science*, 49(4):784–795.

Adulyasak, Y., Cordeau, J.-F., and Jans, R. (2014a). Formulations and branch-and-cut algorithms for multivehicle production and inventory routing problems. *INFORMS Journal on Computing*, 26(1):103–120.

Adulyasak, Y., Cordeau, J.-F., and Jans, R. (2014b). Optimization-based adaptive large neighborhood search for the production routing problem. *Transportation Science*, 48(1):20–45.

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

Adulyasak, Y., Cordeau, J.-F., and Jans, R. (2015b). The production routing problem: A review of formulations and solution algorithms. *Computers & Operations Research*, 55:141–152.

Alvarez, A., Cordeau, J.-F., Jans, R., Munari, P., and Morabito, R. (2020). Formulations, branch-and-cut and a hybrid heuristic algorithm for an inventory routing problem with perishable products. *European Journal of Operational Research*, 283(2):511–529.

Alvarez, A., Cordeau, J.-F., Jans, R., Munari, P., and Morabito, R. (2021). Inventory routing under stochastic supply and demand. *Omega*, 102:102304.

Alvarez, A., Munari, P., and Morabito, R. (2018). Iterated local search and simulated annealing algorithms for the inventory routing problem. *International Transactions in Operational Research*, 25:1785–1809.

Amorim, P., Belo-Filho, M., Toledo, F. M., Almender, C., and Almada-Lobo, B. (2013). Lot sizing versus batching in the production and distribution planning of perishable goods. *International Journal of Production Economics*, 146(1):208–218.

Applegate, D., Bixby, R., Chvátal, V., and Cook, W. (2018). Concorde TSP solver. `http://www.math.uwaterloo.ca/tsp/concorde.html`. Accessed: 2018-07-20.

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., Paletta, G., and Speranza, M. G. (2011). Analysis of the maximum level policy in a production-distribution system. *Computers & Operations Research*, 38(12):1731–1746.

Armentano, V., Shiguemoto, A., and Løkketangen, A. (2011). Tabu search with path relinking for an integrated production-distribution problem. *Computers & Operations Research*, 38(8):1199–1209.

Avci, M. and Yildiz, S. T. (2019). A matheuristic solution approach for the production routing problem with visit spacing policy. *European Journal of Operational Research*, 279(2):572–588.

Avci, M. and Yildiz, S. T. (2020). A mathematical programming-based heuristic for the production routing problem with transshipments. *Computers & Operations Research*, 123:105042.

Bard, J. F. and Nananukul, N. (2009a). Heuristics for a multiperiod inventory routing problem with production decisions. *Computers & Industrial Engineering*, 57(3):713–723.

Bard, J. F. and Nananukul, N. (2009b). The integrated production-inventory-distribution-routing problem. *Journal of Scheduling*, 12(3):257–280.

Bard, J. F. and Nananukul, N. (2010). A branch-and-price algorithm for an integrated production and inventory routing problem. *Computers & Operations Research*, 37(12):2202–2217.

Belo-Filho, M., Amorim, P., and Almada-Lobo, B. (2015). An adaptive large neighbourhood search for the operational integrated production and distribution problem of perishable products. *International Journal of Production Research*, 53(20):6040–6058.

Bertazzi, L., Paletta, G., and Speranza, M. G. (2005). Minimizing the total cost in an integrated vendor-managed inventory system. *Journal of Heuristics*, 11(5-6):393–419.

Boudia, M., Louly, M., and Prins, C. (2007). A reactive grasp and path relinking for a combined production-distribution problem. *Computers & Operations Research*, 34(11):3402–3419.

Boudia, M. and Prins, C. (2009). A memetic algorithm with dynamic population management for an integrated production-distribution problem. *European Journal of Operational Research*, 195(3):703–715.

Brahimi, N. and Aouam, T. (2016). Multi-item production routing problem with backordering: a milp approach. *International Journal of Production Research*, 54(4):1076–1093.

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

Chitsaz, M., Cordeau, J.-F., and Jans, R. (2019). A unified decomposition matheuristic for assembly, production, and inventory routing. *INFORMS Journal on Computing*, 31(1):134–152.

Coelho, L. and Laporte, G. (2014). Optimal joint replenishment, delivery and inventory management policies for perishable products. *Computers & Operations Research*, 47:42–52.

