Stochastic Inventory Routing for Perishable Products
Heuristic and Route Generation Approaches

Toke C. Zinn, Cormac Hughes, and Thomas Mortensen

October 21 2019

Abstract

Modifications to existing solution methods for the Stochastic Inventory Routing Problem with Perishable products are compared on the parameters profit, waste generation, and computational time. The new methods yield larger profit than the fastest of the existing methods but have slower computational times. Similarly, the new methods perform worse than the most advanced existing methods, but are mostly also faster, at least for small problems.

1 Introduction

The setting of this paper is that of a chain of retail stores. The problem consists of distributing homogeneous goods from a central storage facility to different stores based on their respective demands. The problem is very close to the well known *inventory routing problem* abbreviated IRP. The IRP, however, makes some assumptions that do not quite fit our setting. Firstly, demand is assumed to be known. Furthermore, shelf-life and capacity of trucks is typically assumed to be unbounded. Therefore, extensions are made to the IRP in order to accommodate the added complexity of this problem. The full model describes the *Stochastic Inventory Routing Problem with Perishable Products* - also abbreviated SPIRP - where demand is stochastic and products perish after a known time period. Furthermore, vehicles have a limited capacity.

In the following section we describe the SPIRP mathematically as well as methods for solving it.

1.1 Inventory Routing Problem

First, we consider a scenario where we can drive to any node from any node, where node is either a store or the central depot. That is, we assume the associated graph $G = (V, E)$ to be complete. Here V is the set of vertices and E is the set of edges. Furthermore, we make a distinction between how a node is visited, i.e. it matters which path we took in order to reach node $i \in V$. Specifically, we are interested in the set of routes. Furthermore, any route must start and end at the depot. Assume, without loss of generality, that the depot is indexed by $0 \in V$. Specifying the depot immediately allows us the define the set of stores $S = V \backslash \{0\}$. The set of routes, $\mathcal{R}$, can be constructed from our graph as

$$\mathcal{R} \overset{def}{=} \{(0, r_1, r_2, \ldots, r_N, 0) \mid N \in \mathbb{N}, r_i \in S, r_i \neq r_j,\ \forall i, j \in \{1, 2, \ldots, N\}\}.$$

Finally, we wish to model the inventory over time. Therefore, for $T \in \mathbb{N}$ fixed, let $\mathcal{T}$ be a discrete and finite time index set, i.e.

$$\mathcal{T} \overset{def}{=} \{1, 2, \ldots, T\}.$$

1.2 Data

We now consider the data of the model. First we have the vehicle capacity Q and the maximum distance of each route M. Furthermore, the price of our product is p and we pay a to acquire our product.

Each store has a maximum capacity of $C_i \in \mathbb{N}$ for $i \in S$. For each period $t \in \mathcal{T}$ each store $i \in S$ has a demand of $D_{t,i}$.

We consider a scenario where dispatching a truck is free and we only pay for cost of travelling, which we assume to be equal to the distance travelled. The cost of travelling between node $i \in V$ and node $j \in V$ is $c_{i,j}$ and is subsequently also the distance between the nodes by assumption. We also define the cost of a route. Let $r \in \mathcal{R}$ be represented by $(0, r_1, r_2, \ldots, r_N, 0)$, we then define $c_r = c_{0,r_1} + c_{r_N,0} + \sum_{i=1}^{N-1} c_{r_i,r_{i+1}}$.

1.3 Variables

We first consider the delivery quantities. On route r at time t we wish to decide the delivery quantity $y_{t,i,r}$ to be delivered to store i if $i \in r$. Here $i \in r$ means that if $r = (0, r_1, r_2, \ldots, r_N, 0)$ then $\exists j \in \{1, 2, \ldots, N\} : i = r_j$ for $i \in S$. The sales of store $i \in S$ for time $t \in \mathcal{T}$ is denoted $s_{t,i}$. Consequently, we think of the inventory as a variable although it will be uniquely determined from the sales, delivery quantities, and initial inventory. The inventory for store $i \in S$ at time $t \in \mathcal{T}$ is denoted $I_{t,i}$. We also include a decision variable taking route $r \in \mathcal{R}$ at time $t \in \mathcal{T}$ which we denote $Z_{t,r}$.

1.4 The Objective Function

We wish to maximise profit. The profit of each store for a fixed time disregarding delivery costs is given by

$$p \cdot s_{t,i} - a \cdot y_{t,i}.$$

The delivery cost depends on the route $r \in \mathcal{R}$ and whether we take this route at time t, i.e.

$$c_r z_{t,r}.$$

The full problem can then be given by

$$\sum_{t \in \mathcal{T}} \left(\sum_{i \in S} (p \cdot s_{t,i} - a \cdot y_{t,i}) - \sum_{r \in \mathcal{R}} c_r z_{t,r} \right)$$

1.5 Constraints

We now formulate the constraints. Clearly, we will sell as much as is demanded if it is possible, but we cannot sell more than what is demanded, so

$$s_{t,i} \leq D_{t,i},\ \forall (t,i) \in \mathcal{T} \times S$$

However, we cannot sell more than the current inventory plus what we get delivered, which we can express as

$$s_{t,i} \leq I_{t,i} + y_{t,i},\ \forall (t,i) \in \mathcal{T} \times S$$

For each route we can only deliver if we make use of that route and the sum of quantities is less than our capacity, so

$$\sum_{i \in r} y_{t,i,r} \leq Q \cdot Z_{t,r},\ \forall (t,r) \in \mathcal{T} \times \mathcal{R}.$$

Additionally, each delivery quantity cannot exceed the capacity of a store

$$\sum_{r \in R : i \in r} y_{t,i,r} \leq C_i - I_{t,i},\ \forall (t,i) \in \mathcal{T} \times S.$$

Here the notation $r \in \mathcal{R} : i \in r$ means $r \in \{r \in \mathcal{R} \mid i \in r\}$, i.e. the colon can be read as "such that" as usual. Finally, we update our inventory based on sales and deliveries

$$I_{t+1,i} = I_{t,i} + y_{t,i} - S_{t,i},\ \forall (t,i) \in (\mathcal{T} \setminus \{T\}) \times S,$$

and the maximum length of a route must be at most M

$$Z_{t,r} c_r \leq M,\ \forall (t,r) \in \mathcal{T} \times \mathcal{R}.$$

Note that the final constraint is obsolete, since for any problem you would never generate routes that are too long. We simply include it since we do not define the set of routes under any constraints.

1.6 Full Formulation

We present the full IRP.

$$\begin{array}{lllll}
\max & \sum_{t \in \mathcal{T}} \left(\sum_{i \in S} (p \cdot s_{t,i} - a \cdot y_{t,i}) - \sum_{r \in \mathcal{R}} c_r z_{t,r} \right) & & & \\
\text{s.t.} & s_{t,i} & \leq D_{t,i}, & \forall (t,i) \in \mathcal{T} \times S & (1.a) \\
& s_{t,i} & \leq I_{t,i} + y_{t,i} & \forall (t,i) \in \mathcal{T} \times S & (1.b) \\
& \sum_{i \in r} y_{t,i,r} & \leq Q \cdot Z_{t,r} & \forall (t,r) \in \mathcal{T} \times \mathcal{R} & (1.c) \\
& \sum_{r \in \mathcal{R} : i \in r} y_{t,i,r} & \leq C_i - I_{t,i} & \forall (t,i) \in \mathcal{T} \times S & (1.d) \\
& I_{t+1,i} & = I_{t,i} + y_{t,i} - S_{t,i} & \forall (t,i) \in (\mathcal{T} \setminus \{T\}) \times S & (1.e) \\
& Z_{t,r} c_r & \leq M & \forall (t,r) \in \mathcal{T} \times \mathcal{R} & (1.f) \\
& Z_{t,r} \in \{0,1\}, & s_{t,i}, I_{t,i}, y_{t,i,r} \in \mathbb{N} & \forall (t,i,r) \in \mathcal{T} \times S \times \mathcal{R} & (1.g)
\end{array} \tag{1}$$

Recall, that $y_{t,i} = \sum_{r \in \mathcal{R}} y_{t,i,r}$ for $i \in S$ and $t \in \mathcal{T}$. Below is a provided table of the sets, data, and variables.

Type	Symbol	Description
Set	V	Nodes - stores and depot
Set	S	Stores ($S = V\backslash\{0\}$)
Set	$\mathcal{T}$	Time
Set	$\mathcal{R}$	Routes
Data	Q	Vehicle Capacity
Data	M	Maximum Vehicle Distance
Data	C_i	Capacity of store $i \in S$
Data	$c_{i,j}$	Cost/Distance between nodes i and j in V
Data	c_r	Cost/Distance of a route $r \in \mathcal{R}$
Data	$D_{t,i}$	Demand at store i in time t
Variable	$Z_{t,r}$	Decision variable at time t for route r
Variable	$y_{t,i,r}$	Delivery quantity at time t to store i in route r
Variable	$I_{t,i}$	Inventory of store i at time $t \in \mathcal{T}$

Table 1: Table of Model Elements

To comment on the complexity of the problem consider the amount of variables needed. First we have $|\mathcal{T}|$ and $|V|$ which both grow linearly with each point. However, when we consider $|\mathcal{R}|$ we see why the problem quickly becomes intractable.