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

Fumero, F. and Vercellis, C. (1999). Synchronized development of production, inventory, and distribution schedules. *Transportation Science*, 33:330–340.

Jans, R. and Degraeve, Z. (2008). Modeling industrial lot sizing problems: a review. *International Journal of Production Research*, 46(6):1619–1643.

Lei, L., Liu, S., Ruszczynski, A., and Park, S. (2006). On the integrated production, inventory, and distribution routing problem. *IIE Transactions*, 38(11):955–970.

Li, Y., Chu, F., Chu, C., and Zhu, Z. (2019). An efficient three-level heuristic for the large-scaled multi-product production routing problem with outsourcing. *European Journal of Operational Research*, 272(3):914–927.

Li, Y., Chu, F., Côté, J.-F., Coelho, L. C., and Chu, C. (2020). The multi-plant perishable food production routing with packaging consideration. *International Journal of Production Economics*, 221:107472.

Li, Y., Chu, F., Feng, C., Chu, C., and Zhou, M. (2018). Integrated production inventory routing planning for intelligent food logistics systems. *IEEE Transactions on Intelligent Transportation Systems*, 20(3):867–878.

Lourenço, H. R., Martin, O. C., and Stützle, T. (2019). Iterated local search: Framework and applications. In *Handbook of Metaheuristics*, pages 129–168. Springer.

Manousakis, E. G., Kasapidis, G. A., Kiranoudis, C. T., and Zachariadis, E. E. (2021). An infeasible space exploring matheuristic for the production routing problem. *European Journal of Operational Research*.

Miranda, P. L., Cordeau, J.-F., Ferreira, D., Jans, R., and Morabito, R. (2018a). A decomposition heuristic for a rich production routing problem. *Computers & Operations Research*, 98:211–230.

Miranda, P. L., Morabito, R., and Ferreira, D. (2018b). Optimization model for a production, inventory, distribution and routing problem in small furniture companies. *Top*, 26(1):30–67.

Miranda, P. L., Morabito, R., and Ferreira, D. (2019). Mixed integer formulations for a coupled lot-scheduling and vehicle routing problem in furniture settings. *INFOR: Information Systems and Operational Research*, 57(4):563–596.

Neves-Moreira, F., Almada-Lobo, B., Cordeau, J.-F., Guimarães, L., and Jans, R. (2019). Solving a large multi-product production-routing problem with delivery time windows. *Omega*, 86:154–172.

Pochet, Y. and Wolsey, L. A. (2006). *Production Planning by Mixed Integer Programming*. Springer Science & Business Media.

Qiu, Y., Qiao, J., and Pardalos, P. M. (2019). Optimal production, replenishment, delivery, routing and inventory management policies for products with perishable inventory. *Omega*, 82:193–204.

Qiu, Y., Wang, L., Xu, X., Fang, X., and Pardalos, P. M. (2018a). Formulations and branch-and-cut algorithms for multi-product multi-vehicle production routing problems with startup cost. *Expert Systems with Applications*, 98:1–10.

Qiu, Y., Wang, L., Xu, X., Fang, X., and Pardalos, P. M. (2018b). A variable neighborhood search heuristic algorithm for production routing problems. *Applied Soft Computing*, 66:311–318.

Rohmer, S., Claassen, G., and Laporte, G. (2019). A two-echelon inventory routing problem for perishable products. *Computers & Operations Research*, 107:156–172.

Ruokokoski, M., Solyalı, O., Cordeau, J., Jans, R., and Süral, H. (2010). Efficient formulations and a branch-and-cut algorithm for a production-routing problem. Technical report, G-2010-66, GERAD, Canada.

Shaabani, H. and Kamalabadi, I. N. (2016). An efficient population-based simulated annealing algorithm for the multi-product multi-retailer perishable inventory routing problem. *Computers & Industrial Engineering*, 99:189–201.

Solyalı, O. and Süral, H. (2017). A multi-phase heuristic for the production routing problem. *Computers & Operations Research*, 87:114–124.

Subramanian, A., Uchoa, E., and Ochi, L. S. (2013). A hybrid algorithm for a class of vehicle routing problems. *Computers & Operations Research*, 40(10):2519–2531.

Toth, P. and Vigo, D. (2014). *Vehicle Routing: Problems, Methods, and Applications*, volume 18. SIAM.

Appendix A. Sequential Approach

In this appendix we present a sequential approach to solve the production routing problem for perishable products introduced in Section 2. In this approach, we initially solve an inventory routing problem that aims to maximise the total profit of the problem without considering production and inventory costs at the plant. Afterwards, given the solution to this inventory routing problem, we proceed to solve a lot-sizing problem in order to determine production quantities, inventory levels and setups decisions at the plant. It is worth mentioning that this approach may fail to find a feasible solution to the problem, as the production capacity might not be enough to meet the production requirements that are imposed by the output of the inventory routing problem.

Considering the notation introduced in Section 3, the inventory routing model is stated as follows:

$$\max \quad \sum_{t\in\mathcal{T}}\left(\sum_{i\in\mathcal{C}}\sum_{s\in\mathcal{S}} u_{is}w_{is}^{t} - \sum_{(i,j)\in\mathcal{A}}\sum_{k\in\mathcal{K}} c_{ij}x_{ij}^{kt} - \sum_{i\in\mathcal{C}}\sum_{s\in\mathcal{S}} h_i I_{is}^{t} - \sum_{i\in\mathcal{C}}\sum_{j\in\mathcal{C}:\, j\neq i}\sum_{s\in\mathcal{S}} b_{ij}r_{ijs}^{t}\right) \tag{A.1}$$

$$\text{s.t.} \quad I_{is}^{t} = \sum_{k\in\mathcal{K}} q_{is}^{kt} + \sum_{j\in\mathcal{C}:\, j\neq i} r_{jis}^{t} - \sum_{j\in\mathcal{C}:\, j\neq i} r_{ijs}^{t} - w_{is}^{t} \qquad i\in\mathcal{C}, t\in\mathcal{T}, s=0, \tag{A.2}$$

$$I_{is}^{t} = I_{i,s-1}^{t-1} + \sum_{k\in\mathcal{K}} q_{is}^{kt} + \sum_{j\in\mathcal{C}:\, j\neq i} r_{jis}^{t} - \sum_{j\in\mathcal{C}:\, j\neq i} r_{ijs}^{t} - w_{is}^{t} \qquad i\in\mathcal{C}, t\in\mathcal{T}, s\in\mathcal{S}\backslash\{0\}, \tag{A.3}$$

$$d_i^{t} = \sum_{s\in\mathcal{S}} w_{is}^{t} \qquad i\in\mathcal{C}, t\in\mathcal{T}, \tag{A.4}$$

$$\sum_{s\in\mathcal{S}} I_{is}^{t} \leq C_i - d_i^{t} \qquad i\in\mathcal{C}, t\in\mathcal{T}, \tag{A.5}$$

$$\sum_{s\in\mathcal{S}} q_{is}^{kt} \leq \min\{Q, C_i\} y_i^{kt} \qquad i\in\mathcal{C}, k\in\mathcal{K}, t\in\mathcal{T}, \tag{A.6}$$

$$\sum_{i\in\mathcal{C}}\sum_{s\in\mathcal{S}} q_{is}^{kt} \leq Q y_0^{kt} \qquad k\in\mathcal{K}, t\in\mathcal{T}, \tag{A.7}$$

$$\sum_{j\in\mathcal{N}:j\neq i} x_{ji}^{kt} = y_i^{kt} \qquad i\in\mathcal{N}, k\in\mathcal{K}, t\in\mathcal{T}, \tag{A.8}$$

$$\sum_{j\in\mathcal{N}:j\neq i} x_{ij}^{kt} = y_i^{kt} \qquad i\in\mathcal{N}, k\in\mathcal{K}, t\in\mathcal{T}, \tag{A.9}$$

$$\sum_{i\in\mathcal{B}}\sum_{j\in\mathcal{B}:j\neq i} x_{ij}^{kt} \leq \sum_{i\in\mathcal{B}} y_i^{kt} - y_\ell^{kt} \qquad \forall \mathcal{B}\subseteq\mathcal{C}, |\mathcal{B}|\geq 2, k\in\mathcal{K}, t\in\mathcal{T}, \ell\in\mathcal{B}, \quad \text{(A.10)}$$

$$\sum_{k\in\mathcal{K}} y_i^{kt} \leq 1 \qquad i\in\mathcal{C}, t\in\mathcal{T}, \quad \text{(A.11)}$$