If we choose the number of stores $|S|$ then we get

$$|\mathcal{R}| = \sum_{s=1}^{|S|} \binom{|S|}{s} s! \tag{2}$$

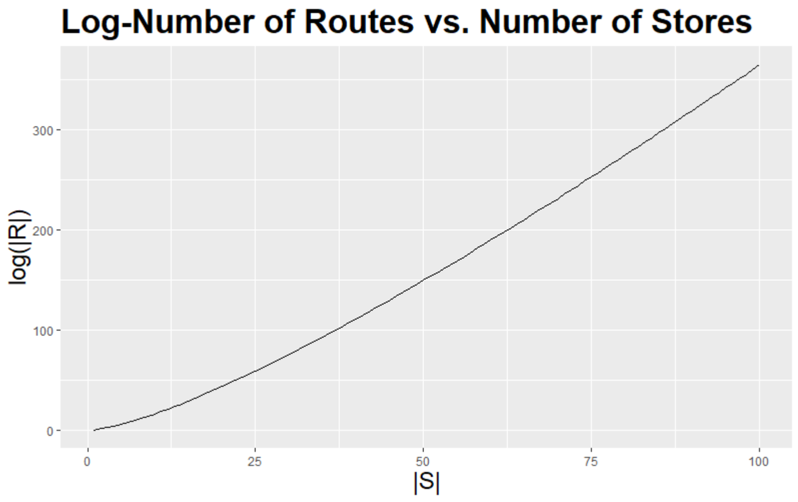


Figure 1: $\log(|\mathcal{R}|)$ vs. $|S|$

Figure 1 shows that the growth seems to be greater than exponential growth. For instance, if $|S| = 25$ we have $|\mathcal{R}| \approx e^{59}$ which is already a very large number of routes. However, it is not unnatural for the number of stores to be much larger than 25 as noted by (Crama et al., 2018).

1.7 The Stochastic Inventory Routing Problem with Perishable Products

We now alter Problem (1) to include stochastic demand. In the IRP we assumed $D_{t,i} \in \mathbb{N}$ for every $(t,i) \in \mathcal{T} \times S$. Now we assume there is some underlying probability space $(\Omega, \mathcal{F}, \{\mathcal{F}_t\}_{t\in\mathcal{T}}, \mathbb{P})$ where $D_{t,i} : \Omega \to \mathbb{N}$ is a random variable on said probability space. The Problem (1) can remain unchanged if we then solve the problem ω-by-ω, i.e. pointwise for every single scenario of the value of $D_{t,i}(\omega)$. This formulation does not make particular sense, since we will never be able to determine the ω that we are observing. Instead it is natural to formulate the problem based on either expected values or some level of probability.

Furthermore, we now also consider perishing products. In order to describe a product perish we need to distinguish between the remaining life-time of our product. We introduce the set $\mathcal{L}$ to be the set of states of "decay" our product can be in with $L = |\mathcal{L}|$ being the freshest product. A product is decayed it it reaches remaining life-time 0. Therefore, we also introduce the parameter $I_{t,i,l}$ to be the inventory at time $t \in \mathcal{T}$ for store $i \in S$ of product with remaining lifetime $l \in \mathcal{L}$. Clearly, we have $I_{t,i} = \sum_{l\in\mathcal{L}\setminus\{0\}} I_{t,i,l}$.

The different approaches to this problem typically requires a reformulation, and we will therefore not provide a full formulation since it will change depending on the solution method.

1.7.1 Reducing the Complexity

We now introduce some ways to make the problem less computationally dense. The first method is to disaggregate the problem into two separate problems; an inventory problem and a routing problem. By disaggregating the problem we obtain a heuristic solution. First, we solve an inventory problem to determine the delivery amounts and then solve a VRP to optimally deliver the determined quantities. This means we know $y_{t,i}$ as a constant before solving the vehicle routing part of the problem. Decomposition is used to some extent in all methods presented by (Crama et al., 2018).

This allows us to restrict the number of routes in a natural way; either explicitly in a route-based formulation or implicitly in a flow-based formulation. Specifically, if we know what we're going to deliver, then we can define the set of feasible routes

$$\mathcal{R}_t^{M,Q} \overset{def}{=} \left\{ r \in \mathcal{R} \;\middle|\; \sum_{i\in r} y_{t,i} \leq Q, c_r \leq M \right\}. \tag{3}$$

By defining $\mathcal{R}_t^{M,Q}$ we both reduce the number of variables and the number of constraints. If the values Q and M provide strict bounds, then it may greatly reduce the number of routes.

Note that $\mathcal{R}_t^{M,Q}$ may not be completely determined yet; if only $y_{t,i} = k$ is predetermined, then all we know is that $\sum_{r\in\mathcal{R}} y_{t,i,r} = k$ and we still have yet to determine how much to deliver on each route. However, approximations to $R^{M,Q}$ are available by heuristically enforcing that we want to deliver as much as possible on each visit to a store.

Consider the traditional VRP. Here, each node must only be visited once, and if $y_{t,i}$ is predetermined, then there exists a *unique* $r \in R$ such that $y_{t,i,r} = y_{t,i}$, i.e. the entire delivery to a given store is made on a single route and by extension our generation of $\mathcal{R}_t^{M,Q}$ is exact. If we are allowed to visit each store multiple times in different routes then we enforce deliveries to be as large as possible, i.e. assume $y_{t,i,r} = y_{t,i}$ for some $r \in \mathcal{R}$. We then add a margin to the capacity. Doing this will only affect the length of the generated routes, not total quantities delivered or the capacity since the route is constrained in our VRP. Hence, we define

$$\mathcal{R}_t^{M,Q} \overset{def}{=} \left\{ r \in \mathcal{R} \;\middle|\; \sum_{i \in r} y_{t,i} \leq Q - 1 + \min_{i \in S}\{y_{t,i} \mid y_{t,i} > 0\}, c_r \leq M \right\}, \tag{4}$$

where $-1 + \min_{i \in S}\{y_{t,i} \mid y_{t,i} > 0\}$ is the margin. Choosing this margin means that we can add a store if we are only able to make a partial delivery, i.e. no further full deliveries can be added to the route. In this way, if there are spare goods on a truck, it may deliver the rest of its goods to one more store even if it cannot satisfy the total delivery required by that store. Thus two trucks with some spare goods can share the delivery to the final store between them. If we consider single store visits we use the definition in Equation (3), if we allow for multiple visits we use the definition in Equation (4).

Furthermore, in the stochastic setting it is natural maximise the *expected* profit rather than the full problem or some other reduction from a random variable to a number. Choosing some way of reducing the random variables to a number will reduce the problem from requiring and ω-by-ω or point-wise solution to a single solution that is optimal for example in expected value. Since the number of different outcomes of our demand may be infinite such a method is definitely required in order to find a solution.

2 Previous Methods

(Crama et al., 2018) looks at four main methods for solving the SPIRP. These methods are of varying complexity and seek to solve the problem by trying to reach quite different goals. The methods are in order of increasing complexity: The *Expected Value* (EV) method, the *Deliver-up-to-Level* (UL) Method, the *Decomposition* (DE) method, and finally the *Decomposition-Integration* (DI) method. In this section, we briefly look at each of these methods.

2.1 The Expected Value Method

The first method is named the expected value method since the idea of this method is simply to take the expected value of the demand and then use this as you would a deterministic demand. This method therefore seeks simply solve a PIRP. This method as well as the following two methods from (Crama et al., 2018) completely decomposes the problem into finding the delivery amounts and then solving a VRP to best deliver these quantities. The method can be decomposed into 3 steps for each time period.

We see that the delivery quantities are found using a simple formula that greedily sets $y_{ti} = 0$ when there is enough inventory to satisfy the expected demand, and otherwise simply supplies the store until it has goods equal to the expected demand of L periods or until the store is full if this happens first. That means that even though the complexity has been reduced to that of an IRP, the method still employs a quite simple heuristic to solve this problem.

Algorithm 1 Expected Value Method

```
for $t \in \mathcal{T}$ do
    for $i \in \mathcal{S}$ do
        if $I_{ti} \geq E(D_{ti})$ then
            $y_{ti} = 0$
        else
            $y_{ti} = \min\{C_i - I_{ti}, \quad \lfloor E(D_{ti} + \cdots + E(D_{t+L-1,i}) \rfloor - I_{ti}\}$
        end if
    end for
    Solve VRP for the delivery quantities $y_{t,i}$.
    Observe the realised demand $D_{t,i}$.
    Update the system according to observed demand.
end for
```

2.2 The Deliver-up-to-level Method

This next method again replaces anything stochastic with something deterministic. In this case, however, the goal of the method is to have enough products in each store to avoid a stock-out a fixed percentage of the time. This percentage is called the *Target Service Level* denoted TSL. Before we continue we define q_{ti} as the smallest integer quantity which satisfies demand for L periods with probability TSL, i.e. it is given by $q_{ti} = \min\{\lambda \in \mathbb{N} \mid Pr(D_{ti} + \cdots + D_{t+L-1,i} \leq \lambda) \geq TSL\}$.

This method is quite similar to the expected value method as can be seen in Algorithm 2.

Algorithm 2 Deliver-up-to-Level Method

```
for $t \in \mathcal{T}$ do
    for $i \in \mathcal{S}$ do
        if $Pr(D_{ti} \leq I_{ti}) \geq TSL$ then
            $y_{ti} = 0$
        else
            $y_{ti} = \min\{C_i - I_{ti}, \quad q_{ti} - I_{ti}\}$
        end if
    end for
    Solve VRP for the delivery quantities $y_{t,i}$.
    Observe the realised demand $D_{t,i}$.
    Update the system according to observed demand.
end for
```

2.3 The Decomposition Method

The DE method relies on a Stochastic Dynamic Programming model in order to actually take revenue and delivery cost into account when determining what to deliver to each store. Recall that in the previous two methods we do not actually take profitability of a delivery into account when deciding delivery amount. Again the SDP is not solved to optimality in this case either due to computational complexity. Thus heuristic methods are required, and for this method the problem is decomposed to an SDP for each individual store. Specifically the method aims to determine the delivery quantities for each store that maximises the expected revenue of the store over T periods.