$$I_{is}^t \geq 0 \qquad i\in\mathcal{C}, t\in\mathcal{T}, s\in\mathcal{S}, \quad \text{(A.12)}$$

$$q_{is}^{kt} \geq 0 \qquad i\in\mathcal{C}, k\in\mathcal{K}, t\in\mathcal{T}, s\in\mathcal{S}, \quad \text{(A.13)}$$

$$w_{is}^t \geq 0 \qquad i\in\mathcal{C}, t\in\mathcal{T}, s\in\mathcal{S}, \quad \text{(A.14)}$$

$$r_{ijs}^t \geq 0 \qquad i\in\mathcal{C}, j\in\mathcal{C}, t\in\mathcal{T}, s\in\mathcal{S}, \quad \text{(A.15)}$$

$$y_i^{kt} \in \{0,1\} \qquad i\in\mathcal{N}, k\in\mathcal{K}, t\in\mathcal{T}, \quad \text{(A.16)}$$

$$x_{ij}^{kt} \in \{0,1\} \qquad (i,j)\in\mathcal{A}, k\in\mathcal{K}, t\in\mathcal{T}. \quad \text{(A.17)}$$

The objective function (A.1) maximises the total profit, given by the total revenue minus the sum of transportation, transshipment and inventory costs at the customers. Constraints (A.2) and (A.3) balance the inventory at the customers locations, taking into account possible transshipments. Constraints (A.4) guarantee customers demand satisfaction. Constraints (A.5) impose an inventory limit at every customer location. Constraints (A.6) link the delivery and visit variables, not allowing deliveries to customers that are not visited. Constraints (A.7) ensure that vehicles capacity is not exceeded, while constraints (A.8) and (A.9) guarantee the vehicle flow conservation at every site. Constraints (A.10) correspond to subtour elimination constraints, while constraints (A.11) limit the number of visits to each customer in each period. Finally, the decision variables domain is given by constraints (A.12)–(A.17).

Let $\bar{q}_{is}^{kt}$ be the optimal delivery quantity of product of age $s\in\mathcal{S}$ to customer $i\in\mathcal{C}$ by vehicle $k\in\mathcal{K}$ in period $t\in\mathcal{T}$, determined by solving the IRP model (A.1)–(A.17). Then, the lot-sizing model is stated as follows:

$$\min \quad \sum_{t\in\mathcal{T}}(fz^t + vp^t) + \sum_{t\in\mathcal{T}}\sum_{s\in\mathcal{S}} h_0 I_{0s}^t \quad \text{(A.18)}$$

$$\text{s.t.} \quad I_{0s}^t = p^t - \sum_{i\in\mathcal{C}}\sum_{k\in\mathcal{K}} \bar{q}_{is}^{kt} \qquad t\in\mathcal{T}, s=0, \quad \text{(A.19)}$$

$$I_{0s}^t = I_{0,s-1}^{t-1} - \sum_{i\in\mathcal{C}}\sum_{k\in\mathcal{K}} \bar{q}_{is}^{kt} \qquad t\in\mathcal{T}, s\in\mathcal{S}\backslash\{0\}, \quad \text{(A.20)}$$

$$p^t \leq \min\left\{P, \sum_{i\in\mathcal{C}}\sum_{\tau=t}^{|\mathcal{T}|} d_i^\tau\right\} z^t \qquad t\in\mathcal{T}, \quad \text{(A.21)}$$

$$\sum_{s\in\mathcal{S}} I_{0s}^t \leq C_0 \qquad t\in\mathcal{T}, \quad \text{(A.22)}$$

$$p^t \geq 0 \qquad t \in \mathcal{T}, \tag{A.23}$$

$$I_{0s}^t \geq 0 \qquad t \in \mathcal{T}, s \in \mathcal{S}, \tag{A.24}$$

$$z^t \in \{0,1\} \qquad t \in \mathcal{T}, \tag{A.25}$$

The objective function (A.18) aims to minimise production and inventory costs at the plant. Constraints (A.19) and (A.20) guarantee the inventory balance at the plant, taking into account the delivery quantities that must be made available at customer locations. Constraints (A.21) impose production capacity and make the production setup variables to be equal to one if production takes place. Constraints (A.22) impose an inventory limit at the plant. Finally, the variables domain is stated by (A.23)–(A.25).