(Crama et al., 2018) propose the following recursive equation for the revenue over the planning horizon:

$$\begin{aligned} f_{ti}(X_{ti}, y_{ti}) = &-F_i \cdot \mathbb{1}(y_{ti} > 0) - a y_{ti} \\ &+ Pr(D_{ti} > I_{ti} + y_{ti})(p(I_{ti} + y_{ti}) + f^*_{t+1,i}(0, \ldots, 0)) \\ &+ \sum_{d=0}^{I_{ti}+y_{ti}} Pr(D_{ti} = d)(pd + f^*_{t+1,i}(X_{t+1,i})), \end{aligned} \tag{5}$$

where F_i is the *cost-to-serve* of each store at each time. (Crama et al., 2018) propose two different measures of this, one which is route based and one that is based on the distance from store i to its nearest neighbours. We show only the simpler version which is the distance based cost to serve, since the two methods perform quite similarly with the simple one performing slightly better. The cost to serve is given by

$$F_i = \sum_{j \in J_i} c_{ij}/|J_i|,$$

where J_i is the set of neighbours. The first term of (5) is thus just the cost-to-serve part of the profit for the store. The second term is the acquisition cost of the goods delivered to the store, while the final terms add up to the expected revenue of the store. The aim of the method is now to find

$$y_{ti} = \underset{q_{ti} \leq x \leq C_i - I_{ti}}{\arg\max} f_{ti}(X_{ti}, x) \tag{6}$$

for each store individually at each time period. The method can be summarised as follows:

Algorithm 3 Decomposition Method

```
for $i \in \mathcal{S}$ do
    $F_i = \sum_{j \in J_i} c_{ij}/|J_i|$
end for
for $t \in \mathcal{T}$ do
    for $i \in \mathcal{S}$ do
        Find $y_{ti}$ as given in (6) via (5).
    end for
    Solve VRP for the delivery quantities $y_{ti}$.
    Observe the realised demand $D_{ti}$.
    Update the system according to observed demand.
end for
```

2.4 The Decomposition-Integration Method

The final method shown in (Crama et al., 2018) builds upon the DE method. However, the expected profit of each store over the planning horizon is now calculated in a more realistic, but also far more complex way. The idea is to use the actual routing costs in period t rather than the simple cost to serve while also refining the estimate of the routing costs in the following periods. To maximise the expected profit over the planning horizon, (Crama et al., 2018) use a matheuristic to find both the routes and delivery amounts together for each individual period. We make no further descriptions of this method as it would take up a lot of space to properly introduce the method and it would diverge from the purpose of this paper.

3 Route-based Expected Value Method

In this paper, we direct our focus towards improving upon the EV method described in Section 2.1. This method does not solve (1) directly, but rather considers each time period individually, sets the delivery amounts as outline in Section 2.1, and then solve a VRP that satisfies these delivery amounts for the given time period. Assigning the delivery amounts is almost trivial, and it thus remains to solve the VRP which for a fixed time t is given by:

$$\begin{array}{lll} \max & \sum_{r\in\mathcal{R}} -c_r Z_{t,r} & \\ \text{s.t.} & \sum_{r\in\mathcal{R}} y_{t,i,r} \; = \overline{y}_{t,i} & \forall i \in S \\ & \sum_{i\in r} y_{t,i,r} \; \le Z_{t,r} Q & \forall r \in \mathcal{R} \\ & Z_{t,r} c_r \; \le M & \forall r \in \mathcal{R} \\ & y_{t,i,r} \in \mathbb{N}, Z_{t,r} \in \{0,1\} & \forall (r,i) \in \mathcal{R} \times S. \end{array} \tag{7}$$

Where $\overline{y}_{ti}$ denotes the delivery amounts predetermined by the EV method. We also note that nowhere in this problem is it stated that every store can only be visited once. Traditional VRPs typically have this rule, but it does not make much sense as a rule in this setup of trying to maximise profits for a chain of retail stores. To solve this problem, we choose to pre-generate all feasible routes described in Section 1.7.1. When considering only routes in $\mathcal{R}^{M,Q}$, the optimisation problem reduces to

$$\begin{array}{lll} \max & \sum_{r\in\mathcal{R}^{M,Q}} -c_r Z_{t,r} & \\ \text{s.t.} & \sum_{r\in\mathcal{R}^{M,Q}} y_{t,i,r} \; = \overline{y}_{t,i} & \forall i \in S \\ & \sum_{i\in r} y_{t,i,r} \; \le Z_{t,r} Q & \forall r \in \mathcal{R}_t^{M,Q} \\ & y_{t,i,r} \in \mathbb{N}, Z_{t,r} \in \{0,1\} & \forall (r,i) \in \mathcal{R}_t^{M,Q} \times S \end{array} \tag{8}$$

While this change did not seem to change the size of the program drastically, as we write it here on paper, we note that the set $\mathcal{R}^{M,Q}$ is usually much smaller than $\mathcal{R}$ which in turn leads to much fewer constraints and variables in the problem.

3.1 Myopic Polishing Procedure

When solving the inventory part of the problem we use a very "sharp" decision boundary for when to deliver or not. However, sometimes deliveries may not be *immediately* profitable. We now construct an example where a delivery in this period seems unnecessary.

Assume that store i has an inventory of 19 and the expected demand is 20. In this case we *expect* to sell 1 additional unit this period, i.e. our immediate expected profit from making a delivery is $(p-a)\left(\mathbb{E}[D_{t,i}] - I_{t,i}\right)$.

Now, let $r = (0, r_1, r_2, \ldots, r_{n-1}, i, r_{n+1} \ldots r_N, 0)$, i.e. we visit the store of interest in this period. We define $r \backslash \{i\} = (0, r_1, r_2, \ldots, r_{n-1}, r_{n+1}, \ldots, r_N, 0)$, i.e. we remove store i from the route r. Then the routing cost difference is given by $\Delta_{r,i} \overset{def}{=} c_r - c_{r\backslash\{i\}}$. Therefore, if

$$(p-a)\left(\mathbb{E}[D_{t,i}] - I_{t,i}\right) - \Delta_{r,i} < 0,$$

then we do not expect any immediate profit from making this delivery. Hence, we wish to postpone the delivery.

We fix a $t \in \mathcal{T}$. We first wish to identify stores with negative immediate profit and then remove the worst. We then rerun the problem to see if there are still nodes with negative immediate profit until no additional store is found.

Given a solution to the relaxed VRP we define R_t^* as the set of routes in the solution for $t \in \mathcal{T}$.

Algorithm 4 Myopic Polishing Procedure

```
while True do
  $\mathcal{H} = \emptyset$
  for $r \in \mathcal{R}_t^*$ do
    for $i \in r$ do
      if $(p-a)\left(\mathbb{E}[D_{t,i}] - I_{t,i}\right) - \Delta_{r,i} < 0$ then
        $\mathcal{H} = \mathcal{H} \cup \{(r,i)\}$
      end if
    end for
  end for
  if $\mathcal{H} \neq \emptyset$ then
    $(r^*, i^*) = \arg\min_{(r,i)\in\mathcal{H}} (p-a)\left(\mathbb{E}[D_{t,i}] - I_{t,i}\right) - \Delta_{r,i}$
    $\mathcal{R}_t^* = (\mathcal{R}_t^* \backslash \{r^*\}) \cup \{r^* \backslash \{i^*\}\}$
  else
    break
  end if
end while
```

The goal of Algorithm 4 is not necessarily to increase profit but rather provide insight into whether or not myopic decision making yield better results for the given problem.

4 Discussion

In this section, we analyse and discuss the results of the computational study. We compare and contrast our results with those of (Crama et al., 2018) and extend their work by considering other factors that could improve on their solutions. First, however, we quickly outline the parameters of the computational study.

4.1 Simulation

For the sake of comparison, we use the same parameters for the computational studies as those used by (Crama et al., 2018). We use the same $N = 40$ stores and consider $T = 30$ periods of time.

The demands are are i.i.d. binomial: $D_{ti} \sim \text{bin}(200, 0.1)$. The capacity of each store is equal and given by $C_i = 20L$. The capacity of each truck is fixed at $Q = 120$.

The cost $c_{i,j}$ of travelling from i to j is the Euclidean distance. Finally, acquisition costs are set to $a = 6$ while the sales price is $p = 10$.

Finally, the initial inventory of a store follows a discrete uniform distribution $I_{0,i} \sim \text{unif}(0, 20L-10)$, and the initial inventory has shelf life $L - 1$. We note however, that due to limited time, our computational studies will only be 10 scenarios for each experiment and will therefore be subject to more variance than the results of (Crama et al., 2018).

4.2 Results

Our aim for this project was to first recreate, and then hopefully improve upon, at least one of the methods shown by (Crama et al., 2018). To give ourselves the best chance success, we decided to implement the simplest method, the Expected Value Method, outlined in Section 2.1. The simplicity of the inventory control algorithm allowed us to focus our attention on the VRP with the goal of matching the profit figures of (Crama et al., 2018) for 30 periods while attempting to close in on the run-time targets. (Crama et al., 2018) also recorded data for a number of key metrics including waste, customer service level and freshness. However these figures should not change in a re-implementation of the problem - as long as the profit is the same over a number of demand scenario simulations, then our solution should be the same. We note that (Crama et al., 2018) appear to have included the TSP constraint that every store can only be visited once per time period in their implementation despite never mentioning it in the problem formulation. For this reason, we choose to implement the problem both with and without this constraint.

Geographic data was provided by the authors of (Crama et al., 2018) and averaging over various demand scenarios generated from the same distribution gave us comparable results. We show these results in Table 2.

	L=2		L=3		L=4	
Method	Time (sec)	Profit ($)	Time (sec)	Profit ($)	Time (sec)	Profit ($)
EV	4	66,542	3	72,320	0	72,182
EV_{RG}	-	-	11	72,767	0	72,005
EV_{IR}	-	-	602	73,021	40	74,615
EV_{MP}	-	-	-	-	50	75,048

Table 2: Table of Results. The first row are results from (Crama et al., 2018) while the following three rows are the EV method with pregenerated routes for the VRP, EV_{RG}, the EV method with intersecting routes, EV_{IR}, and the myopically polished EV method, EV_{MP}, respectively.

As can be seen, we have not run any of the methods for $L = 2$. This is because the method with pregenerated routes we use for the VRP explodes in memory usage in this case and as such, we would need a bigger machine to run calculations on; and even in that case it would likely be very very slow.

As expected for our route-generation approach, we acquire a similar profit figure for $L = 3$ and $L = 4$, but fall short on computational speeds when solving larger, non-trivial VRPs such as is the case for $L = 3$. The more advanced methods of (Crama et al., 2018) obtain even higher profits, but are also slower again. For $L = 4$ the most advanced and best performing method of (Crama et al., 2018) takes 1597 seconds per iteration on average, but it does provide almot 12% higher profits than the EV method. In contrast, the EV_{MP} method surpasses the EV method by only $(75,048 - 72,005)/72,005 \approx= 4.23\%$, but also used only 50 seconds on average per iteration.

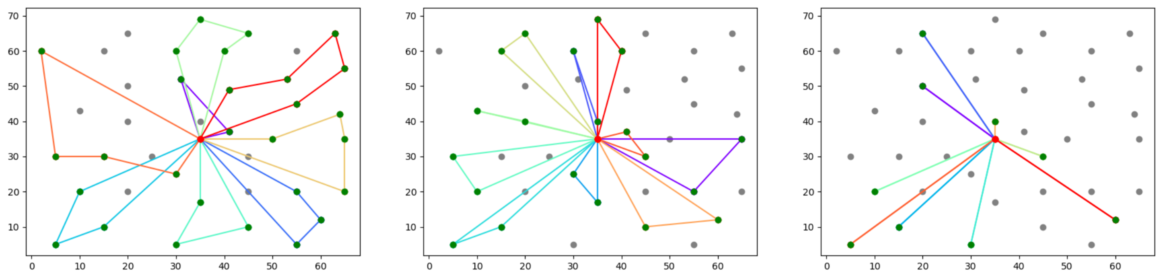
Shelf-Life	Method	Waste
L=3	EV	0.103
	EV_{RG}	0.050
	EV_{IR}	0.050
	EV_{MP}	
L=4	EV	0.084
	EV_{RG}	0.041
	EV_{IR}	0.041
	EV_{MP}	0.039

Table 3: Table showing products going to waste for various methods and shelf lives.

We see, from Table 3 that the waste generation from the methods introduced here are quite a bit smaller than the ones of (Crama et al., 2018), however, we suspect that it is simply a coincidence that the EV_{RG} method generates less waste on the 10 randomly generated scenarios we consider. This is because the allocation of inventory follows the exact same method as the EV method. Further, we note that the waste of EV_{RG} and EV_{IR} are the same in both cases, which is as expected given that the same goods are delivered to the same stores at the same time for both of these methods. The only thing that sets these methods apart are the routes used to deliver the goods. Finally, we see that the EV_{MP} method produces slightly less waste than any of the others which is not very surprising giving that the method strategically chooses not to deliver at certain times, but is otherwise the same. Finally, it should be noted that the more advanced methods of (Crama et al., 2018) produce much less waste than any method here.

Next, let us consider the impact of shelf life on routing within the setup. Figure 2 shows that the number of stores per route is greatly dependent on the shelf-life which of course is because the delivery sizes are much larger for products with longer shelf lives. We also see that the number of deliveries in each period decreases as shelf life increases which is for a very similar reason; less frequent deliveries are required when deliveries are larger.

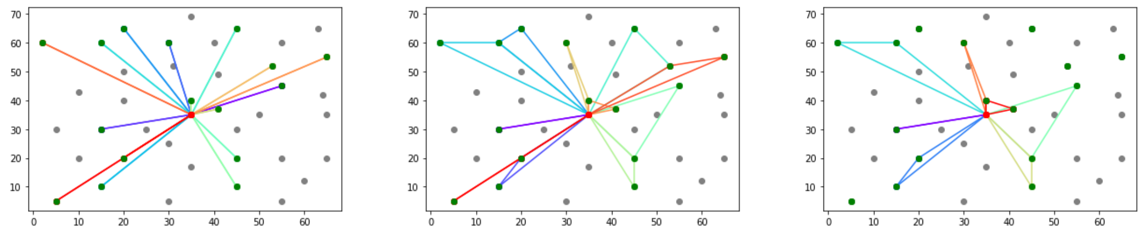
Figure 3 shows the difference between the EV_{RG}, EV_{IR} and EV_{MP} for a single period. We see that initially each (active) store is visited exactly once. Then we allow for multiple visits, which means that some routes are eliminated, reducing distribution costs. Finally, by myopically polishing the solution we



(a) L=2 (b) L=3 (c) L=4

Figure 2: Single period solutions to VRP for varying shelf lives with $Q = 120$.

see that some nodes simply aren't worth visiting right now, at least under expected value assumptions.



(a) Single Visit Solution (b) Multiple Visits Solution (c) Myopically Polished Solution

Figure 3: Difference between the EV_{RG}, EV_{IR}, and EV_{MP} for $L = 4$ and $Q = 120$.

Remark: We note that the Myopic Polishing Procedure is an extension that can be added to any solution. However, if we only allow for single store visits, i.e. we solve an explicit VRP in the expected value method rather than a relaxed VRP that allows for multiple visits, then the method will tend to remove too many stores. In general if we are only driving to a single store, then the travel cost of each route is quite large, and the procedure will therefore eliminate too many stores. If we allow partial deliveries, then it may be worth to visit the store, since it is close to a nearby store. We therefore recommend that one only applies the Myopic Polishing Procedure if the routes visit multiple stores, as it will otherwise tend to remove most deliveries.

4.3 Managerial Insights

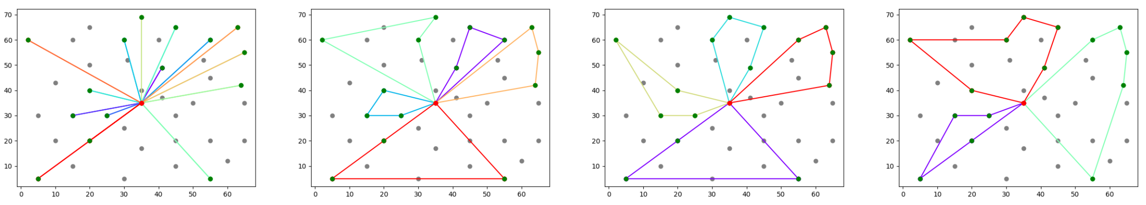
(Crama et al., 2018) test over three different product shelf lives - two, three, and four periods - which they claim impacts the store capacity and therefore the delivery quantities since smaller sections are set aside for products with lower shelf-lives while larger inventory space and delivery quantities are reserved for nonperishables. However, the *vehicle* capacities are fixed regardless of the product they are carrying. This is a realistic assumption but means trucks dedicated to delivering products with long shelf lives are only ever able to serve one store per run as shown in Figure 2c which depicts a trivial VRP solution with unutilised truck capacity. This is addressed by either:

- Only looking at products with L=2 or L=3 shelf lives, and solving the non-trivial VRPs.

- Arbitrarily increasing the truck capacities to allow for multiple customers per route (Figure 4).
- Deviating from TSP assumptions and allowing 'split-deliveries' to make use of surplus vehicle capacity, i.e. multiple trucks can serve the same customer (Figure 3).

In fact, while appearing as the obvious solution, splitting customer orders between routes only exacerbates the issue of computational complexity to the point where run times explode as seen for EV_{IR} for $L = 3$ in Table 2. It does significantly reduce the number of routes needed to service all active customers and while there is no explicit cost assigned to each route, you can imagine reducing the number of trucks would decrease both fixed and variable costs for the period and would be desirable for any manager of a company. Simultaneously, the EV_{IR} and EV_{MP} methods decrease the total distance travelled which *does* have explicit costs assigned to it. This is seen in the higher profit figures obtained and outlined in Table 2.

The setup we consider is a very ideal one. We have a single homogeneous product and all available capacity in the truck is used for this single product. However, outside the confines of academia it is safe to assume that a truck will not only deliver a single product, but rather multiple products with varying shelf-lives. The translation between our results and the generalised multi-product problem can be thought of as pre-assigning a part of the vehicles capacity to each product. However, an integration procedure over the different solution routes for each product must be applied in order to obtain a solution in the multi-product setting.



(a) Q=120 (b) Q=240 (c) Q=320 (d) Q=400

Figure 4: Single period solutions to VRP for varying capacities with $L = 4$.

5 Conclusion

We saw improvements to the expected value method without increasing the complexity of the solution method to a significant degree. The improvement to the profits, however, are accompanied by an increase in computational complexity as well as increased memory requirements as we saw especially for $L = 2$ where the number of feasible routes explode, especially when allowing intersecting routes. Further experimentation with a myopic polishing procedure of solution also proved somewhat effective in producing profits for $L = 4$ again without very large increases in computational time. However, these rather simple improvements do not quite follow the improvements seen for the DI method in (Crama et al., 2018). However, that method does also mostly use more time. In conclusion, there is merit to simply using extensions of the EV method in certain situations, as well as using pregenerated routes for the VRP. However, the pregeneration cannot be widely used unless the problem is of limited size.

Furthermore, there are better methods than extensions of the EV method if closing the optimality gap and other parameters such as waste and service level are prioritised higher than quicker run times.

References

Crama, Yves et al. (2018). “Stochastic Inventory Routing for Perishable Products”. In: *Transportation Science* 52.3, pp. 526–546.